Creating a Flutter Web Hello World Application

Estimated time: **15** minutes

Welcome to the lab on building your first Flutter web application. You will create a simple "Hello World" app in this lab that runs in a web browser. This is a great way to get started with Flutter's web development capabilities.

Objectives

After completing this lab, you will be able to:

- Set up Flutter for web development
- Create a basic Flutter web application
- Modify the main Dart file to display "Hello World"
- Run the Flutter application in a web browser

About Cloud IDE

Running the lab

This lab is designed to be completed using a Cloud IDE environment. You will be writing and executing Dart code directly within the IDE.

About Skills Network Cloud IDE

Skills Network Cloud IDE (based on Theia and Docker) provides an environment for hands on labs for course and project-related labs. It is an open-source Integrated Development Environment (IDE)).

Important notice about this lab environment

Please be aware that sessions for this lab environment are not persisted. Every time you connect to this lab, a new environment is created for you. Any data you may have saved in the earlier session would get lost. Plan to complete these labs in a single session to avoid losing your data.

Key terms

Key terms

- **Flutter**: An open-source UI software development kit created by Google for building natively compiled applications for mobile, web, and desktop from a single codebase.
- **Dart**: A client-optimized programming language for fast apps on any platform. Dart is the programming language used in Flutter.
- **Widget**: The basic building block of a Flutter app's user interface.
- **Scaffold**: A top-level container in Flutter that provides an app structure.
- **Hot Reload**: A feature in Flutter that allows you to instantly see the results of your changes without restarting the application.

Components of Flutter web development

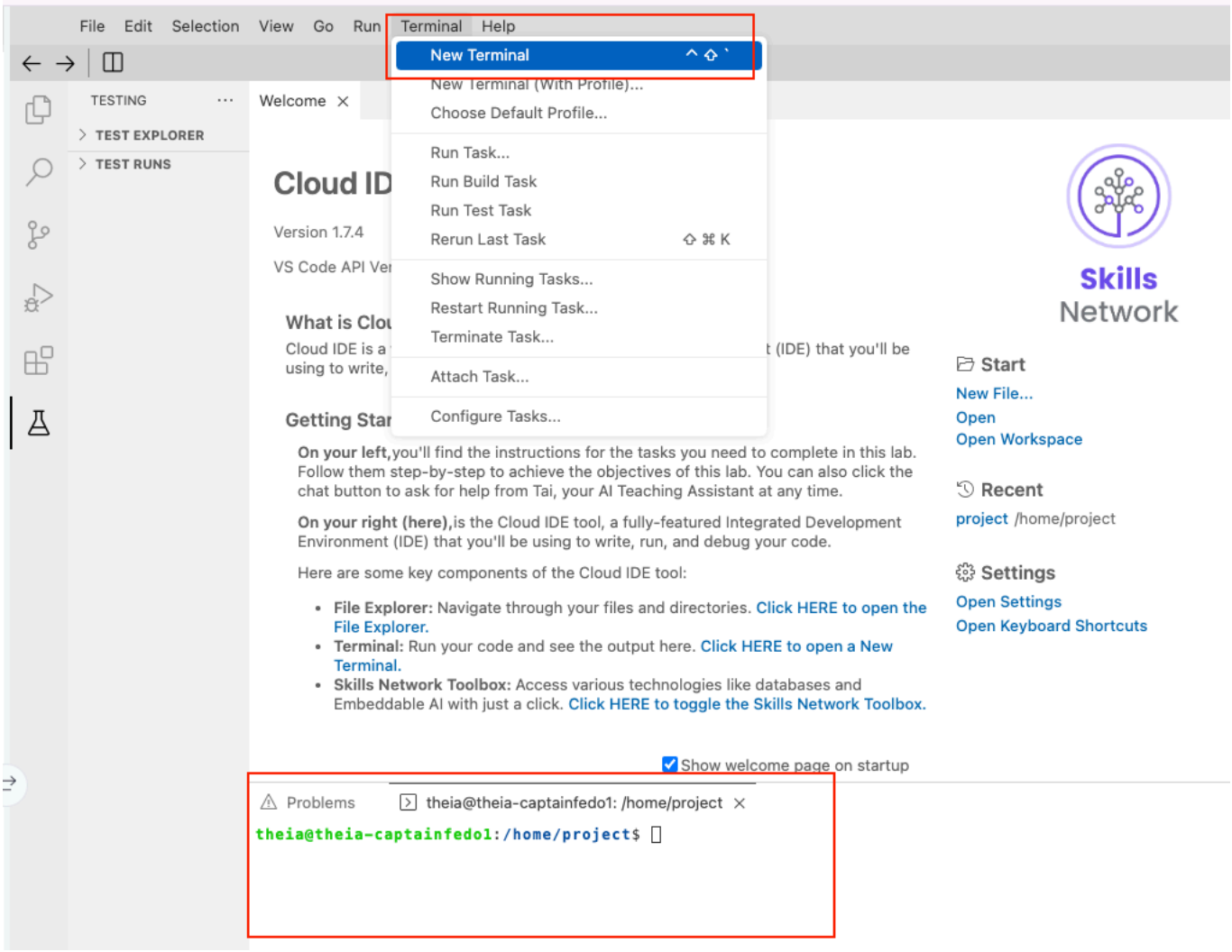
- **Cross-platform**: Build and deploy web applications that work across different browsers and platforms.
- **Single codebase**: Use a single codebase to build apps for mobile, web, and desktop.
- **Fast development**: Experience fast development with Flutter's Hot Reload feature, which allows you to instantly see changes in your code.

Different widgets available

- **Text**: Displays a string of text with a single style.
- **Center**: Aligns its child widget to the center of the screen.
- **Scaffold**: Provides a structure for the visual interface, including an AppBar, Body, and more.
- **AppBar**: A material design app bar that can be placed at the top of the Scaffold.
- **MaterialApp**: A convenience widget that wraps several widgets commonly required for Material Design applications.

Step 1: Confirm lab environment

1. Open your terminal or command prompt. Select `Terminal -> New Terminal`.



2. Run the following command to enable web development in Flutter:

```
flutter config --enable-web
```

3. Verify that Flutter recognizes the web environment by running:

```
flutter devices
```

You should see `Chrome (web)` listed as one of the devices.

```
theia@theia-captainfedo1:/home/project$ flutter devices
Found 2 connected devices:
  Linux (desktop) • linux  • linux-x64      • Ubuntu 22.04.4 LTS 5.4.0-193-generic
  Chrome (web)    • chrome • web-javascript • Google Chrome for Testing 128.0.6613.84
Run "flutter emulators" to list and start any available device emulators.
If you expected another device to be detected, please run "flutter doctor" to diagnose potential issues.
You may also try increasing the time to wait for connected devices with the "--device-timeout" flag. Visit
https://flutter.dev/setup/ for troubleshooting tips.                • chrome                • web-javascript • Google Chrome 127.0.6533.122
```

4. Finally, run `flutter doctor` utility to ensure your environment is ready to start coding. Flutter doctor checks the flutter installation by running diagnostics checks. Since you are testing the apps in the web version, we just want to ensure the following items are marked checked.

    - Flutter
    - Chrome
    - Connected Devices

    We don't care if the other items are marked with a cross as we are not using those features. Here is the output from the lab environment:

```
[✓] Flutter (Channel stable, 3.24.1, on Ubuntu 22.04.5 LTS 5.4.0-196-generic, locale en_US)
[✗] Android toolchain - develop for Android devices
    ✗ Unable to locate Android SDK.
      Install Android Studio from: https://developer.android.com/studio/index.html
      On first launch it will assist you in installing the Android SDK components.
      (or visit https://flutter.dev/to/linux-android-setup for detailed instructions).
      If the Android SDK has been installed to a custom location, please use
      `flutter config --android-sdk` to update to that location.
[✓] Chrome - develop for the web
[✗] Linux toolchain - develop for Linux desktop
    ✗ ninja is required for Linux development.
      It is likely available from your distribution (e.g.: apt install ninja-build), or can be downloaded from https://github.com/ninja-build/n
    ✗ GTK 3.0 development libraries are required for Linux development.
      They are likely available from your distribution (e.g.: apt install libgtk-3-dev)
[!] Android Studio (not installed)
[✓] Connected device (2 available)
[!] Network resources
✗ A network error occurred while checking "https://cocoapods.org/": Connection timed out
! Doctor found issues in 4 categories.
```

Step 2: Create a new Flutter project

1. In your terminal, navigate to the directory where you want to create your new project.
2. Run the following command to create a new Flutter project named `hello_world_web`:

```
flutter create hello_world_web
```

You should see output as follows:

```
theia@theia-captainfedo1:/home/project$ flutter create hello_world_web
Creating project hello_world_web...
Resolving dependencies in `hello_world_web`...
Downloading packages...
Got dependencies in `hello_world_web`.
Wrote 129 files.
All done!
You can find general documentation for Flutter at: https://docs.flutter.dev/
Detailed API documentation is available at: https://api.flutter.dev/
If you prefer video documentation, consider: https://www.youtube.com/c/flutterdev
```

3. Navigate into your project directory:

```
cd hello_world_web
```

4. Update all libraries

```
flutter pub get
```

This command updates the packages used in your flutter application. The output should look as follows:

```
theia@theia-captainfedo1:/home/project/hello_world_web$ flutter pub get
 Resolving dependencies...
 Downloading packages...
   collection 1.18.0 (1.19.0 available)
   leak_tracker 10.0.5 (10.0.7 available)
   leak_tracker_flutter_testing 3.0.5 (3.0.8 available)
   material_color_utilities 0.11.1 (0.12.0 available)
   string_scanner 1.2.0 (1.3.0 available)
   test_api 0.7.2 (0.7.3 available)
 Got dependencies!
 6 packages have newer versions incompatible with dependency constraints.
 Try `flutter pub outdated` for more information.
```

Step 3: Modify the main Dart file

1. Open the `lib/main.dart` file in your code editor.

2. Replace the existing code with the following to display “Hello World” in the center of the web page:

```
import 'package:flutter/material.dart';
void main() {
  runApp(MyApp());
}
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Hello World Web',
      home: Scaffold(
        appBar: AppBar(
          title: Text('Hello World Web App'),
        ),
        body: Center(
          child: Text(
            'Hello World',
            style: TextStyle(fontSize: 24),
          ),
        ),
      ),
    );
  }
}
```

- **MaterialApp**: Wraps the entire app and provides material design styling.
- **Scaffold**: Provides a basic structure, including an `AppBar` and a `Body`.
- **AppBar**: Displays the title of the application.
- **Center**: Aligns the "Hello World" text to the center of the screen.
- **Text**: Displays the "Hello World" message with a font size of 24.

Step 4: Run the application in a web browser

Ideally we would use `flutter run` command to run the application. However since you are runnig this lab in a Cloud IDE, we will instead build the web version of the app and then start a python server to launch the application.

Flutter provides `hot reload` that makes it very easy to test the code as you are developing it. That functionality however does not work in the Cloud IDE. We have developed a `hot_reload.sh` script that essentially does the same thing. Please follow these steps to run the flutter application:

1. Change to the home directory

   ```
   cd /home/project
   ```

2. Set the `PROJECT_DIR` variable. This should be set to the `lib` directory of your flutter app folder.

   ```
   export PROJECT_DIR=/home/project/hello_world_web/lib
   ```

3. Get the script and save to the `/home/project` folder.

   ```
   wget -O /home/project/hot_reload.sh https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/S3cw83zxbxBhfixSSMK1Uw/hot-reload.sh
   ```

4. Make the script executable by using the `chmod` command.

   ```
   chmod +x ./hot_reload.sh
   ```

5. Run the command.

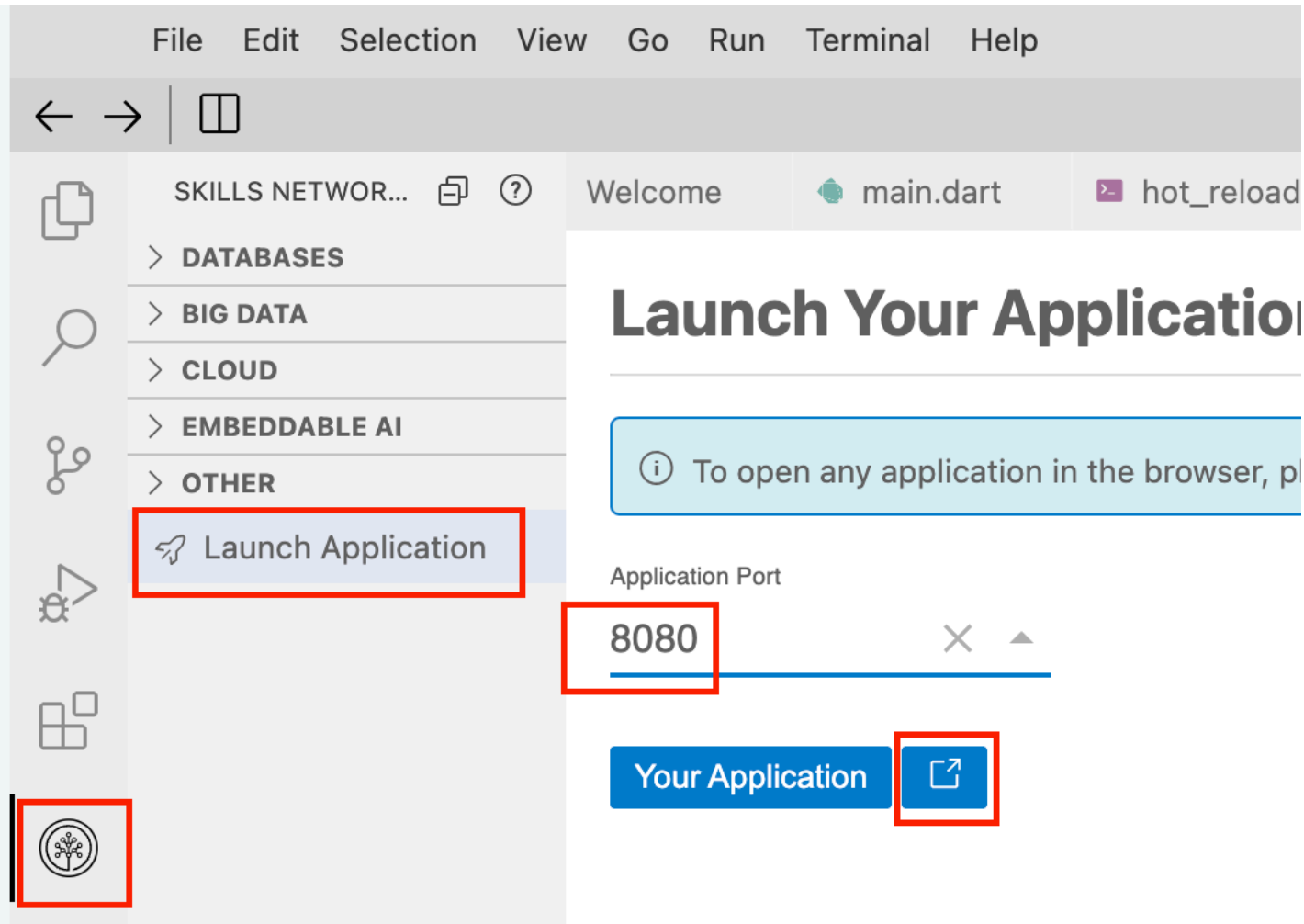
   ```
   ./hot_reload.sh
   ```

You should see output like this:

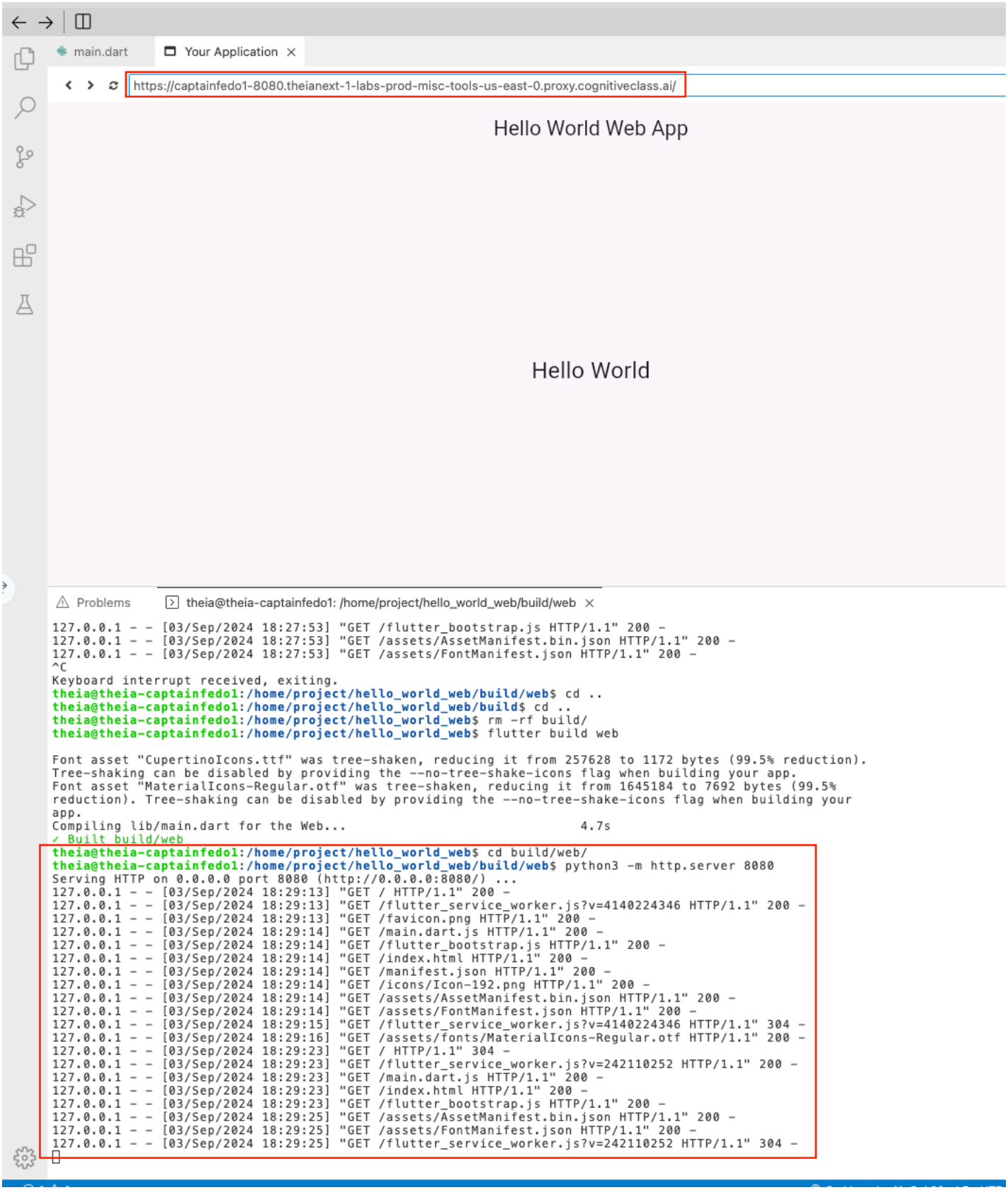
```
theia@theia-captainfedo1:/home/project$ ./hot_reload.sh
Using PROJECT_DIR: /home/project/hello_world_web/lib
inotify-tools is already installed.
lsof is already installed.
Starting Flutter web server...
Port 8080 is free.
Compiling lib/main.dart for the Web...                             984ms
✓ Built build/web
Flutter server started with PID 17675
Setting up watches.  Beware: since -r was given, this may take a while!
Watches established.
Serving HTTP on 0.0.0.0 port 8080 (http://0.0.0.0:8080/) ...
```

6. Launch the application

   - open the `Skills Network Toolbox` in the let bar of the Cloud IDE.
   - click on `Launch Application` button
   - enter `8080` as the port
   - select the `launch in new window` to open the applicaiton in a new tab



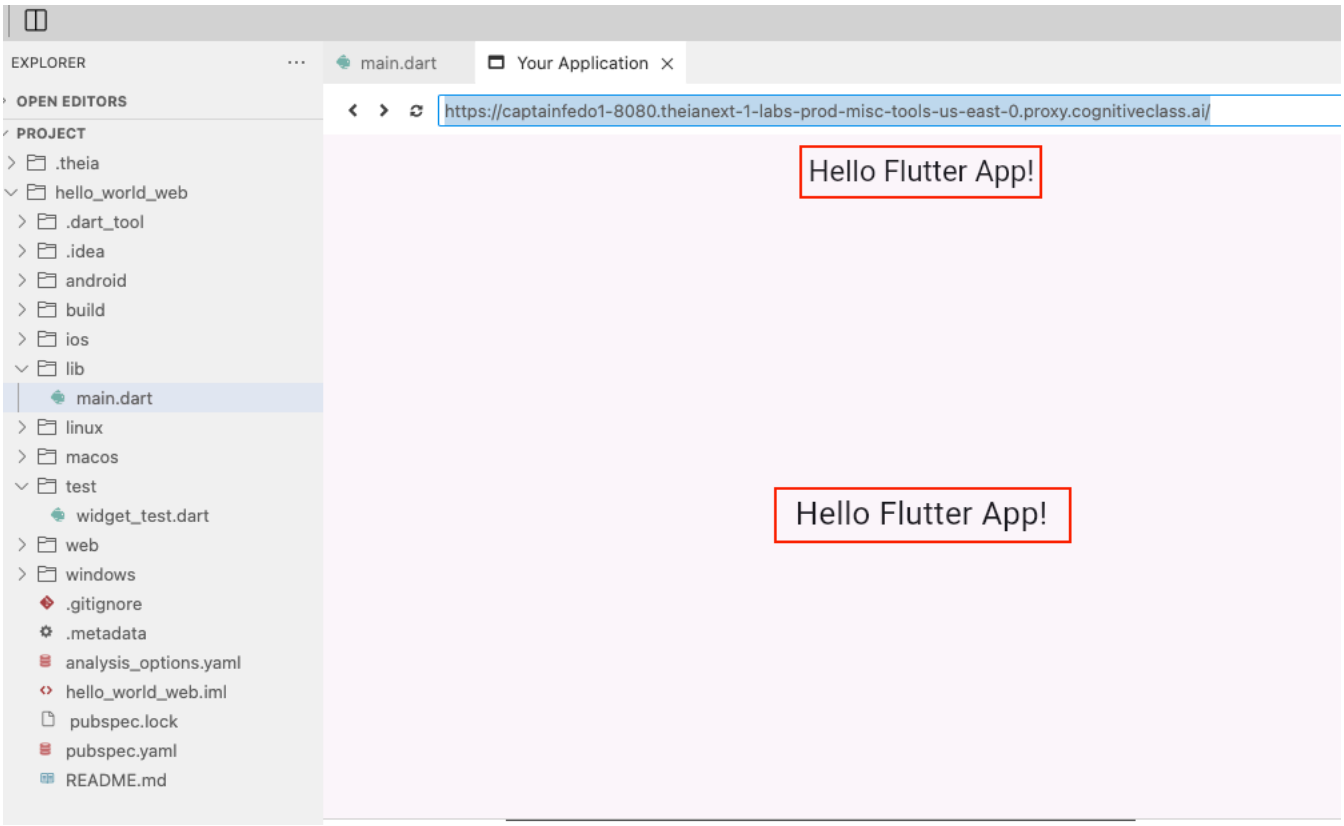
7. Verify that the text "Hello World" appears centered on the page with the title "Hello World Web App" in the AppBar.



8. Experiment by changing the text in the `Text` widget to something else, like "Welcome to Flutter Web!". Since you are using a special cloud IDE environment, you will have to wait a little for the app to rebuild and the server to restart. If you quit that process for some reason, run the script again from the `/home/project` folder.

```
cd /home/project && ./hot_reload.sh
```

Once the app starts, you can refresh the page if the tab is already open, or follow the instructions to launch application on port `8080` as before.



Complete solution

To access the entire code for this lab, here is the code snippet from the main Dart file:

```
import 'package:flutter/material.dart';
void main() {
  runApp(MyApp());
}
class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Hello World Web',
      home: Scaffold(
        appBar: AppBar(
          title: Text('Hello World Web App'),
        ),
        body: Center(
          child: Text(
            'Hello World',
            style: TextStyle(fontSize: 24),
          ),
        ),
      ),
    );
  }
}
```

- **MaterialApp**: Wraps the entire app and provides material design styling.
- **Scaffold**: Provides a basic structure, including an `AppBar` and a `Body`.
- **AppBar**: Displays the title of the application.
- **Center**: Aligns the "Hello World" text to the center of the screen.
- **Text**: Displays the "Hello World" message with a font size of 24.

Next steps

Congratulations on completing the "Hello World" web application lab! Here are some suggestions for what you can do next:

1. **Explore other widgets**: Try adding different widgets, such as `Button`, `Image`, and `Column` to your app.
2. **Style your app**: Experiment with different styles and themes to customize the appearance of your app.
3. **Deploy your app**: Once you are comfortable with your app, explore the steps to deploy your Flutter web application to a hosting service.

You have successfully created and run a basic Flutter web application. This foundational skill will be crucial as you continue to develop more complex web applications using Flutter.

Author(s)

Skills Network