

Session 4:

The image shows two screenshots of a web browser. The top screenshot is a Google search for 'dartpad.dartlang'. The search results show 'dartpad.dev' as the top result, with a description: 'DartPad. code New Pad refresh Reset format_align_left Format get_app Install SDK. ancient-fashion-0893. Samples expand_more. more_vert. Dart HTML CSS.' Below this, there is a section 'People also ask' with the question 'What is DartPad?'. The bottom screenshot shows the DartPad web application interface. The header includes the DartPad logo, navigation links (New Pad, Reset, Format, Install SDK), the username 'bold-sunset-5209', and a 'Samples' dropdown. The main area is split into two panels: the left panel contains a code editor with the following Dart code:

```
void main() {  
  for(int i=0;i<5;i++) {  
    print ('hello ${i+1}');  
  }  
}
```

The right panel is divided into 'Console' and 'Documentation' sections, both of which are currently empty. At the bottom of the interface, there is a footer with 'Privacy notice', 'Send feedback', 'no issues', and 'Based on Dart SDK 2.8.4'.

Browser window showing DartPad interface. The URL is `dartpad.dev`. The code editor contains the following Dart code:

```
main() {  
  print("hii vishesh");  
}
```

The **RUN** button is visible. The console output shows:

```
hii vishesh
```

The bottom status bar indicates "no issues Based on Dart SDK 2.8.4".

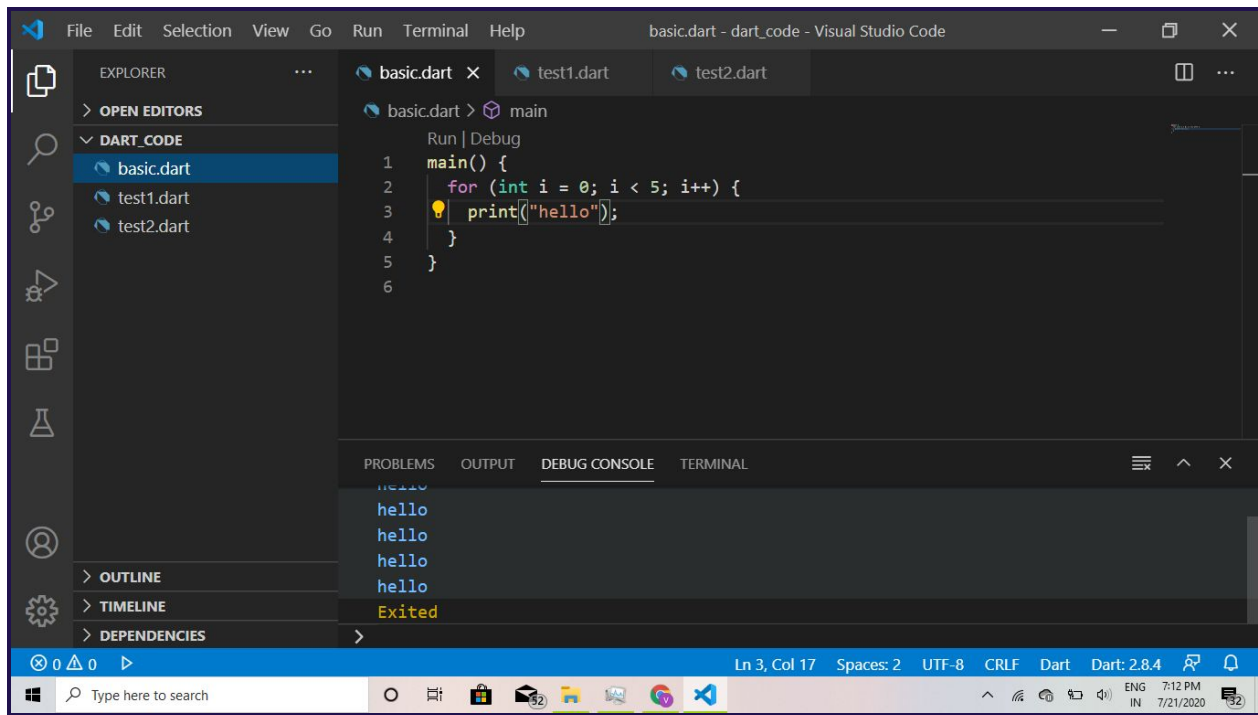
Visual Studio Code window showing the Dart code editor. The Explorer sidebar shows the file structure:

- EXPLORER
 - OPEN EDITORS
 - DART_CODE
 - basic.dart
 - test1.dart
 - test2.dart

The code editor displays the following Dart code:

```
1 main() {  
2   for (int i = 0; i < 5; i++) {  
3     print("hii vishesh");  
4   }  
5 }  
6
```

The bottom status bar shows "Ln 4, Col 4 Spaces: 2 UTF-8 CRLF Dart Dart: 2.8.4".



in programming world box/bucket is known as variables

```
main() {  
  String x;  
  x = "hello";  
  for (int i = 0; i < 5; i++) {  
    print(x);  
  }  
}
```

The screenshot shows the Visual Studio Code interface with a Dart file named `basic.dart` open. The code defines a `main` function that declares a `String` variable `x`, assigns it the value `"hello"`, and prints it five times in a loop. The `DEBUG CONSOLE` at the bottom shows the output: `hello` printed five times, followed by `Exited`. The status bar at the bottom indicates the file is at line 7, column 2, and the Dart SDK version is 2.8.4.

```
1 main() {  
2   String x;  
3   x = "hello";  
4   for (int i = 0; i < 5; i++) {  
5     print(x);  
6   }  
7 }  
8
```

DEBUG CONSOLE
hello
hello
hello
hello
hello
Exited

The screenshot shows the same Visual Studio Code interface, but with a different code snippet in `basic.dart`. The variable `x` is now assigned the integer value `7`. The `PROBLEMS` panel at the bottom shows a red error message: `basic.dart:3:7: Error: A value of type 'int' can't be assigned to a variable of type 'String'.` The status bar at the bottom shows the file is at line 3, column 8, and the Dart SDK version is 2.8.4.

```
1 main() {  
2   String x;  
3   x = 7;  
4   for (int i = 0; i < 5; i++) {  
5     print(x);  
6   }  
7 }  
8
```

PROBLEMS 1
basic.dart:3:7: Error: A value of type 'int' can't be assigned to a variable of type 'String'.
x = 7;
^
Exited (254)

data types:

int(number) , float(decimal or real number) , string(eg. 'hii') etc.....

The screenshot shows the Visual Studio Code interface with a Dart file named `basic.dart` open. The code defines a `main` function that prints the string `"x"` five times in a loop. The `DEBUG CONSOLE` at the bottom shows the output of the program, which is five `x` characters followed by `Exited`.

```
1 main() {  
2   String x;  
3   x = "pop";  
4   for (int i = 0; i < 5; i++) {  
5     print("x");  
6   }  
7 }  
8
```

DEBUG CONSOLE

```
x  
x  
x  
x  
x  
Exited
```

Ln 5, Col 14 Spaces: 2 UTF-8 CRLF Dart Dart: 2.8.4

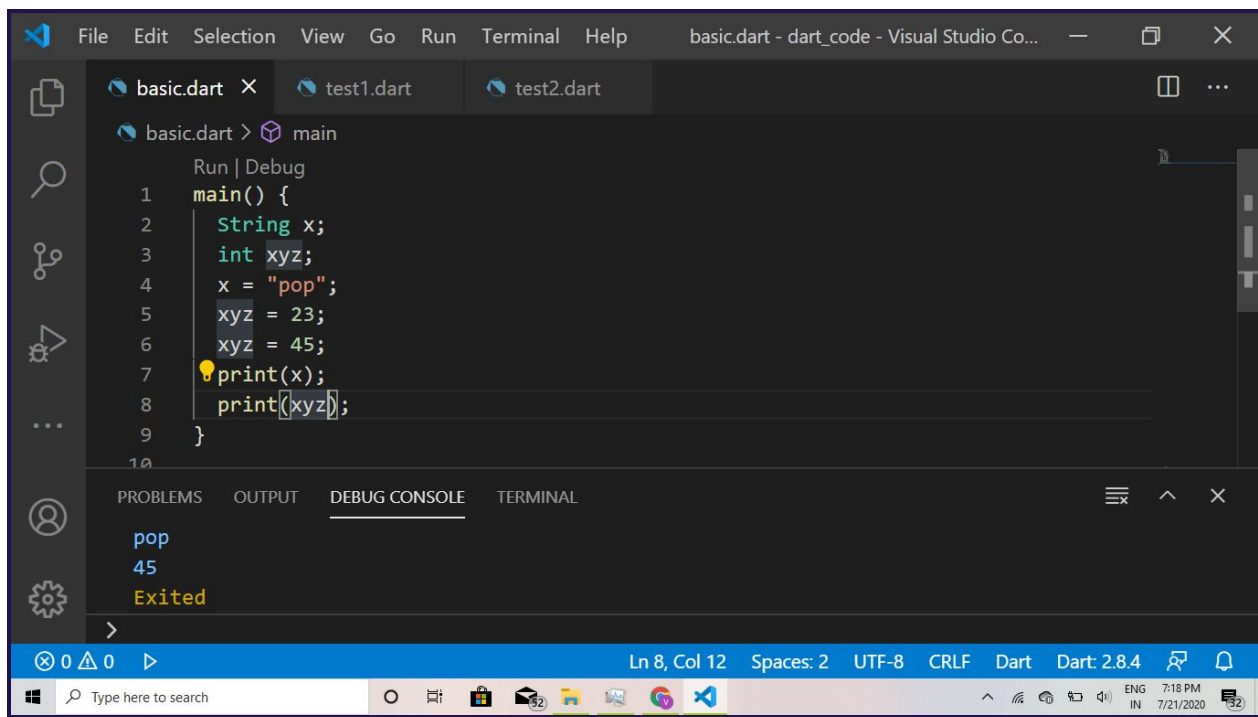
The screenshot shows the Visual Studio Code interface with a Dart file named `basic.dart` open. The code defines a `main` function that declares two `String` variables, `x` and `xyz`. `x` is assigned the value `"pop"`, and `xyz` is assigned `null`. The program prints the values of `x` and `xyz`. The `DEBUG CONSOLE` at the bottom shows the output, which is `pop` followed by `null` and then `Exited`.

```
1 main() {  
2   String x, xyz;  
3   x = "pop";  
4   print(x);  
5   print(xyz);  
6 }  
7
```

DEBUG CONSOLE

```
pop  
null  
Exited
```

Ln 5, Col 14 Spaces: 2 UTF-8 CRLF Dart Dart: 2.8.4



u can't store a string in int data type

The screenshot shows the Visual Studio Code interface with a Dart file named `basic.dart` open. The code defines a `main` function that declares a variable `x` with the value 23, a variable `xyz` with the value "lavya", and prints both. The `DEBUG CONSOLE` panel at the bottom shows the output: `23`, `lavya`, and `Exited`. The status bar at the bottom indicates the cursor is at line 3, column 21.

```
basic.dart > main
Run | Debug
1  main() {
2      var x = 23;
3      var xyz = "lavya";
4      print(x);
5      print(xyz);
6  }
7
```

23
lavya
Exited

Ln 3, Col 21 Spaces: 2 UTF-8 CRLF Dart Dart: 2.8.4

The screenshot shows the same Visual Studio Code interface, but the code in `basic.dart` has been modified. The variable `x` is declared twice: first with the value 23 on line 2, and then with the value "lavya" on line 3. The `DEBUG CONSOLE` panel now shows a red error message: `basic.dart:3:7: Error: 'x' is already declared in this scope.` with a caret pointing to the second declaration. Below the error, it shows the context: `basic.dart:2:7: Context: Previous declaration of 'x'.` with a caret pointing to the first declaration. The status bar at the bottom indicates the cursor is at line 4, column 12.

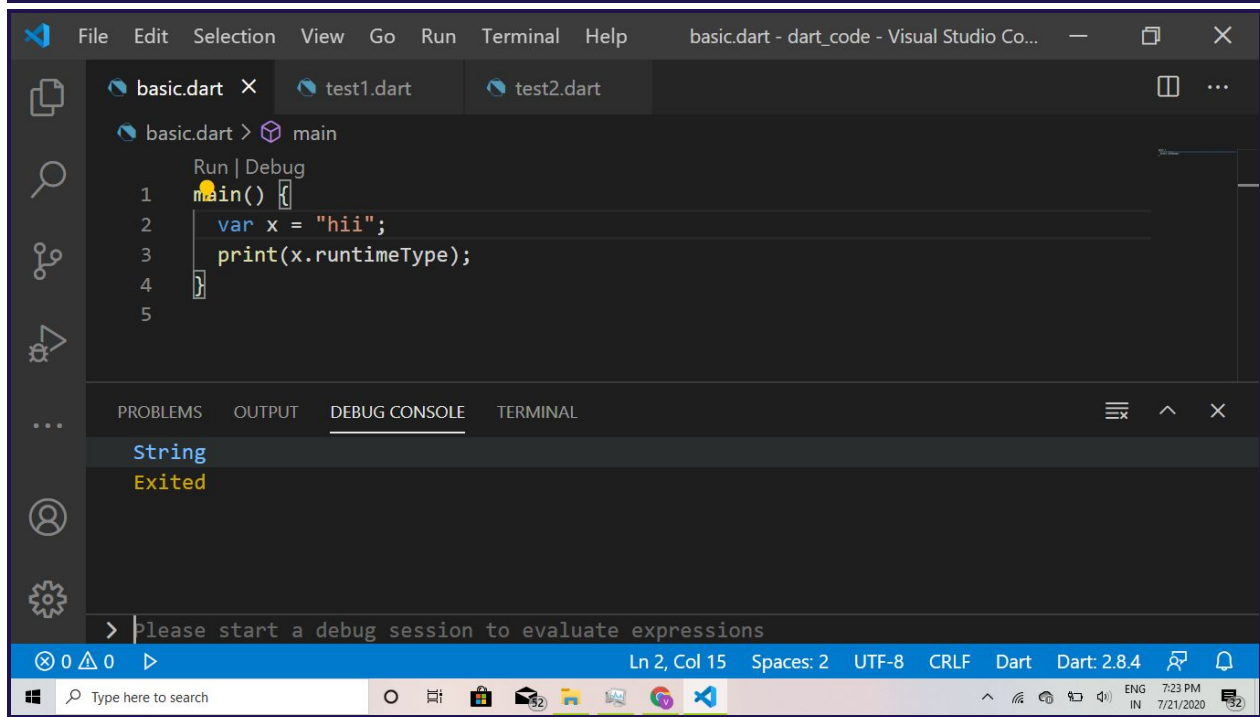
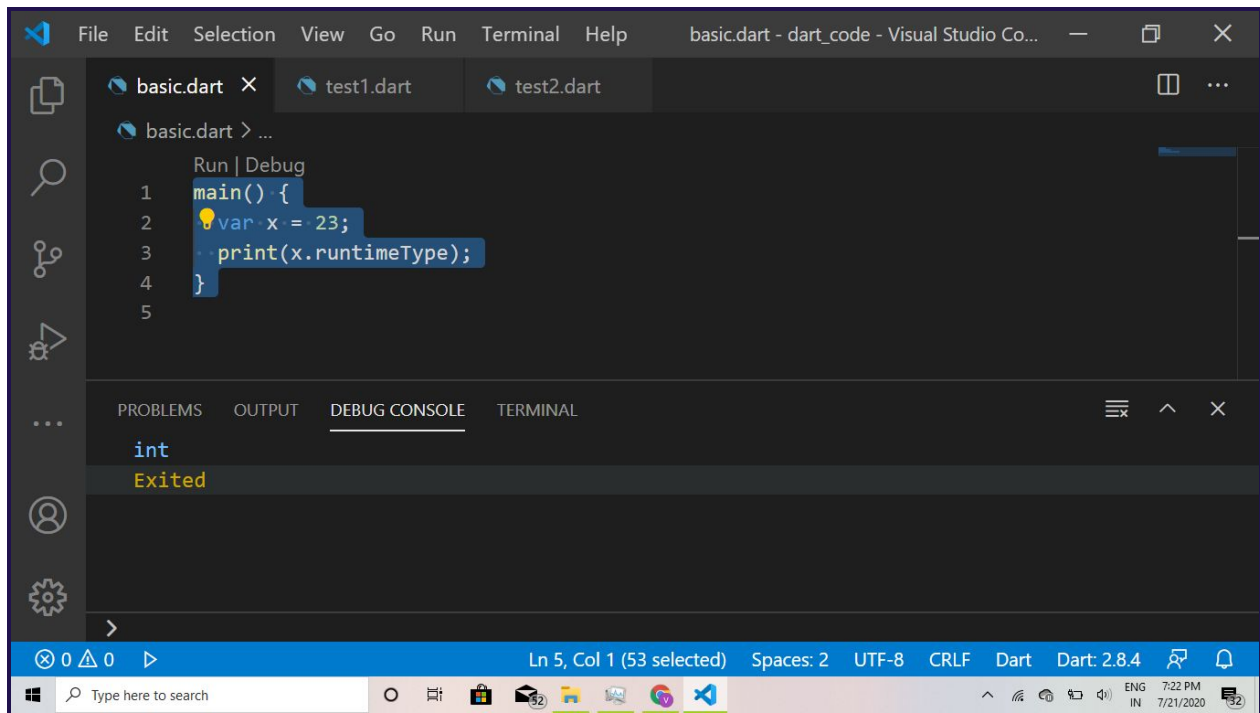
```
basic.dart > main
Run | Debug
1  main() {
2      var x = 23;
3      var x = "lavya";
4      print(x);
5  }
6
```

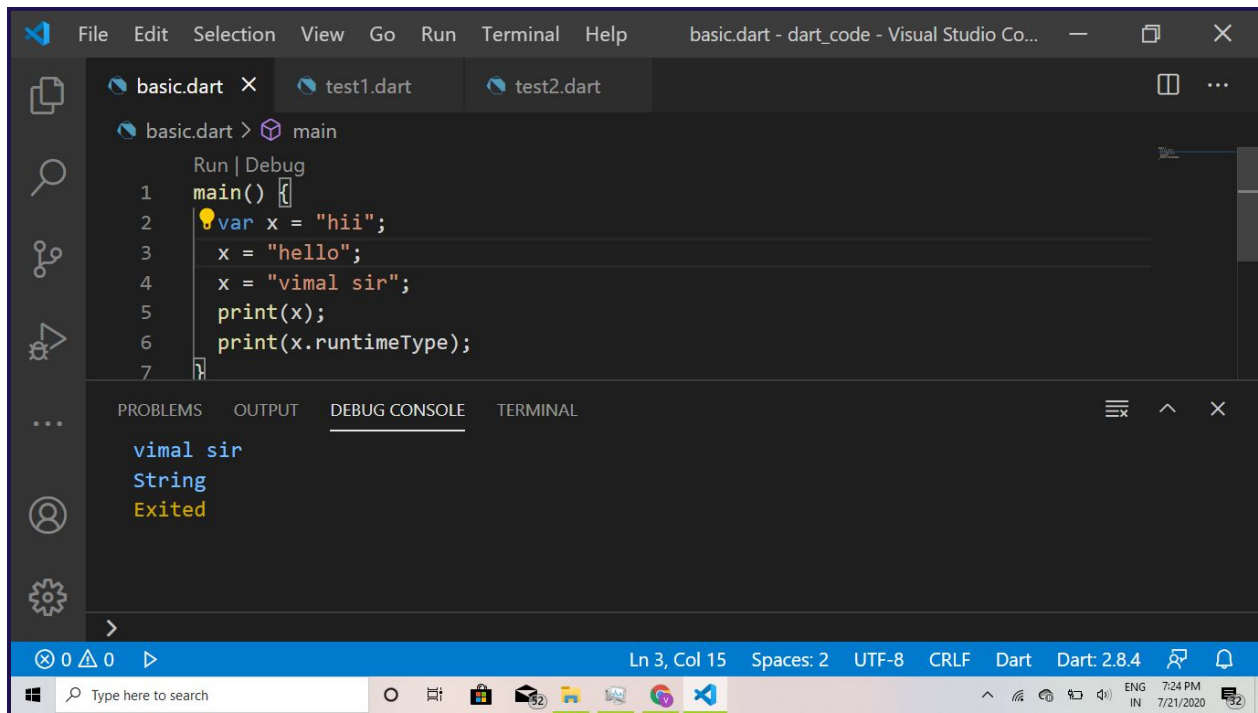
basic.dart:3:7: Error: 'x' is already declared in this scope.
var x = "lavya";
^

basic.dart:2:7: Context: Previous declaration of 'x'.
var x = 23;
^

Exited (254)

Ln 4, Col 12 Spaces: 2 UTF-8 CRLF Dart Dart: 2.8.4



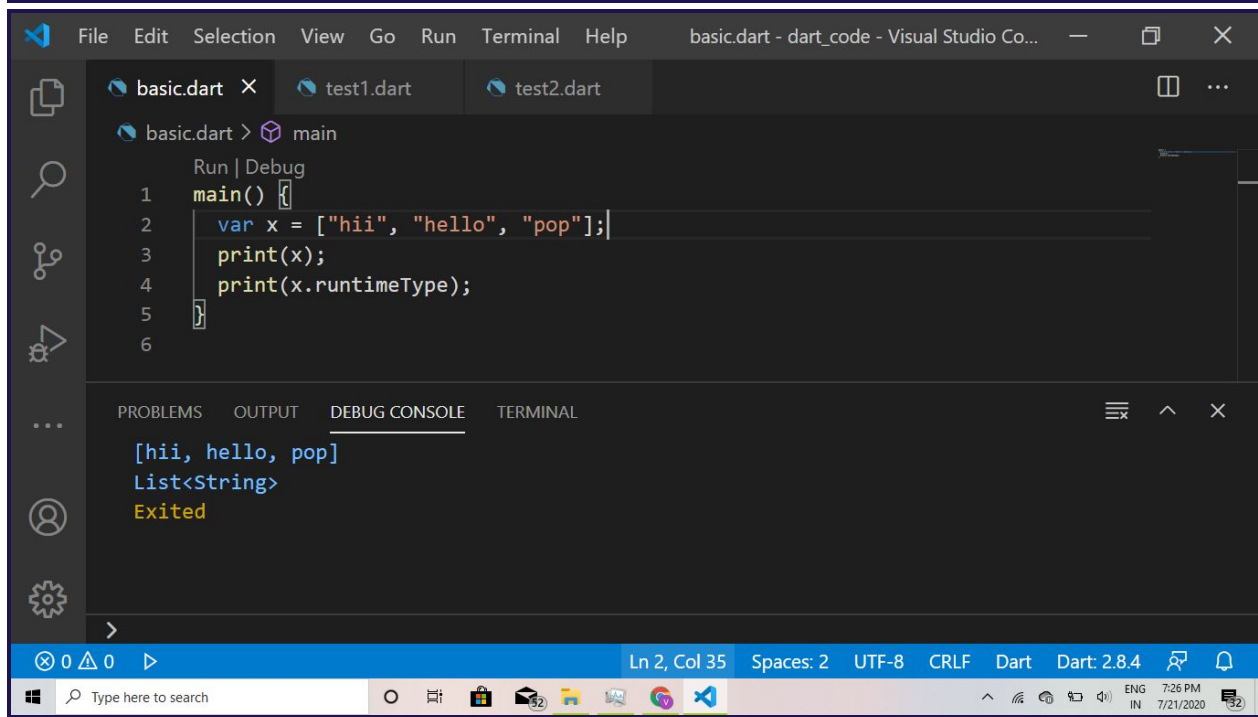


```
basic.dart > main
Run | Debug
1  main() {
2    var x = "hii";
3    x = "hello";
4    x = "vimal sir";
5    print(x);
6    print(x.runtimeType);
7  }
```

PROBLEMS OUTPUT **DEBUG CONSOLE** TERMINAL

vimal sir
String
Exited

Ln 3, Col 15 Spaces: 2 UTF-8 CRLF Dart Dart: 2.8.4



```
basic.dart > main
Run | Debug
1  main() {
2    var x = ["hii", "hello", "pop"];
3    print(x);
4    print(x.runtimeType);
5  }
6  }
```

PROBLEMS OUTPUT **DEBUG CONSOLE** TERMINAL

[hii, hello, pop]
List<String>
Exited

Ln 2, Col 35 Spaces: 2 UTF-8 CRLF Dart Dart: 2.8.4

list data type

Visual Studio Code interface showing a Dart file named `basic.dart` with the following code:

```
1 main() {  
2   var x = [3, 5, 6];  
3   print(x);  
4   print(x.runtimeType);  
5 }  
6
```

The output in the DEBUG CONSOLE is:

```
[3, 5, 6]  
List<int>  
Exited
```

The status bar at the bottom indicates the current position is Ln 2, Col 19, Spaces: 2, UTF-8, CRLF, Dart, Dart: 2.8.4.

Visual Studio Code interface showing a Dart file named `basic.dart` with the following code:

```
1 main() {  
2   var x = [3, 5, 6, "hii"];  
3   print(x);  
4   print(x.runtimeType);  
5 }  
6
```

The output in the DEBUG CONSOLE is:

```
[3, 5, 6, hii]  
List<Object>  
Exited
```

The status bar at the bottom indicates the current position is Ln 2, Col 25, Spaces: 2, UTF-8, CRLF, Dart, Dart: 2.8.4.

The screenshot shows the Visual Studio Code interface with a Dart file named `basic.dart` open. The code defines a `main` function that creates a `List` with three elements: `3`, `5`, and `6`, and a string `"hii"`. It then prints the list and its runtime type. The `DEBUG CONSOLE` tab at the bottom shows the output: `[3, 5, 6, hii]`, `List<dynamic>`, and `Exited`. The status bar at the bottom indicates the current position is `Ln 2, Col 7` and the Dart SDK version is `Dart: 2.8.4`.

```
basic.dart > main
Run | Debug
1 main() {
2   List x = [3, 5, 6, "hii"];
3   print(x);
4   print(x.runtimeType);
5 }
6
```

DEBUG CONSOLE

```
[3, 5, 6, hii]
List<dynamic>
Exited
```

Ln 2, Col 7 Spaces: 2 UTF-8 CRLF Dart Dart: 2.8.4

The screenshot shows the same Visual Studio Code interface, but with a type error. The code has been modified so that the `List` is typed as `List<int>`. Since the list contains a string `"hii"`, the compiler has flagged this as an error. The `DEBUG CONSOLE` tab now shows a red error message: `basic.dart:2:27: Error: A value of type 'String' can't be assigned to a variable of type 'int'.` The status bar at the bottom indicates the current position is `Ln 2, Col 12` and the Dart SDK version is `Dart: 2.8.4`.

```
basic.dart > main
Run | Debug
1 main() {
2   List<int> x = [3, 5, 6, "hii"];
3   print(x);
4   print(x.runtimeType);
5 }
6
```

DEBUG CONSOLE

```
basic.dart:2:27: Error: A value of type 'String' can't be assigned to a variable of type 'int'.
  List<int> x = [3, 5, 6, "hii"];
                        ^
Exited (254)
```

Ln 2, Col 12 Spaces: 2 UTF-8 CRLF Dart Dart: 2.8.4

The screenshot shows the Visual Studio Code editor with a Dart file named `basic.dart`. The code defines a `main` function that creates a `List<int>` named `x` with the values `[3, 5, 6]`. It then prints the list and its runtime type. The output in the Debug Console shows the list `[3, 5, 6]`, the type `List<int>`, and the program exited successfully.

```
basic.dart > main
Run | Debug
1  main() {
2      List<int> x = [3, 5, 6];
3      print(x);
4      print(x.runtimeType);
5  }
6

[3, 5, 6]
List<int>
Exited
```

array start with position number zero by default

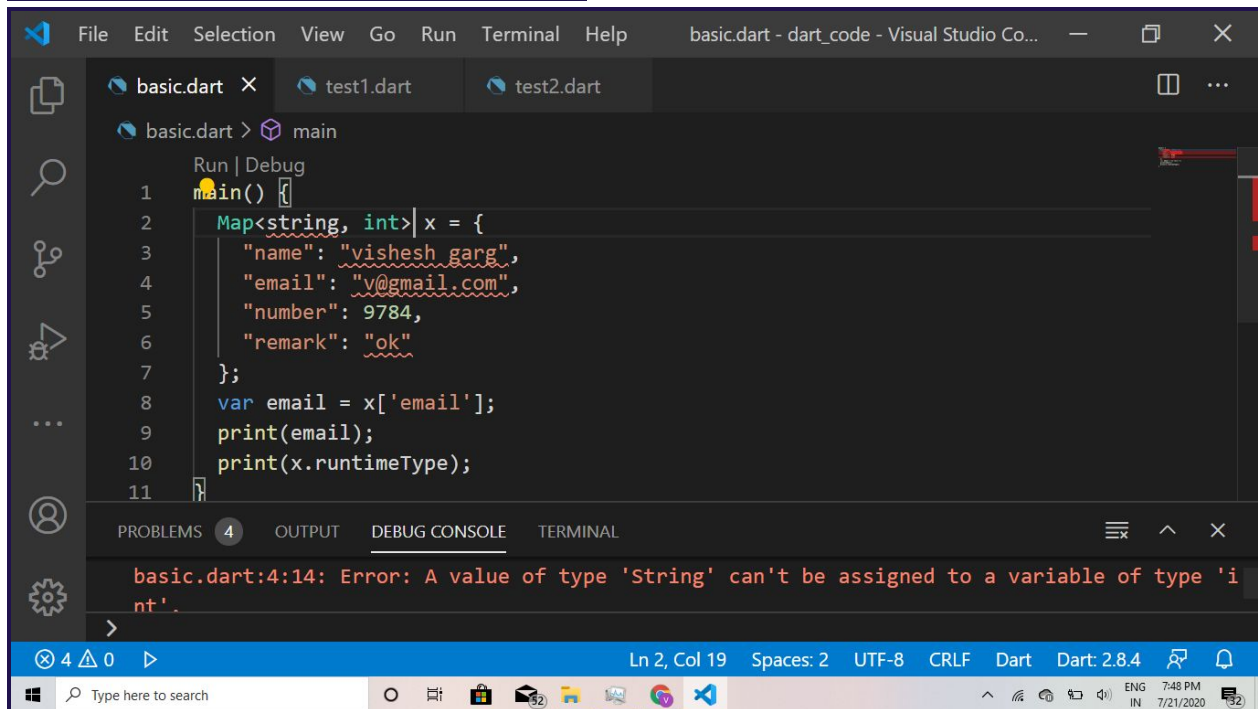
The screenshot shows the Visual Studio Code editor with a Dart file named `basic.dart`. The code defines a `main` function that creates a `List` named `x` with mixed types: `[1, "v@gmail.com", 9784, "ok"]`. It then accesses the element at index `1` (which is `"v@gmail.com"`) and prints it along with its runtime type. The output in the Debug Console shows the string `v@gmail.com`, the type `String`, and the program exited successfully.

```
basic.dart > main
Run | Debug
1  main() {
2      var x = [1, "v@gmail.com", 9784, "ok"];
3      var y = x[1];
4      print(y);
5      print(y.runtimeType);
6  }
7

v@gmail.com
String
Exited
```

list can be append ,update,delete it is like a read-writeable (mutable)

So it is better to use mapping concept



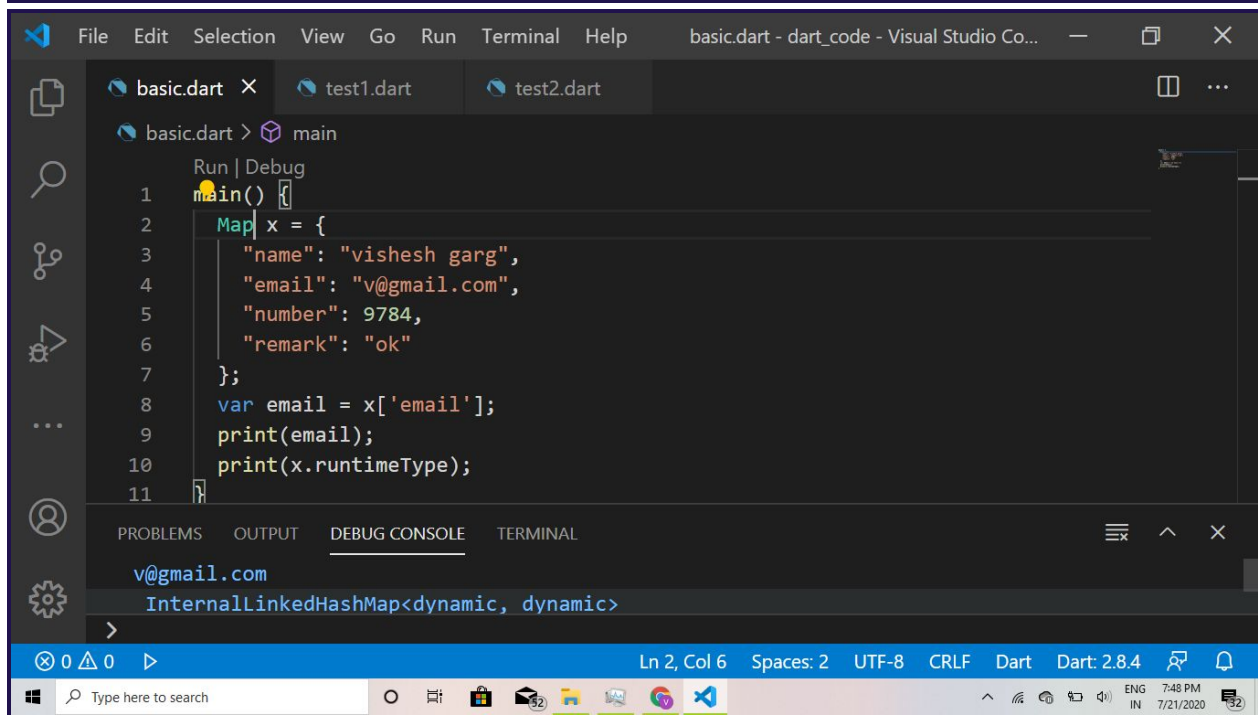
The screenshot shows the Visual Studio Code editor with a Dart file named `basic.dart`. The code defines a `Map<String, int>` and attempts to assign a string value to an integer variable. The error message in the console states: `basic.dart:4:14: Error: A value of type 'String' can't be assigned to a variable of type 'int'.`

```
1 main() {  
2   Map<String, int> x = {  
3     "name": "vishesh garg",  
4     "email": "v@gmail.com",  
5     "number": 9784,  
6     "remark": "ok"  
7   };  
8   var email = x['email'];  
9   print(email);  
10  print(x.runtimeType);  
11 }
```

PROBLEMS 4 OUTPUT DEBUG CONSOLE TERMINAL

basic.dart:4:14: Error: A value of type 'String' can't be assigned to a variable of type 'int'.

Ln 2, Col 19 Spaces: 2 UTF-8 CRLF Dart Dart: 2.8.4



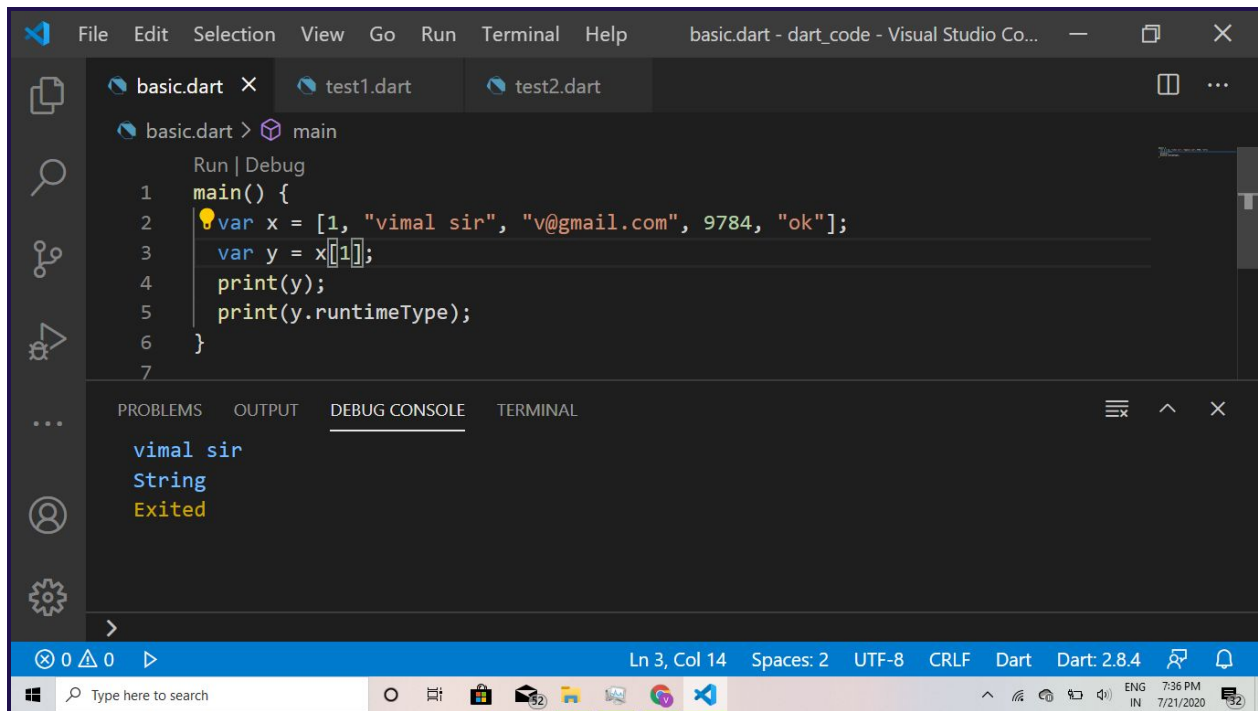
The screenshot shows the same Visual Studio Code editor with the Dart file `basic.dart`. The type annotation `Map<String, int>` has been removed from line 2. The code now compiles successfully, and the output in the console shows the email address `v@gmail.com` and the runtime type `InternallinkedHashMap<dynamic, dynamic>`.

```
1 main() {  
2   Map x = {  
3     "name": "vishesh garg",  
4     "email": "v@gmail.com",  
5     "number": 9784,  
6     "remark": "ok"  
7   };  
8   var email = x['email'];  
9   print(email);  
10  print(x.runtimeType);  
11 }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

v@gmail.com
InternallinkedHashMap<dynamic, dynamic>

Ln 2, Col 6 Spaces: 2 UTF-8 CRLF Dart Dart: 2.8.4



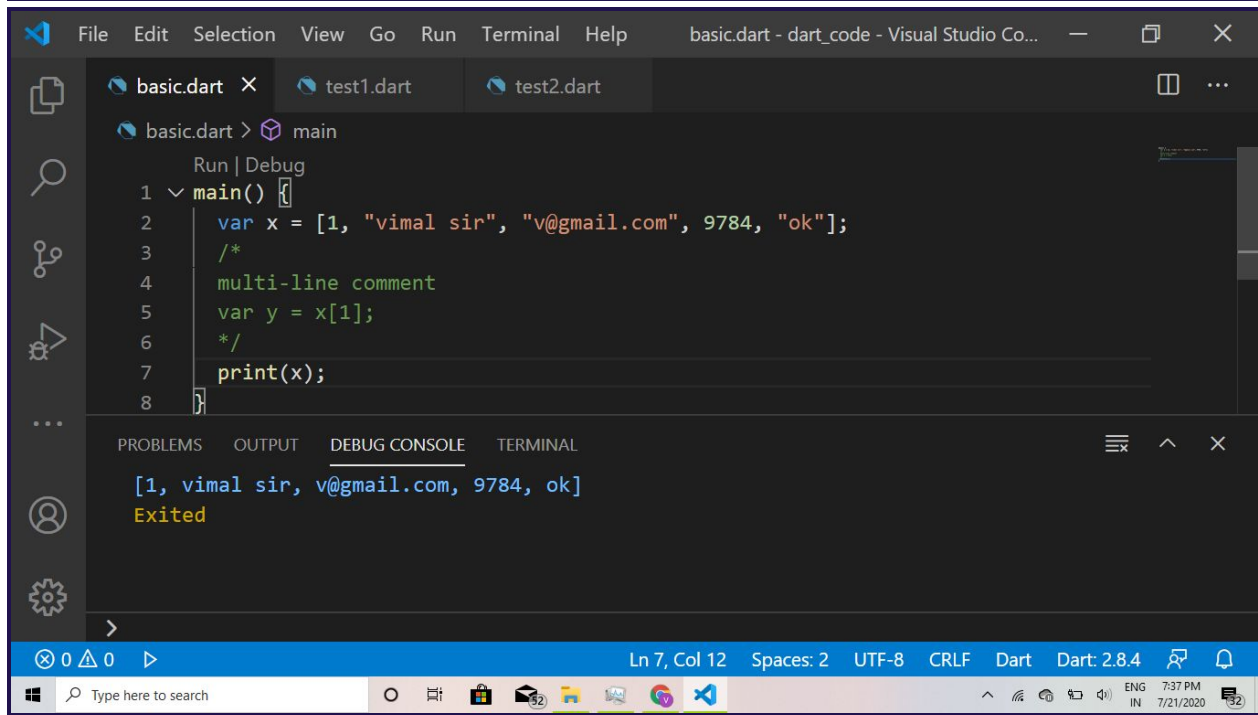
Visual Studio Code interface showing a Dart program execution. The code in `basic.dart` is as follows:

```
1 main() {  
2   var x = [1, "vimal sir", "v@gmail.com", 9784, "ok"];  
3   var y = x[1];  
4   print(y);  
5   print(y.runtimeType);  
6 }  
7
```

The output in the DEBUG CONSOLE is:

```
vimal sir  
String  
Exited
```

The status bar at the bottom indicates the cursor is at Ln 3, Col 14. The system tray shows the date and time as 7:36 PM on 7/21/2020.



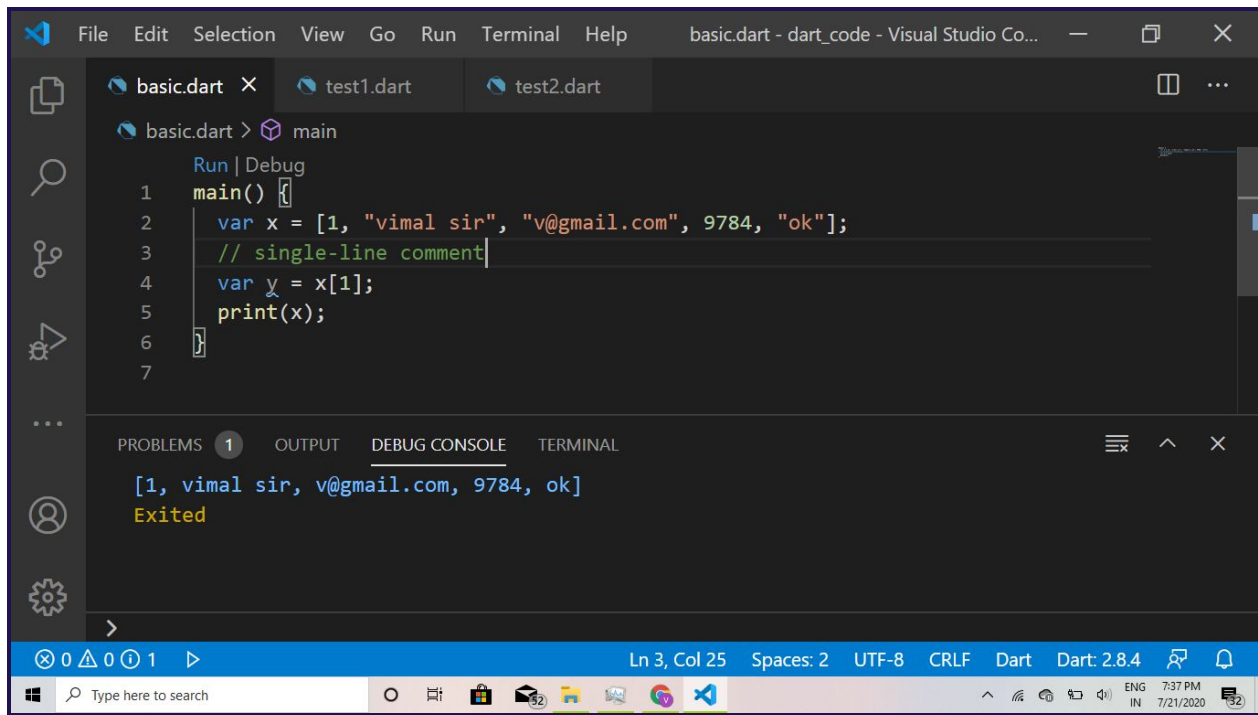
Visual Studio Code interface showing a Dart program execution. The code in `basic.dart` is as follows:

```
1 main() {  
2   var x = [1, "vimal sir", "v@gmail.com", 9784, "ok"];  
3   /*  
4    multi-line comment  
5   */  
6   var y = x[1];  
7   print(x);  
8 }
```

The output in the DEBUG CONSOLE is:

```
[1, vimal sir, v@gmail.com, 9784, ok]  
Exited
```

The status bar at the bottom indicates the cursor is at Ln 7, Col 12. The system tray shows the date and time as 7:37 PM on 7/21/2020.



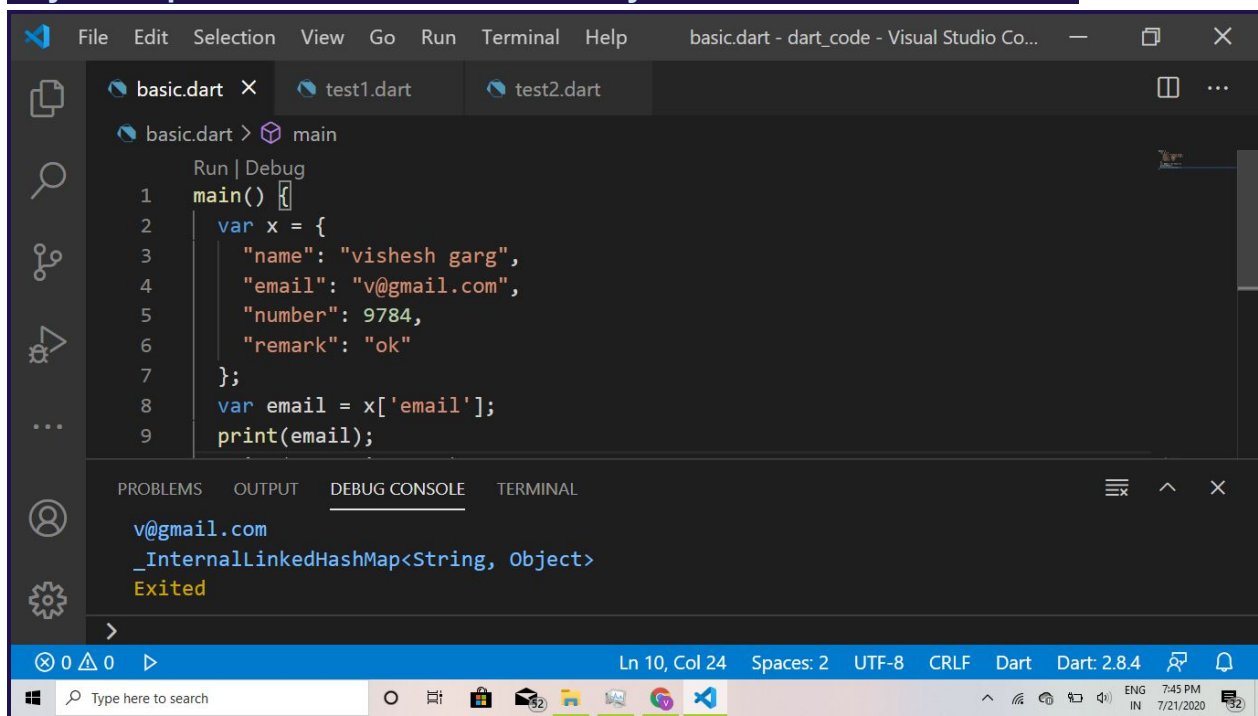
```
basic.dart > main
Run | Debug
1  main() {
2      var x = [1, "vimal sir", "v@gmail.com", 9784, "ok"];
3      // single-line comment
4      var y = x[1];
5      print(x);
6  }
7

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL
[1, vimal sir, v@gmail.com, 9784, ok]
Exited
```

map data type:

Variable:value

Key-value pair here email is known as key and value is the the email id



```
basic.dart > main
Run | Debug
1  main() {
2      var x = {
3          "name": "vishesh garg",
4          "email": "v@gmail.com",
5          "number": 9784,
6          "remark": "ok"
7      };
8      var email = x['email'];
9      print(email);
10 }

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
v@gmail.com
_InternalLinkedHashMap<String, Object>
Exited
```

```
main() {
  var x = {
    "name": "vishesh garg",
```



```
    "email": "v@gmail.com",  
    "number": 9784,  
    "remark": "ok"  
};  
var email = x['email'];  
print(email);  
print(x.runtimeType);  
}
```

If u want to auto format ur code just place a comma on the last-key value pair example:

```
Map x = {  
    "name": "vishesh garg",  
    "email": "v@gmail.com",  
    "number": 9784,  
    "remark": "ok",  
};
```

The screenshot shows the Visual Studio Code editor with a Dart file named `basic.dart`. The code defines a `main` function that calls `vishesh()` twice, and a `vishesh` function that prints "hii vishesh". The `DEBUG CONSOLE` at the bottom shows the output: "hii vishesh", "hii vishesh", and "Exited".

```
basic.dart > main
Run | Debug
1  main() {
2    vishesh();
3    vishesh();
4  }
5
6  vishesh() {
7    print("hii vishesh");
8  }
9
10
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
hii vishesh
hii vishesh
Exited
```

Ln 3, Col 13 Spaces: 2 UTF-8 CRLF Dart Dart: 2.8.4

The screenshot shows the Visual Studio Code editor with a Dart file named `basic.dart`. The code defines a `main` function that calls `vishesh("hi")` and `vishesh("hello")`, and a `vishesh` function that takes an argument `i` and prints it. The `DEBUG CONSOLE` at the bottom shows the output: "hi", "hello", and "Exited".

```
basic.dart > vishesh
Run | Debug
1  main() {
2    vishesh("hi");
3    vishesh("hello");
4  }
5
6  vishesh(i) {
7    print(i);
8  }
9
10
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
hi
hello
Exited
```

Ln 6, Col 13 Spaces: 2 UTF-8 CRLF Dart Dart: 2.8.4

The screenshot shows the Visual Studio Code interface with the file `basic.dart` open. The code defines a `main` function that calls `vishesh("hi", 4)` and `vishesh("hello", "vishesh")`. The `vishesh` function is defined with parameters `i` and `j`, and it prints both. The `DEBUG CONSOLE` at the bottom shows the output: `hi`, `4`, `hello`, and `vishesh`. The status bar at the bottom indicates the cursor is at line 8, column 12.

```
basic.dart > vishesh
Run | Debug
1  main() {
2      vishesh("hi", 4);
3      vishesh("hello", "vishesh");
4  }
5
6  vishesh(i, j) {
7      print(i);
8      print(j);
9  }
```

PROBLEMS OUTPUT **DEBUG CONSOLE** TERMINAL

```
hi
4
hello
vishesh
>
```

Ln 8, Col 12 Spaces: 2 UTF-8 CRLF Dart Dart: 2.8.4

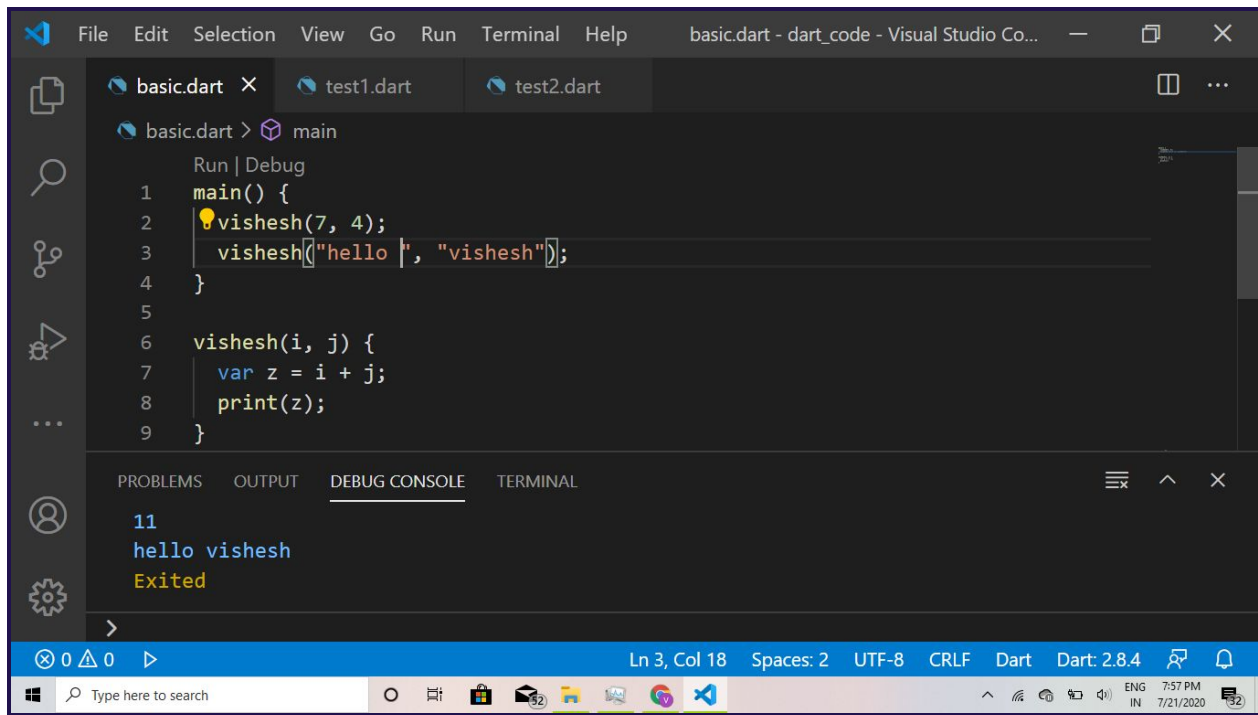
The screenshot shows the same Visual Studio Code interface, but the code in `basic.dart` has been modified. The `vishesh` function now calculates `var z = i + j;` before printing. The `DEBUG CONSOLE` shows an `Unhandled exception: type 'int' is not a subtype of type 'String' of 'other'`, indicating a runtime error because the function is trying to concatenate a string and an integer. The status bar at the bottom indicates the cursor is at line 7, column 7.

```
basic.dart > vishesh
Run | Debug
1  main() {
2      vishesh("hi", 4);
3      vishesh("hello", "vishesh");
4  }
5
6  vishesh(i, j) {
7      var z = i + j;
8      print(z);
9  }
```

PROBLEMS OUTPUT **DEBUG CONSOLE** TERMINAL

```
Unhandled exception:
type 'int' is not a subtype of type 'String' of 'other'
#0      vishesh                                basic.dart:7
#1      main                                    basic.dart:2
>
```

Ln 7, Col 7 Spaces: 2 UTF-8 CRLF Dart Dart: 2.8.4



function signature

Visual Studio Code interface showing a Dart file named `basic.dart` with the following code:

```
1 main() {  
2   vishesh(7, 4);  
3   vishesh("hello ", "vishesh");  
4   vishesh  
5 }  
6  
7 vishesh {  
8   var z = i + j;  
9   print(z);  
}
```

The `DEBUG CONSOLE` shows the output:

```
11  
hello vishesh  
Exited
```

The status bar indicates the cursor is at `Ln 4, Col 8`.

Visual Studio Code interface showing a Dart file named `basic.dart` with the following code:

```
1 main() {  
2   var p = vishesh(7, 4);  
3   print(p);  
4 }  
5  
6 vishesh(i, j) {  
7   var z = i + j;  
8   print(z);  
9 }
```

The `DEBUG CONSOLE` shows the output:

```
11  
null  
Exited
```

The status bar indicates the cursor is at `Ln 4, Col 2`.

Visual Studio Code interface showing a Dart file named `basic.dart`. The code defines a function `vishesh(i, j)` that returns the sum of `i` and `j`, and a `main()` function that calls `vishesh(7, 4)` and prints the result. The status bar indicates the cursor is at line 7, column 17.

```
basic.dart > vishesh
Run | Debug
1  main() {
2      var p = vishesh(7, 4);
3      print(p);
4  }
5
6  vishesh(i, j) {
7      var z = i + j;
8      return z;
9  }
```

11 Exited

Ln 7, Col 17 Spaces: 2 UTF-8 CRLF Dart Dart: 2.8.4

Visual Studio Code interface showing the same Dart file `basic.dart` with an additional `import "test1.dart";` statement at the top. The status bar indicates the cursor is at line 1, column 21.

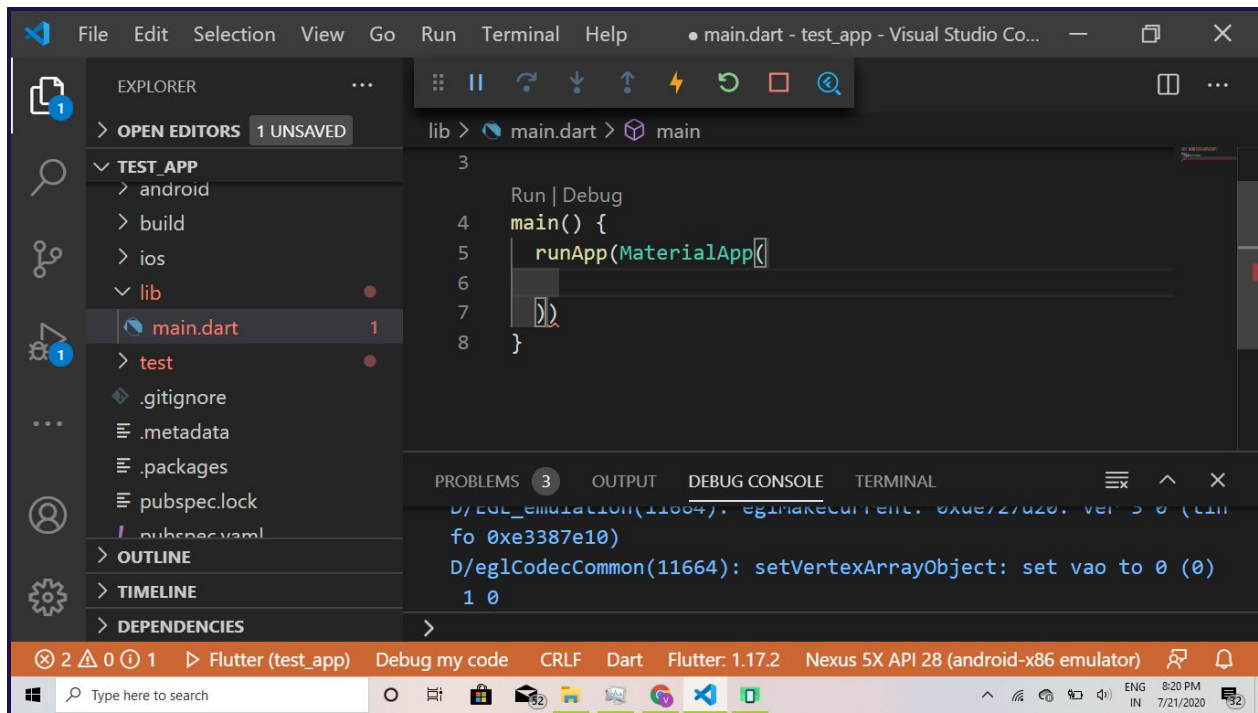
```
basic.dart > ...
1  import "test1.dart";
2
3  Run | Debug
4  main() {
5      var p = vishesh(7, 4);
6      print(p);
7  }
```

11 Exited

Ln 1, Col 21 Spaces: 2 UTF-8 CRLF Dart Dart: 2.8.4

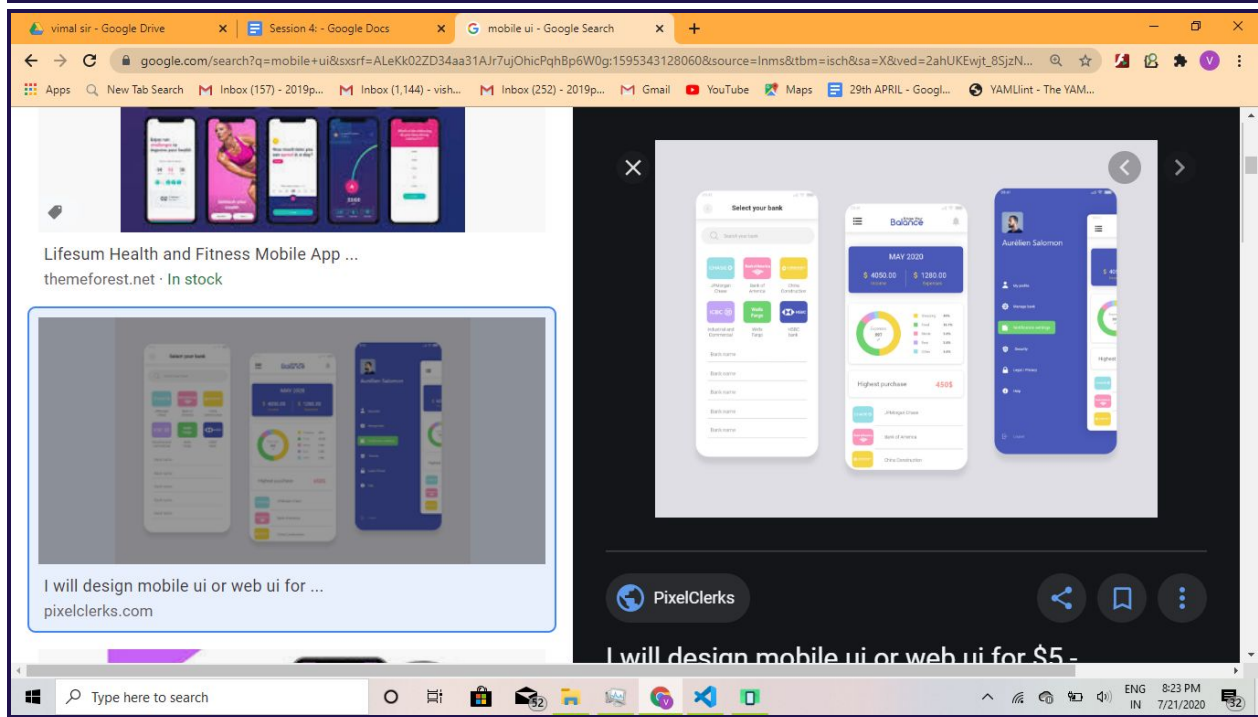
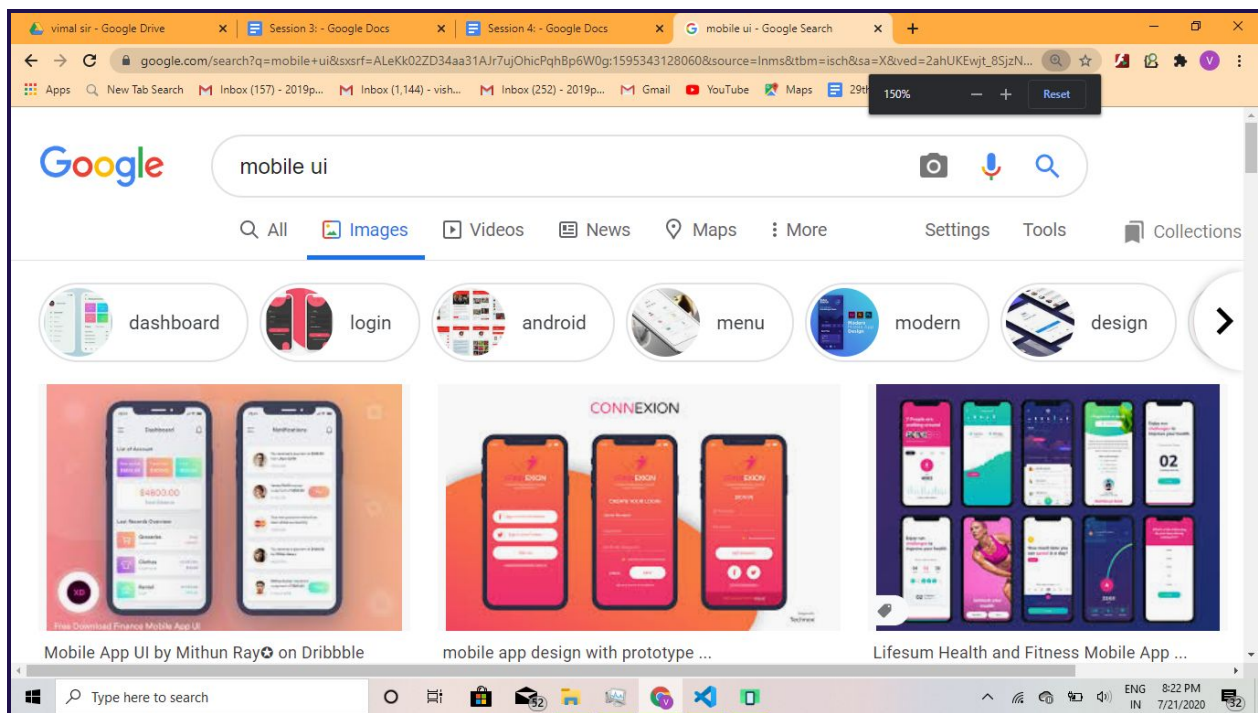
if we replace the file name main.dart with xyz.dart then flutter won't debug it will give us the json file and show us no device is connected

Flutter -> lib -> main.dart



User interface

What is UX ???



```
main() {
  runApp(user_interface())
}
```

Design language:

Varies from company to company sometimes it is known as design icon or design frame

Eg: google,microsoft etc.

Language name is material design

So here we use MaterialApp which has iconed and other look and feels by google

