



Google Maps

# Google Maps Platform Key Services

- **Maps:**

- Embed **interactive maps**, satellite imagery, and Street View in apps with full customization

- **Routes API** :

- Find the **best route** for driving, walking, biking, or public transport
- Compute **travel times** and **distances**
- Get real-time **traffic updates** for selected routes

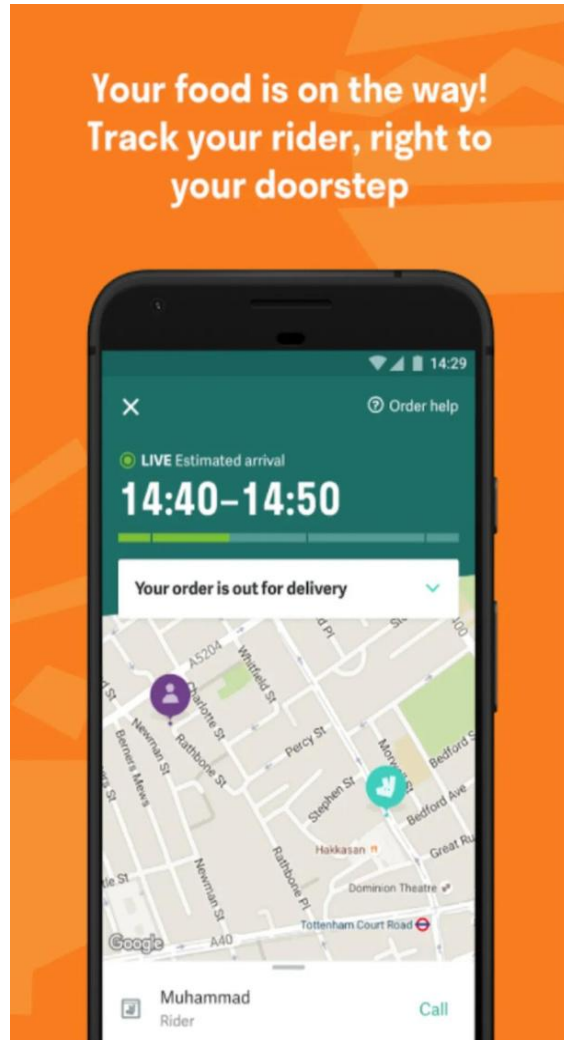
- **Places API** :

- Access details for millions of **points of interest** worldwide
- Includes names, addresses, photos, contact info, and reviews.

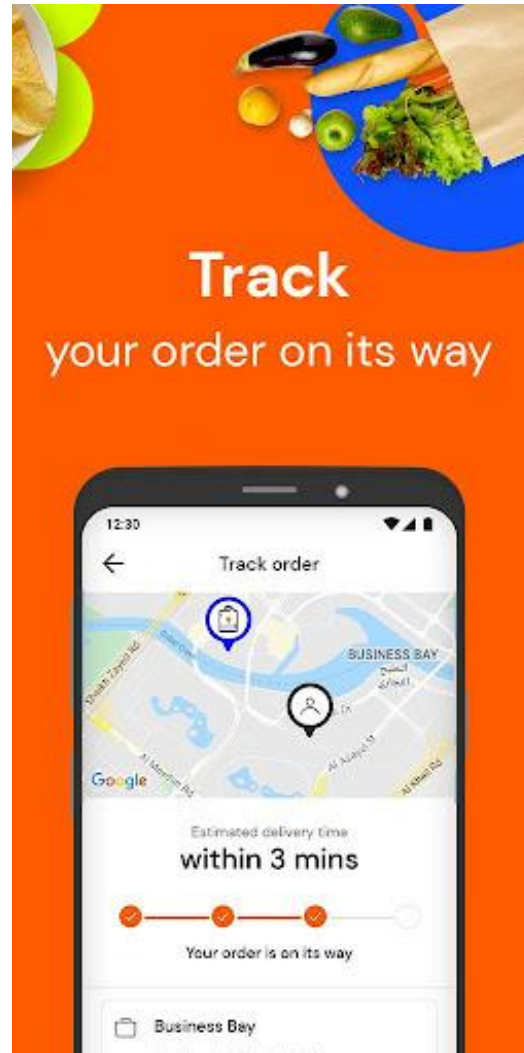


# What's driving growth of Map Apps?

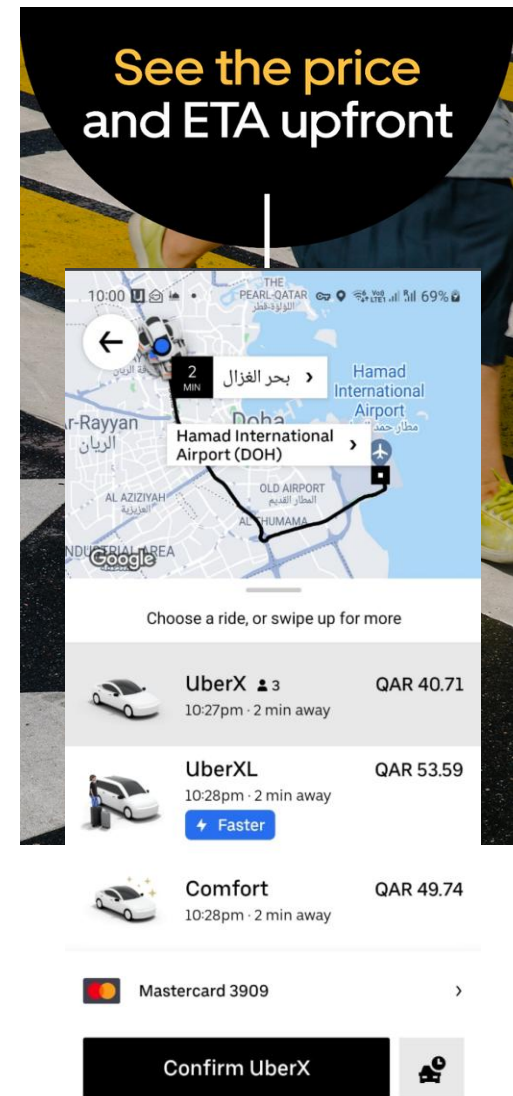
## Grocery delivery apps



## Food delivery apps



## Ride hailing apps



# Dependencies

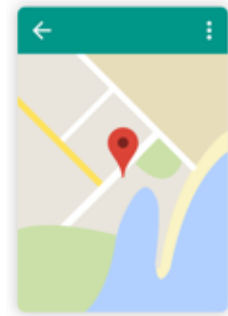
- Add this package to **pubspec.yaml**  
`google_maps_flutter`
- Get Google Maps API Key. See further details at this link  
<https://developers.google.com/maps/documentation/android-sdk/get-api-key>
- Add Google Maps API key to  
`\android\app\src\main\AndroidManifest.xml` file

# Typical Programming Tasks in Location-aware App

- **Visualize Data on a Map:** display custom markers, overlays, and shapes to represent data
- Get the device **geolocation** (latitude & longitude)
- **Geocoding:** convert an address into geographic coordinates
  - E.g., Qatar University → (25.377, 51.491)
- **Reverse Geocoding:** Convert coordinates into a human-readable address
  - E.g., (25.377, 51.491) → "Qatar University, Doha"
- **Location tracking:** continuously update user position (e.g., Uber ride tracking)
- **Geofencing:** trigger an action/notification when entering/exiting a defined area
  - E.g., Turn on lights when near home

# Display a Map

- Use **GoogleMap** widget to display an interactive Google Maps
- Customization Options:
  - **Add markers** to highlight locations
  - **Add overlays** (e.g., images or shapes over the map)
  - **Change zoom level** for different views (world → buildings)
  - **Handle events** such as Point of Interest (PoI) clicks



```
const LatLng quPosition = LatLng(25.377, 51.491);  
const double zoomLevel = 20.0; // Buildings
```

```
GoogleMap(  
  initialCameraPosition: CameraPosition(  
    target: quPosition,  
    zoom: zoomLevel,  
  ),  
  markers: {  
    Marker(  
      markerId: MarkerId('quMarker'),  
      position: quPosition,  
      infoWindow: InfoWindow(  
        title: "QU",  
        snippet: "Qatar University",  
      ),  
    ),  
  },  
)
```

# Zoom to a Location

- Set the initial view of the Map at a specific location and zoom level
  - **GoogleMap** widget **initialCameraPosition** parameter defines where the map should **look** and the **zoom level**
  - Look at a particular **geo coordinates** and change the zoom level

- Zoom levels:

- 1: World
- 5: Continent
- 10: City
- 15: Streets
- 20: Buildings

```
// Define initial position and zoom level
final LatLng quPosition = const
  LatLng(25.377, 51.491);
final double zoomLevel = 20.0;
```

```
GoogleMap(
  initialCameraPosition:
    CameraPosition(
      target: quPosition,
      zoom: zoomLevel,
    )
)
```

# Add Marker

- Marker identify a specific location on the map using geographic coordinates
  - Use the **Marker** widget to add a marker with title and snippet (short description)
  - When clicked, an **InfoWindow** displays the marker's details

```
// A Snippet is a text displayed below the title
final snippetText = "Lat: ${quLocation.latitude},
                    Long: ${quLocation.longitude}"
final marker = Marker(
  markerId: MarkerId('quMarker'),
  position: quLocation,
  infoWindow: InfoWindow(
    title = "Qatar University",
    snippet = snippetText
  ),
)
```



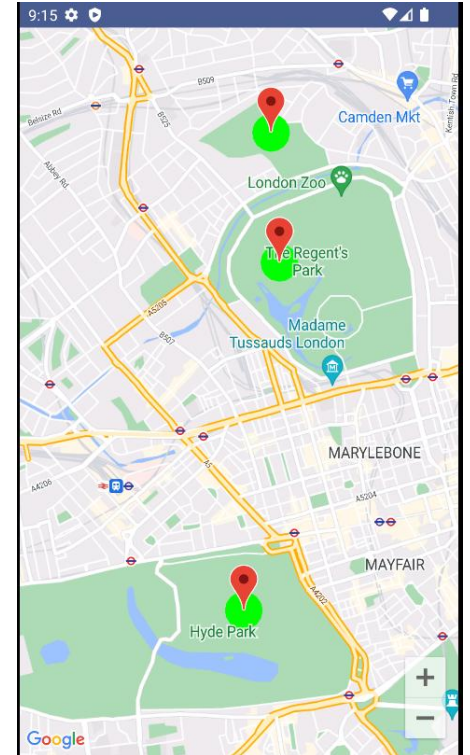
# Map UI Customization

- Customize the look and feel of Google Maps in your app. Key Options:
  - **mapType** could be `normal`, `satellite`, `terrain`, `hybrid`
  - UI Controls: Enable/disable zoom controls, map toolbar, and myLocation button

```
GoogleMap(  
    cameraPositionState = cameraPositionState,  
    mapType: MapType.hybrid,  
    myLocationEnabled: true,  
    mapToolbarEnabled: true,  
    zoomControlsEnabled: true,  
)
```

# Drawing Shapes

```
GoogleMap(...  
  polygons: {  
    Polygon(  
      polygonId: PolygonId('quPolygon'),  
      points: [  
        LatLng(25.376, 51.490),  
        LatLng(25.378, 51.490),  
        LatLng(25.378, 51.492),  
        LatLng(25.376, 51.492),  
      ],  
      fillColor: Colors.green.withOpacity(0.5),  
      strokeWidth: 2,  
      strokeColor: Colors.green,  
    ),  
  },  
  circles: {  
    Circle(  
      circleId: CircleId('quCircle'),  
      center: quPosition,  
      radius: 500,  
      fillColor: Colors.blue.withOpacity(0.5),  
      strokeWidth: 2,  
      strokeColor: Colors.blue,  
    ),  
  },  
),
```



# Marker Clustering in Google Maps

- **Purpose:** Improves map readability when displaying many markers => Enhances performance and user experience
  - Use [google maps cluster manager](#) package to manage marker grouping dynamically
  - At **high zoom levels**, individual markers are shown
  - At **low zoom levels**, markers automatically group into clusters

```
GoogleMap(  
  initialCameraPosition: const CameraPosition(  
    // Center on Doha  
    target: LatLng(25.2854, 51.5310), zoom: 11 ),  
    // Display clustered markers  
    markers: _markers,  
    // These callbacks allow ClusterManager to  
    // recalculate clusters when map moves  
    onCameraMove: _clusterManager.onCameraMove,  
    onCameraIdle: _clusterManager.updateMap,  
    onMapCreated: (controller) {  
      _clusterManager.setMapId(controller.mapId);  
    })  
)
```



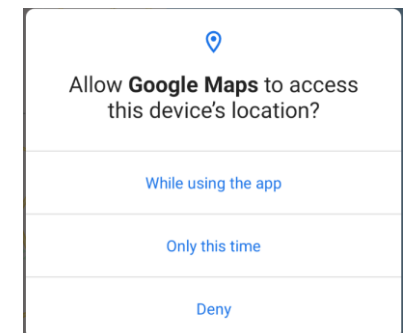
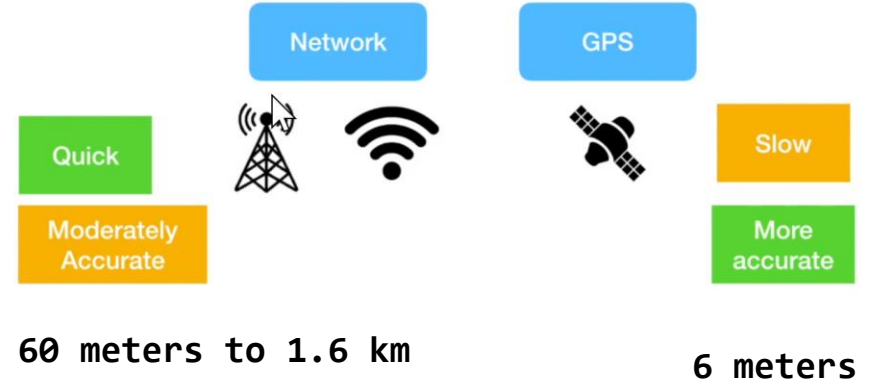
# Get User Location

- Retrieve the device's last known location
  - Uses **Wi-Fi**, **cellular towers**, and/or **GPS** via the [Geolocator package](#)

- Key Steps:**

- Request runtime permission** to access location
- If granted, **Fetch location** using `Geolocator.getCurrentPosition()` with desired accuracy

```
// Request location permission
LocationPermission permission = await Geolocator.requestPermission();
if (permission == LocationPermission.denied ||
    permission == LocationPermission.deniedForever) {
  print("Location permissions are denied.");
  return;
}
// Get the last known location
final position = await Geolocator.getCurrentPosition(
  locationSettings: const LocationSettings(
    accuracy: LocationAccuracy.high,
  ));
```



# Geocoding

- **Geocoding** Converts an address or location name (e.g., street address) into geographic coordinates (**latitude, longitude**). Usage:
  - Place markers on a map
  - Zoom to a specific location
  - Enable location-based features in apps

Hamad International Airport @ Lat:  
25.2608759 & Long: 51.6138416999999995

```
/*  
    Geocoding = converting an address or location name (like a street address) into  
    geographic coordinates (lat, lng) using geocoding package  
*/  
List<Location> locations = await locationFromAddress(address);  
if (locations.isNotEmpty) {  
    final loc = locations.first;  
    return GeoLocation(loc.latitude, loc.longitude);  
} else {  
    return null; // No result  
}
```

# Reverse Geocoding

Lat: 25.2609 & Long: 51.6138 is Hamad International Airport, Doha, Qatar

- **Reverse geocoding** Converts geographic coordinates (latitude, longitude) into a human-readable address. Usage:
  - Display location names on maps
  - Provide address details for GPS coordinates
  - Enable location-based features in apps

```
/*  
    Reverse geocoding = converting geographic coordinates (lat, lng)  
    into a human-readable location address using geocoding package  
*/  
// Get a list of places from latitude and longitude  
List<Placemark> placemarks = await  
    placemarkFromCoordinates(lat, lng);  
if (placemarks.isNotEmpty) {  
    final place = placemarks.first;  
    String name = place.name ?? "";  
    String city = place.locality ?? "";  
    String country = place.country ?? "";  
}
```

# Resources

- Adding Google Maps to a Flutter app
  - <https://codelabs.developers.google.com/codelabs/google-maps-in-flutter>
- A Comprehensive Guide to Using Google Maps in Flutter
  - <https://medium.com/@samra.sajjad0001/a-comprehensive-guide-to-using-google-maps-in-flutter-3fbc0f7d469e>