LAB 1

**A.1 WAP to print your name in console.**

import 'dart:io';

void main() {

  print("XYZ");

  stdout.write("XYZ");

}

**A.2 WAP to Print addition of 2 number.**

import 'dart:io';

void main() {

  stdout.write('Enter an integer: ');

  int a = int.parse(stdin.readLineSync()!);

  stdout.write('Enter an integer: ');

  int b = int.parse(stdin.readLineSync()!);

  print('Sum :${a + b}');

}

**A.3 WAP to convert temperature from Fahrenheit to Celsius.**

import 'dart:io';

void main() {

  stdout.write('Enter a Fahrenheit: ');

  int a = int.parse(stdin.readLineSync()!);

  print('Celsius :${((a - 32) \* 5) / 9}');

}

**A.4 WAP to find percentage of 5 subject.**

import 'dart:io';

void main(List<String> args) {

  int sum = 0, num = 0;

  for (int i = 0; i < 5; i++) {

    print("Enter marks: ");

    num = int.parse(stdin.readLineSync()!);

    sum += num;

  }

  print('Percentage : ${(sum \* 100) / 500}');

}

**A.5 WAP that reads a number in meters, converts it to feet, and display the result.**

import 'dart:io';

void main() {

  stdout.write('Enter a meter: ');

  int m = int.parse(stdin.readLineSync()!);

  double f = double.parse(m.toString()) \* 3.28084;

  print('Feet :$f');

}

**A.6 Body Mass Index (BMI) is measure of health on weight. It can be calculated by taking your weight in kilograms and dividing by square of your height in meters. Write a program that prompts the user to enter a weight in pounds and height in inches and display the BMI.**

**Note: 1 pound=.45359237 Kg and 1 inch=0.254 meters.**

import 'dart:io';

void main() {

  stdout.write("Enter your weight in pounds: ");

  double weight = double.parse(stdin.readLineSync()!) \* 0.45359237;

  stdout.write("Enter your height in inches: ");

  double height = double.parse(stdin.readLineSync()!) \* 0.0254;

  double bmi = weight / (height \* height);

  print("Your Body Mass Index (BMI) is: ${bmi.toStringAsFixed(2)}");

}

**B.1 Write a program to find the subarray with the largest sum from a given integer array. Example: Input: nums = [-2, 1, -3, 4, -1, 2, 1, -5, 4], Output: 6 (subarray [4, -1, 2, 1]).**

int maxSubArray(List<int> nums) {

int currentSum = nums[0];

int maxSum = nums[0];

for (int i = 1; i < nums.length; i++) {

// Decide whether to extend the current subarray or start a new one

currentSum = (currentSum + nums[i]).clamp(currentSum + nums[i], nums[i]);

// Update maxSum if the current subarray sum is greater

if (currentSum > maxSum) {

maxSum = currentSum;

}

}

return maxSum;

}

void main() {

List<int> nums = [-2, 1, -3, 4, -1, 2, 1, -5, 4];

print('Maximum Subarray Sum: ${maxSubArray(nums)}');

}

**B.2 Write a program to check if a given positive integer is an ugly number (a number whose only prime factors are 2, 3, and 5). Example: Input: n = 6, Output: true (6 = 2 × 3); Input: n = 14, Output: false (14 includes the prime factor 7).**

bool isUgly(int n) {

if (n <= 0) return false; // Ugly numbers are positive integers

// Continuously divide by 2, 3, and 5 if divisible

for (int factor in [2, 3, 5]) {

while (n % factor == 0) {

n ~/= factor; // Integer division

}

}

// If the resulting number is 1, then it's an ugly number

return n == 1;

}

oid main() {

int n1 = 6;

int n2 = 14;

print('$n1 is ugly? ${isUgly(n1)}'); // Output: true

print('$n2 is ugly? ${isUgly(n2)}'); // Output: false

}

**B.3 Write av program to find all the prime numbers within a given range [start, end]. Example: Input: start = 10, end = 20, Output: [11, 13, 17, 19]; Input: start = 1, end = 10, Output: [2, 3, 5, 7].**

bool isPrime(int n) {

if (n <= 1) return false;

if (n == 2) return true;

if (n % 2 == 0) return false;

// Check for factors from 3 to the square root of n

for (int i = 3; i <= (n ~/ i); i += 2) {

if (n % i == 0) {

return false;

}

}

return true;

}

List<int> findPrimesInRange(int start, int end) {

List<int> primes = [];

for (int i = start; i <= end; i++) {

if (isPrime(i)) {

primes.add(i);

}

}

return primes;

}

void main() {

int start1 = 10, end1 = 20;

int start2 = 1, end2 = 10;

print('Primes between $start1 and $end1: ${findPrimesInRange(start1, end1)}'); // Output: [11, 13, 17, 19]

print('Primes between $start2 and $end2: ${findPrimesInRange(start2, end2)}'); // Output: [2, 3, 5, 7]

}

LAB 2

**A.1 Write a dart program to check whether the given number is positive or negative.**

import"dart:io";

voidmain(){

  print("Enter any number");

  intnum = int.parse(stdin.readLineSync()!);

  if(num>0){

    print("$num : is positive");

  }

  else{

    print("$num : is nrgative");

  }

}

**A.2 Write a dart program to perform addition, subtraction, multiplication, division based on user choice using if, if..else..if & switch.**

import"dart:io";

voidmain(){

//if

  //if-else

  print("Enter 1st number : ");

  double a = double.parse(stdin.readLineSync()!);

  print("Enter 2nd number : ");

  double b = double.parse(stdin.readLineSync()!);

  double ans;

  print("Enter choice : ");

  int choice = int.parse(stdin.readLineSync()!);

  if(choice == 1){

    ans = a + b;

    print("Addition of $a and $b is : $ans");

  }

  elseif(choice == 2){

    ans = a - b;

    print("Substraction of $a and $b is : $ans");

  }

  elseif(choice == 3){

    ans = a \* b;

    print("Multiplication of $a and $b is : $ans");

  }

  elseif(choice == 4){

    ans = a / b;

    print("Division of $a and $b is : $ans");

  }

  else{

    print("Enter valid choice");

  }

  //switch-case

  print("Enter 1st number : ");

  double x = double.parse(stdin.readLineSync()!);

  print("Enter 2nd number : ");

  double y = double.parse(stdin.readLineSync()!);

  double ans1;

  print("Enter choice : ");

  int choice1 = int.parse(stdin.readLineSync()!);

  switch(choice){

    case1 :

      ans1 = x + y;

      print("Addition of $x and $y is : $ans1");

      break;

    case2 :

      ans1 = x - y;

      print("Substraction of $x and $y is : $ans1");

      break;

    case3 :

      ans1 = x \* y;

      print("Multiplication of $x and $y is : $ans1");

      break;

    case4 :

      ans1 = x / y;

      print("Division of $x and $y is : $ans1");

      break;

    default :

      print("Enter valid choice");

  }

}

**A.3 Write a dart program to find out largest number from given three numbers without using logical operator.**

import'dart:io';

voidmain() {

  stdout.write('Enter an integer: ');

  int a = int.parse(stdin.readLineSync()!);

  stdout.write('Enter an integer: ');

  int b = int.parse(stdin.readLineSync()!);

  print(

      'Enter 1 for Addition, 2 for Subtraction, 3 for Multipliction, 4 for division.');

  int c = int.parse(stdin.readLineSync()!);

  if (a > b) {

    if (a > c) {

      print("Largest number is $a");

    } else {

      print("Largest number is $c");

    }

  } elseif (b > c) {

    print("Largest number is $b");

  } else {

    print("Largest number is $c");

  }

}

**A.4 Write a dart program to read marks of five subjects. Calculate percentage and print class accordingly. Fail below 35, Pass Class between 35 to 45, Second Class between 45 to 60, First Class between 60 to 70, Distinction if more than 70.**

import 'dart:io';

void main() {

print("Enter marks for five subjects:");

stdout.write("Subject 1: ");

int subject1 = int.parse(stdin.readLineSync()!);

stdout.write("Subject 2: ");

int subject2 = int.parse(stdin.readLineSync()!);

stdout.write("Subject 3: ");

int subject3 = int.parse(stdin.readLineSync()!);

stdout.write("Subject 4: ");

int subject4 = int.parse(stdin.readLineSync()!);

stdout.write("Subject 5: ");

int subject5 = int.parse(stdin.readLineSync()!);

int totalMarks = subject1 + subject2 + subject3 + subject4 + subject5;

double percentage = totalMarks / 5;

print("\nTotal Marks: $totalMarks");

print("Percentage: ${percentage.toStringAsFixed(2)}%");

if (percentage < 35) {

print("Result: Fail");

} else if (percentage < 45) {

print("Result: Pass Class");

} else if (percentage < 60) {

print("Result: Second Class");

} else if (percentage < 70) {

print("Result: First Class");

} else {

print("Result: Distinction");

}

}

**A.5 Write a dart program to find out largest number from given 3 numbers using conditional operator.**

import"dart:io";

voidmain(){

  print("Enter 1st number : ");

  int a = int.parse(stdin.readLineSync()!);

  print("Enter 2nd number : ");

  int b = int.parse(stdin.readLineSync()!);

  print("Enter 3rd number : ");

  int c = int.parse(stdin.readLineSync()!);

  int max = (a>b?(a>c?a:c):(b>c?b:c));

  print("$max is gratest");

}

**A.6 Write a dart program to make a Simple Calculator using switch...case.**

Void main() {

  stdout.write("Enter The Number: ");

  int? a = int.parse(stdin.readLineSync()!);

  stdout.write("Enter The Number: ");

  int? b = int.parse(stdin.readLineSync()!);

  print("Enter 1 for Addition, 2 for Subtraction, 3 for Multiplication, 4 for Division");

  int choice = int.parse(stdin.readLineSync()!);

  switch (choice) {

    case1:

      print("$a + $b = ${a+b}");

      break;

    case2:

      print("$a - $b = ${a-b}");

      break;

    case3:

      print("$a x $b = ${a\*b}");

      break;

    case4:

      print("$a / $b = ${a/b}");

      break;

    default:

      print("Invalid choice!");

  }

}

**B.1 Write a program to find the length of the last word in a given string, where a word is defined as a maximal substring consisting of non-space characters. Example: Input: s = "Hello World", Output: 5 (last word is "World"); Input: s = " fly me to the moon ", Output: 4 (last word is "moon").**

int lengthOfLastWord(String s) {

// Trim leading and trailing spaces

s = s.trim();

// Find the position of the last space

int lastSpaceIndex = s.lastIndexOf(' ');

// Extract the last word and return its length

return s.substring(lastSpaceIndex + 1).length;

}

void main() {

String s1 = "Hello World";

String s2 = " fly me to the moon ";

print('Length of last word: ${lengthOfLastWord(s1)}'); // Output: 5

print('Length of last word: ${lengthOfLastWord(s2)}'); // Output: 4

}

**B.2 Write a program to calculate the angle between the hour and minute hands of a clock, where the hours and minutes are taken from the user. Example: Input: hours = 3, minutes = 15, Output: 7.5 (angle between the hands is 7.5 degrees).**

double calculateClockAngle(int hours, int minutes) {

// Ensure hours are within 1-12

hours = hours % 12;

// Calculate the angles of the hour hand and minute hand

double hourAngle = (hours \* 30) + (minutes \* 0.5); // Each hour is 30 degrees, each minute moves the hour hand by 0.5 degrees

double minuteAngle = minutes \* 6; // Each minute is 6 degrees (360/60)

// Calculate the absolute difference between the two angles

double angle = (hourAngle - minuteAngle).abs();

// Return the smaller angle between the two possible angles

return angle > 180 ? 360 - angle : angle;

}

void main() {

int hours = 3;

int minutes = 15;

print('Angle between hour and minute hands: ${calculateClockAngle(hours, minutes)} degrees'); // Output: 7.5

}

**B.3 Write a program to find the majority element in a given array, where the majority element is the one that appears more than [n / 2] times. You may assume that the majority element always exists in the array. Example: Input: nums = [3, 2, 3], Output: 3; Input: nums = [2, 2, 1, 1, 1, 2, 2], Output: 2.**

int majorityElement(List<int> nums) {

int count = 0;

int candidate = 0;

// Boyer-Moore Voting Algorithm

for (int num in nums) {

if (count == 0) {

candidate = num;

}

count += (num == candidate) ? 1 : -1;

}

return candidate;

}

void main() {

List<int> nums1 = [3, 2, 3];

List<int> nums2 = [2, 2, 1, 1, 1, 2, 2];

print('Majority element in nums1: ${majorityElement(nums1)}'); // Output: 3

print('Majority element in nums2: ${majorityElement(nums2)}'); // Output: 2

}

LAB 3

**A.1 WAP to print numbers between two given numbers which is divisible by 2 but not divisible by 3.**

import 'dart:io';

void main() {

  print("Enter the first number:");

  int start = int.parse(stdin.readLineSync()!);

  print("Enter the second number:");

  int end = int.parse(stdin.readLineSync()!);

  print("Numbers between $start and $end that are divisible by 2 but not

by 3:");

  for (int i = start; i <= end; i++) {

    if (i % 2 == 0 && i % 3 != 0) {

      print(i);

    }

  }

}

**A.2 WAP to find factorial of the given number.**

import 'dart:io';

void main() {

  print("Enter a number to find its factorial:");

  int num = int.parse(stdin.readLineSync()!);

if (num < 0) {

    print("Factorial is not defined for negative numbers.");

  } else {

    int factorial = 1;

    // Calculate factorial using a loop

    for (int i = 1; i <= num; i++) {

      factorial \*= i;

    }

    print("The factorial of $num is: $factorial");

  }

}

**A.3 WAP to find whether the given number is prime or not.**

import 'dart:io';

void main() {

  print("Enter a number to check if it is prime:");

  int num = int.parse(stdin.readLineSync()!);

  if (num <= 1) {

    print("$num is not a prime number.");

  } else {

    bool isPrime = true;

   for (int i = 2; i < num; i++) {

      if (num % i == 0) {

        isPrime = false;

        break;

      }

    }

if (isPrime) {

      print("$num is a prime number.");

    } else {

      print("$num is not a prime number.");

    }

  }

}

**A.4 WAP to print given number in reverse order.**

import 'dart:io';

void main() {

  print("Enter a number:");

  int num = int.parse(stdin.readLineSync()!);

int reversed = 0;

// Reverse the number using a loop

  while (num > 0) {

    int lastDigit = num % 10;

    reversed = reversed \* 10 + lastDigit;

    num = num ~/ 10; // Remove the last digit from the number

  }

print("The reversed number is: $reversed");

}

**A.5 WAP to print reverse string.**

import 'dart:io';

void main() {

print("Enter a string:");

String original = stdin.readLineSync()!

String reversed = original.split('').reversed.join(''); // Reverse the string

print("Original String: $original");

print("Reversed String: $reversed");

//OR

print("Enter a string:");

String original = stdin.readLineSync()!;

String reversed = '';

for (int i = original.length - 1; i >= 0; i--) {

reversed += original[i];

}

print("Original String: $original");

print("Reversed String: $reversed");

}

**A.6 WAP program to calculate the sum of all positive even numbers and the sum of all negative odd numbers from a set of numbers. you can enter 0 (zero) to quit the program and thus it displays the result.**

import 'dart:io';

void main() {

  int sumPositiveEven = 0;

  int sumNegativeOdd = 0;

while (true) {

    // Read a number from the user

    print("Enter a number (Enter 0 to quit):");

    int num = int.parse(stdin.readLineSync()!);

    if (num == 0) {

      break;

    }

    // Check if the number is positive even or negative odd

    if (num > 0 && num % 2 == 0) {

      sumPositiveEven += num;

    } else if (num < 0 && num % 2 != 0) {

      sumNegativeOdd += num;

    }

  }

  // Display the results

  print("Sum of all positive even numbers: $sumPositiveEven");

  print("Sum of all negative odd numbers: $sumNegativeOdd");

}

**B.1 Write a program to find the element that appears only once in a non-empty array of integers, where every other element appears twice. The solution should have linear runtime complexity and use only constant extra space. Example: Input: nums = [2, 2, 1], Output: 1; Input: nums = [4, 1, 2, 1, 2], Output: 4; Input: nums = [1], Output: 1.**

int singleNumber(List<int> nums) {

int result = 0;

// XOR all elements in the array

for (int num in nums) {

result ^= num;

}

return result;

}

void main() {

List<int> nums1 = [2, 2, 1];

List<int> nums2 = [4, 1, 2, 1, 2];

List<int> nums3 = [1];

print('Element that appears only once in nums1: ${singleNumber(nums1)}'); // Output: 1

print('Element that appears only once in nums2: ${singleNumber(nums2)}'); // Output: 4

print('Element that appears only once in nums3: ${singleNumber(nums3)}'); // Output: 1

}

**B.2 Write a Dart program to count the occurrences of each word in a given sentence. Return a map where the keys are the words, and the values are the number of occurrences of each word. Example: Input: "this is a test this is a", Output: {'this': 2, 'is': 2, 'a': 2, 'test': 1}.**

Map<String, int> countWordOccurrences(String sentence) {

// Convert sentence to lowercase and split it into words

List<String> words = sentence.toLowerCase().split(RegExp(r'\s+'));

// Initialize an empty map to store word counts

Map<String, int> wordCount = {};

// Iterate over each word in the list

for (String word in words) {

// If the word is already in the map, increment its count

// Otherwise, add the word to the map with a count of 1

wordCount[word] = (wordCount[word] ?? 0) + 1;

}

return wordCount;

}

void main() {

String sentence = "this is a test this is a";

Map<String, int> result = countWordOccurrences(sentence);

print(result); // Output: {this: 2, is: 2, a: 2, test: 1}

}

**B.3 Write a program to insert an element at a specific position in an array. The program should modify the existing array without creating a new one. Example: Input: array = [1, 2, 4, 5], element = 3, position = 2, Output: [1, 2, 3, 4, 5]; Input: array = [10, 20, 30], element = 25, position = 1, Output: [10, 25, 20, 30].**

void insertElementAtPosition(List<int> array, int element, int position) {

// Check if the position is valid (within the bounds of the array)

if (position < 0 || position > array.length) {

print("Invalid position");

return;

}

// Shift elements to the right to make space for the new element

for (int i = array.length; i > position; i--) {

array[i] = array[i - 1];

}

// Insert the new element at the specified position

array[position] = element;

}

void main() {

List<int> array1 = [1, 2, 4, 5];

insertElementAtPosition(array1, 3, 2);

print(array1); // Output: [1, 2, 3, 4, 5]

List<int> array2 = [10, 20, 30];

insertElementAtPosition(array2, 25, 1);

print(array2); // Output: [10, 25, 20, 30]

}

LAB 4

**A.1 WAP to calculate simple interest using method.**

import 'dart:io';

// Method to calculate simple interest

double calculateSimpleInterest(

    {required double principal, required double rate, required double time}) {

  return (principal \* rate \* time) / 100;

}

void main() {

  // Input from the user

  print("Enter the Principal amount:");

  double principal = double.parse(stdin.readLineSync()!);

  print("Enter the Rate of interest:");

  double rate = double.parse(stdin.readLineSync()!);

  print("Enter the Time period (in years):");

  double time = double.parse(stdin.readLineSync()!);

  // Calculate simple interest

  double interest =

      calculateSimpleInterest(principal: principal, rate: rate, time: time);

  // Display the result

  print("The Simple Interest is: \$${interest}");

}

**A.2 WAP to find maximum number from given two numbers using method.**

import 'dart:io';

// Method to find the maximum of two numbers

int findMaximum(int num1, int num2) {

  return (num1 > num2) ? num1 : num2;

}

void main() {

  // Input from the user

  print("Enter the first number:");

  int number1 = int.parse(stdin.readLineSync()!);

  print("Enter the second number:");

  int number2 = int.parse(stdin.readLineSync()!);

  // Find the maximum number

  int maxNumber = findMaximum(number1, number2);

  // Display the result

  print("The maximum number between $number1 and $number2 is: $maxNumber");

}

**A.3 WAP to generate Fibonacci series of N given number using method.**

import 'dart:io';

// Method to generate Fibonacci series up to N terms

void generateFibonacci(int n) {

  int first = 0, second = 1;

  print("Fibonacci series up to $n terms:");

  for (int i = 0; i < n; i++) {

    // Print the current number in the series

    stdout.write("$first ");

// Update values for the next iteration

    int next = first + second;

    first = second;

    second = next;

  }

}

void main() {

  // Input from the user

  print("Enter the number of terms for the Fibonacci series:");

  int n = int.parse(stdin.readLineSync()!);

  if (n <= 0) {

    print("Please enter a positive integer greater than 0.");

  } else {

    // Generate the Fibonacci series

    generateFibonacci(n);

  }

}

**A.4 WAP to accept a number and check whether the number is prime or not. Use method name check (int n). The method returns 1, if the number is prime otherwise, it returns 0.**

import 'dart:io';

int check(int n) {

  if (n <= 1) {

    return 0;

}

for (int i = 2; i <= n / 2; i++) {

    if (n % i == 0) {

      return 0;

    }

  }

  return 1;

}

void main() {

  print("Enter a number to check if it's prime:");

  int number = int.parse(stdin.readLineSync()!);

  int result = check(number);

  if (result == 1) {

    print("$number is a prime number.");

  } else {

    print("$number is not a prime number.");

  }

}

**A.5 WAP to count number of even or odd number from an array of n numbers.**

import 'dart:io';

void countEvenOdd(List<int> numbers) {

  int evenCount = 0;

  int oddCount = 0;

for (int number in numbers) {

    if (number % 2 == 0) {

      evenCount++;

    } else {

      oddCount++;

    }

  }

  print("Number of even numbers: $evenCount");

  print("Number of odd numbers: $oddCount");

}

void main() {

  print("Enter the number of elements in the array:");

  int n = int.parse(stdin.readLineSync()!);

  List<int> numbers = [];

print("Enter $n numbers:");

  for (int i = 0; i < n; i++) {

    numbers.add(int.parse(stdin.readLineSync()!));

  }

// Count even and odd numbers

  countEvenOdd(numbers);

}

**B.1 Write a program to sort an array of names based on the corresponding heights in descending order. The names and heights are given as two separate arrays, and the heights are distinct positive integers. Example: Input: names = ["Mary","John","Emma"], heights = [180,165,170], Output: ["Mary","Emma","John"]; Input: names = ["Alice","Bob","Bob"], heights = [155,185,150], Output: ["Bob","Alice","Bob"].**

List<String> sortNamesByHeights(List<String> names, List<int> heights) {

// Create a list of tuples (name, height)

List<MapEntry<String, int>> nameHeightPairs = [];

for (int i = 0; i < names.length; i++) {

nameHeightPairs.add(MapEntry(names[i], heights[i]));

}

// Sort the pairs by height in descending order

nameHeightPairs.sort((a, b) => b.value.compareTo(a.value));

// Extract the sorted names based on the sorted height order

List<String> sortedNames = nameHeightPairs.map((pair) => pair.key).toList();

return sortedNames;

}

void main() {

List<String> names1 = ["Mary", "John", "Emma"];

List<int> heights1 = [180, 165, 170];

List<String> sortedNames1 = sortNamesByHeights(names1, heights1);

print(sortedNames1); // Output: ["Mary", "Emma", "John"]

List<String> names2 = ["Alice", "Bob", "Bob"];

List<int> heights2 = [155, 185, 150];

List<String> sortedNames2 = sortNamesByHeights(names2, heights2);

print(sortedNames2); // Output: ["Bob", "Alice", "Bob"]

}

**B.2 Write a program to remove duplicates from a sorted integer array in-place such that each unique element appears only once. The relative order of the elements should be kept the same. The function should return the number of unique elements in the array. Example: Input: nums = [1, 1, 2], Output: 2, nums = [1, 2, \_]; Input: nums = [0, 0, 1, 1, 1, 2, 2, 3, 3, 4], Output: 5, nums = [0, 1, 2, 3, 4, \_, \_, \_, \_, \_].**

int removeDuplicates(List<int> nums) {

if (nums.isEmpty) return 0;

int slowPointer = 0; // Slow pointer to track the position for unique elements

// Iterate over the array with the fast pointer

for (int fastPointer = 1; fastPointer < nums.length; fastPointer++) {

if (nums[fastPointer] != nums[slowPointer]) {

slowPointer++;

nums[slowPointer] = nums[fastPointer]; // Place the unique element at the correct position

}

}

// The length of the unique elements will be slowPointer + 1

return slowPointer + 1;

}

void main() {

List<int> nums1 = [1, 1, 2];

int uniqueCount1 = removeDuplicates(nums1);

print('Unique count: $uniqueCount1, Array: ${nums1.sublist(0, uniqueCount1)}'); // Output: Unique count: 2, Array: [1, 2]

List<int> nums2 = [0, 0, 1, 1, 1, 2, 2, 3, 3, 4];

int uniqueCount2 = removeDuplicates(nums2);

print('Unique count: $uniqueCount2, Array: ${nums2.sublist(0, uniqueCount2)}'); // Output: Unique count: 5, Array: [0, 1, 2, 3, 4]

}

**B.3 Write a program to find the intersection of two integer arrays, nums1 and nums2. Each element in the result should appear as many times as it appears in both arrays, and the result should be returned in sorted order. Example: Input: nums1 = [1, 2, 2, 1], nums2 = [2, 2], Output: [2, 2]; Input: nums1 = [4, 9, 5], nums2 = [9, 4, 9, 8, 4], Output: [4, 9].**

List<int> intersect(List<int> nums1, List<int> nums2) {

// Create a frequency map for nums1

Map<int, int> freqMap = {};

for (int num in nums1) {

freqMap[num] = (freqMap[num] ?? 0) + 1;

}

// Find the intersection

List<int> result = [];

for (int num in nums2) {

if (freqMap.containsKey(num) && freqMap[num]! > 0) {

result.add(num);

freqMap[num] = freqMap[num]! - 1; // Decrease the count in the map

}

}

// Sort the result

result.sort();

return result;

}

void main() {

List<int> nums1 = [1, 2, 2, 1];

List<int> nums2 = [2, 2];

print(intersect(nums1, nums2)); // Output: [2, 2]

List<int> nums3 = [4, 9, 5];

List<int> nums4 = [9, 4, 9, 8, 4];

print(intersect(nums3, nums4)); // Output: [4, 9]

}

LAB 5

**A.1 WAP that prompts the user to enter 5 numbers, stores them in a List, and displays them in increasing order.**

import 'dart:io';

void main() {

  List<int> numbers = [];

print("Enter 5 numbers:");

for (int i = 1; i <= 5; i++) {

    int number = int.parse(stdin.readLineSync()!);

    numbers.add(number);

  }

// Sort the numbers in increasing order

  numbers.sort();

  // for (int i = 0; i < numbers.length; i++) {

  //   for (int j = i + 1; j < numbers.length; j++) {

  //     if (numbers[i] > numbers[j]) {

  //       int temp = numbers[j];

  //       numbers[j] = numbers[i];

  //       numbers[i] = temp;

  //     }

  //   }

  // }

  // Display the sorted numbers

  print("Numbers in increasing order:");

  for (int number in numbers) {

    print(number);

  }

}

**A.2 WAP to read 2 list and return list that contains only the elements that are common between them.**

import 'dart:io';

List<int> findCommonElements(List<int> list1, List<int> list2) {

  List<int> commonElements = [];

for (int element in list1) {

    if (list2.contains(element) && !commonElements.contains(element)) {

      commonElements.add(element);

    }

  }

return commonElements;

}

void main() {

  print("Enter the number of elements for the first list:");

  int n1 = int.parse(stdin.readLineSync()!);

  List<int> list1 = [];

  print("Enter $n1 elements for the first list:");

  for (int i = 0; i < n1; i++) {

    int number = int.parse(stdin.readLineSync()!);

    list1.add(number);

  }

// Input for the second list

  print("Enter the number of elements for the second list:");

  int n2 = int.parse(stdin.readLineSync()!);

  List<int> list2 = [];

  print("Enter $n2 elements for the second list:");

  for (int i = 0; i < n2; i++) {

    int number = int.parse(stdin.readLineSync()!);

    list2.add(number);

  }

// Find common elements

  List<int> common = findCommonElements(list1, list2);

// Display the common elements

  if (common.isEmpty) {

    print("No common elements between the two lists.");

  } else {

    print("Common elements between the two lists: $common");

  }

}

**A.3 WAP that creates List with following value:“Delhi”, “Mumbai”, “Bangalore”, “Hyderabad” and “Ahmedabad” Replace “Ahmedabad” with “Surat” in above List.**

void main() {

List<String> cities = [

    "Delhi",

    "Mumbai",

    "Bangalore",

    "Hyderabad",

    "Ahmedabad"

  ];

print("Original List: $cities");

  int index = cities.indexOf("Ahmedabad");

  cities[index] = "Surat";

print("Modified List: $cities");

}

**A.4 WAP to create and read phonebook dictionary.**

import 'dart:io';

void main() {

  // Create a phonebook using Map

  Map<String, String> phonebook = {

    'John': '123-456-7890',

    'Jane': '987-654-3210',

    'Bob': '555-123-4567',

    'Alice': '555-765-4321'

  };

  print("Phonebook:");

  phonebook.forEach((name, phone) {

    print('$name: $phone');

  });

print("\nEnter a name to search for their phone number:");

  String nameToSearch = stdin.readLineSync()!;

if (phonebook.containsKey(nameToSearch)) {

    print("$nameToSearch's phone number is: ${phonebook[nameToSearch]}");

  } else {

    print("Contact not found.");

  }

}

**A.5 WAP to find friends detail by their name using dictionary. (Create local dictionary and search from it using Map<String,Object?> & Model Class).**

import 'dart:io';

class Friend {

  String name;

  int age;

  String phoneNumber;

  Friend({

    required this.name,

    required this.age,

    required this.phoneNumber,

  });

  void displayDetails() {

    print('Name: $name');

    print('Age: $age');

    print('Phone Number: $phoneNumber');

  }

}

void main() {

  Map<String, Friend> friendsMap = {

    'John': Friend(name: 'John', age: 25, phoneNumber: '123-456-7890'),

    'Alice': Friend(name: 'Alice', age: 30, phoneNumber: '987-654-3210'),

    'Bob': Friend(name: 'Bob', age: 22, phoneNumber: '555-123-4567'),

    'Eve': Friend(name: 'Eve', age: 28, phoneNumber: '555-765-4321'),

  };

print('Enter the name of the friend you want to search for:');

  String nameToSearch = stdin.readLineSync()!;

  if (friendsMap.containsKey(nameToSearch)) {

    print("\nDetails for $nameToSearch:");

    friendsMap[nameToSearch]?.displayDetails();

    // print(friendsMap[nameToSearch]?.age);

  } else {

    print("Friend not found.");

  }

}

**A.6 WAP to accept n numbers in an array. Display the sum of all the numbers which are divisible by either 3 or 5.**

import 'dart:io';

void main() {

  print("Enter the number of elements:");

  int n = int.parse(stdin.readLineSync()!);

List<int> numbers = [];

print("Enter $n numbers:");

  for (int i = 0; i < n; i++) {

    int number = int.parse(stdin.readLineSync()!);

    numbers.add(number);

  }

int sum = 0;

  for (int number in numbers) {

    if (number % 3 == 0 || number % 5 == 0) {

      sum += number;

    }

  }

print("The sum of all numbers divisible by either 3 or 5 is: $sum");

}

**B.1 Write a program to find the indices of two numbers in an integer array nums that add up to a given target. You may assume that each input has exactly one solution, and you cannot use the same element twice. The answer can be returned in any order. Example: Input: nums = [2, 7, 11, 15], target = 9, Output: [0, 1]; Input: nums = [3, 2, 4], target = 6, Output: [1, 2]; Input: nums = [3, 3], target = 6, Output: [0, 1].**

List<int> twoSum(List<int> nums, int target) {

// Create a map to store the numbers we've seen and their indices

Map<int, int> numToIndex = {};

// Iterate through the array

for (int i = 0; i < nums.length; i++) {

int complement = target - nums[i];

// If complement is found in the map, return the indices

if (numToIndex.containsKey(complement)) {

return [numToIndex[complement]!, i];

}

// Otherwise, store the current number and its index in the map

numToIndex[nums[i]] = i;

}

// In case there's no solution (not needed per problem constraints)

return [];

}

void main() {

List<int> nums1 = [2, 7, 11, 15];

int target1 = 9;

print(twoSum(nums1, target1)); // Output: [0, 1]

List<int> nums2 = [3, 2, 4];

int target2 = 6;

print(twoSum(nums2, target2)); // Output: [1, 2]

List<int> nums3 = [3, 3];

int target3 = 6;

print(twoSum(nums3, target3)); // Output: [0, 1]

}

**B.2 Write a program to find the largest odd integer (as a string) that is a non-empty substring of a given string num, which represents a large integer. If no odd integer exists, return an empty string "". Example: Input: num = "52", Output: "5"; Input: num = "4206", Output: ""; Input: num = "35427", Output: "35427".**

String largestOddSubstring(String num) {

// Start checking from the longest substring down to the shortest

for (int i = num.length; i > 0; i--) {

String substring = num.substring(0, i);

// Check if the last digit of the substring is odd

if (int.parse(substring[substring.length - 1]) % 2 != 0) {

return substring;

}

}

// If no odd substring found, return an empty string

return "";

}

void main() {

print(largestOddSubstring("52")); // Output: "5"

print(largestOddSubstring("4206")); // Output: ""

print(largestOddSubstring("35427")); // Output: "35427"

}

**B.3 Write a Dart program to remove duplicates from a list of integers. Return a new list that contains only the unique elements, with duplicates removed, using a map. Example: Input: nums = [1, 2, 2, 3, 4, 4], Output: [1, 2, 3, 4].**

List<int> removeDuplicates(List<int> nums) {

// Use a Map to store unique elements as keys

Map<int, bool> uniqueMap = {};

// Add each element to the map (duplicates will be automatically removed)

for (int num in nums) {

uniqueMap[num] = true;

}

// Return the keys of the map as a list (this will contain unique elements)

return uniqueMap.keys.toList();

}

void main() {

List<int> nums1 = [1, 2, 2, 3, 4, 4];

print(removeDuplicates(nums1)); // Output: [1, 2, 3, 4]

List<int> nums2 = [5, 5, 5, 5];

print(removeDuplicates(nums2)); // Output: [5]

List<int> nums3 = [7, 8, 9, 7, 9, 10];

print(removeDuplicates(nums3)); // Output: [7, 8, 9, 10]

}

LAB 6

**A.1 Write a flutter code to divide horizontal space of screen in 3 different equal parts with different colors.**

import "package:flutter/material.dart";

class ThreeHorizontal extends StatelessWidget {

  const ThreeHorizontal({super.key});

@override

  Widget build(BuildContext context) {

    return Scaffold(

      body: Row(

        children: [

          Expanded(

            child: Container(

              color: Colors.blueAccent,

            ),

          ),

          Expanded(

            child: Container(

              color: Colors.orange,

            ),

          ),

          Expanded(

            child: Container(

              color: Colors.purple,

            ),

          ),

        ],

      ),

    );

  }

}

**A.2 Write a flutter code to divide the vertical space of the screen in 3 different equal parts with different colors.**

import "package:flutter/material.dart";

class ThreeHorizontal extends StatelessWidget {

  const ThreeHorizontal({super.key});

@override

  Widget build(BuildContext context) {

    return Scaffold(

      body: Column(

        children: [

          Expanded(

            child: Container(

              color: Colors.blueAccent,

            ),

          ),

          Expanded(

            child: Container(

              color: Colors.orange,

            ),

          ),

          Expanded(

            child: Container(

              color: Colors.purple,

            ),

          ),

        ],

      ),

    );

  }

}

**B.1 Write a flutter code to divide the whole screen into 9 (equal size and the different size) with different colors.**

import 'package:flutter/material.dart';

class EqualSize extends StatelessWidget {

  const EqualSize({super.key});

@override

  Widget build(BuildContext context) {

    return Scaffold(

      body: Column(

        children: [

          Expanded(

              child: Row(

            children: [

              Comman(color: Colors.red),

              Comman(color: Colors.green),

              Comman(color: Colors.pink),

            ],

          )),

          Expanded(

              child: Row(

            children: [

              Comman(color: Colors.black),

              Comman(color: Colors.yellow),

              Comman(color: Colors.white30),

            ],

          )),

          Expanded(

              child: Row(

            children: [

              Comman(color: Colors.yellow),

              Comman(color: Colors.green),

              Comman(color: Colors.pink),

            ],

          )),

        ],

      ),

    );

  }

Widget Comman({required Color color}) {

    return Expanded(

      child: Container(

        color: color,

      ),

    );

  }

}

**OR**

import 'package:flutter/material.dart';

class DiffrentSize extends StatelessWidget {

  const DiffrentSize({super.key});

@override

  Widget build(BuildContext context) {

    return Scaffold(

      body: Row(

        children: [

          Expanded(

              child: Column(

            children: [

              Comman(color: Colors.red),

              Comman(color: Colors.green),

              Comman(color: Colors.pink),

            ],

          )),

          Expanded(

              child: Column(

            children: [

              Comman(color: Colors.black, flex: 2),

              Comman(color: Colors.yellow, flex: 2),

              Comman(color: Colors.white30),

            ],

          )),

          Expanded(

              child: Column(

            children: [

              Comman(color: Colors.yellow),

              Comman(color: Colors.green, flex: 2),

              Comman(color: Colors.pink, flex: 2),

            ],

          )),

        ],

      ),

    );

  }

Widget Comman({required Color color, int? flex = 1}) {

    return Expanded(

      flex: flex!,

      child: Container(

        color: color,

      ),

    );

  }

}

LAB 7

**A.1 Write a flutter code to display “hello world” using Text widget. Change color & size of text using different properties.**

import 'package:flutter/material.dart';

class TextWidget extends StatelessWidget {

  const TextWidget({super.key});

@override

  Widget build(BuildContext context) {

    return const Scaffold(

      body: Center(

        child: Text(

          "Hello World",

          style: TextStyle(

            fontSize: 24.0, // Set the size of the text

            color: Colors.blue, // Set the color of the text

            fontWeight: FontWeight.bold, // Optional: make the text bold

          ),

        ),

      ),

    );

  }

}

**A.2 Write a flutter code to create a custom Text widget with different fonts & use it.**

import 'package:flutter/material.dart';

class CustomTextWidgetDemo extends StatelessWidget {

  const CustomTextWidgetDemo({super.key});

@override

  Widget build(BuildContext context) {

    return Scaffold(

      body: Column(

        children: [

          CustomText(text: 'paresh', color: Colors.red, fontSize: 50),

          CustomText(text: 'ABC', fontSize: 100),

          CustomText(

              text: 'paresh',

              color: Colors.green,

              fontFamily: 'EduAUVICWANTArrows',

              fontSize: 200),

        ],

      ),

    );

  }

Widget CustomText(

      {required String text,

      Color? color,

      double? fontSize,

      String? fontFamily}) {

    return Text(

      text,

      style:

          TextStyle(fontFamily: fontFamily, fontSize: fontSize, color: color),

    );

  }

}

**A.3 Write a flutter code to use TextField and print the input value into the *terminal* using the controller.**

import 'package:flutter/material.dart';

class TextFieldExample extends StatelessWidget {

  TextFieldExample({super.key});

final TextEditingController \_controller = TextEditingController();

@override

  Widget build(BuildContext context) {

    return Scaffold(

      body: Column(

        mainAxisAlignment: MainAxisAlignment.center,

        children: [

          // TextField with a controller

          TextField(

            controller: \_controller,

          ),

          SizedBox(height: 16),

          // Button to print input value to terminal

          ElevatedButton(

            onPressed: () {

              // Print the value from the controller

              print('Input Value: ${\_controller.text}');

            },

            child: Text('Print to Terminal'),

          ),

        ],

      ),

    );

  }}

**A.4 Write a flutter code to use TextField & apply different borders, floating labels, hint text etc.,**

import 'package:flutter/material.dart';

class TextFieldCustomization extends StatelessWidget {

  const TextFieldCustomization({super.key});

@override

  Widget build(BuildContext context) {

    return Scaffold(

      body: Padding(

        padding: const EdgeInsets.all(16.0),

        child: Column(

          mainAxisAlignment: MainAxisAlignment.center,

          children: [

            TextField(

              decoration: InputDecoration(

                labelText: 'Floating Label',

                hintText: 'Enter your text here',

                hintStyle: TextStyle(color: Colors.grey),

                border: OutlineInputBorder(

                  borderRadius: BorderRadius.circular(8.0),

                  borderSide: BorderSide(

                    color: Colors.blue,

                    width: 2.0,

                  ),

                ),

                enabledBorder: OutlineInputBorder(

                  borderRadius: BorderRadius.circular(8.0),

                  borderSide: BorderSide(

                    color: Colors.grey,

                    width: 1.0,

                  ),

                ),

                focusedBorder: OutlineInputBorder(

                  borderRadius: BorderRadius.circular(8.0),

                  borderSide: BorderSide(

                    color: Colors.blue,

                    width: 2.0,

                  ),

                ),

              ),

            ),

            SizedBox(height: 16),

            // TextField with underline border

            TextField(

              decoration: InputDecoration(

                labelText: 'Underline Border',

                hintText: 'Enter your text here',

                hintStyle: TextStyle(color: Colors.grey),

                border: UnderlineInputBorder(

                  borderSide: BorderSide(

                    color: Colors.green,

                    width: 2.0,

                  ),

                ),

                focusedBorder: UnderlineInputBorder(

                  borderSide: BorderSide(

                    color: Colors.green,

                    width: 2.0,

                  ),

                ),

              ),

            ),

            SizedBox(height: 16),

            // TextField with no border

            TextField(

              decoration: InputDecoration(

                labelText: 'No Border',

                hintText: 'Enter your text here',

                hintStyle: TextStyle(color: Colors.grey),

                border: InputBorder.none,

                filled: true,

                fillColor: Colors.grey[200],

              ),

            ),

          ],

        ),

      ),

    );

  }

}

**A.5 Write a flutter code to print TextField value into Text widget on a click of a Button.**

import 'package:flutter/material.dart';

class TextFieldToTextWidget extends StatefulWidget {

  const TextFieldToTextWidget({super.key});

@override

  State<TextFieldToTextWidget> createState() => \_TextFieldToTextWidgetState();

}

class \_TextFieldToTextWidgetState extends State<TextFieldToTextWidget> {

  final TextEditingController \_controller = TextEditingController();

  String \_displayedText = "";

@override

  Widget build(BuildContext context) {

    return Scaffold(

      body: Padding(

        padding: const EdgeInsets.all(16.0),

        child: Column(

          mainAxisAlignment: MainAxisAlignment.center,

          children: [

            // TextField to take input

            TextField(

              controller: \_controller,

              decoration: InputDecoration(

                border: OutlineInputBorder(),

                labelText: "Enter your text",

              ),

            ),

            SizedBox(height: 16),

            // Button to update the displayed text

            ElevatedButton(

              onPressed: () {

                setState(() {

                  \_displayedText = \_controller.text;

                });

              },

              child: Text("Display Text"),

            ),

            SizedBox(height: 16),

            // Text widget to show the input value

            Text(

              \_displayedText,

              style: TextStyle(fontSize: 20, color: Colors.blue),

            ),

          ],

        ),

      ),

    );

  }

}

LAB 8

**A.1 Write a flutter code to display an Image into the Image asset widget from the asset folder.**

**Write a flutter code to display an Image from a WEB URL using a cached network Image.**

import 'package:flutter/material.dart';

class ImageAsset extends StatelessWidget {

  const ImageAsset({super.key});

@override

  Widget build(BuildContext context) {

    return Scaffold(

      body: Image.asset('assets/img/image.png'),

    );

  }

}

**AND**

import 'package:cached\_network\_image/cached\_network\_image.dart';

import 'package:flutter/material.dart';

class ImageNetwork extends StatelessWidget {

  const ImageNetwork({super.key});

@override

  Widget build(BuildContext context) {

    const String imageUrl =

        'https://flutter.dev/assets/homepage/carousel/slide\_1-bg-4e2fcef1ae4cdb32e1ac92702a3b90bcf6d0aa8c6e24818415543a5d02ab2cbf.png'; // Replace with your desired image URL.

    return Scaffold(

      body: CachedNetworkImage(

        imageUrl: imageUrl,

        placeholder: (context, url) =>

            CircularProgressIndicator(), // Loading indicator

        errorWidget: (context, url, error) => Icon(Icons.error), // Error widget

        fit: BoxFit.cover, // Adjust the image fit

        width: 300,

        height: 200,

      ),

    );

  }

}

**A.2 Write a flutter code to use Stack widget and show text upon an Image.**

import 'package:flutter/material.dart';

class MyApp extends StatelessWidget {

const MyApp({super.key});

@override

Widget build(BuildContext context) {

Return Scaffold(

body: Center(

child: Stack(

alignment: Alignment.center,

children: [

// Background Image

Image.network(

'https://example.com/your-image-url.jpg', // Replace with your image URL or asset

width: 300, // Adjust the width and height as needed

height: 300,

fit: BoxFit.cover,

),

// Text overlay

Container(

padding: const EdgeInsets.all(8.0),

color: Colors.black54, // Semi-transparent background for text

child: const Text(

'Text Over Image',

style: TextStyle(

color: Colors.white,

fontSize: 24,

fontWeight: FontWeight.bold,

),

),

),

],

),

),

),

);

}

}

**A.3 Write a flutter code to create a birthday card using different widgets.**

import 'package:flutter/material.dart';

class BirthdayCardApp extends StatelessWidget {

const BirthdayCardApp({super.key});

@override

Widget build(BuildContext context) {

Return Scaffold(

backgroundColor: Colors.purple[50], // Light background

appBar: AppBar(

title: const Text('Birthday Card'),

backgroundColor: Colors.purple,

),

body: Center(

child: Card(

elevation: 10,

shadowColor: Colors.purpleAccent,

shape: RoundedRectangleBorder(

borderRadius: BorderRadius.circular(20),

),

child: SizedBox(

width: 300,

height: 450,

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: [

// Birthday image

ClipRRect(

borderRadius: BorderRadius.circular(20),

child: Image.network(

'https://example.com/birthday\_image.jpg', // Replace with a valid image URL

height: 200,

width: 300,

fit: BoxFit.cover,

),

),

const SizedBox(height: 20),

// Birthday message

const Text(

'Happy Birthday!',

style: TextStyle(

fontSize: 28,

fontWeight: FontWeight.bold,

color: Colors.purple,

),

),

const SizedBox(height: 10),

const Text(

'Wishing you a day filled with love, laughter, and happiness!',

textAlign: TextAlign.center,

style: TextStyle(

fontSize: 16,

color: Colors.black54,

),

padding: EdgeInsets.symmetric(horizontal: 16),

),

],

),

),

),

),

),

);

}

}

**B.1 Write a flutter code to roll a dice on the Button click event.**

import 'dart:math';

import 'package:flutter/material.dart';

class DiceApp extends StatefulWidget {

const DiceApp({super.key});

@override

State<DiceApp> createState() => \_DiceAppState();

}

class \_DiceAppState extends State<DiceApp> {

int diceNumber = 1; // Initial dice face

void rollDice() {

setState(() {

diceNumber = Random().nextInt(6) + 1; // Generates a number between 1 and 6

});

}

@override

Widget build(BuildContext context) {

Return Scaffold(

backgroundColor: Colors.blueAccent[100],

appBar: AppBar(

title: const Text('Roll the Dice'),

backgroundColor: Colors.blueAccent,

),

body: Center(

child: Column(

mainAxisAlignment: MainAxisAlignment.center,

children: [

// Dice image

Image.asset(

'assets/dice$diceNumber.png', // Make sure you have dice images named dice1.png to dice6.png

width: 150,

height: 150,

),

const SizedBox(height: 30),

// Roll button

ElevatedButton(

onPressed: rollDice,

style: ElevatedButton.styleFrom(

backgroundColor: Colors.blueAccent,

padding: const EdgeInsets.symmetric(horizontal: 30, vertical: 15),

),

child: const Text(

'Roll Dice',

style: TextStyle(fontSize: 20, color: Colors.white),

),

),

],

),

),

),);}}

LAB 9

**A.1 Write a flutter code to use TabView and display different pages on different tab clicks.**

import 'package:flutter/material.dart';

class Tabbar extends StatelessWidget {

  @override

  Widget build(BuildContext context) {

    return DefaultTabController(

      length: 3,

      child: Scaffold(

        appBar: AppBar(

          title: Text('TabView Example'),

          bottom: TabBar(

            tabs: [

              Tab(text: 'Tab 1'),

              Tab(text: 'Tab 2'),

              Tab(text: 'Tab 3'),

            ],

          ),

        ),

        body: TabBarView(

          children: [

            Center(child: Text('Page 1')),

            Center(child: Text('Page 2')),

            Center(child: Text('Page 3')),

          ],

        ),

      ),

    );

  }

}

**A.2 Write a flutter code to use the NavigationDrawer & display different pages on different menu clicks.**

import 'package:flutter/material.dart';

class DrawerExample extends StatelessWidget {

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      drawer: Drawer(

        child: ListView(

          padding: EdgeInsets.zero,

          children: <Widget>[

            DrawerHeader(

              decoration: BoxDecoration(

                color: Colors.blue,

              ),

              child: Text(

                'Menu',

                style: TextStyle(

                  color: Colors.white,

                  fontSize: 24,

                ),

              ),

            ),

            ListTile(

              title: Text('Page 1'),

              onTap: () {

                Navigator.pop(context);

                Navigator.push(context, MaterialPageRoute(builder: (context) => Page1()));

              },

            ),

            ListTile(

              title: Text('Page 2'),

              onTap: () {

                Navigator.pop(context);

                Navigator.push(context, MaterialPageRoute(builder: (context) => Page2()));

              },

            ),

            ListTile(

              title: Text('Page 3'),

              onTap: () {

                Navigator.pop(context);

                Navigator.push(context, MaterialPageRoute(builder: (context) => Page3()));

              },

            ),

          ],

        ),

      ),

      body: Center(

        child: Text('Home Page'),

      ),

    );

  }

}

class Page1 extends StatelessWidget {

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('Page 1'),

      ),

      body: Center(

        child: Text('Page 1 Here'),

      ),

    );

  }

}

class Page2 extends StatelessWidget {

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('Page 2'),

      ),

      body: Center(

        child: Text('Page 2 Here'),

      ),

    );

  }

}

class Page3 extends StatelessWidget {

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('Page 3'),

      ),

      body: Center(

        child: Text('Page 3 Here'),

      ),

    );

  }

}

**A.3 Write a flutter code to use the ActionBar widget and display menu in it and display Alert Dialog on menu click.**

import 'package:flutter/cupertino.dart';

import 'package:flutter/material.dart';

class ActionAndDialog extends StatelessWidget {

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('ActionBar Example'),

        actions: [

          IconButton(

            icon: Icon(Icons.menu),

            onPressed: () {

              showDialog(

                context: context,

                builder: (BuildContext context) {

                  return AlertDialog(

                    title: Text('Menu'),

                    content: Text('This is Manu'),

                    actions: [

                      TextButton(

                        onPressed: () {

                          Navigator.of(context).pop();

                        },

                        child: Text('OK'),

                      ),

                    ],

                  );

                },

              );

            },

          ),

        ],

      ),

      body: Center(

        child: Text('Home Page'),

      ),

    );

  }

}

LAB 10

**A.1 Write a flutter code to create login screen login for username & password using Textfield, Button etc.,**

import 'package:flutter/material.dart';

import 'package:google\_fonts/google\_fonts.dart';

class LABA1 extends StatefulWidget {

  @override

  State<LABA1> createState() => \_LABA1State();

}

class \_LABA1State extends State<LABA1> {

  TextEditingController uname = new TextEditingController();

  TextEditingController ename = new TextEditingController();

  String output = " ";

  String outputs = " ";

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        backgroundColor: Colors.teal.shade300,

        title: Center(

          child: Text(

            "Welcome",

            style: GoogleFonts.labrada(),

          ),

        ),

      ),

      body: Column(

        children: [

          Expanded(

              child: Container(

                margin: EdgeInsets.all(30),

                decoration: BoxDecoration(

                  border: Border.all(color: Colors.black26),

                  borderRadius: BorderRadius.circular(20),

                ),

                child: Column(

                  mainAxisAlignment: MainAxisAlignment.center,

                  children: [

                    Padding(

                      padding: const EdgeInsets.all(10.0),

                      child: Center(

                        child: Text("Login Page", style: GoogleFonts.pacifico(textStyle:TextStyle(fontSize: 40)),),

                      ),

                    ),

                    Padding(

                      padding: const EdgeInsets.all(10.0),

                      child: TextField(

                        controller: uname,

                        decoration: InputDecoration(

                          border: OutlineInputBorder(

                            borderRadius: BorderRadius.all(Radius.circular(10)),

                          ),

                          hintText: "Username",

                          labelText: "Enter the Name",

                          labelStyle: TextStyle(fontWeight: FontWeight.bold),

                        ),

                      ),

                    ),

                    Padding(

                      padding: const EdgeInsets.all(10.0),

                      child: TextField(

                        controller: ename,

                        decoration: InputDecoration(

                          border: OutlineInputBorder(

                              borderRadius: BorderRadius.all(Radius.circular(10))),

                          hintText: "abc@gmail.com",

                          labelText: "Enter Your E-mail",

                          labelStyle: TextStyle(fontWeight: FontWeight.bold),

                        ),

                      ),

                    ),

                    Padding(

                      padding: const EdgeInsets.all(20),

                      child: ElevatedButton(

                        onPressed: () {

                          print(uname.text);

                          print(ename.text);

                          setState(() {

                          });

                        },

                        child: Text(

                          "Login",

                          style: TextStyle(color: Colors.black),

),),),],),),),],),);}}

**A.2 Write a flutter code to do validation in the login screen. (Email Validation & Password Validation) on Button click.**

import 'package:flutter/material.dart';

import 'package:google\_fonts/google\_fonts.dart';

class Validation extends StatefulWidget {

  @override

  State<Validation> createState() => \_ValidationState();

}

class \_ValidationState extends State<Validation> {

  Widget Txtstyle(String name, double? n) {

    return Text(

      name,

      style: TextStyle(

        fontSize: n,

        fontFamily: "myName",

      ),

    );

  }

  TextEditingController uname = new TextEditingController();

  TextEditingController ename = new TextEditingController();

  TextEditingController password = new TextEditingController();

  String output = " UserName";

  String outputs = " Mail";

  String outputp = " Password";

  final \_formKey = GlobalKey<FormState>();

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      body: Form(

        key: \_formKey,

        child: Column(

          children: [

            Expanded(

                child: Container(

                  margin: EdgeInsets.all(30),

                  decoration: BoxDecoration(

                    border: Border.all(color: Colors.black26),

                    borderRadius: BorderRadius.circular(20),

                  ),

                  child: Column(

                    mainAxisAlignment: MainAxisAlignment.center,

                    children: [

                      Padding(

                        padding: const EdgeInsets.all(10.0),

                        child: Center(

                          child: Txtstyle("Login Page", 40),

                        ),

                      ),

                      Padding(

                        padding: const EdgeInsets.all(10.0),

                        child: TextFormField(

                          validator: (value) {

                            if (value!.isEmpty) {

                              return "Enter the Name";

                            }

                            return null;

                          },

                          controller: uname,

                          decoration: InputDecoration(

                            border: OutlineInputBorder(

                              borderRadius: BorderRadius.all(Radius.circular(10)),

                            ),

                            hintText: "Username",

                            labelText: "Enter the Name",

                            labelStyle: TextStyle(fontWeight: FontWeight.bold),

                          ),

                        ),

                      ),

                      Padding(

                        padding: const EdgeInsets.all(10.0),

                        child: TextFormField(

                          validator: (value) {

                            if (value!.isEmpty) {

                              return "Enter the E-mail";

                            }

                            String pattern = "[a-z]+[0-9]\*@gmail.com";

                            var emailRegExp = RegExp(pattern);

                            if (!emailRegExp.hasMatch(value!)) {

                              return "Enter The Valid Mail Address";

                            }

                            return null;

                          },

                          controller: ename,

                          decoration: InputDecoration(

                            border: OutlineInputBorder(

                                borderRadius:

                                BorderRadius.all(Radius.circular(10))),

                            hintText: "abc@Email.com",

                            labelText: "Enter Your Email",

                            labelStyle: TextStyle(fontWeight: FontWeight.bold),

                          ),

                        ),

                      ),

                      Padding(

                        padding: const EdgeInsets.all(10.0),

                        child: TextFormField(

                          validator: (value) {

                            if (value!.isEmpty) {

                              return "Enter the Password";

                            }

                            String pattern = r"^(?=.\*[a-zA-Z])(?=.\*\d).+$";

                            var passwordRegExp = RegExp(pattern);

                            if (!passwordRegExp.hasMatch(value)) {

                              return "Enter The Valid Password";

                            }

                            return null;

                          },

                          controller: password,

                          decoration: InputDecoration(

                            border: OutlineInputBorder(

                                borderRadius:

                                BorderRadius.all(Radius.circular(10))),

                            hintText: "Password",

                            labelText: "Enter Your Password",

                            labelStyle: TextStyle(fontWeight: FontWeight.bold),

                          ),

                        ),

                      ), Padding(

                        padding: const EdgeInsets.all(20),

                        child: ElevatedButton(

                          onPressed: () {

                            if (\_formKey.currentState!.validate()) {

                              setState(() {

                                output = uname.text;

                                uname.text = "";

                                outputs = ename.text;

                                ename.text = "";

                                outputp = password.text;

                                password.text = "";

                              });

                            }

                            // print(bname.text),

                            // print(uname.text),

                          },

                          child: Text(

                            "Login",

                            style: TextStyle(color: Colors.black),

                          ),

                        ),

                      ),

                      Center(

                        child: Column(

                          children: [

                            Container(

                              child: Text(

                                "Username: $output",

                                style: TextStyle(fontWeight: FontWeight.bold),

                              ),

                            ),

                            Container(

                              child: Text(

                                "Mail: $outputs",

                                style: TextStyle(fontWeight: FontWeight.bold),

                              ),

                            ),

                            Container(

                              child: Text(

                                "Password: $outputp",

                                style: TextStyle(fontWeight: FontWeight.bold),

                              ),

                            ),

                          ],

                        ),

                      ),

                    ],

                  ),

                )),

          ],

        ),

      ),

    );

  }

}

**B.1 Write a flutter code to create a registration screen using different widgets.**

import 'package:flutter/material.dart';

import 'package:google\_fonts/google\_fonts.dart';

class LAB\_B1 extends StatefulWidget {

  @override

  State<LAB\_B1> createState() => \_LAB\_B1State();

}

class \_LAB\_B1State extends State<LAB\_B1> {

  TextEditingController uname = new TextEditingController();

  TextEditingController ename = new TextEditingController();

  TextEditingController password = new TextEditingController();

  TextEditingController cpassword = new TextEditingController();

  String output = " UserName";

  String outputs = " Mail";

  String outputp = " Password";

  String outputcp = " Confirm Password";

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        backgroundColor: Colors.teal.shade300,

        title: Center(

          child: Text(

            "Welcome",

            style: GoogleFonts.labrada(),

          ),

        ),

      ),

      body: Column(

        children: [

          Expanded(

              child: Container(

                margin: EdgeInsets.all(30),

                decoration: BoxDecoration(

                  border: Border.all(color: Colors.black26),

                  borderRadius: BorderRadius.circular(20),

                ),

                child: Column(

                  mainAxisAlignment: MainAxisAlignment.center,

                  children: [

                    Padding(

                      padding: const EdgeInsets.all(10.0),

                      child: Center(

                        child: Text("Login Page", style: GoogleFonts.pacifico(textStyle:TextStyle(fontSize: 40)),),

                      ),

                    ),

                    Padding(

                      padding: const EdgeInsets.all(10.0),

                      child: TextField(

                        controller: uname,

                        decoration: InputDecoration(

                          border: OutlineInputBorder(

                            borderRadius: BorderRadius.all(Radius.circular(10)),

                          ),

                          hintText: "Username",

                          labelText: "Enter the Name",

                          labelStyle: TextStyle(fontWeight: FontWeight.bold),

                        ),

                      ),

                    ),

                    Padding(

                      padding: const EdgeInsets.all(10.0),

                      child: TextField(

                        controller: ename,

                        decoration: InputDecoration(

                          border: OutlineInputBorder(

                              borderRadius: BorderRadius.all(Radius.circular(10))),

                          hintText: "abc@gmail.com",

                          labelText: "Enter Your E-mail",

                          labelStyle: TextStyle(fontWeight: FontWeight.bold),

                        ),

                      ),

                    ),

                    Padding(

                      padding: const EdgeInsets.all(10.0),

                      child: TextFormField(

                        controller: password,

                        decoration: InputDecoration(

                          border: OutlineInputBorder(

                              borderRadius:

                              BorderRadius.all(Radius.circular(10))),

                          hintText: "Password",

                          labelText: "Enter Password",

                          labelStyle: TextStyle(fontWeight: FontWeight.bold),

                        ),

                      ),

                    ),

                    Padding(

                      padding: const EdgeInsets.all(10.0),

                      child: TextFormField(

                        controller: cpassword,

                        decoration: InputDecoration(

                          border: OutlineInputBorder(

                              borderRadius:

                              BorderRadius.all(Radius.circular(10))),

                          hintText: "Password",

                          labelText: "Enter Confirm Password",

                          labelStyle: TextStyle(fontWeight: FontWeight.bold),

                        ),

                      ),

                    ),

                    Padding(

                      padding: const EdgeInsets.all(20),

                      child: ElevatedButton(

                        onPressed: () {

                          print(uname.text);

                          print(ename.text);

                          print(password.text);

                          print(cpassword.text);

                          setState(() {

                            uname.text = "";

                            ename.text = "";

                            password.text = "";

                            cpassword.text = "";

                          });

                        },

                        child: Text(

                          "Login",

                          style: TextStyle(color: Colors.black),

                        ),

                      ),

                    ),

                  ],

                ),

              )),

        ],

      ),

    );

  }

}

**B.2 Write a flutter code to do validation in the registration screen.**

import 'package:flutter/material.dart';

import 'package:google\_fonts/google\_fonts.dart';

class Validation2 extends StatefulWidget {

  @override

  State<Validation2> createState() => \_Validation2State();

}

class \_Validation2State extends State<Validation2> {

  Widget Txtstyle(String name, double? n) {

    return Text(

      name,

      style: TextStyle(

        fontSize: n,

        fontFamily: "myName",

      ),

    );

  }

  TextEditingController uname = new TextEditingController();

  TextEditingController ename = new TextEditingController();

  TextEditingController password = new TextEditingController();

  TextEditingController cpassword = new TextEditingController();

  String output = " UserName";

  String outputs = " Mail";

  String outputp = " Password";

  String outputcp = " Confirm Password";

  final \_formKey = GlobalKey<FormState>();

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      body: Form(

        key: \_formKey,

        child: Column(

          children: [

            Expanded(

                child: Container(

                  margin: EdgeInsets.all(10),

                  decoration: BoxDecoration(

                    border: Border.all(color: Colors.black26),

                    borderRadius: BorderRadius.circular(20),

                  ),

                  child: Column(

                    mainAxisAlignment: MainAxisAlignment.center,

                    children: [

                      Padding(

                        padding: const EdgeInsets.all(7.0),

                        child: Center(

                          child: Text("Login Page",style: TextStyle(fontSize: 30),)

                        ),

                      ),

                      Padding(

                        padding: const EdgeInsets.all(3.0),

                        child: TextFormField(

                          validator: (value) {

                            if (value!.isEmpty) {

                              return "Enter the Name";

                            }

                            return null;

                          },

                          controller: uname,

                          decoration: InputDecoration(

                            border: OutlineInputBorder(

                              borderRadius: BorderRadius.all(Radius.circular(10)),

                            ),

                            hintText: "Username",

                            labelText: "Enter the Name",

                            labelStyle: TextStyle(fontWeight: FontWeight.bold),

                          ),

                        ),

                      ),

                      Padding(

                        padding: const EdgeInsets.all(3.0),

                        child: TextFormField(

                          validator: (value) {

                            if (value!.isEmpty) {

                              return "Enter the E-mail";

                            }

                            String pattern = "[a-z]+[0-9]\*@gmail.com";

                            var emailRegExp = RegExp(pattern);

                            if (!emailRegExp.hasMatch(value!)) {

                              return "Enter The Valid Mail Address";

                            }

                            return null;

                          },

                          controller: ename,

                          decoration: InputDecoration(

                            border: OutlineInputBorder(

                                borderRadius:

                                BorderRadius.all(Radius.circular(10))),

                            hintText: "abc@Email.com",

                            labelText: "Enter Your Email",

                            labelStyle: TextStyle(fontWeight: FontWeight.bold),

                          ),

                        ),

                      ),

                      Padding(

                        padding: const EdgeInsets.all(3.0),

                        child: TextFormField(

                          validator: (value) {

                            if (value!.isEmpty) {

                              return "Enter the Password";

                            }

                            String pattern = r"^(?=.\*[a-zA-Z])(?=.\*\d).+$";

                            var passwordRegExp = RegExp(pattern);

                            if (!passwordRegExp.hasMatch(value)) {

                              return "Enter The Valid Password";

                            }

                            return null;

                          },

                          controller: password,

                          decoration: InputDecoration(

                            border: OutlineInputBorder(

                                borderRadius:

                                BorderRadius.all(Radius.circular(10))),

                            hintText: "Password",

                            labelText: "Enter Your Password",

                            labelStyle: TextStyle(fontWeight: FontWeight.bold),

                          ),

                        ),

                      ),

                      Padding(

                        padding: const EdgeInsets.all(3.0),

                        child: TextFormField(

                          validator: (value) {

                            if (value!.isEmpty) {

                              return "Confirm Password";

                            }

                            // Check for password match and minimum requirements

                            if (value != password.text || !RegExp(r"^(?=.\*[a-zA-Z])(?=.\*\d).+$").hasMatch(value)) {

                              if (value != password.text) {

                                return "Passwords do not match";

                              } else {

                                return "Password must contain at least one letter and one digit";

                              }

                            }

                            return null;

                          },

                          controller: cpassword,

                          decoration: InputDecoration(

                            border: OutlineInputBorder(

                                borderRadius: BorderRadius.all(Radius.circular(10))),

                            hintText: "Password",

                            labelText: "Confirm Password",

                            labelStyle: TextStyle(fontWeight: FontWeight.bold),

                          ),

                        ),

                      ),

                      ElevatedButton(

                        onPressed: () {

                          if (\_formKey.currentState!.validate()) {

                            setState(() {

                              output = uname.text;

                              uname.text = "";

                              outputs = ename.text;

                              ename.text = "";

                              outputp = password.text;

                              password.text = "";

                              outputcp = password.text;

                              cpassword.text = "";

                            });

                          }

                        },

                        child: Text(

                          "Login",

                          style: TextStyle(color: Colors.black),

                        ),

                      ),

                      Center(

                        child: Column(

                          children: [

                            Container(

                              child: Text(

                                "Username: $output",

                                style: TextStyle(fontWeight: FontWeight.bold),

                              ),

                            ),

                            Container(

                              child: Text(

                                "Mail: $outputs",

                                style: TextStyle(fontWeight: FontWeight.bold),

                              ),

                            ),

                            Container(

                              child: Text(

                                "Password: $outputp",

                                style: TextStyle(fontWeight: FontWeight.bold),

                              ),

                            ),

                            Container(

                              child: Text(

                                "Confirm Password: $outputcp",

                                style: TextStyle(fontWeight: FontWeight.bold),

                              ),

                            ),

                          ],

                        ),

                      ),

                    ],

                  ),

                )),

          ],

        ),

      ),

    );

  }

}

**B.3 Add password visibility icon in the Text field.**

import 'package:flutter/material.dart';

import 'package:google\_fonts/google\_fonts.dart';

class LabC extends StatefulWidget {

  @override

  State<LabC> createState() => \_LabCState();

}

class \_LabCState extends State<LabC> {

  TextEditingController uname = new TextEditingController();

  TextEditingController ename = new TextEditingController();

  TextEditingController password = new TextEditingController();

  TextEditingController cpassword = new TextEditingController();

  bool \_isVisible = false;

  bool \_confirmPasswordVisible = false;

  String output = " UserName";

  String outputs = " Mail";

  String outputp = " Password";

  String outputcp = " Confirm Password";

  final \_formKey = GlobalKey<FormState>();

  @override

  Widget build(BuildContext context) {

    return SafeArea(

      child: Scaffold(

        body: Form(

          key: \_formKey,

          child: Column(

            children: [

              Expanded(

                  child: Container(

                    margin: EdgeInsets.all(10),

                    decoration: BoxDecoration(

                      border: Border.all(color: Colors.black26),

                      borderRadius: BorderRadius.circular(20),

                    ),

                    child: Column(

                      mainAxisAlignment: MainAxisAlignment.center,

                      children: [

                        Padding(

                          padding: const EdgeInsets.all(7.0),

                          child: Center(

                            child: Text("Login Page", style: GoogleFonts.pacifico(textStyle:TextStyle(fontSize: 40)),),

                          ),

                        ),

                        Padding(

                          padding: const EdgeInsets.all(7.0),

                          child: TextFormField(

                            validator: (value) {

                              if (value!.isEmpty) {

                                return "Enter the Name";

                              }

                              return null;

                            },

                            controller: uname,

                            decoration: InputDecoration(

                              border: OutlineInputBorder(

                                borderRadius: BorderRadius.all(Radius.circular(10)),

                              ),

                              hintText: "Username",

                              labelText: "Enter the Name",

                              labelStyle: TextStyle(fontWeight: FontWeight.bold),

                            ),

                          ),

                        ),

                        Padding(

                          padding: const EdgeInsets.all(7.0),

                          child: TextFormField(

                            validator: (value) {

                              if (value!.isEmpty) {

                                return "Enter the E-mail";

                              }

                              String pattern = "[a-z]+[0-9]\*@gmail.com";

                              var emailRegExp = RegExp(pattern);

                              if (!emailRegExp.hasMatch(value!)) {

                                return "Enter The Valid Mail Address";

                              }

                              return null;

                            },

                            controller: ename,

                            decoration: InputDecoration(

                              border: OutlineInputBorder(

                                  borderRadius:

                                  BorderRadius.all(Radius.circular(10))),

                              hintText: "abc@Email.com",

                              labelText: "Enter Your Email",

                              labelStyle: TextStyle(fontWeight: FontWeight.bold),

                            ),

                          ),

                        ),

                        Padding(

                          padding: const EdgeInsets.all(10.0),

                          child: TextFormField(

                            validator: (value) {

                              if (value!.isEmpty) {

                                return "Enter the Password";

                              }

                              String pattern = r"^(?=.\*[a-zA-Z])(?=.\*\d).+$";

                              var emailRegExp = RegExp(pattern);

                              if (!emailRegExp.hasMatch(value)) {

                                return "Enter The Valid Password";

                              }

                              return null;

                            },

                            controller: password,

                            decoration: InputDecoration(

                              border: OutlineInputBorder(

                                  borderRadius:

                                  BorderRadius.all(Radius.circular(10))),

                              hintText: "Password",

                              labelText: "Enter Your Password",

                              labelStyle: TextStyle(fontWeight: FontWeight.bold),

                              suffixIcon: IconButton(

                                icon: Icon(

                                  \_isVisible ? Icons.visibility : Icons.visibility\_off,

                                ),

                                onPressed: () {

setState(() {

                                    \_isVisible = !\_isVisible;

                                  });

                                },

                              ),

                            ),

                            obscureText: !\_isVisible,

                          ),

                        ),

                        Padding(

                          padding: const EdgeInsets.all(10.0),

                          child: TextFormField(

                            validator: (value) {

                              if (value!.isEmpty) {

                                return "Enter the Password";

                              }

                              if (value != password.text) {

                                return "Passwords do not match";

                              }

                              return null;

                            },

                            controller: cpassword,

                            decoration: InputDecoration(

                              border: OutlineInputBorder(

                                  borderRadius: BorderRadius.all(Radius.circular(10))),

                              hintText: "Confirm Password",

                              labelText: "Confirm Your Password",

                              labelStyle: TextStyle(fontWeight: FontWeight.bold),

                              suffixIcon: IconButton(

                                icon: Icon(

                                  \_confirmPasswordVisible ? Icons.visibility : Icons.visibility\_off,

                                ),

                                onPressed: () {

                                                                   setState(() {

                                    \_confirmPasswordVisible = !\_confirmPasswordVisible;

                                  });

                                },

                              ),

                            ),

                            obscureText: !\_confirmPasswordVisible, // Toggle password visibility

                          ),

                        ),

                        Padding(

                          padding: const EdgeInsets.all(20),

                          child: ElevatedButton(

                            onPressed: () {

                              if (\_formKey.currentState!.validate()) {

                                print(uname.text);

                                print(ename.text);

                                print(password.text);

                                print(cpassword.text);

                                setState(() {

                                  uname.text = "";

                                  ename.text = "";

                                  password.text = "";

                                  cpassword.text = "";

                                });

                              }

                            },

                            child: Text(

                              "Login",

                              style: TextStyle(color: Colors.black),

                            ),

                          ),

                        ),

                    ),

                  )),

            ],

          ),

        ),

      ),

    );}}

LAB 11

**A.1 Write a flutter code to display list of city in to listview. (Static list in**

**List<String>)**

class CityListScreen extends StatelessWidget {

  // Static list of cities

  final List<String> cities = [

    'New York',

    'London',

    'Tokyo',

    'Paris',

    'Sydney',

    'Berlin',

    'Mumbai',

    'Dubai',

    'Toronto',

    'Singapore'

  ];

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('City List'),

      ),

      body: ListView.builder(

        itemCount: cities.length,

        itemBuilder: (context, index) {

          return ListTile(

            leading: Icon(Icons.location\_city),

            title: Text(cities[index]),

          );

        },

      ),

    );

  }

}

**A.2 Write a flutter code to display list of image in gridview with static data.**

class ImageGridScreen extends StatelessWidget {

  // Static list of image URLs

  final List<String> images = [

    'https://via.placeholder.com/150/FF0000',

    'https://via.placeholder.com/150/00FF00',

    'https://via.placeholder.com/150/0000FF',

    'https://via.placeholder.com/150/FFFF00',

    'https://via.placeholder.com/150/FF00FF',

    'https://via.placeholder.com/150/00FFFF',

  ];

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('Image Grid'),

      ),

      body: Padding(

        padding: const EdgeInsets.all(8.0),

        child: GridView.builder(

          gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(

            crossAxisCount: 2, // Number of columns

            crossAxisSpacing: 10.0,

            mainAxisSpacing: 10.0,

          ),

          itemCount: images.length,

          itemBuilder: (context, index) {

            return Container(

              decoration: BoxDecoration(

                borderRadius: BorderRadius.circular(10),

                color: Colors.grey[200],

              ),

              child: ClipRRect(

                borderRadius: BorderRadius.circular(10),

                child: Image.network(

                  images[index],

                  fit: BoxFit.cover,

                ),

              ),

            );

          },

        ),

      ),

    );

  }

}

**A.3 Write a flutter code to display image and text in gridview with static data.**

class ImageTextGridScreen extends StatelessWidget {

  // Static list of image URLs and their corresponding texts

  final List<Map<String, String>> items = [

    {'image': 'https://via.placeholder.com/150/FF0000', 'text': 'Red'},

    {'image': 'https://via.placeholder.com/150/00FF00', 'text': 'Green'},

    {'image': 'https://via.placeholder.com/150/0000FF', 'text': 'Blue'},

    {'image': 'https://via.placeholder.com/150/FFFF00', 'text': 'Yellow'},

    {'image': 'https://via.placeholder.com/150/FF00FF', 'text': 'Pink'},

    {'image': 'https://via.placeholder.com/150/00FFFF', 'text': 'Cyan'},

  ];

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('Image & Text Grid'),

      ),

      body: Padding(

        padding: const EdgeInsets.all(8.0),

        child: GridView.builder(

          gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(

            crossAxisCount: 2, // Number of columns

            crossAxisSpacing: 10.0,

            mainAxisSpacing: 10.0,

          ),

          itemCount: items.length,

          itemBuilder: (context, index) {

            return Container(

              decoration: BoxDecoration(

                borderRadius: BorderRadius.circular(10),

                color: Colors.grey[200],

              ),

              child: Column(

                crossAxisAlignment: CrossAxisAlignment.center,

                children: [

                  Expanded(

                    child: ClipRRect(

                      borderRadius: BorderRadius.circular(10),

                      child: Image.network(

                        items[index]['image']!,

                        fit: BoxFit.cover,

                        width: double.infinity,

                      ),

                    ),

                  ),

                  Padding(

                    padding: const EdgeInsets.all(8.0),

                    child: Text(

                      items[index]['text']!,

                      style: TextStyle(fontSize: 16, fontWeight: FontWeight.bold),

                    ),

                  ),

                ],

              ),

            );

          },

        ),

      ),

    );

  }

}

**A.4 Write a flutter code to display data in listview. Place switch button at top bar. Click on switch button change view as gridview to listview and vice versa.**

class ToggleListGridScreen extends StatefulWidget {

  @override

  \_ToggleListGridScreenState createState() => \_ToggleListGridScreenState();

}

class \_ToggleListGridScreenState extends State<ToggleListGridScreen> {

  bool \_isGridView = false; // Toggle state for GridView or ListView

  // Static data to display

  final List<Map<String, String>> items = List.generate(

    10,

    (index) => {

      'image': 'https://via.placeholder.com/150/${index \* 111111.toRadixString(16)}',

      'text': 'Item ${index + 1}',

    },

  );

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('Toggle View'),

        actions: [

          Row(

            children: [

              Text('Grid', style: TextStyle(color: Colors.white)),

              Switch(

                value: \_isGridView,

                onChanged: (value) {

                  setState(() {

                    \_isGridView = value;

                  });

                },

              ),

              Text('List', style: TextStyle(color: Colors.white)),

            ],

          ),

        ],

      ),

      body: \_isGridView

          ? GridView.builder(

              gridDelegate: SliverGridDelegateWithFixedCrossAxisCount(

                crossAxisCount: 2, // Number of columns

                crossAxisSpacing: 10.0,

                mainAxisSpacing: 10.0,

              ),

              padding: EdgeInsets.all(8.0),

              itemCount: items.length,

              itemBuilder: (context, index) {

                return \_buildItem(items[index]);

              },

            )

          : ListView.builder(

              padding: EdgeInsets.all(8.0),

              itemCount: items.length,

              itemBuilder: (context, index) {

                return \_buildItem(items[index]);

              },

            ),

    );

  }

  Widget \_buildItem(Map<String, String> item) {

    return Card(

      elevation: 4.0,

      child: Row(

        children: [

          Container(

            width: 100,

            height: 100,

            decoration: BoxDecoration(

              borderRadius: BorderRadius.circular(10),

              color: Colors.grey[200],

            ),

            child: ClipRRect(

              borderRadius: BorderRadius.circular(10),

              child: Image.network(

                item['image']!,

                fit: BoxFit.cover,

              ),

            ),

          ),

          Expanded(

            child: Padding(

              padding: const EdgeInsets.all(8.0),

              child: Text(

                item['text']!,

                style: TextStyle(fontSize: 16, fontWeight: FontWeight.bold),

              ),

            ),

          ),

        ],

      ),

    );

  }

}

LAB 12

**A.1 Write a flutter code scroll whole screen content using single child scrollview.**

class ScrollableScreen extends StatelessWidget {

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('SingleChildScrollView Example'),

      ),

      body: SingleChildScrollView(

        child: Padding(

          padding: const EdgeInsets.all(16.0),

          child: Column(

            crossAxisAlignment: CrossAxisAlignment.start,

            children: [

              Text(

                'Header',

                style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),

              ),

              SizedBox(height: 20),

              Text(

                'This is a sample paragraph to demonstrate scrolling. You can add as much content as you need, and the screen will remain scrollable. Here’s some sample text to fill the space.',

                style: TextStyle(fontSize: 16),

              ),

              SizedBox(height: 20),

              Image.network(

                'https://via.placeholder.com/400x200',

                fit: BoxFit.cover,

              ),

              SizedBox(height: 20),

              for (int i = 1; i <= 10; i++)

                Padding(

                  padding: const EdgeInsets.symmetric(vertical: 10.0),

                  child: Text(

                    'Item $i',

                    style: TextStyle(fontSize: 18),

                  ),

                ),

              SizedBox(height: 20),

              ElevatedButton(

                onPressed: () {},

                child: Text('Click Me'),

              ),

            ],

          ),

        ),

      ),

    );

  }

}

**A.2 Write a flutter code to use listview/gridview inside single child scrollview.**

class ScrollableWithListView extends StatelessWidget {

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('SingleChildScrollView with ListView'),

      ),

      body: SingleChildScrollView(

        child: Column(

          crossAxisAlignment: CrossAxisAlignment.start,

          children: [

            Padding(

              padding: const EdgeInsets.all(16.0),

              child: Text(

                'Scrollable Content Above ListView',

                style: TextStyle(fontSize: 20, fontWeight: FontWeight.bold),

              ),

            ),

            Padding(

              padding: const EdgeInsets.all(16.0),

              child: Text(

                'Here is some content above the ListView. The ListView itself is constrained to fit properly within the SingleChildScrollView.',

                style: TextStyle(fontSize: 16),

              ),

            ),

            SizedBox(

              height: 300, // Constrain the height of the ListView

              child: ListView.builder(

                itemCount: 10,

                itemBuilder: (context, index) {

                  return ListTile(

                    leading: Icon(Icons.list),

                    title: Text('Item ${index + 1}'),

                  );

                },

              ),

            ),

            Padding(

              padding: const EdgeInsets.all(16.0),

              child: Text(

                'Scrollable Content Below ListView',

                style: TextStyle(fontSize: 20, fontWeight: FontWeight.bold),

              ),

            ),

            Padding(

              padding: const EdgeInsets.all(16.0),

              child: Text(

                'Additional content goes here. You can scroll past the ListView and see this content below.',

                style: TextStyle(fontSize: 16),

              ),

            ),

          ],

        ),

      ),

    );

  }

}

**A.3 Write a flutter code to get current date from system and format into different date formats. Ex.**

**dd/MM/yyyy**

**dd-MM-yyyy**

**dd-MMM-yyyy**

**dd-MM-yy**

**dd MMM, yyyy**

**etc.**

class DateFormatsScreen extends StatelessWidget {

  @override

  Widget build(BuildContext context) {

    DateTime now = DateTime.now(); // Get current date and time

    // Format the current date into different formats

    String format1 = DateFormat('dd/MM/yyyy').format(now);

    String format2 = DateFormat('dd-MM-yyyy').format(now);

    String format3 = DateFormat('dd-MMM-yyyy').format(now);

    String format4 = DateFormat('dd-MM-yy').format(now);

    String format5 = DateFormat('dd MMM, yyyy').format(now);

    return Scaffold(

      appBar: AppBar(

        title: Text('Date Formats'),

      ),

      body: Padding(

        padding: const EdgeInsets.all(16.0),

        child: Column(

          crossAxisAlignment: CrossAxisAlignment.start,

          children: [

            Text(

              'Current Date in Different Formats:',

              style: TextStyle(fontSize: 20, fontWeight: FontWeight.bold),

            ),

            SizedBox(height: 16),

            Text('dd/MM/yyyy: $format1', style: TextStyle(fontSize: 16)),

            Text('dd-MM-yyyy: $format2', style: TextStyle(fontSize: 16)),

            Text('dd-MMM-yyyy: $format3', style: TextStyle(fontSize: 16)),

            Text('dd-MM-yy: $format4', style: TextStyle(fontSize: 16)),

            Text('dd MMM, yyyy: $format5', style: TextStyle(fontSize: 16)),

          ],

        ),

      ),

    );

  }

}

**B.1 Write a flutter code get date frod date picker dialog and display in textview.**

class DatePickerExample extends StatefulWidget {

  @override

  \_DatePickerExampleState createState() => \_DatePickerExampleState();

}

class \_DatePickerExampleState extends State<DatePickerExample>

  DateTime \_selectedDate = DateTime.now();

  Future<void> \_selectDate(BuildContext context) async {

    final DateTime picked = await showDatePicker(

      context: context,

      initialDate: \_selectedDate,

      firstDate: DateTime(2000),

      lastDate: DateTime(2101),

    ) ?? \_selectedDate;

    if (picked != null && picked != \_selectedDate) {

      setState(() {

        \_selectedDate = picked;

      });

    }

  }

  @override

  Widget build(BuildContext context) {

    String formattedDate = "${\_selectedDate.day}/${\_selectedDate.month}/${\_selectedDate.year}";

    return Scaffold(

      appBar: AppBar(

        title: Text('Date Picker Example'),

      ),

      body: Padding(

        padding: const EdgeInsets.all(16.0),

        child: Column(

          children: [

            Text(

              'Selected Date: $formattedDate',

              style: TextStyle(fontSize: 20, fontWeight: FontWeight.bold),

            ),

            SizedBox(height: 20),

            ElevatedButton(

              onPressed: () => \_selectDate(context),

              child: Text('Pick a Date'),

            ),

          ],),),);}}

LAB 13

**A.1 Write a flutter code to open bottom sheet on button click.**

class BottomSheetExample extends StatelessWidget {

  // Function to show the bottom sheet

  void \_showBottomSheet(BuildContext context) {

    showModalBottomSheet(

      context: context,

      builder: (BuildContext context) {

        return Container(

          padding: EdgeInsets.all(16),

          height: 200, // Height of the bottom sheet

          child: Column(

            crossAxisAlignment: CrossAxisAlignment.start,

            children: [

              Text(

                'This is a Bottom Sheet',

                style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),

              ),

              SizedBox(height: 16),

              Text('You can add any widget here.'),

              SizedBox(height: 16),

              ElevatedButton(

                onPressed: () {

                  Navigator.pop(context); // Close the bottom sheet

                },

                child: Text('Close'),

              ),

            ],

          ),

        );

      },

    );

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('Bottom Sheet Example'),

      ),

      body: Center(

        child: ElevatedButton(

          onPressed: () => \_showBottomSheet(context), // Open bottom sheet on button click

          child: Text('Open Bottom Sheet'),

        ),

      ),

    );

  }

}

**A.2 Write a flutter code to create and use navigation drawer.**

import 'package:flutter/material.dart';

class HomePage extends StatelessWidget {

const HomePage({super.key});

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: const Text('Navigation Drawer Demo'),

),

drawer: Drawer(

child: ListView(

padding: EdgeInsets.zero,

children: [

const UserAccountsDrawerHeader(

accountName: Text('John Doe'),

accountEmail: Text('john.doe@example.com'),

currentAccountPicture: CircleAvatar(

backgroundImage: AssetImage('assets/profile.png'), // Add your image in assets

),

decoration: BoxDecoration(

color: Colors.blueAccent,

),

),

ListTile(

leading: const Icon(Icons.home),

title: const Text('Home'),

onTap: () {

Navigator.pop(context); // Close the drawer

// Handle navigation

},

),

ListTile(

leading: const Icon(Icons.person),

title: const Text('Profile'),

onTap: () {

Navigator.pop(context);

// Handle navigation

},

),

ListTile(

leading: const Icon(Icons.settings),

title: const Text('Settings'),

onTap: () {

Navigator.pop(context);

// Handle navigation

},

),

const Divider(), // Adds a divider

ListTile(

leading: const Icon(Icons.exit\_to\_app),

title: const Text('Logout'),

onTap: () {

Navigator.pop(context);

// Handle logout

},

),

],

),

),

body: const Center(

child: Text(

'Welcome to the Home Page!',

style: TextStyle(fontSize: 24),

),

),

);

}

}

**A.3 Write a flutter code to use the bottom NavigationBar & on click display different pages.**

class BottomNavigationBarExample extends StatefulWidget {

  @override

  \_BottomNavigationBarExampleState createState() =>

      \_BottomNavigationBarExampleState();

}

class \_BottomNavigationBarExampleState extends State<BottomNavigationBarExample> {

  // Current index of the selected page

  int \_selectedIndex = 0;

  // List of pages to display

  final List<Widget> \_pages = [

    HomePage(),

    SearchPage(),

    NotificationsPage(),

    ProfilePage(),

  ];

  // Function to handle item selection in the Bottom Navigation Bar

  void \_onItemTapped(int index) {

    setState(() {

      \_selectedIndex = index; // Update the selected index

    });

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('Bottom Navigation Bar Example'),

      ),

      body: \_pages[\_selectedIndex], // Display the selected page

      bottomNavigationBar: BottomNavigationBar(

        currentIndex: \_selectedIndex, // Highlight the selected item

        onTap: \_onItemTapped, // Handle tap event

        items: const <BottomNavigationBarItem>[

          BottomNavigationBarItem(

            icon: Icon(Icons.home),

            label: 'Home',

          ),

          BottomNavigationBarItem(

            icon: Icon(Icons.search),

            label: 'Search',

          ),

          BottomNavigationBarItem(

            icon: Icon(Icons.notifications),

            label: 'Notifications',

          ),

          BottomNavigationBarItem(

            icon: Icon(Icons.person),

            label: 'Profile',

          ),

        ],

      ),

    );

  }

}

// Sample Pages for demonstration

class HomePage extends StatelessWidget {

  @override

  Widget build(BuildContext context) {

    return Center(

      child: Text(

        'Home Page',

        style: TextStyle(fontSize: 24),

      ),

    );

  }

}

class SearchPage extends StatelessWidget {

  @override

  Widget build(BuildContext context) {

    return Center(

      child: Text(

        'Search Page',

        style: TextStyle(fontSize: 24),

      ),

    );

  }

}

class NotificationsPage extends StatelessWidget {

  @override

  Widget build(BuildContext context) {

    return Center(

      child: Text(

        'Notifications Page',

        style: TextStyle(fontSize: 24),

      ),

    );

  }

}

class ProfilePage extends StatelessWidget {

  @override

  Widget build(BuildContext context) {

    return Center(

      child: Text(

        'Profile Page',

        style: TextStyle(fontSize: 24),

      ),

    );

  }

}

**A.4 Write a flutter code to create form using different widgets and do validation on it.**

class FormExample extends StatefulWidget {

  @override

  \_FormExampleState createState() => \_FormExampleState();

}

class \_FormExampleState extends State<FormExample> {

  // GlobalKey to identify the form and validate it

  final \_formKey = GlobalKey<FormState>();

  // Controllers for the form fields

  final TextEditingController \_nameController = TextEditingController();

  final TextEditingController \_emailController = TextEditingController();

  final TextEditingController \_passwordController = TextEditingController();

  // Focus nodes for field focus management

  final FocusNode \_emailFocusNode = FocusNode();

  final FocusNode \_passwordFocusNode = FocusNode();

  // Function to handle form submission

  void \_submitForm() {

    if (\_formKey.currentState!.validate()) {

      // If the form is valid, display the form data

      ScaffoldMessenger.of(context).showSnackBar(

        SnackBar(content: Text('Form Submitted!')),

      );

    }

  }

  @override

  void dispose() {

    // Clean up the controllers and focus nodes

    \_nameController.dispose();

    \_emailController.dispose();

    \_passwordController.dispose();

    \_emailFocusNode.dispose();

    \_passwordFocusNode.dispose();

    super.dispose();

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('Form Validation Example'),

      ),

      body: Padding(

        padding: const EdgeInsets.all(16.0),

        child: Form(

          key: \_formKey, // Assign the form key to the form

          child: Column(

            children: [

              // Name Field

              TextFormField(

                controller: \_nameController,

                decoration: InputDecoration(

                  labelText: 'Name',

                  border: OutlineInputBorder(),

                ),

                validator: (value) {

                  if (value == null || value.isEmpty) {

                    return 'Name cannot be empty';

                  }

                  return null;

                },

                textInputAction: TextInputAction.next,

                onFieldSubmitted: (\_) {

                  FocusScope.of(context).requestFocus(\_emailFocusNode);

                },

              ),

              SizedBox(height: 16),

              // Email Field

              TextFormField(

                controller: \_emailController,

                focusNode: \_emailFocusNode,

                decoration: InputDecoration(

                  labelText: 'Email',

                  border: OutlineInputBorder(),

                ),

                validator: (value) {

                  if (value == null || value.isEmpty) {

                    return 'Email cannot be empty';

                  } else if (!RegExp(r'\S+@\S+\.\S+').hasMatch(value)) {

                    return 'Enter a valid email';

                  }

                  return null;

                },

                textInputAction: TextInputAction.next,

                onFieldSubmitted: (\_) {

                  FocusScope.of(context).requestFocus(\_passwordFocusNode);

                },

              ),

              SizedBox(height: 16),

              // Password Field

              TextFormField(

                controller: \_passwordController,

                focusNode: \_passwordFocusNode,

                obscureText: true,

                decoration: InputDecoration(

                  labelText: 'Password',

                  border: OutlineInputBorder(),

                ),

                validator: (value) {

                  if (value == null || value.isEmpty) {

                    return 'Password cannot be empty';

                  } else if (value.length < 6) {

                    return 'Password must be at least 6 characters';

                  }

                  return null;

                },

              ),

              SizedBox(height: 16),

              // Submit Button

              ElevatedButton(

                onPressed: \_submitForm,

                child: Text('Submit'),

              ),

            ],

          ),

        ),

      ),

    );

  }

}

**A.5 Write a flutter code create form using bottom sheet dialog.**

class BottomSheetFormExample extends StatelessWidget {

  // GlobalKey to identify the form and validate it

  final \_formKey = GlobalKey<FormState>();

  // Controllers for the form fields

  final TextEditingController \_nameController = TextEditingController();

  final TextEditingController \_emailController = TextEditingController();

  // Function to show the bottom sheet and create the form

  void \_showBottomSheet(BuildContext context) {

    showModalBottomSheet(

      context: context,

      builder: (BuildContext context) {

        return Padding(

          padding: const EdgeInsets.all(16.0),

          child: SingleChildScrollView(

            child: Form(

              key: \_formKey, // Assign the form key to the form

              child: Column(

                children: [

                  // Name Field

                  TextFormField(

                    controller: \_nameController,

                    decoration: InputDecoration(

                      labelText: 'Name',

                      border: OutlineInputBorder(),

                    ),

                    validator: (value) {

                      if (value == null || value.isEmpty) {

                        return 'Name cannot be empty';

                      }

                      return null;

                    },

                  ),

                  SizedBox(height: 16),

                  // Email Field

                  TextFormField(

                    controller: \_emailController,

                    decoration: InputDecoration(

                      labelText: 'Email',

                      border: OutlineInputBorder(),

                    ),

                    validator: (value) {

                      if (value == null || value.isEmpty) {

                        return 'Email cannot be empty';

                      } else if (!RegExp(r'\S+@\S+\.\S+').hasMatch(value)) {

                        return 'Enter a valid email';

                      }

                      return null;

                    },

                  ),

                  SizedBox(height: 16),

                  // Submit Button

                  ElevatedButton(

                    onPressed: () {

                      if (\_formKey.currentState!.validate()) {

                        // If the form is valid, display the form data

                        ScaffoldMessenger.of(context).showSnackBar(

                          SnackBar(content: Text('Form Submitted!')),

                        );

                        Navigator.pop(context); // Close the bottom sheet

                      }

                    },

                    child: Text('Submit'),

                  ),

                ],

              ),

            ),

          ),

        );

      },

    );

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('Bottom Sheet Form Example'),

      ),

      body: Center(

        child: ElevatedButton(

          onPressed: () => \_showBottomSheet(context), // Open bottom sheet on button click

          child: Text('Open Form'),

        ),

      ),

    );

  }

}

LAB 14

**A.1 Write a flutter code to do navigation between two different pages using a material page route.**

class HomePage extends StatelessWidget {

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('Home Page'),

),

body: Center(

child: ElevatedButton(

onPressed: () {

// Navigating to the SecondPage using MaterialPageRoute

Navigator.push(

context,

MaterialPageRoute(builder: (context) => SecondPage()),

);

},

child: Text('Go to Second Page'),

),

),

);

}

}

class SecondPage extends StatelessWidget {

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: Text('Second Page'),

),

body: Center(

child: ElevatedButton(

onPressed: () {

// Navigating back to the HomePage

Navigator.pop(context);

},

child: Text('Back to Home Page'),

),

),

);

}

}

**A.2 Write a flutter code to do navigation between two different pages using a name route.**

import 'package:flutter/material.dart';

class MyApp extends StatelessWidget {

  @override

  Widget build(BuildContext context) {

    return MaterialApp(

      // Defining the named routes

      initialRoute: '/', // Default route

      routes: {

        '/': (context) => HomePage(),

        '/second': (context) => SecondPage(),

      },

    );

  }

}

class HomePage extends StatelessWidget {

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('Home Page'),

      ),

      body: Center(

        child: ElevatedButton(

          onPressed: () {

            // Navigating to the SecondPage using named route

            Navigator.pushNamed(context, '/second');

          },

          child: Text('Go to Second Page'),

        ),

      ),

    );

  }

}

class SecondPage extends StatelessWidget {

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('Second Page'),

      ),

      body: Center(

        child: ElevatedButton(

          onPressed: () {

            // Navigating back to the HomePage

            Navigator.pop(context);

          },

          child: Text('Back to Home Page'),

        ),

      ),

    );

  }

}

**A.3 Write a flutter code to create a dynamic Birthday card.**

import 'package:flutter/material.dart';

class Bday extends StatefulWidget {

  const Bday({super.key});

  @override

  \_BdayState createState() => \_BdayState();

}

class \_BdayState extends State<Bday> {

  // Controller to get the name input

  TextEditingController nameController = TextEditingController();

  // Variable to store the entered name

  String name = '';

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      body: Stack(

        fit: StackFit.expand,

        children: [

          // Background image for the card

          Image.asset(

            'assets/images/bday.jpg', // Make sure this image path is correct

            fit: BoxFit.cover,

          ),

          Center(

            child: Column(

              mainAxisAlignment: MainAxisAlignment.center,

              children: [

                // TextField to input the name

                Padding(

                  padding: const EdgeInsets.symmetric(horizontal: 20.0),

                  child: TextField(

                    controller: nameController,

                    decoration: InputDecoration(

                      labelText: 'Enter Name',

                      border: OutlineInputBorder(),

                      filled: true,

                      fillColor: Colors.white.withOpacity(0.7),

                    ),

                    onChanged: (value) {

                      setState(() {

                        name = value; // Update the name variable as the user types

                      });

                    },

                  ),

                ),

                SizedBox(height: 20),

                // Birthday text with dynamic name

                Padding(

                  padding: const EdgeInsets.symmetric(vertical: 20.0),

                  child: Text(

                    name.isEmpty ? "Happy Birthday" : "Happy Birthday, $name",

                    style: TextStyle(

                      fontSize: 70,

                      fontFamily: 'LongCang-Regular', // Make sure the font is included

                      color: Colors.indigo,

                      fontWeight: FontWeight.bold,

                    ),

                  ),

                ),

                // Cake image below the text

                Image.asset(

                  'assets/images/cake1.png', // Ensure this path is correct

                  fit: BoxFit.contain, // Adjust the box fit for the cake image

                  height: 200, // Limit the height of the cake image

                ),

              ],

            ),

          ),

        ],),);}}

LAB 15

**A.1 Create different tables for the to-do list app in SQLite and attach a .db file with the flutter project.**

import 'dart:io';

import 'package:sqflite/sqflite.dart';

import 'package:path/path.dart';

import 'package:flutter/services.dart';

class MyDatabase {

  Future<Database> loadDatabase() async {

    final dbPath = await getDatabasesPath();

    final path = join(dbPath, 'todo.db');

    // Copy the database from assets if it doesn't exist

    final exists = await databaseExists(path);

    if (!exists) {

      ByteData data = await rootBundle.load('assets/todo.db');

      List<int> bytes =

          data.buffer.asUint8List(data.offsetInBytes, data.lengthInBytes);

      // Write the copied bytes to the database file

      await File(path).writeAsBytes(bytes, flush: true);

    }

    // Open the database

    return await openDatabase(path);

  }

  Future<List<Map<String, dynamic>>> getAllTasks() async {

    Database db = await loadDatabase();

    return await db.query('tasks');

  }

  Future<void> addTask(String title, int categoryId) async {

    Database db = await loadDatabase();

    await db.insert('tasks', {'title': title, 'category\_id': categoryId});

  }

  Future<void> addCategories(String name) async {

    Database db = await loadDatabase();

    await db.insert('categories', {'name': name});

  }

  Future<List<Map<String, dynamic>>> getTasksWithCategories() async {

    Database db = await loadDatabase();

    return await db.rawQuery('''

      select t.id , t.title , t.category\_id , c.name from tasks as t

      inner join categories as c

      on t.category\_id = c.id

      ''');

  }

}

**B.1 Query table and display data in terminal using print.**

import 'package:flutter/material.dart';

import 'package:lebs/lab-15/database.dart';

class SelectAllTasks extends StatefulWidget {

  const SelectAllTasks({super.key});

  @override

  State<SelectAllTasks> createState() => \_SelectAllTasksState();

}

class \_SelectAllTasksState extends State<SelectAllTasks> {

  @override

  void initState() {

    // TODO: implement initState

    super.initState();

    fetchData();

  }

  void fetchData() async {

    print("loading..");

    MyDatabase db = new MyDatabase();

    var tasks = await db.getAllTasks();

    print('All Tasks: $tasks');

    print('All Tasks: ');

    //add addCategories

    await db.addCategories("abc");

    var tasksWithCategories = await db.getTasksWithCategories();

    print('Tasks with Categories: $tasksWithCategories');

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold();

  }

}

LAB 16

**A.1 Write a flutter code to design To-Do list insert screen and do validation on it.**

import 'package:flutter/material.dart';

class addOrEditTodu extends StatefulWidget {

  String? title;

  String? dec;

  addOrEditTodu({this.title, this.dec});

  @override

  State<addOrEditTodu> createState() => \_addOrEditToduState();

}

class \_addOrEditToduState extends State<addOrEditTodu> {

  TextEditingController titleCon = TextEditingController();

  TextEditingController decCon = TextEditingController();

  @override

  void initState() {

    super.initState();

    titleCon.text = widget.title ?? '';

    decCon.text = widget.dec ?? '';

  }

  @override

  Widget build(BuildContext context) {

    return AlertDialog(

      title: Text(widget.title == null ? 'add' : 'edit'),

      content: Container(

        height: MediaQuery.of(context).size.height \* 0.2,

        child: Column(

          children: [

            TextField(

              controller: titleCon,

              decoration: const InputDecoration(

                  border: OutlineInputBorder(), hintText: 'enter title'),

            ),

            const SizedBox(

              height: 20,

            ),

            TextField(

              controller: decCon,

              decoration: const InputDecoration(

                  border: OutlineInputBorder(), hintText: 'enter des'),

            )

          ],

        ),

      ),

      actions: [

        ElevatedButton(

            style: ButtonStyle(

              backgroundColor: MaterialStateProperty.all(Colors.redAccent),

            ),

            onPressed: () {

              Navigator.pop(context);

            },

            child: const Text(

              'cancel',

            )),

        ElevatedButton(

            onPressed: () {

              // Todu(id: 1 , title: titleCon.text.toString() , dec:  decCon.text.toString())

              // print(todu.length.toString());

              Navigator.pop(context, {

                'title': titleCon.text.toString(),

                'dec': decCon.text.toString()

              });

            },

            child: Text(widget.title == null ? 'create' : 'update'))

      ],

    );

  }

}

**A.2 Write a flutter code to insert data into the table.**

import 'package:flutter/material.dart';

import 'package:lebs/lab-16/addOrEditTodu.dartWidget.dart';

import 'package:lebs/lab-16/myDatabase.dart';

class HomePage extends StatefulWidget {

  @override

  State<HomePage> createState() => \_HomePageState();

}

class \_HomePageState extends State<HomePage> {

  MyDatabase myDatabase = MyDatabase();

  // List<Todu> todu = [];

  TextEditingController titleCon = TextEditingController();

  TextEditingController decCon = TextEditingController();

  @override

  void initState() {

    // TODO: implement initState

    super.initState();

    // insertData();

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        backgroundColor: Theme.of(context).primaryColor,

        title: const Text(

          'All Todu',

          style: TextStyle(color: Colors.white),

        ),

        centerTitle: true,

      ),

      body: FutureBuilder(

        future: myDatabase.getAllTodu(),

        // future: futureTodu,

        builder: (context, snapshot) {

          if (snapshot.connectionState == ConnectionState.waiting) {

            return const Center(child: CircularProgressIndicator());

          } else if (snapshot.hasData) {

            final toduList = snapshot.data as List<Map<String, dynamic>> ?? [];

            if (toduList.isEmpty) {

              return Center(

                  child: Text(

                'No data',

                style: TextStyle(color: Colors.grey, fontSize: 20),

              ));

            }

            return Expanded(

              child: ListView.builder(

                scrollDirection: Axis.vertical,

                itemCount: toduList.length,

                // itemCount: 20,

                itemBuilder: (context, index) {

                  Map<String, dynamic> currTodu = toduList[index];

                  return Card(

                    margin:

                        const EdgeInsets.symmetric(horizontal: 10, vertical: 8),

                    elevation: 8,

                    child: ListTile(

                      leading: CircleAvatar(

                        child: Text('${index + 1}'),

                      ),

                      title: Text(currTodu['title']),

                      trailing: Container(

                        width: 50,

                        child: Row(

                          children: [

                            Expanded(

                                child: InkWell(

                                    onTap: () {

                                      showDialog(

                                          context: context,

                                          builder: (context) {

                                            return addOrEditTodu(

                                              title: currTodu['title'],

                                              dec: currTodu['dec'],

                                            );

                                          }).then((value) async => {

                                            await myDatabase.updateTodu(

                                                value['title'],

                                                value['dec'],

                                                currTodu['id']),

                                            setState(() {})

                                          });

                                    },

                                    child: const Icon(Icons.edit))),

                            const SizedBox(

                              width: 10,

                            ),

                            Expanded(

                                child: InkWell(

                                    onTap: () {

                                      showDialog(

                                        context: context,

                                        builder: (context) {

                                          return AlertDialog(

                                            title:

                                                Text('Are you sure delete ?'),

                                            actions: [

                                              TextButton(

                                                  onPressed: () {

                                                    Navigator.of(context)

                                                        .pop(true);

                                                  },

                                                  child: Text('Yes')),

                                              TextButton(

                                                  onPressed: () {

                                                    Navigator.of(context)

                                                        .pop(false);

                                                  },

                                                  child: Text('No'))

                                            ],

                                          );

                                        },

                                      ).then((value) async {

                                        if (value) {

                                          myDatabase.deleteTodu(currTodu['id']);

                                          setState(() {});

                                        }

                                      });

                                    },

                                    child: const Icon(

                                      Icons.delete,

                                      color: Colors.redAccent,

                                    ))),

                          ],

                        ),

                      ),

                    ),

                  );

                },

              ),

            );

          } else {

            return const Text('error');

          }

        },

      ),

      floatingActionButton: FloatingActionButton(

        onPressed: () {

          showDialog(

              context: context,

              builder: (context) {

                // return Text("");

                return addOrEditTodu();

              }).then((value) => {

                setState(() {

                  myDatabase.insert(value);

                  setState(() {});

                  // futureTodu = myDatabase.getAllTodu();

                }),

              });

        },

        child: const Icon(Icons.add),

      ),

    );

  }

}

// Database…

import 'dart:io';

import 'package:path\_provider/path\_provider.dart';

import 'package:sqflite/sqflite.dart';

import 'package:path/path.dart';

class MyDatabase {

  static const tableName = "Todu";

  Future<Database> initDatabase() async {

    Directory directory = await getApplicationCacheDirectory();

    String path = join(directory.path, 'curd.db');

    var db = await openDatabase(path, onCreate: (db, version) async {

      await db.execute('''

          create table $tableName(

           id INTEGER PRIMARY KEY AUTOINCREMENT,

           title TEXT NOT NULL,

           dec TEXT NOT NULL

          )

        ''');

    }, onUpgrade: (db, oldVersion, newVersion) {

      // if(newVersion > oldVersion){

      //   db.execute('ALTER TABLE TblUser()');

      // }

    }, version: 1);

    return db;

  }

  Future<int> insert(Map<String, dynamic> todu) async {

    Database db = await initDatabase();

    return await db.insert(tableName, todu);

  }

  Future<List<Map<String, dynamic>>> getAllTodu() async {

    Database db = await initDatabase();

    return db.query(tableName);

  }

  Future<int> updateTodu(String title, String dec, int id) async {

    Database db = await initDatabase();

    return await db.update(tableName, {'title': title, 'dec': dec},

        where: 'id = ?', whereArgs: [id]);

  }

  Future<int> deleteTodu(int id) async {

    Database db = await initDatabase();

    return await db.delete(tableName, where: 'id = ?', whereArgs: [id]);

  }

}

**B.1 Write a flutter code to fill country list in dropdown and then fill Satelist based on selected country and then fill city dropdown based on state.**

import 'package:flutter/material.dart';

class CountryStateCityDropdown extends StatefulWidget {

  @override

  \_CountryStateCityDropdownState createState() =>

      \_CountryStateCityDropdownState();

}

class \_CountryStateCityDropdownState extends State<CountryStateCityDropdown> {

  // Data for country, states, and cities

  final Map<String, List<String>> countryStateMap = {

    'USA': ['California', 'Texas', 'Florida'],

    'India': ['Maharashtra', 'Delhi', 'Karnataka'],

    'Canada': ['Ontario', 'Quebec', 'British Columbia'],

  };

  final Map<String, List<String>> stateCityMap = {

    'California': ['Los Angeles', 'San Francisco', 'San Diego'],

    'Texas': ['Houston', 'Dallas', 'Austin'],

    'Florida': ['Miami', 'Orlando', 'Tampa'],

    'Maharashtra': ['Mumbai', 'Pune', 'Nagpur'],

    'Delhi': ['New Delhi', 'Dwarka', 'Noida'],

    'Karnataka': ['Bangalore', 'Mysore', 'Mangalore'],

    'Ontario': ['Toronto', 'Ottawa', 'Hamilton'],

    'Quebec': ['Montreal', 'Quebec City', 'Laval'],

    'British Columbia': ['Vancouver', 'Victoria', 'Kelowna'],

  };

  String? selectedCountry;

  String? selectedState;

  String? selectedCity;

  List<String> states = [];

  List<String> cities = [];

  // Update states when a country is selected

  void updateStates(String country) {

    setState(() {

      selectedCountry = country;

      states = countryStateMap[country] ?? [];

      selectedState = null;

      cities = [];

      selectedCity = null;

    });

  }

  // Update cities when a state is selected

  void updateCities(String state) {

    setState(() {

      selectedState = state;

      cities = stateCityMap[state] ?? [];

      selectedCity = null;

    });

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('Country-State-City Dropdown'),

      ),

      body: Padding(

        padding: const EdgeInsets.all(16.0),

        child: Column(

          crossAxisAlignment: CrossAxisAlignment.start,

          children: [

            // Country Dropdown

            DropdownButton<String>(

              isExpanded: true,

              value: selectedCountry,

              hint: Text('Select Country'),

              items: countryStateMap.keys.map((country) {

                return DropdownMenuItem<String>(

                  value: country,

                  child: Text(country),

                );

              }).toList(),

              onChanged: (value) {

                if (value != null) updateStates(value);

              },

            ),

            SizedBox(height: 16),

            // State Dropdown

            DropdownButton<String>(

              isExpanded: true,

              value: selectedState,

              hint: Text('Select State'),

              items: states.map((state) {

                return DropdownMenuItem<String>(

                  value: state,

                  child: Text(state),

                );

              }).toList(),

              onChanged: (value) {

                if (value != null) updateCities(value);

              },

            ),

            SizedBox(height: 16),

            // City Dropdown

            DropdownButton<String>(

              isExpanded: true,

              value: selectedCity,

              hint: Text('Select City'),

              items: cities.map((city) {

                return DropdownMenuItem<String>(

                  value: city,

                  child: Text(city),

                );

              }).toList(),

              onChanged: (value) {

                setState(() {

                  selectedCity = value;

                });

              },

            ),

          ],

        ),

      ),

    );

  }

}

LAB 17

**A.1 Write a flutter code to list data into the List view.**

import 'package:flutter/material.dart';

void main() {

runApp(const MyApp());

}

class MyApp extends StatelessWidget {

const MyApp({super.key});

@override

Widget build(BuildContext context) {

return MaterialApp(

title: 'ListView Demo',

theme: ThemeData(primarySwatch: Colors.blue),

home: const ListViewExample(),

);

}

}

class ListViewExample extends StatelessWidget {

const ListViewExample({super.key});

// Sample data for the ListView

final List<String> items = const [

'Item 1',

'Item 2',

'Item 3',

'Item 4',

'Item 5',

'Item 6',

'Item 7',

'Item 8',

'Item 9',

'Item 10',

];

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: const Text('ListView Example'),

),

body: ListView.builder(

itemCount: items.length,

itemBuilder: (context, index) {

return Card(

margin: const EdgeInsets.symmetric(horizontal: 10, vertical: 5),

child: ListTile(

leading: const Icon(Icons.label),

title: Text(items[index]),

subtitle: Text('Description for ${items[index]}'),

trailing: const Icon(Icons.arrow\_forward\_ios),

onTap: () {

// Handle item tap event

ScaffoldMessenger.of(context).showSnackBar(

SnackBar(content: Text('${items[index]} tapped')),

);

},

),

);

},

),

);

}

}

**A.2 Write a flutter code to display a list with different colors with different priority.**

import 'package:flutter/material.dart';

class PriorityListApp extends StatelessWidget {

  final List<Map<String, dynamic>> tasks = [

    {'title': 'Buy groceries', 'priority': 'High'},

    {'title': 'Check emails', 'priority': 'Medium'},

    {'title': 'Call mom', 'priority': 'Low'},

    {'title': 'Work out', 'priority': 'Medium'},

    {'title': 'Pay bills', 'priority': 'High'},

  ];

  Color getPriorityColor(String priority) {

    switch (priority) {

      case 'High':

        return Colors.red;

      case 'Medium':

        return Colors.orange;

      case 'Low':

        return Colors.green;

      default:

        return Colors.grey;

    }

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('Priority List'),

      ),

      body: ListView.builder(

        itemCount: tasks.length,

        itemBuilder: (context, index) {

          final task = tasks[index];

          return Card(

            color: getPriorityColor(task['priority']),

            child: ListTile(

              title: Text(task['title']),

              subtitle: Text('Priority: ${task['priority']}'),

            ),

          );

        },

      ),

    );

  }

}

**B.1 Write a flutter code to sort data by options: A->Z & Z-A.**

import 'package:flutter/material.dart';

class SortDataApp extends StatefulWidget {

  @override

  \_SortDataAppState createState() => \_SortDataAppState();

}

class \_SortDataAppState extends State<SortDataApp> {

  List<String> items = ['Banana', 'Apple', 'Orange', 'Mango', 'Grapes'];

  void sortData(bool ascending) {

    setState(() {

      items.sort((a, b) => ascending ? a.compareTo(b) : b.compareTo(a));

    });

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('Sort Data Example'),

        actions: [

          PopupMenuButton<String>(

            onSelected: (value) {

              if (value == 'A→Z') {

                sortData(true);

              } else if (value == 'Z→A') {

                sortData(false);

              }

            },

            itemBuilder: (context) => [

              PopupMenuItem(

                value: 'A→Z',

                child: Text('Sort A→Z'),

              ),

              PopupMenuItem(

                value: 'Z→A',

                child: Text('Sort Z→A'),

              ),

            ],

          ),

        ],

      ),

      body: ListView.builder(

        itemCount: items.length,

        itemBuilder: (context, index) {

          return ListTile(

            title: Text(items[index]),

          );

        },

      ),

    );

  }

}

LAB 18

**A.1 Write a flutter code pass data from list to edit page and edit record in table.**

import 'package:flutter/material.dart';

// Data model class

class Item {

String name;

String description;

Item({required this.name, required this.description});

}

class ListPage extends StatefulWidget {

const ListPage({super.key});

@override

\_ListPageState createState() => \_ListPageState();

}

class \_ListPageState extends State<ListPage> {

// Sample data list

List<Item> items = [

Item(name: 'Item 1', description: 'Description 1'),

Item(name: 'Item 2', description: 'Description 2'),

Item(name: 'Item 3', description: 'Description 3'),

];

// Refresh the list after editing

void \_updateItem(int index, Item updatedItem) {

setState(() {

items[index] = updatedItem;

});

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: const Text('Item List'),

),

body: ListView.builder(

itemCount: items.length,

itemBuilder: (context, index) {

return Card(

child: ListTile(

title: Text(items[index].name),

subtitle: Text(items[index].description),

trailing: const Icon(Icons.edit),

onTap: () async {

// Navigate to EditPage and wait for result

final updatedItem = await Navigator.push(

context,

MaterialPageRoute(

builder: (context) => EditPage(

item: items[index],

),

),

);

if (updatedItem != null) {

\_updateItem(index, updatedItem);

}

},

),

);

},

),

);

}

}

class EditPage extends StatefulWidget {

final Item item;

const EditPage({super.key, required this.item});

@override

\_EditPageState createState() => \_EditPageState();

}

class \_EditPageState extends State<EditPage> {

late TextEditingController \_nameController;

late TextEditingController \_descriptionController;

@override

void initState() {

super.initState();

\_nameController = TextEditingController(text: widget.item.name);

\_descriptionController = TextEditingController(text: widget.item.description);

}

@override

void dispose() {

\_nameController.dispose();

\_descriptionController.dispose();

super.dispose();

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: const Text('Edit Item'),

),

body: Padding(

padding: const EdgeInsets.all(16.0),

child: Column(

children: [

TextField(

controller: \_nameController,

decoration: const InputDecoration(labelText: 'Name'),

),

TextField(

controller: \_descriptionController,

decoration: const InputDecoration(labelText: 'Description'),

),

const SizedBox(height: 20),

ElevatedButton(

onPressed: () {

// Update item and return to previous page

Navigator.pop(

context,

Item(

name: \_nameController.text,

description: \_descriptionController.text,

),

);

},

child: const Text('Save'),

),

],

),

),

);

}

}

**A.2 Write a flutter code to place delete button in list page and on click delete button show alert dialog for confirmation for delete and delete data from table and refresh the page with updated data.**

import 'package:flutter/material.dart';

// Data model class

class Item {

String name;

String description;

Item({required this.name, required this.description});

}

class ListPage extends StatefulWidget {

const ListPage({super.key});

@override

\_ListPageState createState() => \_ListPageState();

}

class \_ListPageState extends State<ListPage> {

// Sample data list

List<Item> items = [

Item(name: 'Item 1', description: 'Description 1'),

Item(name: 'Item 2', description: 'Description 2'),

Item(name: 'Item 3', description: 'Description 3'),

];

// Function to show delete confirmation dialog

Future<void> \_confirmDelete(int index) async {

bool? confirmDelete = await showDialog<bool>(

context: context,

builder: (BuildContext context) {

return AlertDialog(

title: const Text('Confirm Deletion'),

content: Text('Are you sure you want to delete ${items[index].name}?'),

actions: [

TextButton(

onPressed: () {

Navigator.of(context).pop(false); // Dismiss without deletion

},

child: const Text('Cancel'),

),

ElevatedButton(

onPressed: () {

Navigator.of(context).pop(true); // Confirm deletion

},

style: ElevatedButton.styleFrom(backgroundColor: Colors.red),

child: const Text('Delete'),

),

],

);

},

);

if (confirmDelete == true) {

\_deleteItem(index);

}

}

// Function to delete an item from the list

void \_deleteItem(int index) {

setState(() {

items.removeAt(index); // Remove the item from the list

});

ScaffoldMessenger.of(context).showSnackBar(

const SnackBar(content: Text('Item deleted successfully')),

);

}

@override

Widget build(BuildContext context) {

return Scaffold(

appBar: AppBar(

title: const Text('Delete Item Example'),

),

body: ListView.builder(

itemCount: items.length,

itemBuilder: (context, index) {

return Card(

margin: const EdgeInsets.symmetric(horizontal: 10, vertical: 5),

child: ListTile(

title: Text(items[index].name),

subtitle: Text(items[index].description),

trailing: IconButton(

icon: const Icon(Icons.delete, color: Colors.red),

onPressed: () => \_confirmDelete(index),

),

),

);

},

),

);

}

}

LAB 19

**A.1 Write a flutter code to fetch json file from asset folder & parse json data and display data in widget.**

import 'dart:convert';

import 'package:flutter/material.dart';

class FetchJsonApp extends StatefulWidget {

  @override

  \_FetchJsonAppState createState() => \_FetchJsonAppState();

}

class \_FetchJsonAppState extends State<FetchJsonApp> {

  List<dynamic> jsonData = [];

  @override

  void initState() {

    super.initState();

    loadJsonData();

  }

  Future<void> loadJsonData() async {

    // Load the JSON file from assets

    final String response =

        await DefaultAssetBundle.of(context).loadString('assets/data.json');

    final data = json.decode(response);

    print(data);

    setState(() {

      jsonData = data; // Assuming the JSON contains an "items" array

    });

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('Fetch JSON Example'),

      ),

      body: jsonData.isEmpty

          ? Center(child: CircularProgressIndicator())

          : ListView.builder(

              itemCount: jsonData.length,

              itemBuilder: (context, index) {

                final item = jsonData[index];

                return ListTile(

                  title: Text(item['name']),

                  subtitle: Text('Email: ${item['email']}'),

                );

              },

            ),

    );

  }

}

**B.1 Write and flutter code to generate model class from json.**

import 'dart:convert';

import 'package:lebs/lab-19/UserModel.dart';

void main() {

  String jsonString =

      '{"id": 1, "name": "John Doe", "email": "john.doe@example.com"}';

  // Parse JSON string into a Map

  Map<String, dynamic> userMap = jsonDecode(jsonString);

  // Convert Map to User instance

  User user = User.fromJson(userMap);

  print('User Name: ${user.name}');

  print('User Email: ${user.email}');

  // Convert User instance back to JSON

  String jsonOutput = user.toJson(user).toString();

  print('JSON Output: $jsonOutput');

}

LAB 20

**A.1 Write a flutter code to call rest api (using Mockapi) with GET Method with httpclient library and display data in List widget using future builder.**

import 'dart:convert';

import 'package:flutter/material.dart';

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

import 'package:lebs/lab-20/GetByIdMockApi.dart';

class GetAllMockApi extends StatelessWidget {

  Future<List<dynamic>> fetchPosts() async {

    final response =

        await http.get(Uri.parse('https://jsonplaceholder.typicode.com/posts'));

    if (response.statusCode == 200) {

      // Directly parse the JSON response into a List of dynamic objects

      return json.decode(response.body);

    } else {

      throw Exception('Failed to load posts');

    }

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('Posts List'),

      ),

      body: FutureBuilder<List<dynamic>>(

        future: fetchPosts(),

        builder: (context, snapshot) {

          if (snapshot.connectionState == ConnectionState.waiting) {

            return Center(child: CircularProgressIndicator());

          } else if (snapshot.hasError) {

            return Center(child: Text('Error: ${snapshot.error}'));

          } else if (!snapshot.hasData || snapshot.data!.isEmpty) {

            return Center(child: Text('No posts available.'));

          } else {

            var posts = snapshot.data!;

            return ListView.builder(

              itemCount: posts.length,

              itemBuilder: (context, index) {

                return ListTile(

                  title: Text(posts[index]['title']),

                  subtitle: Text(posts[index]['body']),

                  onTap: () {

                    Navigator.push(

                      context,

                      MaterialPageRoute(

                        builder: (context) => PostDetailScreen(

                          postId: posts[index]['id'],

                          extraParam: 'Hello World',

                        ),

                      ),

                    );

                  },

                );

              },

            );

          }

        },

      ),

    );

  }

}

**A.2 Write a flutter code pass Path Variable and Query Variable as parameter in GET Method and display single recod in full screen page.**

import 'dart:convert';

import 'package:flutter/material.dart';

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

class PostDetailScreen extends StatelessWidget {

  final int postId;

  final String? extraParam;

  const PostDetailScreen({Key? key, required this.postId, this.extraParam})

      : super(key: key);

  @override

  Widget build(BuildContext context) {

    Future<Map<String, dynamic>> fetchPost(int id, {String? extraParam}) async {

      final String url = 'https://jsonplaceholder.typicode.com/posts/$id';

      final Uri uri = Uri.parse(url).replace(queryParameters: {

        if (extraParam != null) 'extra\_param': extraParam,

      });

      final response = await http.get(uri);

      if (response.statusCode == 200) {

        return json.decode(response.body);

      } else {

        throw Exception('Failed to load post');

      }

    }

    return Scaffold(

      appBar: AppBar(

        title: Text('Post Detail'),

      ),

      body: FutureBuilder<Map<String, dynamic>>(

        future: fetchPost(postId, extraParam: extraParam),

        builder: (context, snapshot) {

          if (snapshot.connectionState == ConnectionState.waiting) {

            return Center(child: CircularProgressIndicator());

          } else if (snapshot.hasError) {

            return Center(child: Text('Error: ${snapshot.error}'));

          } else if (!snapshot.hasData) {

            return Center(child: Text('No data found.'));

          } else {

            var post = snapshot.data!;

            return Padding(

              padding: const EdgeInsets.all(16.0),

              child: Column(

                crossAxisAlignment: CrossAxisAlignment.start,

                children: [

                  Text(

                    'Title: ${post['title']}',

                    style: TextStyle(fontSize: 24, fontWeight: FontWeight.bold),

                  ),

                  SizedBox(height: 8),

                  Text(

                    'Body: ${post['body']}',

                    style: TextStyle(fontSize: 16),

                  ),

                ],

              ),

            );

          }

        },

      ),

    );

  }

}

**B.1 Write a flutter code to call rest api and display data in listview.builder with pagination (Lazzy Loading).**

import 'dart:convert';

import 'package:flutter/material.dart';

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

class PaginatedListView extends StatefulWidget {

  @override

  \_PaginatedListViewState createState() => \_PaginatedListViewState();

}

class \_PaginatedListViewState extends State<PaginatedListView> {

  List<dynamic> data = [];

  bool isLoading = false;

  int page = 1;

  bool hasMore = true;

  final ScrollController scrollController = ScrollController();

  @override

  void initState() {

    super.initState();

    scrollController.addListener(() {

      if (scrollController.offset >=

              scrollController.position.maxScrollExtent &&

          !scrollController.position.outOfRange) {

        if (!isLoading && hasMore) {

          fetchData(); // Fetch more data when scrolled to the bottom

        }

      }

    });

    fetchData(); // Initial data load

  }

  // Fetch data from the API

  Future<void> fetchData() async {

    if (isLoading || !hasMore) return;

    setState(() {

      isLoading = true;

    });

    final String url =

        'https://jsonplaceholder.typicode.com/posts?\_page=$page&\_limit=10';

    final response = await http.get(Uri.parse(url));

    if (response.statusCode == 200) {

      List<dynamic> fetchedData = json.decode(response.body);

      setState(() {

        if (fetchedData.isEmpty) {

          hasMore = false;

        } else {

          data.addAll(fetchedData);

          page++;

        }

        isLoading = false;

      });

    } else {

      throw Exception('Failed to load data');

    }

  }

  // Build the list of data

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('Paginated ListView'),

      ),

      body: ListView.builder(

        controller: scrollController,

        itemCount: data.length + 1, // Add 1 for the loading indicator

        itemBuilder: (context, index) {

          if (index == data.length) {

            // Display loading indicator at the bottom

            if (isLoading) {

              return Center(child: CircularProgressIndicator());

            } else if (!hasMore) {

              return Center(child: Text('No more data.'));

            }

            return SizedBox.shrink();

          }

          var post = data[index];

          return ListTile(

            title: Text(post['title']),

            subtitle: Text(post['body']),

          );

        },

      ),

    );

  }

}

LAB 21

**A.1 Write a flutter code to display country list in dropdown on selection of country call state api and fill data in state list after selection state call city api and fill city list in dropdown.**

import 'package:flutter/material.dart';

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

import 'dart:convert';

class LocationDropdowns extends StatefulWidget {

  @override

  \_LocationDropdownsState createState() => \_LocationDropdownsState();

}

class \_LocationDropdownsState extends State<LocationDropdowns> {

  // Lists for dropdown data

  List<dynamic> countries = [];

  List<dynamic> states = [];

  List<dynamic> cities = [];

  // Selected values

  String? selectedCountry;

  String? selectedState;

  String? selectedCity;

  @override

  void initState() {

    super.initState();

    fetchCountries(); // Load country data on start

  }

  // Fetch country list

  Future<void> fetchCountries() async {

    final response = await http.get(Uri.parse('https://yourapi.com/countries'));

    if (response.statusCode == 200) {

      setState(() {

        countries = json.decode(response.body);

      });

    } else {

      throw Exception('Failed to load countries');

    }

  }

  // Fetch state list based on selected country

  Future<void> fetchStates(String countryId) async {

    final response = await http.get(Uri.parse('https://yourapi.com/states?country\_id=$countryId'));

    if (response.statusCode == 200) {

      setState(() {

        states = json.decode(response.body);

        cities = []; // Reset cities

        selectedState = null;

        selectedCity = null;

      });

    } else {

      throw Exception('Failed to load states');

    }

  }

  // Fetch city list based on selected state

  Future<void> fetchCities(String stateId) async {

    final response = await http.get(Uri.parse('https://yourapi.com/cities?state\_id=$stateId'));

    if (response.statusCode == 200) {

      setState(() {

        cities = json.decode(response.body);

        selectedCity = null; // Reset city selection

      });

    } else {

      throw Exception('Failed to load cities');

    }

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(title: Text('Location Dropdowns')),

      body: Padding(

        padding: const EdgeInsets.all(20.0),

        child: Column(

          crossAxisAlignment: CrossAxisAlignment.start,

          children: [

            // Country Dropdown

            DropdownButtonFormField<String>(

              value: selectedCountry,

              hint: Text('Select Country'),

              items: countries.map<DropdownMenuItem<String>>((country) {

                return DropdownMenuItem<String>(

                  value: country['id'].toString(),

                  child: Text(country['name']),

                );

              }).toList(),

              onChanged: (value) {

                setState(() {

                  selectedCountry = value;

                });

                if (value != null) fetchStates(value);

              },

              decoration: InputDecoration(border: OutlineInputBorder()),

            ),

            SizedBox(height: 20),

            // State Dropdown

            DropdownButtonFormField<String>(

              value: selectedState,

              hint: Text('Select State'),

              items: states.map<DropdownMenuItem<String>>((state) {

                return DropdownMenuItem<String>(

                  value: state['id'].toString(),

                  child: Text(state['name']),

                );

              }).toList(),

              onChanged: (value) {

                setState(() {

                  selectedState = value;

                });

                if (value != null) fetchCities(value);

              },

              decoration: InputDecoration(border: OutlineInputBorder()),

            ),

            SizedBox(height: 20),

            // City Dropdown

            DropdownButtonFormField<String>(

              value: selectedCity,

              hint: Text('Select City'),

              items: cities.map<DropdownMenuItem<String>>((city) {

                return DropdownMenuItem<String>(

                  value: city['id'].toString(),

                  child: Text(city['name']),

                );

              }).toList(),

              onChanged: (value) {

                setState(() {

                  selectedCity = value;

                });

              },

              decoration: InputDecoration(border: OutlineInputBorder()),

            ),

          ],

        ),

      ),

    );

  }

}

**B.1 Write a flutter code call nested api (prefill 3 dropdown before).**

import 'package:flutter/material.dart';

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

import 'dart:convert';

class NestedAPIDropdowns extends StatefulWidget {

  @override

  \_NestedAPIDropdownsState createState() => \_NestedAPIDropdownsState();

}

class \_NestedAPIDropdownsState extends State<NestedAPIDropdowns> {

  // Lists to store dropdown data

  List<dynamic> countries = [];

  List<dynamic> states = [];

  List<dynamic> cities = [];

  // Selected values

  String? selectedCountry;

  String? selectedState;

  String? selectedCity;

  @override

  void initState() {

    super.initState();

    fetchPrefilledData(); // Fetch prefilled data for all dropdowns

  }

  // Fetch prefilled data for Country, State, and City

  Future<void> fetchPrefilledData() async {

    try {

      final countryResponse = await http.get(Uri.parse('https://yourapi.com/countries'));

      final stateResponse = await http.get(Uri.parse('https://yourapi.com/states?country\_id=1'));

      final cityResponse = await http.get(Uri.parse('https://yourapi.com/cities?state\_id=1'));

      if (countryResponse.statusCode == 200 &&

          stateResponse.statusCode == 200 &&

          cityResponse.statusCode == 200) {

        setState(() {

          countries = json.decode(countryResponse.body);

          states = json.decode(stateResponse.body);

          cities = json.decode(cityResponse.body);

          selectedCountry = countries.isNotEmpty ? countries[0]['id'].toString() : null;

          selectedState = states.isNotEmpty ? states[0]['id'].toString() : null;

          selectedCity = cities.isNotEmpty ? cities[0]['id'].toString() : null;

        });

      } else {

        throw Exception('Failed to load initial data');

      }

    } catch (e) {

      print(e);

    }

  }

  // Fetch states when a country is selected

  Future<void> fetchStates(String countryId) async {

    final response = await http.get(Uri.parse('https://yourapi.com/states?country\_id=$countryId'));

    if (response.statusCode == 200) {

      setState(() {

        states = json.decode(response.body);

        cities = []; // Reset cities

        selectedState = states.isNotEmpty ? states[0]['id'].toString() : null;

        selectedCity = null;

      });

      if (selectedState != null) fetchCities(selectedState!);

    }

  }

  // Fetch cities when a state is selected

  Future<void> fetchCities(String stateId) async {

    final response = await http.get(Uri.parse('https://yourapi.com/cities?state\_id=$stateId'));

    if (response.statusCode == 200) {

      setState(() {

        cities = json.decode(response.body);

        selectedCity = cities.isNotEmpty ? cities[0]['id'].toString() : null;

      });

    }

  }

@override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(title: Text('Nested API Dropdowns')),

      body: Padding(

        padding: const EdgeInsets.all(16.0),

        child: Column(

          children: [

            // Country Dropdown

            DropdownButtonFormField<String>(

              value: selectedCountry,

              hint: Text('Select Country'),

              items: countries.map<DropdownMenuItem<String>>((country) {

                return DropdownMenuItem<String>(

                  value: country['id'].toString(),

                  child: Text(country['name']),

                );

              }).toList(),

              onChanged: (value) {

                setState(() {

                  selectedCountry = value;

                });

                if (value != null) fetchStates(value);

              },

              decoration: InputDecoration(border: OutlineInputBorder()),

            ),

            SizedBox(height: 20),

            // State Dropdown

            DropdownButtonFormField<String>(

              value: selectedState,

              hint: Text('Select State'),

              items: states.map<DropdownMenuItem<String>>((state) {

                return DropdownMenuItem<String>(

                  value: state['id'].toString(),

                  child: Text(state['name']),

                );

              }).toList(),

              onChanged: (value) {

                setState(() {

                  selectedState = value;

                });

                if (value != null) fetchCities(value);

              },

              decoration: InputDecoration(border: OutlineInputBorder()),

            ),

            SizedBox(height: 20),

            // City Dropdown

            DropdownButtonFormField<String>(

              value: selectedCity,

              hint: Text('Select City'),

              items: cities.map<DropdownMenuItem<String>>((city) {

                return DropdownMenuItem<String>(

                  value: city['id'].toString(),

                  child: Text(city['name']),

                );

              }).toList(),

              onChanged: (value) {

                setState(() {

                  selectedCity = value;

                });

              },

              decoration: InputDecoration(border: OutlineInputBorder()),

            ),

          ],

        ),

      ),

    );

  }

}

LAB 22

**A.1 Write a flutter code to design entry screen with Name, DOB, City, Address, etc.. validate it & save data to webserver using POST method with formurlencoded.**

import 'package:flutter/material.dart';

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

import 'dart:convert';

class EntryScreen extends StatefulWidget {

  @override

  \_EntryScreenState createState() => \_EntryScreenState();

}

class \_EntryScreenState extends State<EntryScreen> {

  final \_formKey = GlobalKey<FormState>();

  // Text controllers for form fields

  final TextEditingController nameController = TextEditingController();

  final TextEditingController dobController = TextEditingController();

  final TextEditingController cityController = TextEditingController();

  final TextEditingController addressController = TextEditingController();

  // Submit data to webserver

  Future<void> submitData() async {

    if (!\_formKey.currentState!.validate()) return;

    // Data to send in POST request

    final Map<String, String> data = {

      'name': nameController.text,

      'dob': dobController.text,

      'city': cityController.text,

      'address': addressController.text,

    };

    try {

      final response = await http.post(

        Uri.parse('https://yourapi.com/submit'),

        headers: {'Content-Type': 'application/x-www-form-urlencoded'},

        body: data,

      );

      if (response.statusCode == 200) {

        final responseData = json.decode(response.body);

        showDialog(

          context: context,

          builder: (\_) => AlertDialog(

            title: Text('Success'),

            content: Text(responseData['message']),

          ),

        );

      } else {

        throw Exception('Failed to submit data');

      }

    } catch (e) {

      showDialog(

        context: context,

        builder: (\_) => AlertDialog(

          title: Text('Error'),

          content: Text(e.toString()),

        ),

      );

    }

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(title: Text('Entry Form')),

      body: Padding(

        padding: const EdgeInsets.all(16.0),

        child: Form(

          key: \_formKey,

          child: Column(

            crossAxisAlignment: CrossAxisAlignment.start,

            children: [

              // Name Field

              TextFormField(

                controller: nameController,

                decoration: InputDecoration(labelText: 'Name', border: OutlineInputBorder()),

                validator: (value) {

                  if (value == null || value.isEmpty) return 'Name is required';

                  return null;

                },

              ),

              SizedBox(height: 16),

              // DOB Field

              TextFormField(

                controller: dobController,

                decoration: InputDecoration(

                  labelText: 'Date of Birth (DD/MM/YYYY)',

                  border: OutlineInputBorder(),

                ),

                validator: (value) {

                  if (value == null || value.isEmpty) return 'DOB is required';

                  final dobRegex = RegExp(r'^\d{2}/\d{2}/\d{4}$');

                  if (!dobRegex.hasMatch(value)) return 'Invalid DOB format';

                  return null;

                },

              ),

              SizedBox(height: 16),

              // City Field

              TextFormField(

                controller: cityController,

                decoration: InputDecoration(labelText: 'City', border: OutlineInputBorder()),

                validator: (value) {

                  if (value == null || value.isEmpty) return 'City is required';

                  return null;

                },

              ),

              SizedBox(height: 16),

              // Address Field

              TextFormField(

                controller: addressController,

                decoration: InputDecoration(labelText: 'Address', border: OutlineInputBorder()),

                maxLines: 3,

                validator: (value) {

                  if (value == null || value.isEmpty) return 'Address is required';

                  return null;

                },

              ),

              SizedBox(height: 20),

              // Submit Button

              Center(

                child: ElevatedButton(

                  onPressed: submitData,

                  child: Text('Submit'),

                ),

              ),

            ],

          ),

        ),

      ),

    );

  }

}

**B.1 Write a flutter code to refresh list with updated data without creating new instance of list screen.**

import 'package:flutter/material.dart';

class RefreshListScreen extends StatefulWidget {

  @override

  \_RefreshListScreenState createState() => \_RefreshListScreenState();

}

class \_RefreshListScreenState extends State<RefreshListScreen> {

  // List of items (example data)

  List<String> items = ['Item 1', 'Item 2', 'Item 3', 'Item 4'];

  // Function to simulate data update

  void updateData() {

    setState(() {

      // Update the list with new data (simulated)

      items.add('Item ${items.length + 1}');

    });

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('Refresh List Example'),

        actions: [

          IconButton(

            icon: Icon(Icons.refresh),

            onPressed: updateData,  // Call updateData when refresh button is pressed

          ),

        ],

      ),

      body: Padding(

        padding: const EdgeInsets.all(16.0),

        child: ListView.builder(

          itemCount: items.length,

          itemBuilder: (context, index) {

            return ListTile(

              title: Text(items[index]),  // Display updated list item

            );

          },

        ),

      ),

    );

  }

}

LAB 23

**A.1 Write a flutter code to request camera permission from user capture image using camera and store in external storage.**

///please read text file in this folder before copying the code.

import 'package:flutter/material.dart';

import 'package:permission\_handler/permission\_handler.dart';

import 'package:camera/camera.dart';

import 'package:path\_provider/path\_provider.dart';

import 'dart:io';

class CameraScreen extends StatefulWidget {

  @override

  \_CameraScreenState createState() => \_CameraScreenState();

}

class \_CameraScreenState extends State<CameraScreen> {

  late CameraController \_controller;

  late List<CameraDescription> cameras;

  late CameraDescription camera;

  bool isCameraInitialized = false;

  @override

  void initState() {

    super.initState();

    \_initializeCamera();

  }

  // Initialize the camera

  Future<void> \_initializeCamera() async {

    // Get available cameras

    cameras = await availableCameras();

    camera = cameras.first; // Use the first camera (usually rear camera)

    \_controller = CameraController(camera, ResolutionPreset.high);

    await \_controller.initialize();

    setState(() {

      isCameraInitialized = true;

    });

  }

  // Request camera permission

  Future<bool> \_requestCameraPermission() async {

    final status = await Permission.camera.request();

    return status.isGranted;

  }

  // Capture and store image

  Future<void> \_captureImage() async {

    final hasPermission = await \_requestCameraPermission();

    if (!hasPermission) {

      // If permission is not granted, show a dialog

      showDialog(

        context: context,

        builder: (\_) => AlertDialog(

          title: Text('Permission Denied'),

          content: Text('Camera permission is required to capture an image.'),

          actions: [

            TextButton(

              child: Text('OK'),

              onPressed: () {

                Navigator.pop(context);

              },

            ),

          ],

        ),

      );

      return;

    }

    try {

      final image = await \_controller.takePicture();

      // Get the external directory to store the image

      final directory = await getExternalStorageDirectory();

      final path = '${directory!.path}/captured\_image\_${DateTime.now().millisecondsSinceEpoch}.jpg';

      // Save image to the file system

      final File file = File(path);

      await file.writeAsBytes(await image.readAsBytes());

      // Show a confirmation message

      showDialog(

        context: context,

        builder: (\_) => AlertDialog(

          title: Text('Image Captured'),

          content: Text('The image has been saved at $path'),

          actions: [

            TextButton(

              child: Text('OK'),

              onPressed: () {

                Navigator.pop(context);

              },

            ),

          ],

        ),

      );

    } catch (e) {

      // Handle error during capture

      print('Error capturing image: $e');

      showDialog(

        context: context,

        builder: (\_) => AlertDialog(

          title: Text('Error'),

          content: Text('There was an error capturing the image. Please try again.'),

          actions: [

            TextButton(

              child: Text('OK'),

              onPressed: () {

                Navigator.pop(context);

              },

            ),

          ],

        ),

      );

    }

  }

  @override

  void dispose() {

    \_controller.dispose();

    super.dispose();

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(title: Text('Capture Image')),

      body: Center(

        child: isCameraInitialized

            ? CameraPreview(\_controller)

            : CircularProgressIndicator(),

      ),

      floatingActionButton: FloatingActionButton(

        onPressed: \_captureImage,

        child: Icon(Icons.camera),

      ),

    );

  }

}

**A.2 Write a flutter code to display image from gallery using image widget.**

import 'package:flutter/material.dart';

import 'package:image\_picker/image\_picker.dart';

import 'dart:io';

class GalleryImagePickerScreen extends StatefulWidget {

  @override

  \_GalleryImagePickerScreenState createState() =>

      \_GalleryImagePickerScreenState();

}

class \_GalleryImagePickerScreenState extends State<GalleryImagePickerScreen> {

  File? \_image;

  // Function to pick an image from the gallery

  Future<void> \_pickImage() async {

    final picker = ImagePicker();

    final pickedFile = await picker.pickImage(source: ImageSource.gallery);

    if (pickedFile != null) {

      setState(() {

        \_image = File(pickedFile.path);

      });

    } else {

      print('No image selected.');

    }

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text('Select Image from Gallery'),

      ),

      body: Center(

        child: Column(

          mainAxisAlignment: MainAxisAlignment.center,

          children: <Widget>[

            // Display the selected image or a placeholder

            \_image != null

                ? Image.file(

                    \_image!,

                    height: 250, // Set height for the image

                    width: 250,  // Set width for the image

                    fit: BoxFit.cover, // Set the image fit style

                  )

                : Text('No image selected'),

            SizedBox(height: 20),

            ElevatedButton(

              onPressed: \_pickImage, // Call the pick image function

              child: Text('Pick Image from Gallery'),

            ),

          ],

        ),

      ),

    );

  }

}

LAB 24

**A.1 Develop UI/UX for the application designs.**

**Login Screen**

**Registration Screen**

//Database

import 'dart:convert';

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

class MyDatabase{

Future<bool> loginUser(String email, String password) async {

    final url = Uri.parse('https://yourapi.com/login');

    try {

      final response = await http.post(

        url,

        headers: {'Content-Type': 'application/json'},

        body: json.encode({'email': email, 'password': password}),

      );

      if (response.statusCode == 200) {

        print('Login Successful: ${response.body}');

        return true;

      } else {

        print('Login Failed: ${response.body}');

        return false;

      }

    } catch (e) {

      print('Error: $e');

      return false;

    }

  }

  Future<bool> registerUser(String name, String email, String phone, String password) async {

    final url = Uri.parse('https://yourapi.com/register');

    try {

      final response = await http.post(

        url,

        headers: {'Content-Type': 'application/json'},

        body: json.encode({

          'name': name,

          'email': email,

          'phone': phone,

          'password': password,

        }),

      );

      if (response.statusCode == 201) {

        print('Registration Successful: ${response.body}');

        return true;

      } else {

        print('Registration Failed: ${response.body}');

        return false;

      }

    } catch (e) {

      print('Error: $e');

      return false;

    }

  }

}

//Login Screen

import 'package:flutter/material.dart';

import 'Database.dart';

class LoginScreen extends StatefulWidget {

  @override

  \_LoginScreenState createState() => \_LoginScreenState();

}

class \_LoginScreenState extends State<LoginScreen> {

  final TextEditingController emailController = TextEditingController();

  final TextEditingController passwordController = TextEditingController();

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      body: Padding(

        padding: const EdgeInsets.all(20.0),

        child: Column(

          mainAxisAlignment: MainAxisAlignment.center,

          crossAxisAlignment: CrossAxisAlignment.stretch,

          children: [

            Text(

              'Login',

              style: TextStyle(fontSize: 32, fontWeight: FontWeight.bold),

              textAlign: TextAlign.center,

            ),

            SizedBox(height: 20),

            TextField(

              controller: emailController,

              decoration: InputDecoration(

                labelText: 'Email',

                border: OutlineInputBorder(),

              ),

            ),

            SizedBox(height: 20),

            TextField(

              controller: passwordController,

              obscureText: true,

              decoration: InputDecoration(

                labelText: 'Password',

                border: OutlineInputBorder(),

              ),

            ),

            SizedBox(height: 20),

            ElevatedButton(

              onPressed: () async {

                bool ans = await MyDatabase().loginUser(

                  emailController.text.trim(),

                  passwordController.text.trim(),

                );

                if(ans){

                  //Navigate to next page

                }

                else{

                  //show snackbar or alertbox

                }

              },

              child: Text('Login'),

            ),

            TextButton(

              onPressed: () {

                Navigator.push(

                  context,

                  MaterialPageRoute(builder: (context) => RegistrationScreen()),

                );

              },

              child: Text('Don\'t have an account? Register'),

            ),

          ],

        ),

      ),

    );

  }

}

// Registration Screen

import 'package:flutter/material.dart';

import 'Database.dart';

class RegistrationScreen extends StatefulWidget {

  @override

  \_RegistrationScreenState createState() => \_RegistrationScreenState();

}

class \_RegistrationScreenState extends State<RegistrationScreen> {

  final TextEditingController nameController = TextEditingController();

  final TextEditingController emailController = TextEditingController();

  final TextEditingController phoneController = TextEditingController();

  final TextEditingController passwordController = TextEditingController();

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      body: Padding(

        padding: const EdgeInsets.all(20.0),

        child: SingleChildScrollView(

          child: Column(

            mainAxisAlignment: MainAxisAlignment.center,

            crossAxisAlignment: CrossAxisAlignment.stretch,

            children: [

              Text(

                'Register',

                style: TextStyle(fontSize: 32, fontWeight: FontWeight.bold),

                textAlign: TextAlign.center,

              ),

              SizedBox(height: 20),

              TextField(

                controller: nameController,

                decoration: InputDecoration(

                  labelText: 'Name',

                  border: OutlineInputBorder(),

                ),

              ),

              SizedBox(height: 20),

              TextField(

                controller: emailController,

                decoration: InputDecoration(

                  labelText: 'Email',

                  border: OutlineInputBorder(),

                ),

              ),

              SizedBox(height: 20),

              TextField(

                controller: phoneController,

                decoration: InputDecoration(

                  labelText: 'Phone',

                  border: OutlineInputBorder(),

                ),

              ),

              SizedBox(height: 20),

              TextField(

                controller: passwordController,

                obscureText: true,

                decoration: InputDecoration(

                  labelText: 'Password',

                  border: OutlineInputBorder(),

                ),

              ),

              SizedBox(height: 20),

              ElevatedButton(

                onPressed: () async {

                  bool ans = await MyDatabase().registerUser(

                    nameController.text.trim(),

                    emailController.text.trim(),

                    phoneController.text.trim(),

                    passwordController.text.trim(),

                  );

                },

                child: Text('Register'),

              ),

              TextButton(

                onPressed: () {

                  Navigator.push(

                    context,

                    MaterialPageRoute(builder: (context) => LoginScreen()),

                  );

                },

                child: Text('Already have an account?'),

              ),

            ],

          ),

        ),

      ),

    );

  }

}

LAB 25

**A.1 Develop UI/UX for the application designs.**

**Listing Screen**

**Add/Edit Screen**

//Database

import 'dart:convert';

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

class MyDatabase{

String baseUrl = 'https://yourapi.com'; // Replace with your API URL

// Fetch items

Future<List<Map<String, dynamic>>> fetchItems() async {

  final response = await http.get(Uri.parse('$baseUrl/items'));

  if (response.statusCode == 200) {

    return List<Map<String, dynamic>>.from(json.decode(response.body));

  } else {

    throw Exception('Failed to load items');

  }

}

// Add item

Future<void> addItem(String title, String description) async {

  final response = await http.post(

    Uri.parse('$baseUrl/items'),

    headers: {'Content-Type': 'application/json'},

    body: json.encode({'title': title, 'description': description}),

  );

  if (response.statusCode != 201) {

    throw Exception('Failed to add item');

  }

}

// Update item

Future<void> updateItem(int id, String title, String description) async {

  final response = await http.put(

    Uri.parse('$baseUrl/items/$id'),

    headers: {'Content-Type': 'application/json'},

    body: json.encode({'title': title, 'description': description}),

  );

  if (response.statusCode != 200) {

    throw Exception('Failed to update item');

  }

}

// Delete item

Future<void> deleteItem(int id) async {

  final response = await http.delete(Uri.parse('$baseUrl/items/$id'));

  if (response.statusCode != 200) {

    throw Exception('Failed to delete item');

  }

}

}

//Listing Screen

import 'package:flutter/material.dart';

import './Database.dart';

class ListingScreen extends StatefulWidget {

  @override

  \_ListingScreenState createState() => \_ListingScreenState();

}

class \_ListingScreenState extends State<ListingScreen> {

  late Future<List<Map<String, dynamic>>> items;

  @override

  void initState() {

    super.initState();

    items = MyDatabase().fetchItems();

  }

  void refreshList() {

    setState(() {

      items = MyDatabase().fetchItems();

    });

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(title: Text('Items')),

      body: FutureBuilder<List<Map<String, dynamic>>>(

        future: items,

        builder: (context, snapshot) {

          if (snapshot.connectionState == ConnectionState.waiting) {

            return Center(child: CircularProgressIndicator());

          } else if (snapshot.hasError) {

            return Center(child: Text('Error: ${snapshot.error}'));

          } else if (!snapshot.hasData || snapshot.data!.isEmpty) {

            return Center(child: Text('No items found.'));

          } else {

            return ListView.builder(

              itemCount: snapshot.data!.length,

              itemBuilder: (context, index) {

                final item = snapshot.data![index];

                return ListTile(

                  title: Text(item['title']),

                  subtitle: Text(item['description']),

                  trailing: Row(

                    mainAxisSize: MainAxisSize.min,

                    children: [

                      IconButton(

                        icon: Icon(Icons.edit),

                        onPressed: () {

                          Navigator.push(

                            context,

                            MaterialPageRoute(

                              builder: (context) => AddEditScreen(

                                item: item,

                                onSaved: refreshList,

                              ),

                            ),

                          );

                        },

                      ),

                      IconButton(

                        icon: Icon(Icons.delete),

                        onPressed: () async {

                          await MyDatabase().deleteItem(item['id']);

                          refreshList();

                        },

                      ),

                    ],

                  ),

                );

              },

            );

          }

        },

      ),

      floatingActionButton: FloatingActionButton(

        onPressed: () {

          Navigator.push(

            context,

            MaterialPageRoute(

              builder: (context) => AddEditScreen(onSaved: refreshList),

            ),

          );

        },

        child: Icon(Icons.add),

      ),

    );

  }

}

//Add/Edit Screen

import 'package:flutter/material.dart';

import 'Database.dart';

class AddEditScreen extends StatefulWidget {

  final Map<String, dynamic>? item;

  final VoidCallback onSaved;

  const AddEditScreen({this.item, required this.onSaved});

  @override

  \_AddEditScreenState createState() => \_AddEditScreenState();

}

class \_AddEditScreenState extends State<AddEditScreen> {

  final TextEditingController titleController = TextEditingController();

  final TextEditingController descriptionController = TextEditingController();

  bool isLoading = false;

  @override

  void initState() {

    super.initState();

    if (widget.item != null) {

      titleController.text = widget.item!['title'];

      descriptionController.text = widget.item!['description'];

    }

  }

  Future<void> saveItem() async {

    setState(() {

      isLoading = true;

    });

    try {

      if (widget.item == null) {

        await MyDatabase().addItem(titleController.text.trim(), descriptionController.text.trim());

      } else {

        await MyDatabase().updateItem(

          widget.item!['id'],

          titleController.text.trim(),

          descriptionController.text.trim(),

        );

      }

      widget.onSaved();

      Navigator.pop(context);

    } catch (e) {

      ScaffoldMessenger.of(context).showSnackBar(SnackBar(content: Text('Error: $e')));

    } finally {

      setState(() {

        isLoading = false;

      });

    }

  }

  @override

  Widget build(BuildContext context) {

    return Scaffold(

      appBar: AppBar(

        title: Text(widget.item == null ? 'Add Item' : 'Edit Item'),

      ),

      body: Padding(

        padding: const EdgeInsets.all(20.0),

        child: Column(

          children: [

            TextField(

              controller: titleController,

              decoration: InputDecoration(labelText: 'Title'),

            ),

            SizedBox(height: 20),

            TextField(

              controller: descriptionController,

              decoration: InputDecoration(labelText: 'Description'),

            ),

            SizedBox(height: 20),

            ElevatedButton(

              onPressed: isLoading ? null : saveItem,

              child: isLoading

                  ? CircularProgressIndicator(color: Colors.white)

                  : Text(widget.item == null ? 'Add' : 'Update'),

            ),

          ],

        ),

      ),

    );

  }

}

LAB 26

**A.1 Create database to do backend programming.**

* **Database Schema**
* **Tables**
* **Stored Procedure**
* //install npm and create backend with node's express framework
* //npm install express mysql body-parser
* const express = require('express');
* const mysql = require('mysql');
* const bodyParser = require('body-parser');
* const app = express();
* app.use(bodyParser.json());
* const db = mysql.createConnection({
* host: 'localhost',
* user: 'root',
* password: 'yourpassword',
* database: 'yourdatabase'
* });
* // Fetch all items
* app.get('/items', (req, res) => {
* db.query('CALL GetAllItems()', (err, results) => {
* if (err) throw err;
* res.json(results[0]);
* });
* });
* // Add item
* app.post('/items', (req, res) => {
* const { title, description } = req.body;
* db.query('CALL AddItem(?, ?)', [title, description], (err, results) => {
* if (err) throw err;
* res.status(201).json({ message: 'Item added successfully' });
* });
* });
* // Update item
* app.put('/items/:id', (req, res) => {
* const { id } = req.params;
* const { title, description } = req.body;
* db.query('CALL UpdateItem(?, ?, ?)', [id, title, description], (err, results) => {
* if (err) throw err;
* res.json({ message: 'Item updated successfully' });
* });
* });
* // Delete item
* app.delete('/items/:id', (req, res) => {
* const { id } = req.params;
* db.query('CALL DeleteItem(?)', [id], (err, results) => {
* if (err) throw err;
* res.json({ message: 'Item deleted successfully' });
* });
* });
* // Start server
* app.listen(3000, () => {
* console.log('Server running on http://localhost:3000');
* });

//SQL

-- Database Schema

-- Table: Items

-- sql

CREATE TABLE Items (

id INT AUTO\_INCREMENT PRIMARY KEY,

title VARCHAR(255) NOT NULL,

description TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP ON UPDATE CURRENT\_TIMESTAMP

);

-- Sample Data for Items Table

-- sql

INSERT INTO Items (title, description)

VALUES

('Sample Item 1', 'This is a sample description for item 1'),

('Sample Item 2', 'This is a sample description for item 2');

-- Stored Procedures

-- 1. Procedure to Fetch All Items

-- sql

CREATE PROCEDURE GetAllItems()

BEGIN

SELECT \* FROM Items ORDER BY created\_at DESC;

END

-- 2. Procedure to Add an Item

-- sql

CREATE PROCEDURE AddItem(IN itemTitle VARCHAR(255), IN itemDescription TEXT)

BEGIN

INSERT INTO Items (title, description) VALUES (itemTitle, itemDescription);

END

-- 3. Procedure to Update an Item

-- sql

CREATE PROCEDURE UpdateItem(IN itemId INT, IN itemTitle VARCHAR(255), IN itemDescription TEXT)

BEGIN

UPDATE Items

SET title = itemTitle, description = itemDescription, updated\_at = CURRENT\_TIMESTAMP

WHERE id = itemId;

END

-- 4. Procedure to Delete an Item

-- sql

CREATE PROCEDURE DeleteItem(IN itemId INT)

BEGIN

DELETE FROM Items WHERE id = itemId;

END