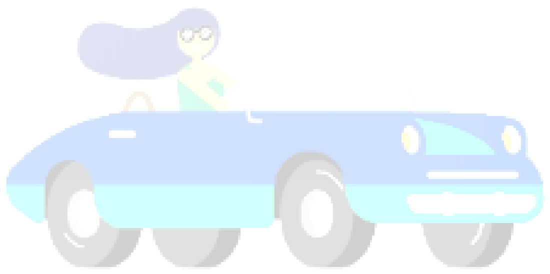
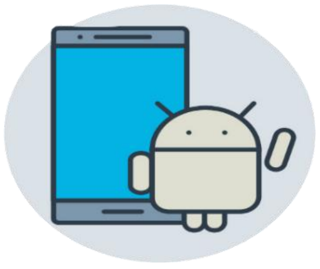
Android Developer Nanodegree

Capstone Project



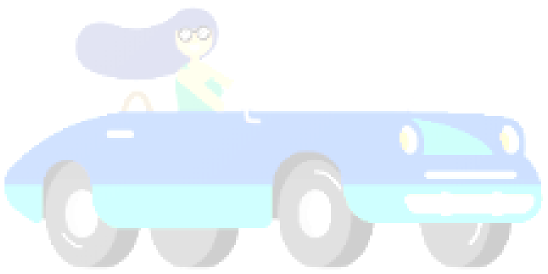
FMW: Free my way

Our website



Abhinandan Trilokia

Contents



Requirements:

- This app is written solely in the Java Programming Language.
- This App utilizes stable release versions of all libraries, Gradle, and Android Studio.

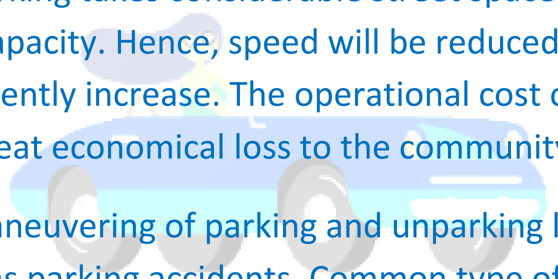
Description

This app is created as a part of **Google India Challenge Scholarship 2018** under Google Udacity Android Developer Nanodegree program.

Special thanks to **Google** & **Udacity** for providing this opportunity. This app is inspired from the **Google Play social impact apps** through this app *we want to solve one the of most common issue faced by all of us i.e. congestion or blocked road because of wrong parked vehicle.*

Wrong parking is the root cause of road congestion, accidents, traffic jams, environmental pollution etc.

Congestion: Wrong parking takes considerable street space leading to the lowering of the road capacity. Hence, speed will be reduced, journey time and delay will also subsequently increase. The operational cost of the vehicle increases leading to great economical loss to the community.



Accidents: Careless maneuvering of parking and unparking leads to accidents which are referred to as parking accidents. Common type of parking accidents occurs while driving out a car from the parking area, careless opening of the doors of parked cars, and while bringing in the vehicle to the parking lot for parking.

Environmental pollution: They also cause pollution to the environment because stopping and starting of vehicles while parking and unparking results in noise and fumes. They also affect the aesthetic beauty of the buildings because cars parked at every available space creates a feeling that building rises from a plinth of cars.

Obstruction to emergency operations: Parked vehicles may obstruct the movement of firefighting vehicles. Sometimes they block access to hydrants and access to buildings.

With the help of our app, we can together solve these problems

Because we believe that transportation solution should be safe, rapid, comfortable, convenient, economical, and eco-friendly.

We take special care while developing this app. No comprises were made while considering the design, user experience of our app

This app does not require any personal info. All it requires an active internet connection to work. Simply search the vehicle number and app will automatically place call to the other person (free of cost) and you can then ask them to aside their vehicle, all this without disclosing your person info.

Simple, easy & convenient.

Intended User

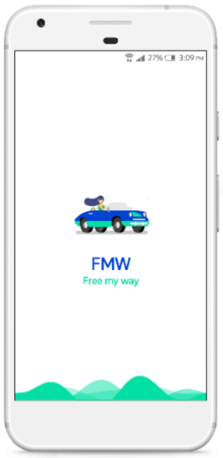
This app is intended for all. Everyone can use it. It's simple user interface make it suitable for all types of user.

Features

- Clean & minimal UI: Featuring material design by Google nothing else.
- No more sign in/sign up: No phone number or email address are required.
- Search vehicle: Manually by entering the vehicle number or using scan and go feature powered by Google mobile vision.
- Top quality VOIP service: Provided by Sinch.
- Uninterrupted background service: Through Google Firebase push message service.
- We respect our user privacy: No useless app permission.
- Guest mode: Share your vehicle with other using Google nearby API.
- Battery friendly: No background process.
- Free for all: No service charges

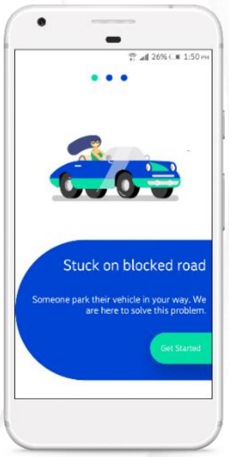
User Interface Mock

Splash Screen



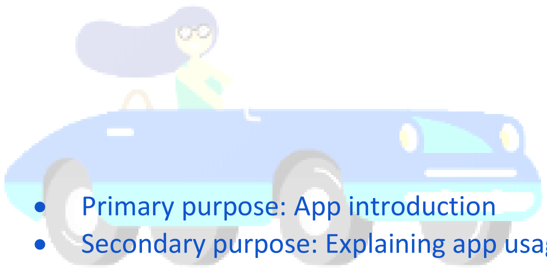
- Primary purpose: Initialize app core components.
- Secondary purpose: Welcoming user with smooth animation.

Get started



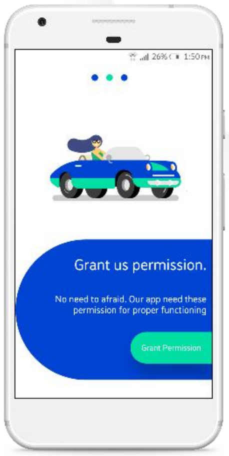
Stuck on blocked road

Someone park their vehicle in your way. We are here to solve this problem.



- Primary purpose: App introduction
- Secondary purpose: Explaining app usage

App permissions

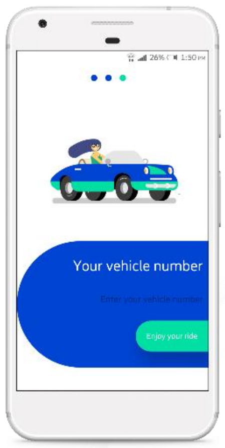


Grant us permission.

No need to afraid. Our app need these permission for proper functioning

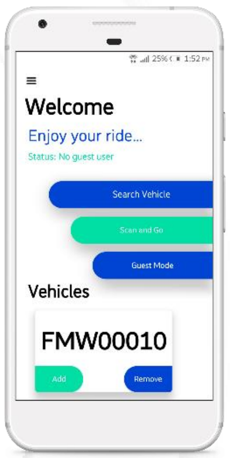
- Primary purpose: Demonstrate android runtime permission using Dexter.
- Secondary purpose: Register the user with UUID.

Final setup



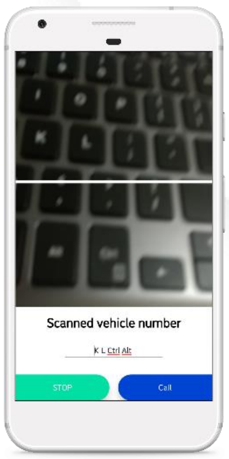
- Primary purpose: Register the user with UUID.
- Secondary purpose: Verify server response & save the data using shared preference

Home screen



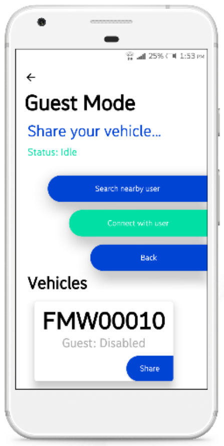
- Primary purpose: Provide various options like search vehicle, scan & go, guest mode etc.
- Secondary purpose: Add or remove vehicles, update item in card view

Scan & go



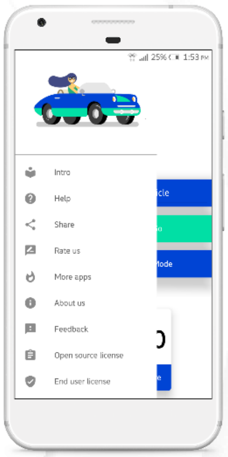
- Primary purpose: Ease the way of manually vehicle searching.
- Secondary purpose: Demonstrate the use of Google mobile vision API

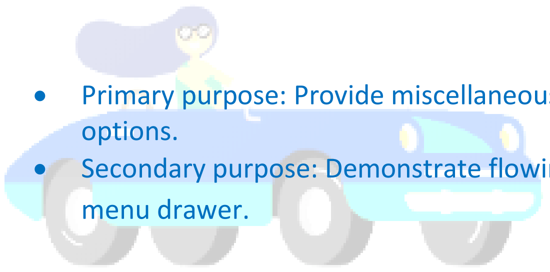
Guest mode



- Primary purpose: Vehicle sharing.
- Secondary purpose: Demonstrate the use of Google nearby API.

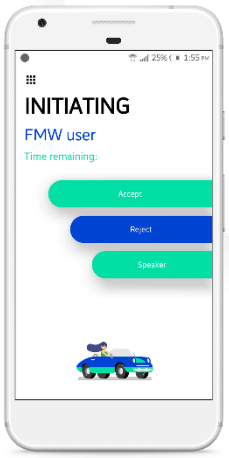
Navigation drawer





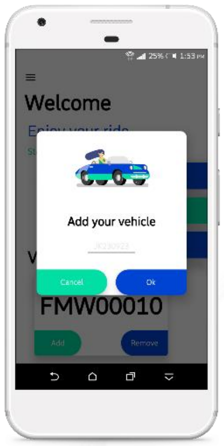
- Primary purpose: Provide miscellaneous options.
- Secondary purpose: Demonstrate flowing menu drawer.

Call screen



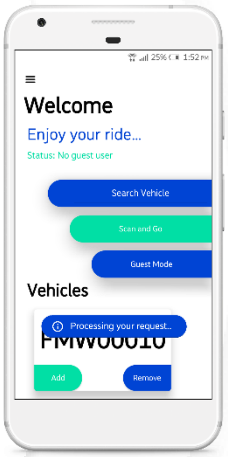
- Primary purpose: Provide the options for accepting & rejecting call
  - Secondary purpose: Demonstrate the Google Firebase push message service.

Add or remove vehicles



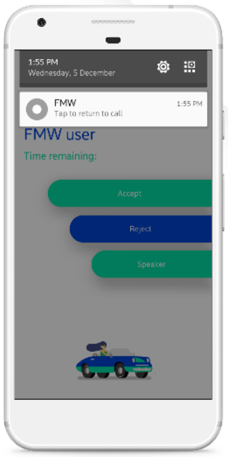
- Primary purpose: Add vehicle and update item in card view.
- Secondary purpose: Demonstrate the implementation of custom android dialog.

Notify user



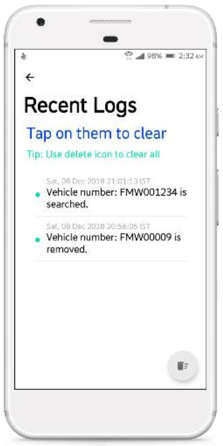
- Primary purpose: Inform user about the various tasks
- Secondary purpose: Demonstrate the implementation of custom toast.

App notification



- Primary purpose: Inform the user about the service.
- Secondary purpose: Provide a mechanism to get back to the call screen after pressing home key.

Recent logs


Recent Logs

Tap on them to clear

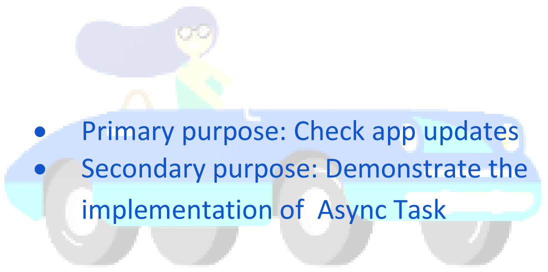
Tip: Use delete icon to clear all

Sat, 08 Dec 2018 21:01:13 IST
- Vehicle number: FMW001234 is searched.

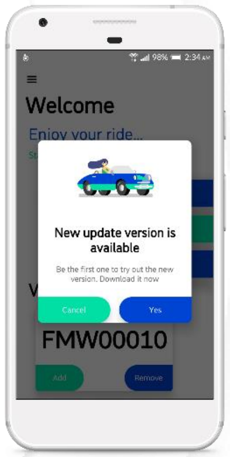
Sat, 08 Dec 2018 20:56:05 IST
- Vehicle number: FMW00009 is removed.

- Primary purpose: Display the recent activities of the user.
- Secondary purpose: Demonstrate the use of data persistence (Content Provider).

Update check



- Primary purpose: Check app updates
- Secondary purpose: Demonstrate the implementation of Async Task

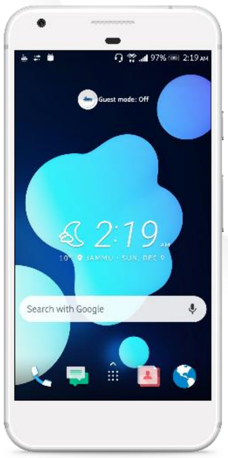

Welcome

Enjoy your ride...

New update version is available

Be the first one to try out the new version. Download it now

App widget



- Primary purpose: App widget to check guest mode status.
- Secondary purpose: Demonstrate the widget usage.

Key Considerations

Handling data persistence:

To ensure our app meets user's expectations with respect to their UI state, and provides a smooth, snappy UI especially after frequently occurring configuration changes, like rotation. We are using SQLite Database for saving all the recent activities of the user in a database , basic operations like Insert & delete are used for it. In the end data is loaded in the recyclerview. **OnSaveInstanceState** is also used for saving text status and **SharedPreferences** is used as caching the Json response(vehicle llist) for avoiding lag while loading data into the UI.

Edge or corner cases:

1) Receiving call even when app is not running. To solve this, we can choose background or foreground service but we decided to go with firebase push message service because of following reasons

   a) No effect of Android 8.0 background process limit.
   b) No effect of Android 8.0 battery optimization.
   c) No need of Android 8.0 notification for foreground service.

2) Status bar issue. Our app uses white theme which hides the status bar icons so we use system UI flags to force light status bar in main activity & full screen flag in scan & go activity.

3) Saving vehicle list retrieved over the API using shared preference for offline purpose.

4) Foreground notification for call activity so that even if user pressed the home key. The notification will inform user about it until the calls ends.

5) Handle the situation when try to calls himself/herself by check the UUID before making call.

6) Handle call screen while device is in sleep state. We use window manager flags to solve it

Libraries:

- **Butterknife**: For binding Android views.
- **Dexter**: For requesting permissions at runtime.
- **Firebase**: For push messaging service.
- **Flowing drawer**: For navigation drawer with animation.
- **Gif drawable**: For displaying gif.
- **Google Play services**: GCM (for push message), Mobile Vision OCR (for scan feature), Nearby (for guest mode).
- **Retrofit**: A type-safe HTTP client for Android.
- **Sinch**: For VOIP feature.
- **Support Library Packages**: AppCompat, Design, Cardview.

Implement Google Play Services or other external services.

The core component our app is VOIP, we are using **Sinch VOIP API** for implementing VOIP feature in our app. For receiving call even in background, instead of using background service. We choose **Firebase** push message to start foreground service only when it is required for specify amount of time to save power & memory consumption. **Google mobile vision API** for OCR and **Google nearby API** for receiving guest user UUID.

Implementations of various tasks

Task 1: Integrating Sinch

- Registration at Sinch.com for the Voice API then register our user with UUID. In our app. We use IMEI for it. This is because Sinch requires phone permission and we use the same permission for getting device IMEI (used as UUID).

Task 2: Android runtime permission

- Using Dexter, we implemented android runtime permission.

Task 3: Setting up app backend

- Backend is developed by Dheeraj Kumar Chalotra. (He is Google India Challenge 2018 Recipient: Mobile web specialist)
- In our backend, Laravel for PHP web framework and MySql for database is used.
- In our database UUID is primary key & vehicle numbers are associated with it.
- In search vehicle option, our app sends the POST request and receives UUID in response (associated with that particular vehicle number) after then app make a VOIP call (using Sinch API) to other user (associated with that UUID).

Task 4: Integrating GCM & Firebase

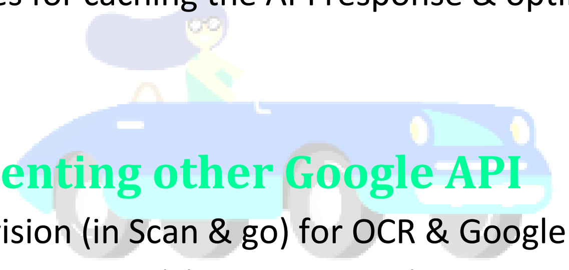
- For receiving call even in the background. Instead of running service in the background, we choose firebase push message to start the Sinch client and call required methods for establishing call between our user.

Task 5: Implementing essential methods

- With the help of retrofit. We send various POST & GET request for registering our users, add/removing their vehicles numbers, setting vehicle sharing/un sharing etc.

Task 6: Implementing ContentProvider ,OnSaveInstanceState & SharePreferences

- ContentProvider for saving the activities of the user like when & which vehicle is added or removed etc.
- OnSaveInstanceState for saving state (of fields like status) and restore it after app rotation.
- SharePreferences for caching the API response & optimizing unnecessary API calls.



Task 7: Implementing other Google API

- Google mobile vision (in Scan & go) for OCR & Google nearby API (cloud.console.google.com) (in guest mode) for receiving guest user UUID.

Task 8: Polishing the app UI & add support for accessibility

- Design the app UI for multiple device and states like landscape.
- Use contentDescription attribute.

Task 9: App testing, use Gradle task & Sign apk

- Testing various app components & edge case before publish it on Google Play.
- Build the app using Gradle task & use the signing configuration

Thanks. Hope you liked it.

Download it from Play Store.
Don't forget to visit our website. It is designed by Dheeraj Kumar Chalotra.