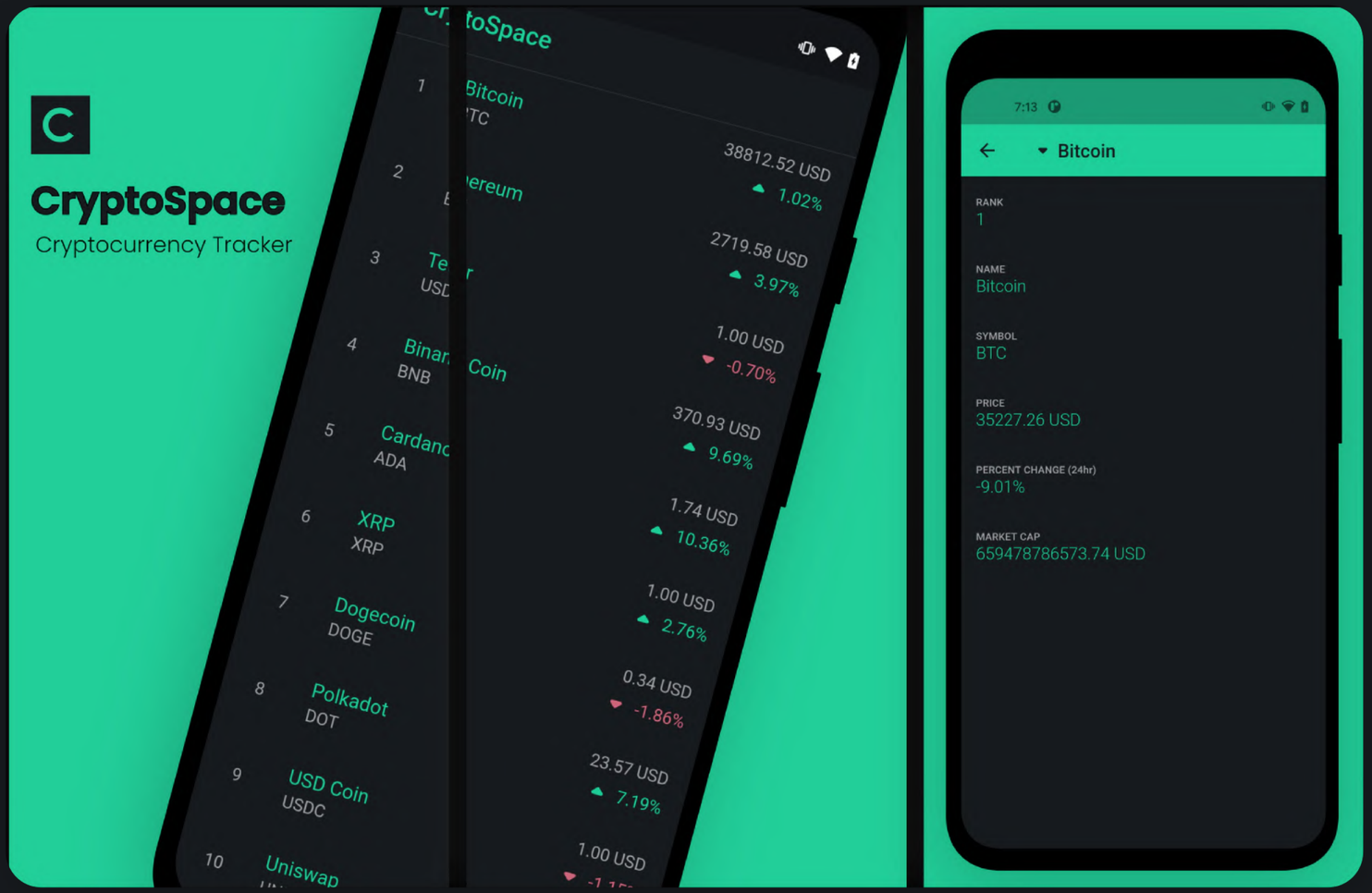
CryptoSpace

An introduction to stateful applications and API requests

Mohamed Ibrahim

What we will build

Cryptocurrency Tracker



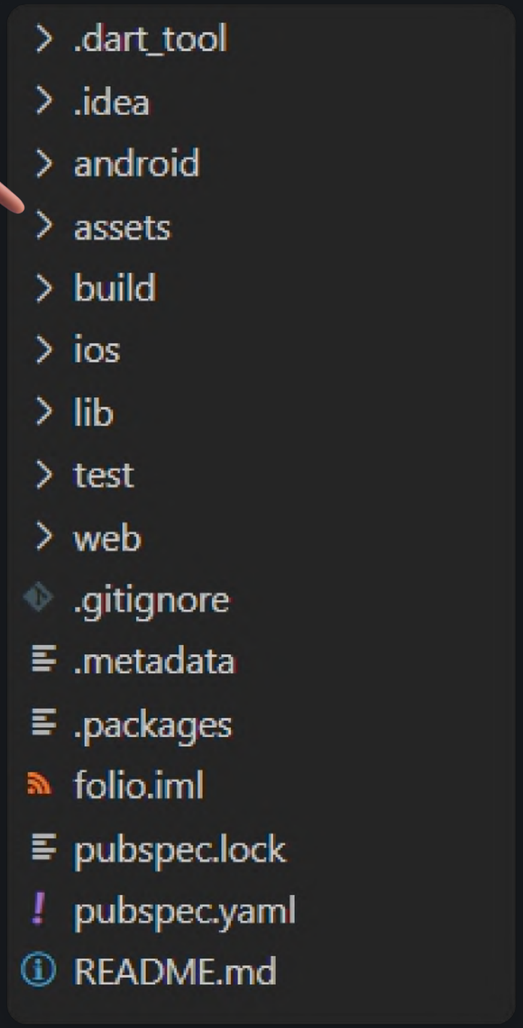
https://cryptospace.surge.sh

Create new project

```
$flutter create cryptospace
$cd cryptospace
$code . // open in vscode
```

Add *logo.png* to *assets/images*

logo.png



Add the necessary dependencies

Step 1

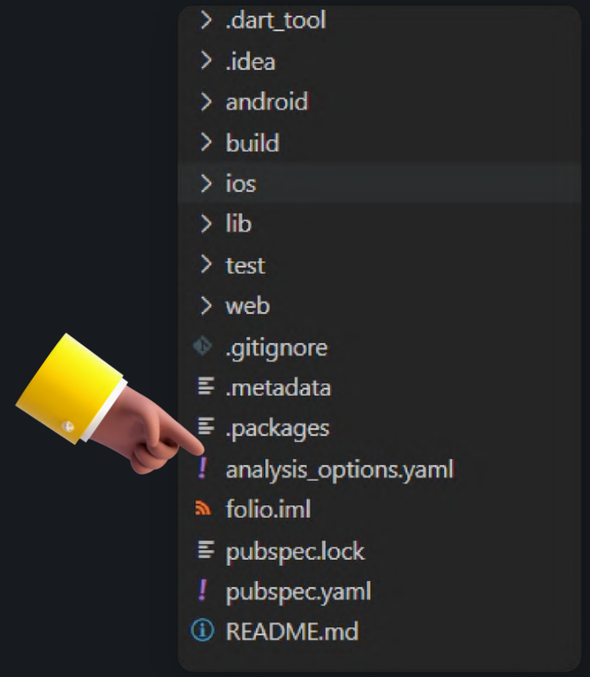
Add the lint package to the *dev_dependencies* in *pubspec.yaml*

```yaml
dependencies:
  flutter:
    sdk: flutter
  dio: ^4.0.0

dev_dependencies:
  flutter_launcher_icons: ^0.9.0
  flutter_native_splash: ^1.1.8+4
  flutter_test:
    sdk: flutter
  lint: ^1.0.0
```

Step 2

Create a new file in the root directory and add this line




```yaml
include: package:lint/analysis_options.yaml
```

Add configurations for *flutter_icons* and *flutter_native_splash*

```yaml
flutter_icons:
  android: "launcher_icon"
  ios: true
  remove_alpha_ios: true
  image_path: "assets/images/cryptoapp_logo.png"
  adaptive_icon_background: "assets/images/cryptoapp_logo.png"
  adaptive_icon_foreground: "assets/images/cryptoapp_logo.png"

flutter_native_splash:
  color: "#171A1E"
  image: "assets/images/cryptoapp_logo.png"
  android: true
  ios: true
```

Configure starter assets in *pubspec.yaml*

```yaml
flutter:
  uses-material-design: true
  # Add this
  assets:
    - assets/images/
```

Let's create a *constants.dart* file and add the colors and *API endpoints* we want to use in our app

```
final String baseUrl = 'api.coincap.io';
final String cryptosPath = '/v2/assets';

final kGreenColor= Color(0×FF21CE99);
final kGreyColor = Color(0×FFA0A2A4);
final kBlackColor = Color(0×FF171A1E);
final kRedColor = Color(0×FFCF6679);
```

Starter Code

```dart
import 'package:flutter/material.dart';

void main() {
  runApp(CryptoSpace());
}

class CryptoSpace extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      theme: ThemeData.dark(),
      home: Scaffold(),
    );
  }
}
```

Let's Add a *HomeScreen* widget and navigate to it

main.dart

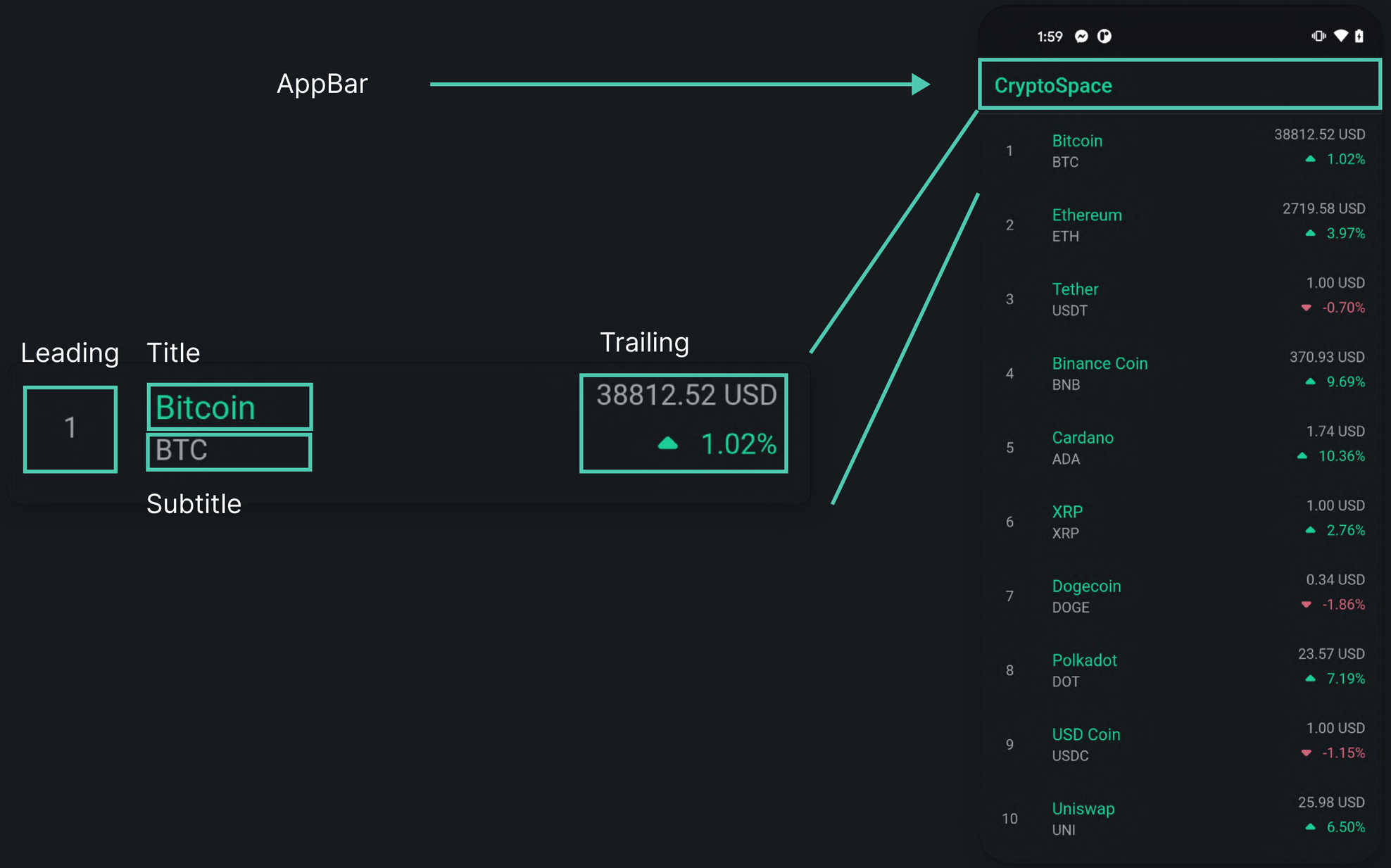
```
class CryptoSpace extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      theme: ThemeData.dark(),
      routes: {
        HomeScreen.id: (context) ⇒ HomeScreen(),
      },
      initialRoute: HomeScreen.id,
    );
  }
}
```

home_screen.dart

```
class HomeScreen extends StatefulWidget {
  static final String id = "home_screen";

  @override
  _HomeScreenState createState() ⇒ _HomeS
creenState();
}

class _HomeScreenState extends State<HomeS
creen> {
  @override
  Widget build(BuildContext context) {
    return Scaffold();
  }
}
```


AppBar
1:59
CryptoSpace
Leading
Title
Trailing
1
Bitcoin
BTC
38812.52 USD
1.02%
Subtitle
1 Bitcoin BTC 38812.52 USD 1.02%
2 Ethereum ETH 2719.58 USD 3.97%
3 Tether USDT 1.00 USD -0.70%
4 Binance Coin BNB 370.93 USD 9.69%
5 Cardano ADA 1.74 USD 10.36%
6 XRP XRP 1.00 USD 2.76%
7 Dogecoin DOGE 0.34 USD -1.86%
8 Polkadot DOT 23.57 USD 7.19%
9 USD Coin USDC 1.00 USD -1.15%
10 Uniswap UNI 25.98 USD 6.50%

AppBar

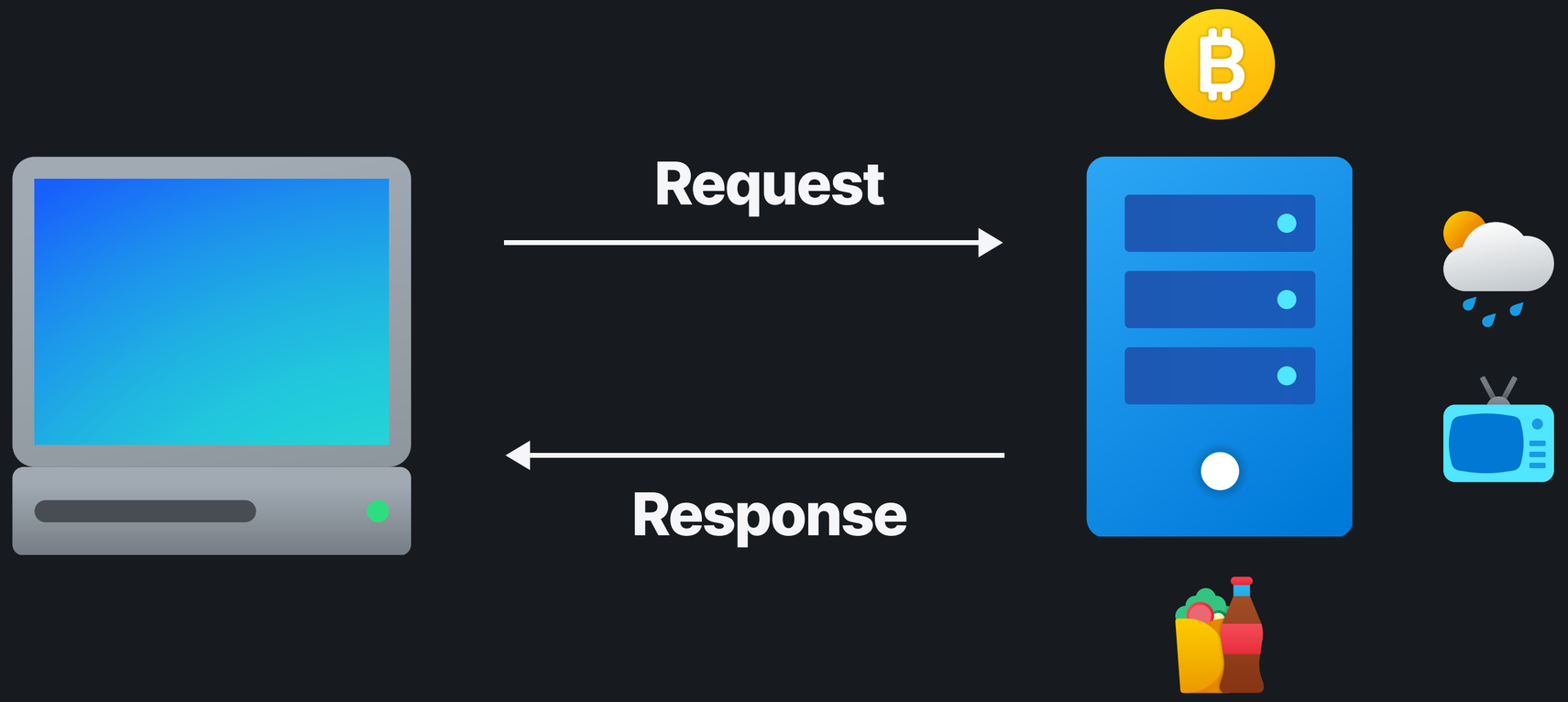
```
appBar: AppBar(
  title: Text(
    'CryptoSpace',
    style: TextStyle(
      color: kGreenColor,
    ),
  ),
  elevation: 0,
  backgroundColor: kBlackColor,
  centerTitle: false,
  bottom: PreferredSize(
    child: Container(
      color: Colors.grey[400],
      height: 0.1,
    ),
    preferredSize: Size.fromHeight(0.1),
  ),
),
```

Let's add a *backgroundColor* and a *ListView.builder*

```
return Scaffold(
  appBar: AppBar(
    ...
  ),
  backgroundColor: kBlackColor,
  body: ListView.builder(
    itemCount: 1,
    itemBuilder: (context, index) {
      return SizedBox.shrink();
    },
  ),
);
```

Introduction to *APIs*

Application Programming Interfaces



https://www.youtube.com/watch?v=s7wmiS2mSXY

https://github.com/public-apis/public-apis

HTTP Requests

Methods:

GET

POST

PUT

HEAD

DELETE

PATCH

OPTIONS

https://www.w3schools.com/tags/ref_httpmethods.asp

https://stackoverflow.com/questions/31089221/what-is-the-difference-between-put-post-and-patch

Get requests using the *Dio* package

```
import 'package:dio/dio.dart';

Future<void> getHttp() async {
  try {
    var response = await Dio().get('http://www.google.com');
    print(response);
  } catch (e) {
    print(e);
  }
}
```

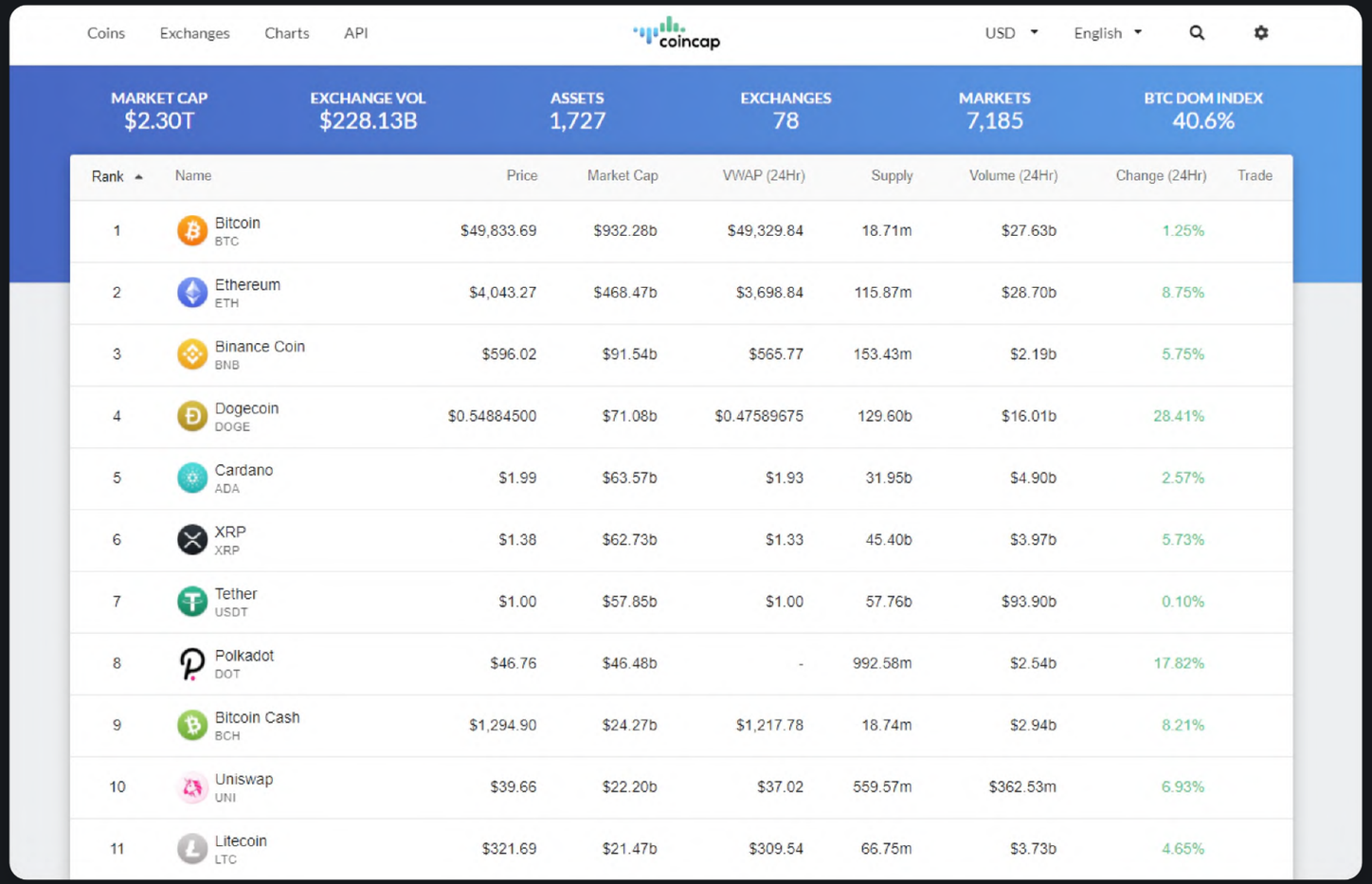
Post requests using the *Dio* package

```dart
import'package:dio/dio.dart';

Future<void> postHttp() async {
  try {
    final response = await Dio().post(
      '/test',
      data: {'id': 12, 'name': 'wendu'},
    );
    print(response);
  } catch (e) {
    print(e);
  }
}
```

Cryptocurrency API

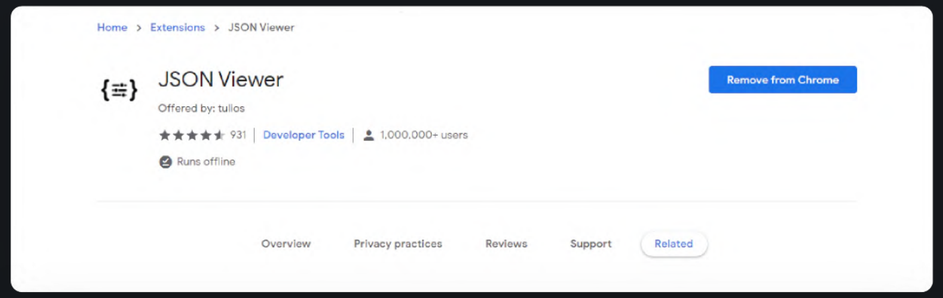
https://coincap.io/



Rank	Name	Price	Market Cap	VWAP (24Hr)	Supply	Volume (24Hr)	Change (24Hr)	Trade
1	Bitcoin BTC	$49,833.69	$932.28b	$49,329.84	18.71m	$27.63b	1.25%	
2	Ethereum ETH	$4,043.27	$468.47b	$3,698.84	115.87m	$28.70b	8.75%	
3	Binance Coin BNB	$596.02	$91.54b	$565.77	153.43m	$2.19b	5.75%	
4	Dogecoin DOGE	$0.54884500	$71.08b	$0.47589675	129.60b	$16.01b	28.41%	
5	Cardano ADA	$1.99	$63.57b	$1.93	31.95b	$4.90b	2.57%	
6	XRP XRP	$1.38	$62.73b	$1.33	45.40b	$3.97b	5.73%	
7	Tether USDT	$1.00	$57.85b	$1.00	57.76b	$93.90b	0.10%	
8	Polkadot DOT	$46.76	$46.48b	-	992.58m	$2.54b	17.82%	
9	Bitcoin Cash BCH	$1,294.90	$24.27b	$1,217.78	18.74m	$2.94b	8.21%	
10	Uniswap UNI	$39.66	$22.20b	$37.02	559.57m	$362.53m	6.93%	
11	Litecoin LTC	$321.69	$21.47b	$309.54	66.75m	$3.73b	4.65%	

JSON Viewer Extension

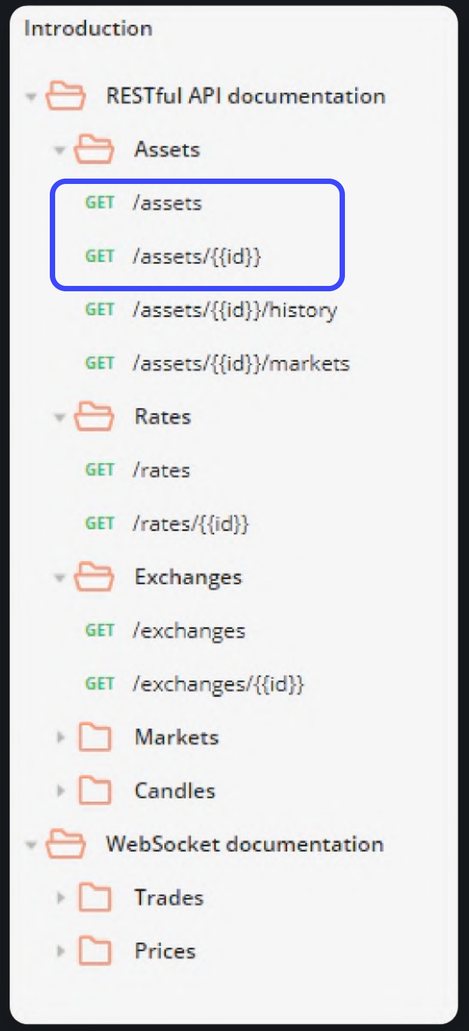
https://chrome.google.com/webstore/detail/json-viewer/gbmdgpbipfallnflgajpaliibnhdgobh/related




```
{
  "data": [
    {
      "id": "bitcoin",
      "rank": "1",
      "symbol": "BTC",
      "name": "Bitcoin",
      "supply": "18710137.0000000000000000",
      "maxSupply": "21000000.0000000000000000",
      "marketCapUsd": "932659234783.4076184556093757",
      "volumeUsd24Hr": "27686101201.2455493744966605",
      "priceUsd": "49847.8036148750604261",
      "changePercent24Hr": "1.2936240323643025",
      "vwap24Hr": "49329.8409663886143720",
      "explorer": "https://blockchain.info/"
    },
    {
      "id": "ethereum",
      "rank": "2",
      "symbol": "ETH",
```

Endpoints

https://docs.coincap.io/#ee0c0be6-513f-4466-bbb0-2016add462e9



https://api.coincap.io/v2/assets

- Provides an array of data regarding multiple assets


```
{
  "data": [
    {
      "id": "bitcoin",
      "rank": "1",
      "symbol": "BTC",
      "name": "Bitcoin",
      "supply": "18710137.0000000000000000",
      "maxSupply": "21000000.0000000000000000",
      "marketCapUsd": "932659234783.4076184556093757",
      "volumeUsd24Hr": "27686101201.2455493744966605",
      "priceUsd": "49847.8036148750604261",
      "changePercent24Hr": "1.2936240323643025",
      "vwap24Hr": "49329.8409663886143720",
      "explorer": "https://blockchain.info/"
    },
    {
      "id": "ethereum",
      "rank": "2",
      "symbol": "ETH",
```


http://api.coincap.io/v2/assets/bitcoin

- Provides data regarding a particular asset


```
{
  "data": {
    "id": "bitcoin",
    "rank": "1",
    "symbol": "BTC",
    "name": "Bitcoin",
    "supply": "18710137.0000000000000000",
    "maxSupply": "21000000.0000000000000000",
    "marketCapUsd": "934317822289.9926691899291408",
    "volumeUsd24Hr": "27722129406.0870488398675555",
    "priceUsd": "49936.4500799749712784",
    "changePercent24Hr": "1.0655903035828561",
    "vwap24Hr": "49329.8409663886143720",
    "explorer": "https://blockchain.info/"
  },
  "timestamp": 1621031479163
}
```

Create a data directory and add a *Crypto* model

```
class Crypto {
  String id;
  String name;
  String symbol;
  double changePercent24hr;
  double priceUsd;
  double marketCapUsd;
  int rank;

  Crypto({
    required this.id,
    required this.name,
    required this.symbol,
    required this.changePercent24hr,
    required this.priceUsd,
    required this.marketCapUsd,
    required this.rank,
  });
}
```

Let's add a *factory* constructor

Factory constructors

Use the factory keyword when implementing **a constructor that doesn't always create a new instance of its class**. For example, a factory constructor might return an instance from a cache, or it might return an instance of a subtype. Another use case for factory constructors is initializing a final variable using logic that can't be handled in the initializer list.

https://dart.dev/guides/language/language-tour#factory-constructors

```
factory Crypto.fromJson(Map<String, dynamic> json) {
  return Crypto(
    id: json['id'],
    name: json['name'],
    symbol: json['symbol'],
    explorer: json['explorer'],
    changePercent24hr: double.parse(json['changePercent24Hr']),
    priceUsd: double.parse(json['priceUsd']),
    marketCapUsd: double.parse(json['marketCapUsd']),
    rank: int.parse(json['rank']),
  );
}
```

Let's add the states that we will need

```
List<Crypto> _cryptos = [];
bool _loading = false;
```

Let's add a function that is called on initialization

```
Future<void> _fetchCryptos() async {}

@override
void initState() {
  super.initState();
  _fetchCryptos();
}
```

Let's add the logic for the *loading state* and for *error handling*

```
try {
  setState(() {
    _loading = true;
  });

} catch (e) {
  ScaffoldMessenger.of(context).showSnackBar(
    SnackBar(
      content: Text('Failed to fetch cryptos'),
    ),
  );
} finally {
  setState(() {
    _loading = false;
  });
}
```

Let's add the logic for the *loading state* and for *error handling*

```
try {
  setState(() {
    _loading = true;
  });

  final response = await Dio().get("https://$baseUrl$cryptosPath");
  List<Crypto> data = response.data['data']
    .map(
      (item) {
        return Crypto.fromJson(item);
      },
    )
    .toList()
    .cast<Crypto>();

  setState(() {
    _cryptos = data;
  });
} catch (e) { ... } finally { ... }
```

Let's Render the *ListTile* Widget

```
itemBuilder: (context, index) {
  final crypto = _cryptos[index];
  return ListTile(
    title: Text(crypto.name),
    subtitle: Text(crypto.symbol),
    leading: SizedBox(
        width: 30.0,
        child: Center(child: Text(crypto.rank.toString()))),
    trailing: Column(
      crossAxisAlignment: CrossAxisAlignment.end,
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        Text("${crypto.priceUsd.toStringAsFixed(2)} USD"),
        Text(
          crypto.changePercent24hr.toStringAsFixed(2),
        )
      ],
    ),
  );
},
```

Let's conditionally render a *CircularProgressIndicator*

```
body: _loading
    ? Container(
        color: kBlackColor,
        child: Center(
          child: CircularProgressIndicator(
            valueColor: AlwaysStoppedAnimation<Color>(
              kGreenColor,
            ),
          ),
        ),
      )
    : ListView.builder(
        ...
```

```
title: Text(
  crypto.name,
  style: TextStyle(color: kGreenColor),
),
...
subtitle: Text(
  crypto.symbol,
  style: TextStyle(color: kGreyColor),
),
...
leading: Text(
  crypto.rank.toString(),
  style: TextStyle(color: kGreyColor),
),
...
trailing: Text(
  "${crypto.priceUsd.toStringAsFixed(2)} USD",
  style: TextStyle(color: kGreyColor),
),
...
Text(
  crypto.changePercent24hr.toStringAsFixed(2),
  style: TextStyle(
    color: crypto.changePercent24hr > 0
        ? kGreenColor
        : kRedColor,
  ),
)
```

Let's create a helper method that returns an *upward/downward* arrow depending on the *percentage change*

```
import 'package:flutter/material.dart';

import 'constants.dart';

Icon getPrefixIcon(double percentChange, {Color? color}) {
  return percentChange ≤ 0
      ? Icon(
          Icons.arrow_drop_down_rounded,
          size: 32,
          color: color ?? kRedColor,
        )
      : Icon(
          Icons.arrow_drop_up_rounded,
          size: 32,
          color: color ?? kGreenColor,
        );
```

Add arrow icon next to percent

```
SizedBox(
  width: 100,
  child: Row(
    mainAxisAlignment: MainAxisAlignment.end,
    children: [
      _getPrefixIcon(crypto.changePercent24hr),
      Text(
        crypto.changePercent24hr.toStringAsFixed(
          2,
        ),
        style: TextStyle(
          color: crypto.changePercent24hr > 0
              ? kGreenColor
              : kRedColor,
        ),
      ),
    ],
  ),
),
```

Add internet permissions

AndroidManifest.xml

```
<uses-permission android:name="android.permission.INTERNET" />
```

macos/Runner/DebugProfile.entitlements

macos/Runner/Release.entitlements

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

Generate launcher icons and splash screen

```
$flutter pub run flutter_launcher_icons:main

$flutter pub run flutter_native_splash:create
```

Create web build

```
$flutter build web
```

Deploy app

```
$surge
```

1. Sign in
2. Type domain and submit

Build release versions for iOS, Android, and MacOS

```
$flutter run --release
```