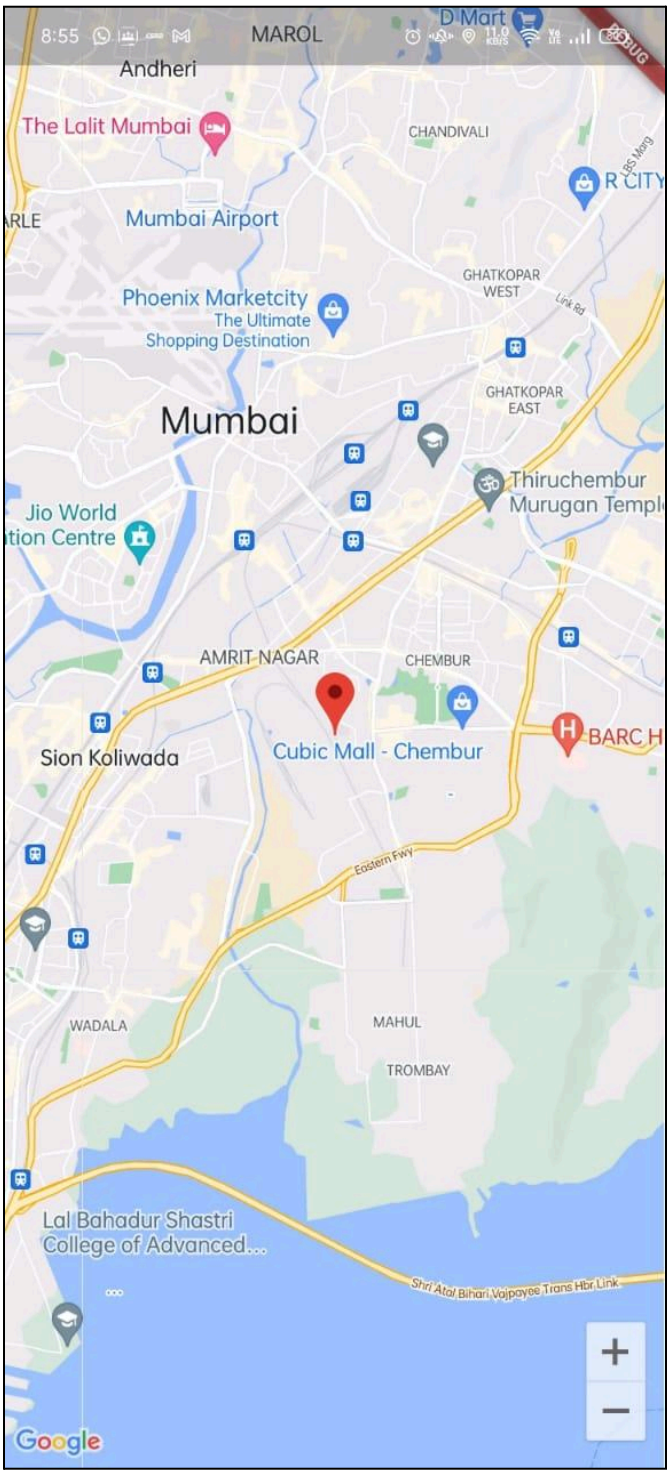
EXP 3: To include icons, images, fonts in Flutter app.

Aim: To include icons, images, fonts in Flutter app

Theory:



Code:

```
import 'dart:async';

import 'package:flutter/material.dart';
import 'package:google_maps_flutter/google_maps_flutter.dart';
import 'package:flutter_polyline_points/flutter_polyline_points.dart';
import 'package:google_maps_yt/consts.dart';
import 'package:location/location.dart';

class MapPage extends StatefulWidget {
  const MapPage({super.key});

  @override
  State<MapPage> createState() => _MapPageState();
}

class _MapPageState extends State<MapPage> {
  Location _locationController = new Location();

  final Completer<GoogleMapController> _mapController =
      Completer<GoogleMapController>();

  static const LatLng _pGooglePlex = LatLng(37.4223, -122.0848);
  static const LatLng _pApplePark = LatLng(37.3346, -122.0090);
  LatLng? _currentP = null;

  Map<PolylineId, Polyline> polylines = {};

  @override
  void initState() {
    super.initState();
    getLocationUpdates().then(
      (_) => {
        getPolylinePoints().then((coordinates) => {
              generatePolyLineFromPoints(coordinates),
            }),
      },
    );
  }

  @override
```

```
  Widget build(BuildContext context) {
    return Scaffold(
      body: _currentP == null
          ? const Center(
              child: Text("Loading..."),
            )
          : GoogleMap(
              onMapCreated: ((GoogleMapController controller) =>
                  _mapController.complete(controller)),
              initialCameraPosition: CameraPosition(
                target: _pGooglePlex,
                zoom: 13,
              ),
              markers: {
                Marker(
                  markerId: MarkerId("_currentLocation"),
                  icon: BitmapDescriptor.defaultMarker,
                  position: _currentP!,
                ),
                Marker(
                    markerId: MarkerId("_sourceLocation"),
                    icon: BitmapDescriptor.defaultMarker,
                    position: _pGooglePlex),
                Marker(
                    markerId: MarkerId("_destionationLocation"),
                    icon: BitmapDescriptor.defaultMarker,
                    position: _pApplePark)
              },
              polylines: Set<Polyline>.of(polylines.values),
            ),
    );
  }

  Future<void> _cameraToPosition(LatLng pos) async {
    final GoogleMapController controller = await _mapController.future;
    CameraPosition _newCameraPosition = CameraPosition(
      target: pos,
      zoom: 13,
    );
    await controller.animateCamera(
      CameraUpdate.newCameraPosition(_newCameraPosition),
    );
```

```
  }

  Future<void> getLocationUpdates() async {
   bool _serviceEnabled;
   PermissionStatus _permissionGranted;

   _serviceEnabled = await _locationController.serviceEnabled();
   if (_serviceEnabled) {
    _serviceEnabled = await _locationController.requestService();
   } else {
    return;
   }

   _permissionGranted = await _locationController.hasPermission();
   if (_permissionGranted == PermissionStatus.denied) {
    _permissionGranted = await _locationController.requestPermission();
    if (_permissionGranted != PermissionStatus.granted) {
     return;
    }
   }

   _locationController.onLocationChanged
       .listen((LocationData currentLocation) {
    if (currentLocation.latitude != null &&
        currentLocation.longitude != null) {
     setState(() {
      _currentP =
          LatLng(currentLocation.latitude!, currentLocation.longitude!);
      _cameraToPosition(_currentP!);
     });
    }
   });
  }

  Future<List<LatLng>> getPolylinePoints() async {
   List<LatLng> polylineCoordinates = [];
   PolylinePoints polylinePoints = PolylinePoints();
   PolylineResult result = await polylinePoints.getRouteBetweenCoordinates(
    GOOGLE_MAPS_API_KEY,
    PointLatLng(_pGooglePlex.latitude, _pGooglePlex.longitude),
    PointLatLng(_pApplePark.latitude, _pApplePark.longitude),
    travelMode: TravelMode.driving,
```

```
    );
    if (result.points.isNotEmpty) {
      result.points.forEach((PointLatLng point) {
        polylineCoordinates.add(LatLng(point.latitude, point.longitude));
      });
    } else {
      print(result.errorMessage);
    }
    return polylineCoordinates;
  }

  void generatePolyLineFromPoints(List<LatLng> polylineCoordinates) async {
    PolylineId id = PolylineId("poly");
    Polyline polyline = Polyline(
        polylineId: id,
        color: Colors.black,
        points: polylineCoordinates,
        width: 8);
    setState(() {
      polylines[id] = polyline;
    });
  }
}
```

Conclusion:

In this Experiment we learnt how to include icons, images and fonts in Flutter. Also overcame the different problems and difficulties faced while doing this experiment.