GUJARAT TECHNOLOGICAL UNIVERSITY

Syllabus for Integrated MCA 9th Semester
Branch: MCA Integrated
Subject Name: Flutter App Development
Subject Code: 2698606

With effective from academic year 2022-23

Teaching Scheme			Credits	Examination Marks				
				Theory		Tutorial/ Practical		Total Marks
L	P	OJT	C	University exams (ESE)	Progressive Assessment (PA)	External Practical /viva Exam(ESE)	Internal evaluation Practical /viva Exam(PA)	
4	2		6	70	30	30	20	150

Pre-requisites:
1. Programming Skill.
2. Basic Understanding of Mobile Applications.

Learning Objectives:
1. Understand the lifecycle of mobile app development.
2. Understand the Widget tree and learn to use pre-made Flutter Widgets for user interface design.
3. Learn to incorporate Image and Text Widgets to create simple user interfaces.
4. Build cross-platform mobile app swiftly.
5. Create full-fledged mobile app and deploy.

Course Outcome (COs):
CO1: Demonstrate the basic primitives in Flutter and Dart framework.
CO2: Model native platform code using Flutter and Dart.
CO3: Examine the use of widgets and user interactions in application development.
CO4: Evaluate application development using the concepts of animation and interactive widgets.
CO5: Construct flutter and dart applications.

Course Content:

Unit No.	Content	Hrs
1	**Introduction to Flutter:** Flutter Framework, Working of Flutter and Dart, installation of Flutter SDK, Using Hot Reload and themes, External packages import, project templates, organizing files and folders.	6
2	**Introduction to Dart:** Declaring and referencing variables, using operators, flow control statements, using functions and classes, implementing asynchronous programming.	6
3	**Flutter Widgets:** Stateless and stateful Widgets, Widget Tree, using common widgets: Scaffold, AppBar, SafeArea, Container, Text, Buttons, column, row, Using images and icons, decorators, form widgets, checking orientation **User Inputs:** Collecting text Inputs, setting text limits, Writing platform native code.	10
4	**Adding Animation to App using Widget:** Using AnimatedContainer, AnimatedCrossfade, AnimatedOpacity, AnimationController, using staggered Animations, using CurveTween. **App's Navigation:** Using Navigator, named navigator route, Hero widget, BottomNavigationBar BottomMapBar, TabBar, TabBarView, Drawer. **Applying Interactivity:** GestureDetector, Dragable and DragTarget Widget, InkWell and InkResponse gestures Dismissible widget.	10
5	**Building layouts:** Scrolling Lists: Card widget, Using ListView and ListTitle, GridView, Stack widget, Customising CustomScrollView using Slivers **Service interaction:** Working	12

	with Future object, working with streams, building widgets based on stream and futures, handling various types of data: JSON, XML, HTML, Sending HTTP request **Saving data with local persistence:** Reading and Writing files, working with key-value pairs, Supporting multiple locales, Introduction to Firebase and cloud Firestore.	
	Total Hours:	**44**

Suggested Specification table with Marks (Theory):

Distribution of Theory Marks				
R Level	U Level	A Level	N Level	E Level
10	40	35	15	0

Legends: R: Remembrance; U: Understanding; A: Application, N: Analyze and E: Evaluate and above Levels (Bloom's Taxonomy)

Text Books:

1. Marco L. Napoli, "Beginning Flutter: A Hands on Guide to App Development", John Wiley & Sons, 1st Edition, 2020.
2. D. Kopec, "Dart for Absolute Beginners", Apress, 1st Edition, 2014.

Reference Books:

1. Eric Windmill, "Flutter in Action", Manning Publications, 1st Edition, 2019.
2. Fu Cheng, "Flutter Recipes", Apress, 1st Edition, 2019.
3. Simone Alessandria, "Flutter Projects: A Practical, Project-based Guide to Building Real-world Cross-platform Mobile Applications and Games", Packt Publishing Limited, 1st Edition, 2020.
4. Rap Payne, "Beginning App development with Flutter", Apress, 1st Edition, 2019.
5. Frank Zammetti, "Practical Flutter: Improve your Mobile Development with Google's Latest Open-Source", Apress, 1st Edition, 2019.

Unit wise Coverage from Main Text Book(s):

Unit No.	**Topics/Subtopics**
I	Chapter 1 to Chapter 4 (From Text Book 1)
II	Chapters 2-5, Chapter 8 and Chapter 10-12 (From Text Book 2)
III	Chapter 5, 6 and 12 (From Text Book 1) and Chapter 6 and 10 (From Text Book 2)
IV	Chapter 7 - 9 and 11 (From Text Book 1)
V	Chapters 10, 13, 14 (From Text Book 1) and Chapters 9, 10, 12 (From Text Book 2)

Suggested List of Practicals as follows (Dart and Flutter):

Dart - exercises

1. Create a program that asks the user to enter their name and their age. Print out a message that tells how many years they have to be 100 years old.

2. Ask the user for a number. Depending on whether the number is even or odd, print out an appropriate message to the user.

3. Take a list, say for example this one: a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
and write a program that prints out all the elements of the list that are less than 5.

4. Create a program that asks the user for a number and then prints out a list of all the divisors of that number.
Tip:if you don't know what a divisor is, it is a number that divides evenly into another number. For example, 13 is a divisor of 26 because 26 / 13 has no remainder.

5. Take two lists, for example:
a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
b = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]
and write a program that returns a list that contains only the elements that are common between them (without duplicates). Make sure your program works on two lists of different sizes.

6. Ask the user for a string and print out whether this string is a palindrome or not.
Tip: A palindrome is a string that reads the same forwards and backwards.

7. Let's say you are given a list saved in a variable:
a = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100].
Write a Dart code that takes this list and makes a new list that has only the even elements of this list in it.

8. Make a two-player Rock-Paper-Scissors game against computer.
Tip: Ask for player's input, compare them, print out a message to the winner.

9. Generate a random number between 1 and 100. Ask the user to guess the number, then tell them whether they guessed too low, too high, or exactly right.
Tip: Keep track of how many guesses the user has taken, and when the game ends, print this out.

10. Write a program that asks the user how many Fibonnaci numbers to generate and then generates them. Take this opportunity to think about how you can use functions.
Tip: Make sure to ask the user to enter the number of numbers in the sequence to generate.

11. Write a program (function) that takes a list and returns a new list that contains all the elements of the first list minus all the duplicates.

12. Write a program (using functions!) that asks the user for a long string containing multiple words. Print back to the user the same string, except with the words in backwards order.
For example, say I type the string:
My name is Michele
Then I would see the string:
Michele is name My

13. Write a password generator in Dart. Be creative with how you generate passwords - strong passwords have a mix of lowercase letters, uppercase letters, numbers, and symbols. The passwords should be random, generating a new password every time the user asks for a new password. Include your run-time code in a main method.

14. Time for some fake graphics! Let's say we want to draw game boards that look like this:

```
 --- --- ---
|  |  |  |
 --- --- ---
|  |  |  |
 --- --- ---
|  |  |  |
 --- --- ---
```

This one is 3x3 (like in tic tac toe).
Ask the user what size game board they want to draw, and draw it for them to the screen using Dart's print statement.

15. Implement a function that takes as input three variables, and returns the largest of the three. Do this without using the Dart max() function!

The goal of this exercise is to think about some internals that Dart normally takes care of for us. All you need is some variables and if statements!

16. For this exercise, we will keep track of when our friend's birthdays are, and be able to find that information based on their name.

Create a dictionary (in your file) of names and birthdays. When you run your program it should ask the user to enter a name, and return the birthday of that person back to them. The interaction should look something like this:

```
>>> Welcome to the birthday dictionary. We know the birthdays of:
Albert Einstein
Benjamin Franklin
Ada Lovelace
>>> Who's birthday do you want to look up?
Benjamin Franklin
>>> Benjamin Franklin's birthday is 01/17/1706.
```

Flutter - exercises

1. Write a program in Flutter to demostrate stateful and stateless widget.
2. Write a program to demostrate Data Model and HTTP.
3. Write a program to create custom widget.
4. Write a program to demostrate ListView and builder pattern with images.
5. Write a program to make the app a little prettier by adding a gradient background.
6. Write a program to builds pages using route builders.
7. Write a program to add a couple built in Flutter classes that keep track of form input, and a function that returns the data to the main page through the router.
8. Write a program to demostrate using sliders and buttons for rating.
9. Write a program to demostrate on AnimatedCrossFade to load widget on state change.
10. Write a program to demostrate on Tween and AnimationController.
11. Write a program to do classic Todo app and it also has 'due date' and 'repeating' task features using bloc features.
12. Create a shopping cart app using concept of flutters.