1. Product.

Product Description. Carpooling is a shared travel practice where several people organize to use the same private vehicle to go to similar destinations, whether for daily commutes, such as to work or university, or for longer trips between cities.

FRIMUV is a carpooling service designed for students of the Faculty of Mathematics (FMAT) who live in the same area and need transportation to or from the faculty. The idea is that students who own private vehicles can coordinate with other students from the faculty who live in the same area so they can travel together both to and from campus.

Users/Clients. FMAT students, both those who own a private car and those who do not have a vehicle and need transportation.

Value Proposition. Our project not only seeks to provide students with a faster and more efficient way to commute to and from the faculty, but also contributes to reducing traffic in the city of Mérida, decreasing the carbon footprint, and strengthening the university community through a collaborative solution that optimizes time, resources, and trust among students. To achieve this, safety is essential to ensure confidence, protection, and peace of mind for all users. The measures that will help us accomplish this include requiring that all students—both drivers and passengers—must have a valid institutional email to register, along with an up-to-date profile picture.

2. Requirements

Functional Requirements

1. **Trip Request:** Passengers will be able to request a ride through the application.
2. **Boarding Rides:** Passengers will request the driver to pick them up at a common point, to avoid deviating from the trip route.
3. **Registration:** Users will be able to register in the application using their institutional email or Institutional ID card.
4. **Login:** Users will be able to log in to the app in different ways (email, phone number, password, etc.), as well as log out securely.
5. **Route Display:** When starting the trip, passengers will see their location to the destination in real time.
6. **Trip Cancellation:** Passengers must have the option to cancel the ride, get off in a safe area, and not reach the destination if they choose.
7. **Available Compensation:** Although trips are free for passengers, the driver must have the option to receive voluntary compensation for their service from the passengers.

8. **Payment Method:** Passengers who wish to compensate for the service may do so in the quickest and most convenient way, such as bank transfers, PayPal, or cash.
9. **Shared Service:** More than one user can join the trip in the same vehicle through the common meting point.

Non-Functional Requirements

Product Requirements.

Efficiency Requirements:

- **Performance:** Ability to handle multiple users simultaneously without performance loss.
- **User location:** The location on the map must update in real time with a maximum delay of 10 seconds.
- **Portability:** It must be compatible with the most widely used mobile operating systems (iOS and Android).
- **Storage:** The application must have a total size of less than 100 MB.

Security Requirements:

- **Driver safety:** Allow drivers to record trips for their safety in case of any incident or accident.
- **User PIN:** Require all users to enter a PIN to start a trip.
- **Privacy:** Location data must only be shared when the user is on an active trip.

Usability Requirements.

- **Usability:** The interface must be intuitive so that a new user can easily create a trip.
- **Ease of use:** Passengers will not require any training to use the application's features.

External Requirements

Legal Requirements:

- **Car license plates:** Require drivers to provide their car's license plates at the time of registration in the application.
- **Driver's license:** Drivers must have a valid driver's license to provide the service.
- **Protection/Security:** Require official identification (INE or driver's license) from users when registering in the application.

Regulatory requirements:

- **Car insurance:** Drivers providing the transportation service are required to have valid car insurance.

Organizational Requirements

Development requirements:

- The code will be developed using Kotlin, an open-source programming language for application development.
- Visual Studio Code will be used to efficiently test and fix the code.
- We will store all code in a GitHub repository.
- Only allow automobiles: For greater user safety, drivers may only use automobiles to provide the service.
- Data deletion: Allow users to delete their account and all associated data.
- Route storage: The application may store information about vehicle routes.

Prioritization.

Must Have

- Trip request: users must be able to request a ride in the application.
- Boarding rides: passengers must gather at a common point to avoid deviating from the route.
- Login: secure login via email, phone number, or password, and secure logout.
- Show route: passengers must be able to see their real-time location until the destination.
- Trip cancellation: option to cancel the trip and get off at a safe area.
- Shared service: multiple users can join the same ride if the pick-up point is common.
- Performance: handle multiple users simultaneously without performance loss.
- User location: real-time location updates with a maximum delay of 10 seconds.
- Portability: compatible with iOS and Android.
- Car license plates: drivers must register their vehicle license plates.
- Driver's license: drivers must have a valid driver's license.
- Protection/Security: users must provide official identification when registering.
- Only allow automobiles: only cars will be allowed to provide the service.
- Data deletion: allow users to delete their account and associated data.

Should Have

- Driver safety: option to record trips for the driver's safety.
- User PIN: all users must enter a PIN to start a trip.
- Privacy: location data must only be shared during active trips.
- Usability: the interface must be intuitive so that a new user can easily create a trip.
- Ease of use: passengers will not require training to use the app.
- Development requirements: use of Kotlin, Visual Studio Code, and GitHub (affects the dev team, not end users directly).
- Route storage: store information about vehicle routes.

Could Have

- Storage: the application must have a total size of less than 100 MB.
- Car insurance: drivers must have valid car insurance.

Won't Have

- Optional remuneration: option for the driver to receive voluntary compensation.
- Payment methods: compensation via bank transfer, PayPal, or cash.

Artifacts

Use Case: Ride Request

Overview: This use case describes how a user interacts with the application to request a ride to a specific destination. The process includes everything from selecting the destination to confirming the ride and the driver's arrival.

Primary Actor: User (Person requesting the ride).

Application System: FRIMUV Application.

Goal in Context: To request a ride to the desired destination.

Preconditions:

The user must have the FRIMUV application installed.

The user must have internet access.

The user must have an active account in the FRIMUV application, be willing to log in, or create a new one.

The system must be ready to enter a destination and display ride options published by drivers.

Trigger: The user needs to go to a specific place, requires transportation, and opens the application to request a ride.

Flow of Events:

1. The user (passenger) opens the application on their mobile device. If they are not logged in, they enter their user information (name or email and password) to access their account.
2. The user enters the destination they want to go to. The destination entered by the user is compared with the destinations offered by drivers in the application. If the user's destination matches one or more of the destinations offered by drivers, the application presents them as available ride options; the options display the driver's name, destination, meeting point, departure time, and the number of

users sharing the ride. If there are no matches, the screen shows that no options are available.

3. The user selects the most convenient option and confirms the ride request. The system sends the ride request to the driver, who decides whether to accept or reject it.
4. The user receives a notification that the ride has been accepted, along with the driver's information (name, photo, license plate number, vehicle model, and features) and the information of other passengers sharing the ride (if any).
5. The user goes to the established meeting point and waits for the driver to arrive. As the driver approaches, the application shows their real-time location on the map, along with the estimated time of arrival.
6. Once the user reaches the destination, the driver completes the ride. If desired, the user can provide compensation for the service via the application or in cash.
7. After the ride is completed, the application asks the user to rate the driver and the service. The user can also share feedback on their experience.

Scenario:

- User: Opens the application on their mobile device.
- User: Enters the destination they want to go to.
- User: Selects their preferred ride option and confirms the request.
- User: Receives the "service accepted" notification.
- User: Goes to the meeting point and waits for the driver's arrival.
- User: Pays the driver (optional).
- User: Rates the driver and the service.

Exceptions:

- If the user is not logged in: they must access their account before continuing.
- If the user has no internet access: the application will show an error message and prompt them to connect before proceeding.
- If the driver does not accept the ride request: the system will notify the user.
- If the user cancels the ride before the driver arrives: a notification is displayed.

- If payment through the application fails: the system notifies the user and asks them to choose another payment method.

Priority: Essential, as it is the core functionality upon which the service provided by the application is built.

Availability: At all times, since users may require the service at any hour of the day.

Frequency of Use: High, as users may request rides multiple times a day depending on their needs, especially during peak hours.

Actor's Channel: Through the application interface.

Secondary Actors: Driver (Person who accepts or rejects the ride request).

Secondary Actors' Channels: Through the driver interface of the application.

Diagram of Use Cases.