

Generate haikus for Google products

[Before you begin](#)

[Prerequisites](#)

[What you'll learn](#)

[What you'll need](#)

[Set up your Flutter development environment](#)

[Get set up](#)

[Download the dependencies for the project](#)

[Step 0: Run the starter app](#)

[Run and explore the app](#)

[Step 1: Set up the Generative Language API access](#)

[Step 2: Add the DropdownButton with a list of Google products](#)

[Step 3: Complete the Flutter app for Android and iOS](#)

[Run it](#)

[Step 4: Try other prompts](#)

[Step 5: Run the Flutter app on the desktop platforms](#)

[Linux](#)

[Mac](#)

[Windows](#)

[Step 5: Run the Flutter app on the web platform](#)

[Congratulations](#)

[Learn more](#)

Before you begin

Duration: 3:00

One of the most exciting machine learning breakthroughs recently is generative AI. Generative AI can produce amazing images, texts, audios, and even videos based on simple text from users (also known as prompts). Specifically with the release of Generative Language API powered by [LaMDA](#), Google is empowering developers to build new categories of apps with delightful user experiences re-imagined with generative language technology.

In this codelab, you learn how to build an app using:

- Generative Language API to generate haikus based on Google product names
- Flutter to create a cross-platform app to display the haikus

Prerequisites

- Basic knowledge of generative language models, i.e., prompting

- Basic knowledge of Flutter development with Dart

What you'll learn

- How to use the new Discuss API from Google
- How to build a cross-platform Flutter app to display the results

What you'll need

- [Flutter SDK](#)
- [Android](#) and [iOS](#) setup for Flutter
- [Desktop](#) setup for Flutter
- [Web](#) setup for Flutter
- [Visual Studio Code \(VS Code\) setup for Flutter and Dart](#)

Set up your Flutter development environment

Duration: 10:00

For Flutter development, you need two pieces of software to complete this codelab—the [Flutter SDK](#) and [an editor](#) (in this codelab, we are going to the Visual Studio Code editor).

You can run the codelab using any of these devices:

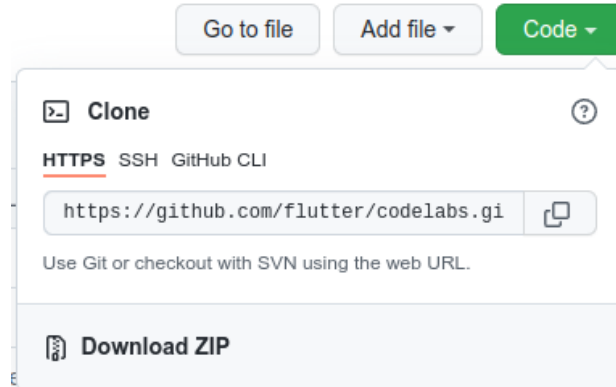
- The [iOS simulator](#) (requires installing Xcode tools).
- The [Android Emulator](#) (requires setup in Android Studio).
- A browser (Chrome is required for debugging).
- As a [Windows](#), [Linux](#), or [macOS](#) desktop application. You must develop on the platform where you plan to deploy. So, if you want to develop a Windows desktop app, you must develop on Windows to access the appropriate build chain. There are operating system-specific requirements that are covered in detail on docs.flutter.dev/desktop.

Get set up

Duration: 2:00

To download the code for this codelab:

1. Navigate to [the GitHub repository](#) for this codelab.
2. Click **Code** > **Download zip** to download all the code for this codelab.



3. Unzip the downloaded zip file to unpack a `codelabs-main` root folder with all the resources that you need.

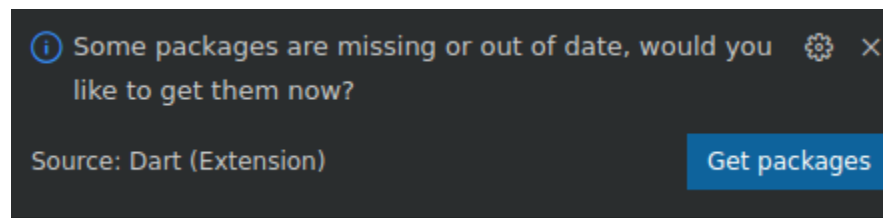
For this codelab, you only need the files in the `haiku-generator/` subdirectory in the repository, which contains multiple folders:

- The `step0` to `step5` folders contain the starter code that you build upon for each step in this codelab.
- The `finished` folder contains the completed code for the finished sample app.

Download the dependencies for the project

Duration: 3:00

1. In VS Code, click **File > Open folder** and then select the `step0` folder from the source code that you downloaded earlier.
2. Open `step0/lib/main.dart` file. If you see a VS Code dialog appear that prompts you to download the required packages for the starter app, click **Get packages**.
3. If you don't see this dialog, open your terminal and then run `flutter pub get` command in the `step0/` folder.




Step 0: Run the starter app

Duration: 3:00

1. Open `step0/lib/main.dart` file in VS Code, ensure that the Android Emulator or iOS Simulator is properly set up and appears in the status bar.

For example, here's what you see when you use Pixel 5 with the Android Emulator:

 Pixel 5 API 30 (android-x86 emulator)

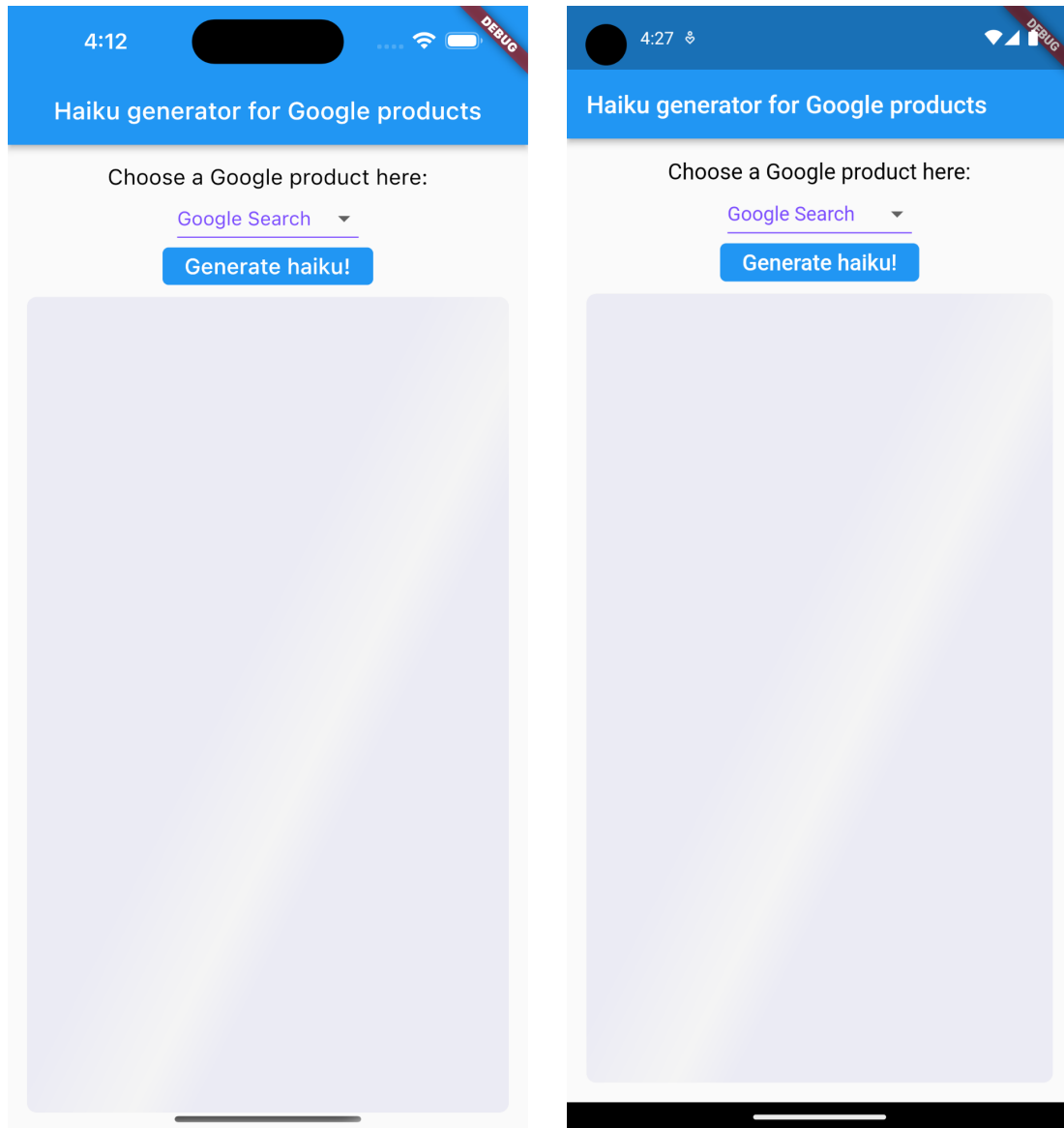
Here's what you see when you use iPhone 13 with the iOS Simulator:

iPhone 13 (ios simulator)

2. Click  **Start debugging**.

Run and explore the app

The app should launch on your Android Emulator or iOS Simulator now. The UI is pretty straightforward. There's a dropdown button that allows users to choose a specific Google product. Once the **Generate haiku!** button is tapped, the Flutter app will send the built-in prompt to the Generative Language API endpoint, which will generate a haiku on the fly. The app will display the generated haiku in the Text widget below after receiving the response.



If you tap **Generate haiku!** now, nothing happens because the app can't communicate with the Generative Language API yet.

Step 1: Set up the Generative Language API access

Duration: 15:00

As of March 2023, the Generative Language API is under Private Preview. Please follow the documentation you have received to set up your access to Generative Language API and generate a working API key.

Step 2: Add the DropdownButton with a list of Google products

Duration: 5:00

Our goal is to generate haikus for Google products. At runtime, the app user can dynamically choose a product from a pre-populated list of product names. This is done through a DropdownButton.

- Add this code as the child of the SizedBox in the `_buildTopView()` function in the `step2/lib/app.dart` file:

TODO: add code

Step 3: Complete the Flutter app for Android and iOS

Duration: 5:00


The product name chosen in the DropdownButton from the last step is combined with a simple prompt template 'You are an awesome haiku writer. Write a cool long haiku about {product name}', and then it is sent to the Generative Language API endpoint to generate a haiku.

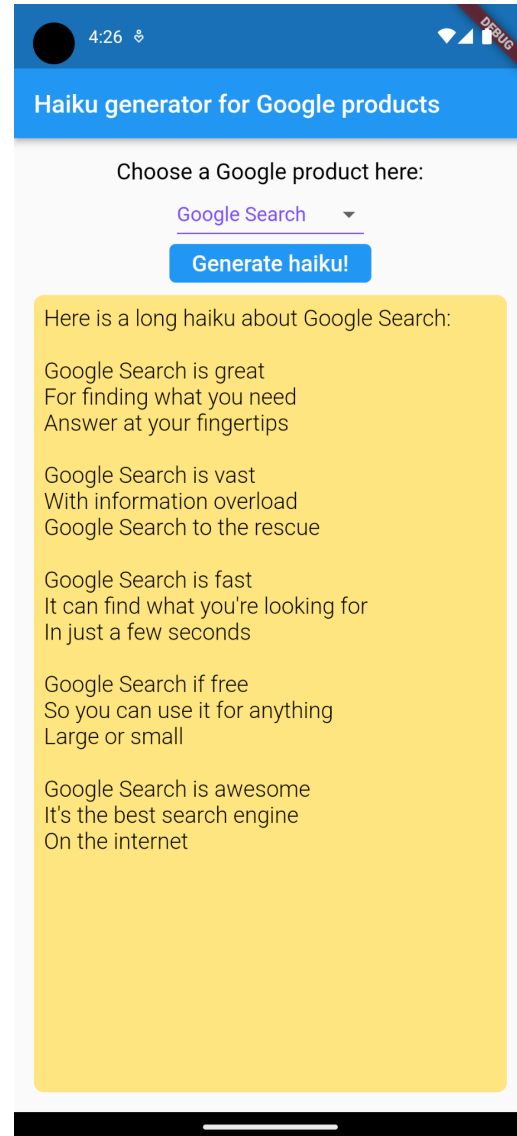
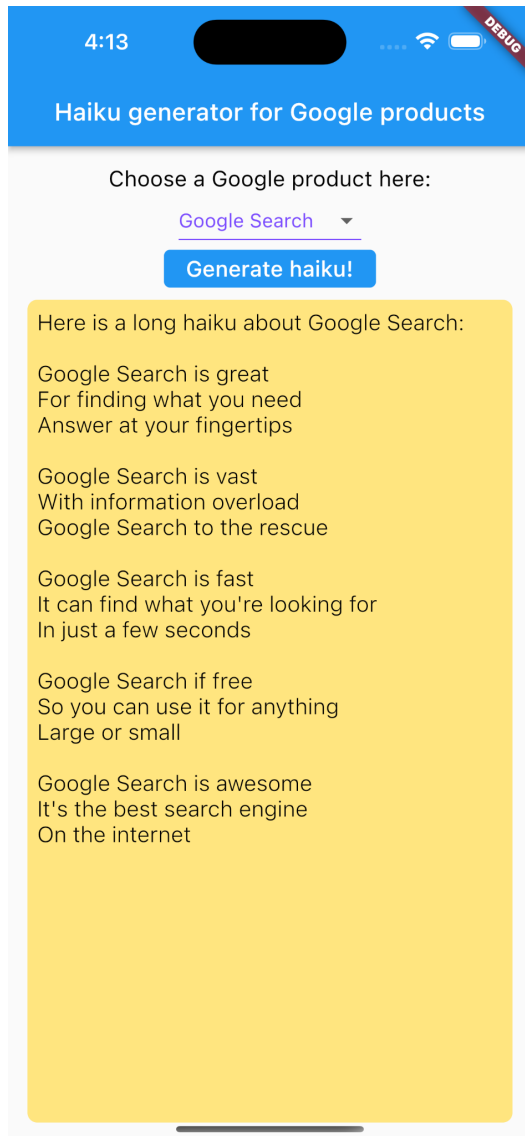
- Add this code to the `generateHaiku()` function in the `step2/lib/repositories/impl/poem_repository_impl.dart` file:

TODO: add code

The Text widget in the UI is wired up to render the generated haiku once the response is received.

Run it

1. Click  **Start debugging** and then wait for the app to load.
2. Choose a product and then tap **Generate haiku!**.



Step 4: Try other prompts

Duration: 10:00

Generative language models leverage the power of Large Language Models (LLMs) to generate text responses based on user prompts. These models have no restriction on what users can input. So you can easily change the prompt template and play with different ideas.

- Change 'haiku' to 'poem' or 'lyrics' in the prompt template of in the `step4/lib/repositories/impl/poem_repository_impl.dart` file

TODO: add code


We are not going to discuss in detail how to tune the prompt or come up with new prompt ideas in this codelab. But there is an entire field of [prompt engineering](#) that studies how to write best prompts for LLMs. Feel free to look for additional resources in this emergent domain.

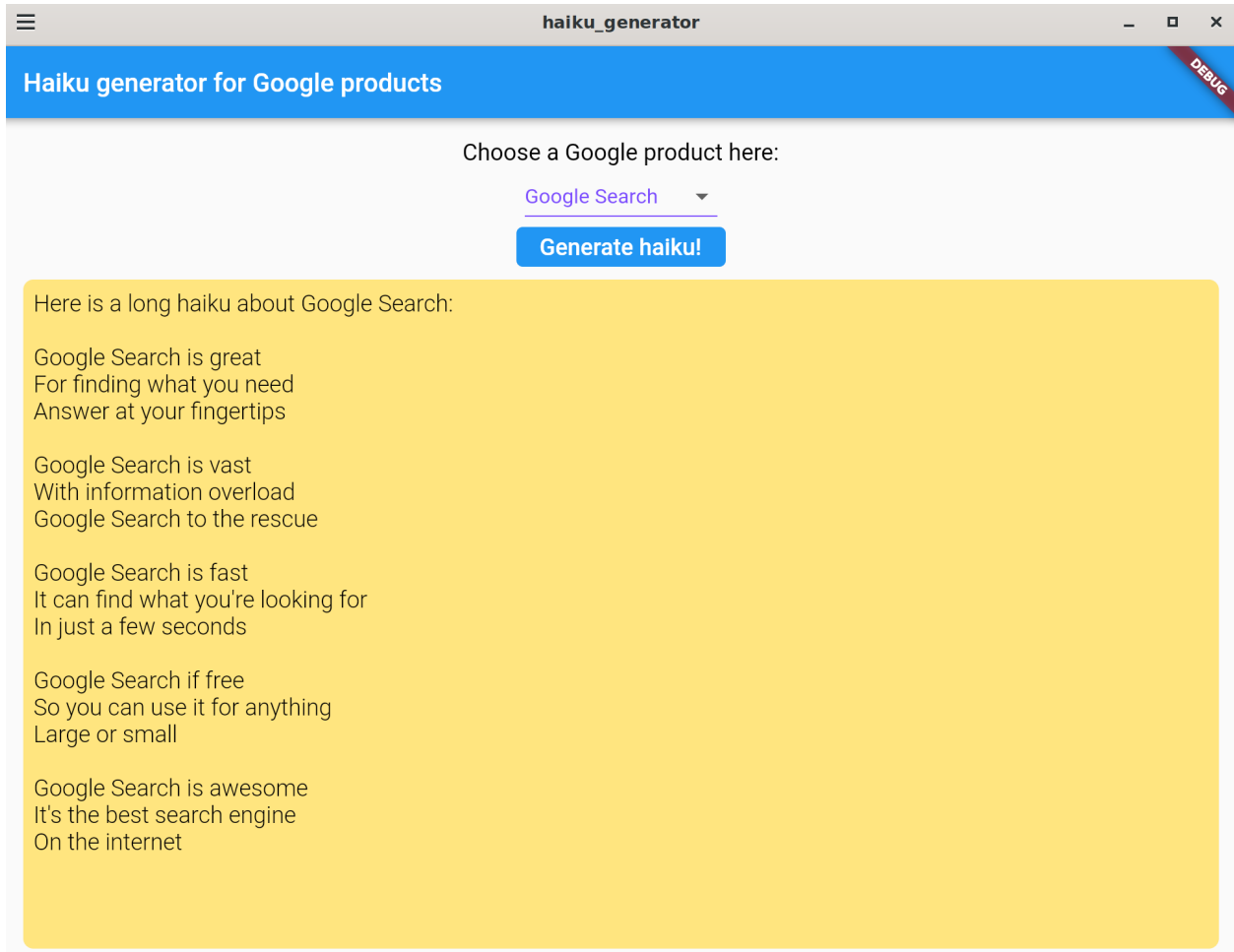
Step 5: Run the Flutter app on the desktop platforms

Duration: 10:00

In addition to Android and iOS, Flutter also supports desktop platforms including Linux, Mac and Windows.

Linux

1. Make sure the target device is set to **Linux (linux-x64)** in the status bar of VSCode.
2. Click  **Start debugging** and then wait for the app to load.
3. Choose a product and then tap **Generate haiku!**.




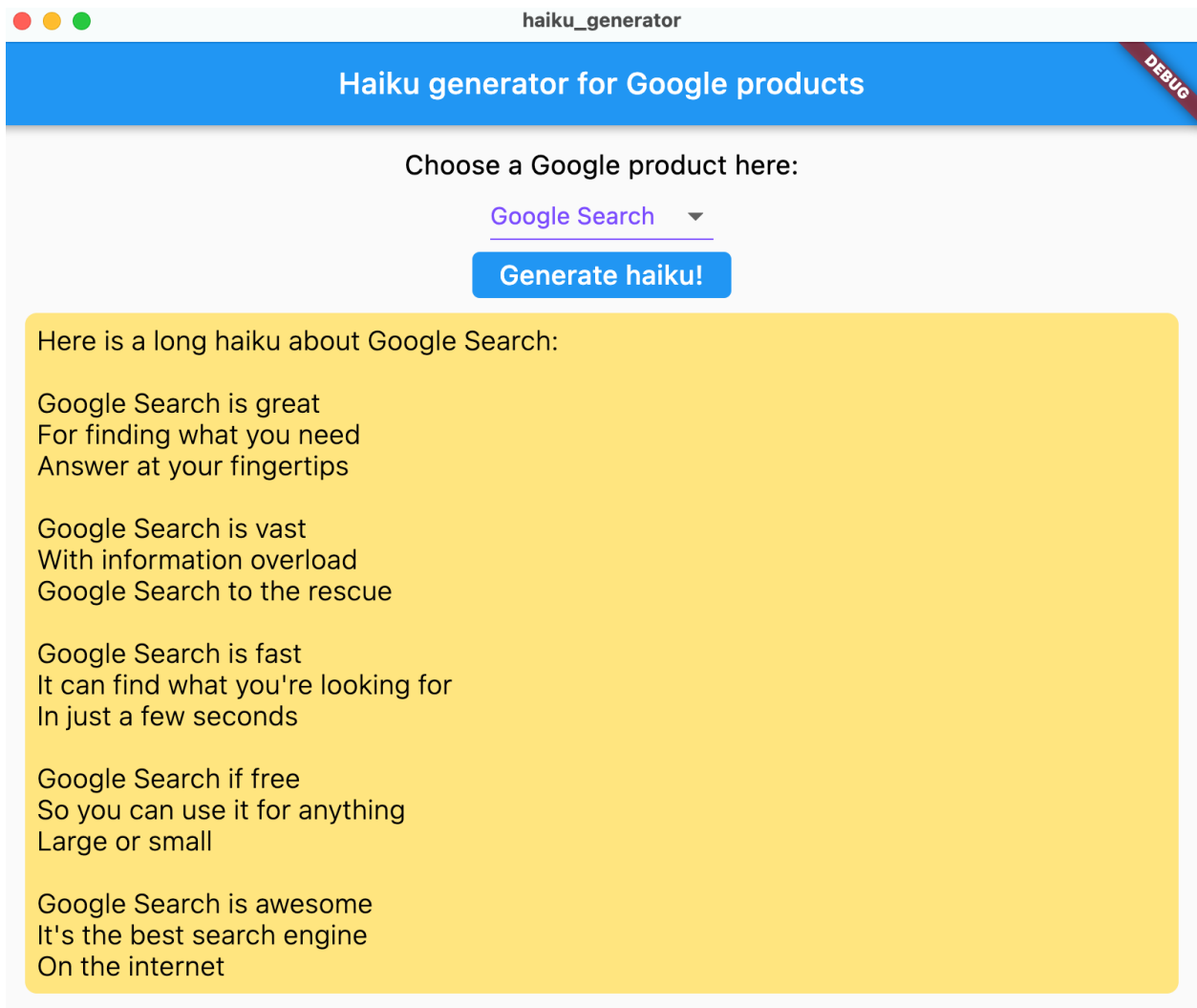
Mac

1. For Mac, you need to set up appropriate entitlements since the app will send HTTP requests to the backend. Please refer to [Entitlements and the App Sandbox](#) for more details.


Add this code to `step5/macOS/Runner/DebugProfile.entitlements` and `step5/macOS/Runner/Release.entitlements` respectively:

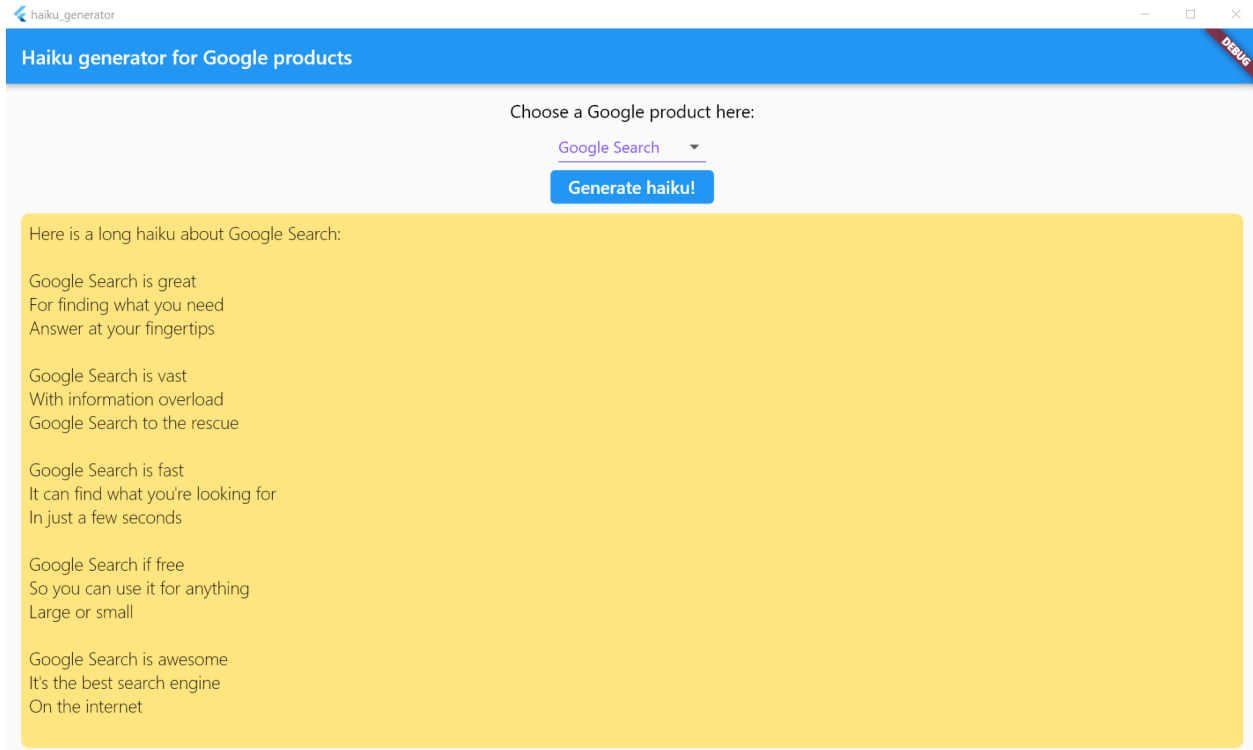
```
<key>com.apple.security.network.client</key>
<true/>
```

2. Make sure the target device is set to `macOS (darwin)` in the status bar of VSCode.
3. Click  **Start debugging** and then wait for the app to load.
4. Choose a product and then tap **Generate haiku!**.



Windows


1. Make sure the target device is set to **Windows (windows-x64)** in the status bar of VSCode.
2. Click  **Start debugging** and then wait for the app to load.
3. Choose a product and then tap **Generate haiku!**

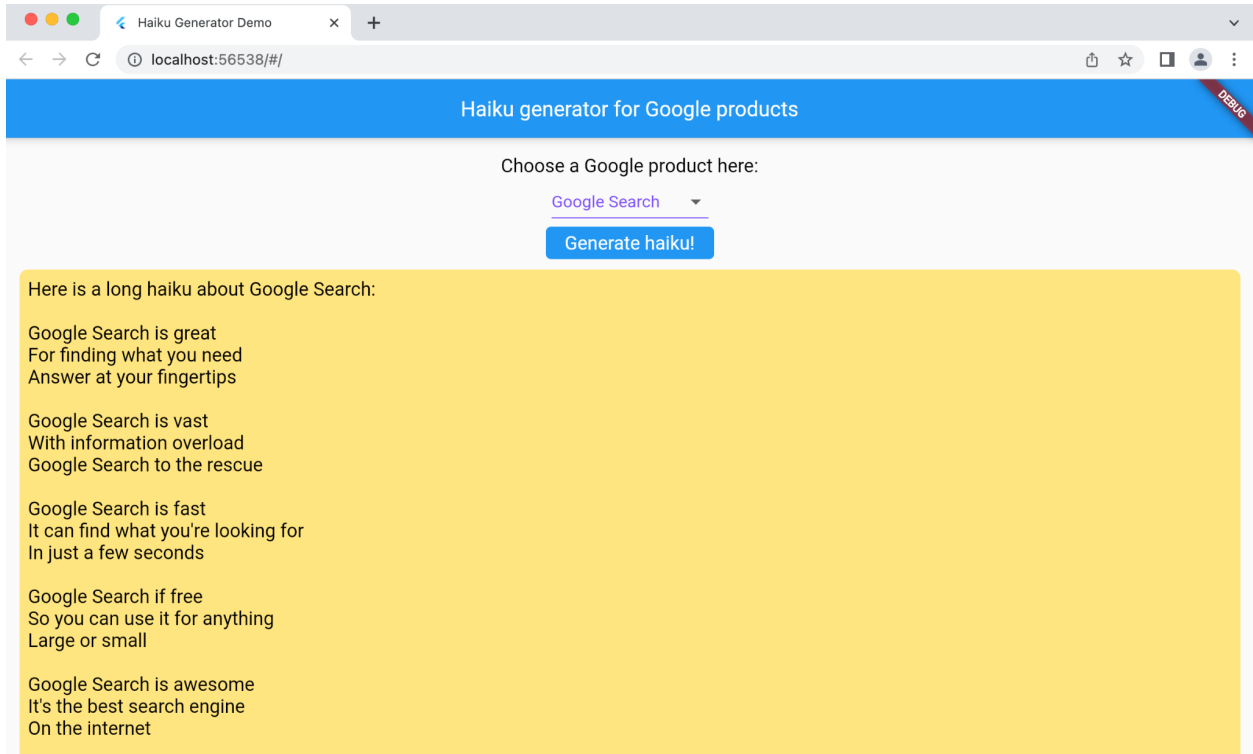


Step 5: Run the Flutter app on the web platform

Duration: 2:00

One more thing you can do is to add web support to the Flutter app. By default the web platform is automatically enabled for Flutter apps, so all you need to do is to launch it.

1. Make sure the target device is set to **Chrome (web-javascript)** in the status bar of VSCode.
2. Click  **Start debugging** and then wait for the app to load in the Chrome browser.
3. Choose a product and then tap **Generate haiku!**.



Congratulations

Duration: 1:00

You built a fullstack app to generate haikus for Google products!

Although the app only generates haikus for select Google products, you can easily change the prompt and generate your desired text. Now you know how to use the new Generative Language API, start building amazing apps with the power of LLMs!

Learn more

- [Generative Language API homepage](#)
- [Generative AI homepage](#)
- [Flutter home page](#)