**X-Car**

Project Plan

Version 1.0

10/19/2023

Authors

Ana Carolina Coelho Robl

Arthur Nicola

Baptiste Turpin

Loïc Thomas

Nicolas Siard

Pierre Bonnecaze

Vialan Louis

Well-John Lu

Abstract

This project plan describes the development and implementation of an assistive vehicle, designed to meet the pressing needs of a diverse public that faces mobility challenges and difficulties transporting heavy objects. The X-Car project was motivated by the growing demand for solutions that improve the quality of life and autonomy of people with physical disabilities, the elderly and individuals with temporarily reduced mobility.

Moreover, there was a need of more explanation of the vehicle’s behavior in order to increase the trust of the user and to pass certification constraints for more critical systems

In this context, the X-Car will offer three distinct operating modes - manual, tracking and autonomous - to accommodate a variety of mobility situations.

It will also feature a web interface for real-time monitoring and a verbal communication system, allowing users to understand and trust the X-Car's actions.

Keywords

Assistive vehicle, real-time monitoring, explainability, embedded system, tracking, obstacle detection, object recognition, ADAS

Table of Contents

[Abstract](#_heading=h.1fob9te)

[Keywords](#_heading=h.3znysh7)

[Table of Contents](#_heading=h.2et92p0)

[Executive summary](#_heading=h.2xcytpi)

[Product description](#_heading=h.3dy6vkm)

[Stakeholders](#_heading=h.1t3h5sf)

[Release vision](#_heading=h.4d34og8)

[Features](#_heading=h.1ci93xb)

[Operational](#_heading=h.17dp8vu) requirements

Validation plan

[Team](#_heading=h.3rdcrjn)

[Milestones and deliverables](#_heading=h.lnxbz9)

[Project planning](#_heading=h.35nkun2)

[First sprint](#_heading=h.1ksv4uv)

[Product backlog](#_heading=h.2jxsxqh)

[Schedule control](#_heading=h.z337ya)

[Risk management](#_heading=h.3j2qqm3)

[Method](#_heading=h.1y810tw)

[Risk and actions](#_heading=h.4i7ojhp)

1. Executive summary
   1. Product description

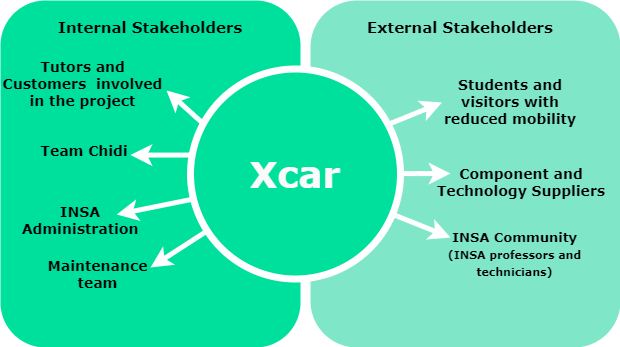
The X-Car project aims to develop an assistive vehicle to cater to the needs of visitors and students at INSA who face mobility challenges and require assistance in transporting heavy or bulky items. These individuals require support to transport their belongings safely and efficiently while navigating the campus.

To address this problem, we will develop a specialized assistive vehicle equipped with advanced features and three different modes:

* Manual mode, in which the vehicle will be controlled by a remote (Xbox controller or Smartphone). This will allow the user to have direct input and control over the vehicle, ensuring it moves as per their specific requirements in a contained environment.
* Tracking mode, in which the vehicle will autonomously follow a person, making it easier for the user to navigate without constantly operating the remote control.
* Autonomous mode which will allow the vehicle to join the user’s position from a parking spot. The vehicle will be programmed to never collide with any obstacles or individuals. This will be vital for the safety of the users and others on the INSA campus.

A Human Machine Interface (HMI) such as a web interface or a mobile app will provide users with real-time monitoring of the vehicle's operational modes. This interface will allow users to keep track of the vehicle's movements, providing a sense of control and oversight. In addition, all decisions made by the vehicle, along with addressing potential bugs that may occur, whether in manual control, tracking mode or autonomous mode, will be explainable to the user. The vehicle will be fitted with a speaker system that will enable it to explain its decisions orally. All the explanations will also be recorded textually in the interface. This feature will enhance user understanding and trust by providing clear and audible explanations for the actions it takes. This is a key feature of our project.

* 1. Stakeholders



* 1. Features

We will implement 3 different operating modes in our product:

For each mode, we will have different functionalities. Each functionality will have a priority that determines which ones are the most essential and have to be treated first. The smallest number corresponding to the highest priority.

| Mode | Description | Functionalities | Priority |
| --- | --- | --- | --- |
| Manual | The user will control the vehicle with a wireless joystick, The vehicle will be able to move according to the direction and the velocity the user applies. The vehicle will stop by itself if the order the user send collides with an obstacle or person. The system will explain why it made this decision through visual and/or sound feedback. | Moving the car | 1 |
| Obstacle detection | 1 |
| Visual feedback | 2 |
| Sound feedback | 7 |
| Stop | 1 |
| Tracking | The system will be initialized according to certain parameters (to be defined) and it will follow the person that did the initialization. The vehicle will track this person and follow it everywhere the system is allowed to go. (to be defined). It will have the same obstacle avoidance system but it will find a way to avoid the obstacle to keep tracking the person (if it’s possible). It will also have to explain each decision it made. When the vehicle tracking is finished, the user must leave the car in a safe place. | Tracking (find a way to follow the person and try to compute another path if it finds an obstacle. Otherwise, stop the vehicle and notify the user) | 3 |
| Avoid obstacle | 3 |
| Obstacle detection ~~(moving or immobile object)~~  (*fixed1*, *moving2*, *unavoidable3* or *insurmountable4* obstacle) | 6 |
| Autonomous | The person who wants to use the vehicle will be able to send a request from a position through the HMI and the vehicle will reach that position by itself if it is possible. Then we can switch the mode to tracking through the interface.  The vehicle will drive back to its departure location if it has a problem (loss of signal, position unreachable) or if the user never used the vehicle. | Automonous moving | 8 |
| Obstacle avoidance | 8 |
| Localisation | 9 |
| Navigation | 10 |

1 - fixed obstacle: an obstacle that does not move. It is all urban environments like panels, posts, rocks etc.

2 - moving obstacle: an obstacle that is moving (at a minimum speed of 0.25m/s) and can hit the X-car, it can be pedestrians, bikes, dogs etc.

3 - unavoidable obstacle: an obstacle that is too close to the vehicle to bypass it. It can be an obstacle that emerges on the vehicle’s trajectory.

4 - insurmountable obstacle: an obstacle that cannot be crossed by the vehicle, such as a path that is too narrow or a change in level that is too high like a step or a sidewalk.

* 1. Release vision

**Sprint 0**

* First version of the state machine
* Project plan done

**Sprint 1**

* Detailed state machine (final version)
* Manual control with the Xbox controller
* Obstacle detection (vehicle stops)
* Emergency stop button on the remote control
* HMI : visual feedback on the computer (console)

**Sprint 2**

* Tracking a special characteristic (qr code or shoes …)
* Follow the tracked person while detecting obstacles
* First version of the phone app. Choose between the two first modes (tracking and manual) and visual feedback

It's difficult to know the objectives that we will set for the sprint 3 but here is a first idea

**Sprint 3**

* Autonomous mode.
* Avoid obstacles in tracking mode
* Final version of the app :
* Choose between the 3 modes (manual, following, autonomous)
* Communication between the car to the user (errors, decision …)
* Control the car with the phone
* Visual Feedback upgraded
  1. Operational requirements

| Priority | Functionalities | Identifiers | Requirements |
| --- | --- | --- | --- |
| 1 | Moving the car | 1A - Time response | ~~The time response of the command must be under 2s~~  The response time of the remote controller command (speed, angle, stop) must be under 0.2s. Note that the response time for stopping here corresponds to the time it takes for the wheels to stop rotating, and not the time it takes for the vehicle to stop, which depends on its speed. |
| 1B - Maximum speed | The maximum speed of the X-car must be at 6km/h |
| 1C - Ability to turn | The X-car must be able to take a 30-degree turn (tbd) |
| 5 | Autonomous moving | 2A - Found the shortest itinerary to reach the user | ~~The X-car should found the shortest path to the user and send it on the App~~  The X-car should find the shortest itinerary to reach the user and display it live on the HMI. |
| 2B - Arriving time to user | When the user calls the X-car remotely via the HMI, the arriving time must be display |
| 1 | Obstacle detection | 3A - Obstacle detection distance | An obstacle (*fixed1*, *moving2*, *unavoidable3* or *insurmountable4*) must be detected within a minimum of 40 cm whatever the state of the vehicle.  An obstacle must also be detectable within a maximum radius of 1m30. |
| 3 | ~~Decision making~~  Avoid obstacle | 4A - The ability of the X-car to avoid obstacles | In tracking and autonomous mode the X-car must avoid a detected obstacle within at least 20 cm if it has enough space to do so. Otherwise the X-car must stop and report it to the user.  In manual mode, the vehicle must stop. |
| 2 | Visual feedback | 5A - Notifications status informations | Every time the X-car changes status (Autonomous driving, Tracking, manual and emergency stop) it must notice it on the HMI. We will also see the battery’s level (in percent), all the records of all the decisions taken and the position of the vehicle (red dot on a map and gps coordinates). |
| 7 | Sound feedback | 6A - Sound status informations | When the X-car changes status (Autonomous driving, tracking, manuel and emergency stop) it must notice it to the user with the speaker. |
| 6B - Low battery sound | When battery capacity is almost empty (30min, equivalent to a low voltage level yet to be defined), the speaker will notice the user. |
| 4 | Object recognition | 7A - Distinguish fixed and moving obstacle | The X-car must distinguish between a *fixed1,* *moving2 unavoidable3* or *insurmountable4* obstacle of a minimum height of 5 cm. |
| 3 | Tracking | 8A -Tracking distance | When the X-car tracks the user it must be at a distance between 1m and 1m30 behind the user when he is moving forward. If the user stops or turns (right or left) the X-Car must stop or turn with him. If the user turns back the X-Car must keep the user in sight and turn if it’s necessary. |
| 6 | Localisation | 9A - Localisation precision | The user must know the X-car position with a precision around 1 m. The localisation of the vehicle and the user will be displayed on a map with the GPS coordinates on the HMI. |
| 7 | Navigation | 10A - Save navigation map | When the user calls the X-car, the navigation to the user must be saved in case of a lost connection problem. |
| 1 | Emergency stop | 11A - Stop the vehicle in a case of emergency | The emergency stop can be triggered by the user with the remote controller in manual mode or via the HMI in tracking mode. It can also be triggered by the vehicle in case of an *unavoidable3* or an *insurmountable4* obstacle. |

1 - fixed obstacle : an obstacle that does not move, it is all urban environment like panels, post, rock etc.

2 - moving obstacle : an obstacle that is moving (at a minimum speed of 0.25m/s) and can hit the X-car, it can be pedestrians, bikes, dogs etc.

3 - unavoidable obstacle : an obstacle that is too close to the vehicle to bypass it. It can be an obstacle that emerges on the vehicle’s trajectory.

4 - insurmountable obstacle : an obstacle that cannot be crossed by the vehicle, such as a path that is too narrow or a change in level that is too high like a step or a sidewalk.

* 1. Reusable features

With previous years projects (Pivoane, Tow Master and Postcar) we have some common features that we can reuse for X-car.

All of these projects have developed a detecting and avoiding obstacle module that we can reuse. As Postacr is a personnel assistant in senior residences it has lots of common features with our X-car like autonomous driving, communication via App and Sound.

Details of all their codes are available on GitHub. We didn't analyze it completely but we thought to reuse some of their drivers and ROS nodes for features above (Autonomous driving, detecting and avoiding obstacles).

We also found open source ROS nodes that we could use for tracking.

* 1. Validation plan

**Sprint 1**

**Test procedure :**

Verify manual control functionality using the remote controller.

* Ensure the system starts and shuts down : By pressing the ON/OFF switch the system must be operational in 10 seconds. That is to say that the user is able to move the car manually. The car can be turned off whenever the user wants by pressing the ON/OFF switch.
* Ensure the vehicle responds to all remote control commands (back, right, left,..) The vehicle should respond well to the command sent by the remote controller within 0.2 seconds. Example of procedure : 5 seconds forward, right turn at around 25 degrees ,5 seconds backward, left turn at around 25 degrees
* Ensure the vehicle stops when it faces an obstacle in the direction maintained (at least 20 cm from the obstacle) : Place an obstacle in the X-Car's path at a distance longer than 70 cm and observe the distance between the obstacle and the vehicle when this last stops. Check that if we remove the obstacle, the vehicle is able to respond to the user command.
* Test the emergency stop button. The vehicle should stop in less than 1s. The vehicle is still operational.
* Check the history on the computer 1 second after the vehicle has detected an obstacle and so has stopped. The user must be able to access the decision history made by the car on the computer from the moment it is turned on until it is turned off.

**User experience :**

Invite users to test manual driving and provide feedback.

* Collect feedback on the user experience.

Invite users to test the visual feedback by adding an obstacle on the vehicle’s path.

* Collect feedback on the user experience

**Sprint 2**

**Test Procedure :**

Validate the web interface for manual control on smartphone/tablet.

* Check wireless connectivity between the device and X-Car by changing modes (tracking and manual) for example. Modifications must be taken into account by the car in 2 seconds maximum. Make the connection lost to check the output. The app should display an error message. In this case the car stays in standby mode waiting for a new connection. After 5 min if no one tried to connect the vehicle comes back to its first location
* Ensure an intuitive and user-friendly interface ( Not too much data displayed, one or two screens per mode).

Test tracking capability.

* Observe the X-Car consistently following a person on a distance of 20 meters without any obstacles. The user must have a max speed of 6 km/h. The vehicle has to be at a distance between 1m and 1m30 behind the user when he is moving forward. The user will follow a linear trajectory.
* Ensure the vehicle stops when it faces an obstacle (20 cm from the obstacle): Place an obstacle in the X-Car's path at a distance larger than 40 cm and observe the distance between the obstacle and the vehicle when this last stops. The vehicle won’t still be able to avoid any obstacles. It will stop automatically.
* Test the emergency stop button in tracking mode. The vehicle should stop in less than 0.2s. The user should be redirected to the Home menu.
* Check that when the vehicle loses the person being followed, the vehicle stops in less than 1s, an error message is displayed on the app within 2 seconds.

**User experience :**

Invite users to test tracking mode and provide feedback. The user must have a max speed of 6 km/h. The vehicle has to be at a distance between 1m and 1m30 behind the user when he is moving forward. The user will follow a linear trajectory.

* Collect feedback on the user experience.

Team

Loïc Thomas : c/ASM microcontroller programming, Real Time systems, object oriented programming (c++, java), GIT (basics).

Pierre Bonnecaze : Microcontroller programming C, object oriented programming (java,C++), machine learning (basics) , data processing ( Python , SQL)

Well-John LU: object oriented object(c++,java,c#), microcontroller programming(stm32,Arduino), WPS application(xaml,csharp)

Baptiste Turpin: Microcontroller programming, object oriented programming (java,C++), Real time systems

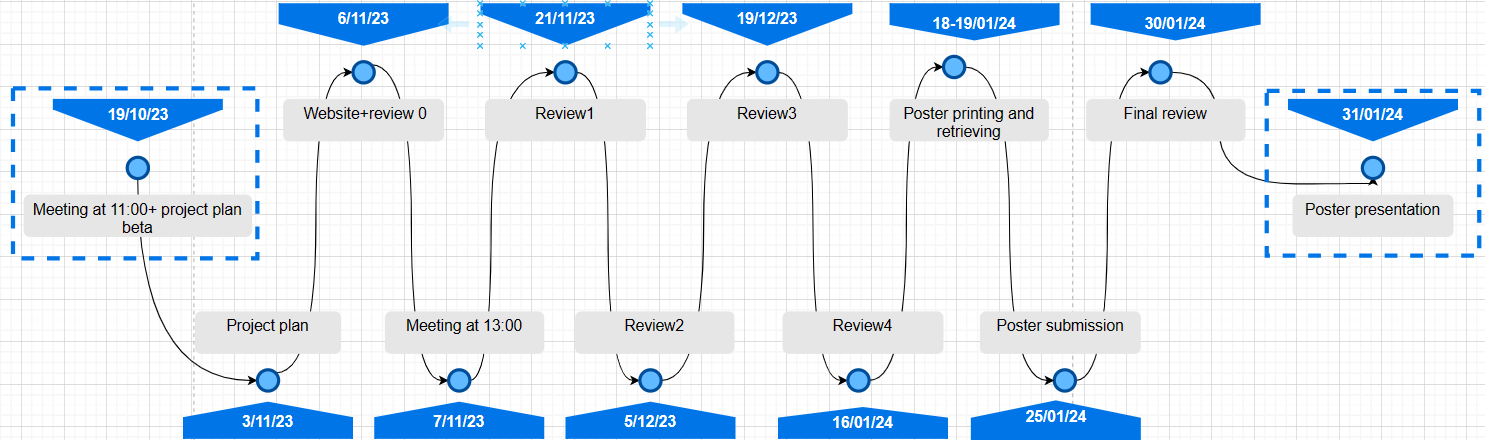
Nicolas Siard: C Programming, SImulink/Matlab, python programming, Microcontroller stm32, AI, Web Programming (html/css/js/php/sql)

Arthur Nicola : Micorcontroller programming (Stm32, Arduino), object-oriented programming (C++, Python), real time system

Ana Carolina Ceolho Robl: Microcontroller programming, Object-oriented programming, GIT.

VIALAN Louis (**SCRUM Master sprint 0**): Micorcontroller programming (Stm32,,C, Arduino), object-oriented programming (C++, Python), real time system, data processing ( Python , SQL), electronics.

1. Milestones and deliverables



1. Project planning
   1. First sprint planning

**Sprint goal**

The goal of the first sprint is to implement the basic functionality of the X-Car, which is to drive it manually with an Xbox controller and to stop it when it faces an obstacle or when the emergency stop button is pressed.

The sprint backlog consists of three user stories, each corresponding to a feature of the X-Car:

- Story 1 Moving the car : As a user, I want to drive the X-Car with an Xbox controller, so that I can control its speed and direction.

- Story 2 Detect obstacle : As a user, I want the X-Car to stop automatically when it faces an obstacle, so that I can avoid collisions and damage.

- Story 3 Emergency Stop : As a user, I want to press an emergency stop button on the controller, so that I can stop the X-Car in less than 1s in case of danger.

- Story 4 Visual Feedback : As a user, I want to see the history of the different operating modes of our vehicle, and the actual mode. I also want to see the different transitions activated.

**Acceptance tests**

For each user story, we define the acceptance tests that will verify if the feature is working as expected:

- Story 1:

- Test 1: Connect the Xbox controller to the X-Car and check if the connection is established.

- Test 2: Move the left joystick on the controller and check if the X-Car moves forward or backward according to the joystick position.

- Test 3: Move the right joystick on the controller and check if the X-Car turns left or right according to the joystick position.

- Story 2:

- Test 1: Place an obstacle at a distance of 70 cm in front of the X-Car and drive it towards the obstacle. Check if the X-Car stops before 20 cm from the obstacle.

- Test 2: Place an obstacle at a distance of 70cm behind the X-Car and drive it backwards. Check if the X-Car stops before 20 cm from the obstacle.

- Test 3: Place an obstacle at a distance of 70 cm with an angle between 20 and 30 ° on the side of the X-Car and drive it sideways. Check if the X-Car stops 20 cm from the obstacle.

- Story 3:

- Test 1: Drive the X-Car in any direction and press the emergency stop button on the controller. Check if the X-Car stops in less than 1s.

- Test 2: Release the emergency stop button and go back to the home menu. The vehicle is still stationary (not moving).

- Story 4:

-Test 1: Display the actual mode on the computer console.

-Test 2:Save the history of the different operating modes in a file “history\_operating\_mode .txt “. Check that it’s the right history.

**Team assignment**

We assign each user story to a subteam of two developers, who will be responsible for implementing and testing it:

- Story 1: Ana Carolina / Arthur

- Story 2: Pierre / Loic

- Story 3: Well-John / Baptiste

-Story 4: Louis / Nicolas

**Task**

For each user story, we break down the tasks into smaller and manageable units, ranked by priority.

Task 0: update state machine and select the part applicable to sprint 1, read previous existing codes that we can reuse.

* Story 1:
  + Task 1: Set up the hardware and software components for connecting the Xbox controller to the X-Car.
    - Start date: November 6
    - End date: November 8
    - Duration: 2 days
  + Task 2: Write a program that reads the input from the controller and converts it into commands for the X-Car’s motors.
    - Start date: November 9
    - End date: November 13
    - Duration: 3 days
  + Task 3: Perform unit testing and integration testing for each task.
    - Start date: November 14
    - End date: November 17
    - Duration: 4 days
* Story 2:
  + Task 1: Set up the hardware and software components for detecting obstacles using ultrasonic sensors and LIDAR.
    - Start date: November 6
    - End date: November 9
    - Duration: 4 days
  + Task 2: Write a program that reads the distance from each sensor and compares it with a threshold value.
    - Start date: November 10
    - End date: November 16
    - Duration: 4 days
  + Task 3: Write a program that stops the motors when any sensor detects an obstacle within the threshold distance.
    - Start date: November 13
    - End date: November 15
    - Duration: 3 days
  + Task 4: Perform unit testing and integration testing for each task.
    - Start date: November 16
    - End date: November 17
    - Duration: 2 days

* Story 3:
  + Task 1: Set up the hardware and software components for activating an emergency stop button on the controller.
    - Start date: November 6
    - End date: November 8
    - Duration: 3 days
  + Task 2: Write a program that reads the state of the button and sends a signal to the X-Car.
    - Start date: November 8
    - End date: November 8
    - Duration: 1 days
  + Task 3: Write a program that stops the motors and disables the input when the signal is received.
    - Start date: November 13
    - End date: November 14
    - Duration: 2 days
  + Task 4: Perform unit testing and integration testing for each task.
    - Start date: November 16
    - End date: November 17
    - Duration: 2 days
* Story 4
  + Task 1: Get the changing mode information in the software
    - Start date: November 9
    - End date: November 13
    - Duration: 4 days
  + Task 2: Identify the place where the information we be displayed
    - Start date: November 13
    - End date: November 14
    - Duration: 1 days
  + Task 3:Display all the information the user needs ( changing mode information, level battery,...)
    - Start date: November 14
    - End date: November 17
    - Duration: 3 days

* AND PERFORM UNIT TESTING AND INTEGRATION ACROSS ALL STORIES: all team members
  1. Product backlog

<Backlog: give the prioritized stories (name, description, feature) that will be planned into the upcoming sprints. >

- Story 1 Moving the car : As a user, I want to drive the X-Car with an Xbox controller, so that I can control its speed and direction. PRIORITY 2

- Story 2 Detect obstacle : As a user, I want the X-Car to stop automatically when it faces an obstacle, so that I can avoid collisions and damage. PRIORITY 3

- Story 3 Emergency Stop: As a user, I want to press an emergency stop button on the controller, so that I can stop the X-Car in less than 1 second in case of danger. PRIORITY 1

- Story 4 Visual Feedback : As a user, I want to see the history of the different operating modes of our vehicle, and the actual mode. I also want to see the different transitions activated. PRIORITY 4

TO SEE LATER

- Story 5 Sensor Data Fusion: As a developer, I should develop the sensor data fusion system ~~that consolidates data from various sensors for more informed decision-making~~ to gather more information for a more accurate decision-making explanation. HIGH PRIORITY

- Story 6 User Training Materials: As a developer, I need to create user training materials and documentation for users to understand how to operate the X-Car effectively. LOW PRIORITY

- Story 7 Explainable Decision-Making Logic: As a customer, I need a logic for explainable decision-making, ensuring that the X-Car can provide clear explanations for its actions in various scenarios. HIGH PRIORITY

- Story 8 User Interface Development: As a user, I want a user-friendly web interface for real-time monitoring and control, enabling us to track the X-Car's movements. HIGH PRIORITY

- Story 9 Continuous Person Tracking: As a user, I need a continuous person tracking system to ensure that the X-Car consistently follows the me without losing track or straying off course. HIGH PRIORITY

- Story 10 Person Recognition and Avoidance: As a user, I want a person recognition system that allows the X-Car to differentiate between the user it is tracking and other individuals, ensuring it avoids collisions with others. HIGH PRIORITY

- Story 11 Reconnection Logic: As a developer, I need to create a reconnection logic that allows the X-Car to automatically return to its initial position or a predefined safe location if it loses connection or in case of a bug, in autonomous mode. LOW PRIORITY

* 1. Schedule control

We will employ a framework that includes techniques such as **Story Points** to estimate the complexity of user stories. We'll also use a **Burndown chart** to visualize progress and hold **daily standup meetings** to keep communication open and adapt to emerging challenges. We'll also do **story scheduling**, which will involve prioritizing stories based on the capacity of the team, with tasks divided up and time estimates assigned.

**Iteration and feedback** loops through sprint reviews and retrospectives will allow us to continually refine our approach, ensuring that the project remains aligned with stakeholder needs, as well as being a crucial part of continuous project improvement.

1. Risk management
   1. Method

Our strategy is based on an initial risk assessment, identifying potential challenges in development and implementation. This is a crucial reflection stage in our project. We prioritize proactive measures, integrating regular team assessments to identify emerging risks quite early. During the development stage, continuous communication within the team regarding our individual progress, coupled with regular updates to stakeholders, ensures ourselves to be aware of every potential risk. This strategy allows us to manage the risks associated with the project.

* 1. Risk and actions

| Risk | Number | Anticipation | Response |
| --- | --- | --- | --- |
| Lose the user (tracking) | 1 | Use different features for the tracking | Stop the car and warn the user (app and audio) |
| Loose connection | 2 | Always check the connection state, maintain a small distance with the user | Stop the car and warn the use |
| Blocked by an object | 3 | None | Stop for a few moments. If the object does not move, avoid it. Otherwise stay stopped and warn the user |
| The car cannot reach the user (autonomous mode) | 4 | The car has to wait in a reachable location | Warn the user and find another meeting point. Or go back to the initial location |
| Some sensors are not working | 5 | Make a process to check sensors behavior. Launch it periodically and during the initialization. Have some spare sensors. Check that the sensors are available with a simple code.  For the main components, prepare spare components. | If the sensors are redundant for some features, try to operate without them. Otherwise if the sensors are important stop and warn the user.  Buy/Order new sensors |
| Late for a release | 6 | Communication in the team, respect the schedule | Inform the clients, rework on the sprint scheduling. Simplify or discuss the lowest priority features to give up with the customer |
| Bad implementation of the state machine (unknown error) | 7 | Follow carefully the state machine (available for us and the client on the drive) while programming. Check the state machine with the tutor. Implement the state machine in an organized manner | Check firstly if the code does not work or if it is a state machine problem. |
| Phone app does not work properly | 8 | Before releasing the app, check if it works well without the car | Check if it is a communication problem between the car and the phone. Otherwise, check separately the car and the app to find the problem |
| No battery in the car | 9 | Think about recharging the car, Have a spare battery | Recharge the car and find tasks which do not need the car to be done. Else find a spare battery to use. |
| Some members are not available for a meeting | 10 | Be clear about availabilities | If too much people are unavailable, postpone the meeting, or do it remotely |
| The product is destroyed | 11 | Take care of the car as it was the apple of our eyes. Make sure to have fund(€10,000) to repay the product | Prepare an apologize message and run away |
| The car hit someone | 12 | Respect security procedures (stay behind the car, check if there are no onlooker) | Help the concerned person. If heavily injured, use the first aid kit situated at the second floor near the secretariat in the GEI building. |
| The only one who knows how a part works is not here for the development of this part | 13 | Everyone should know (a minimum) how every part works (by commenting the code it must be understandable) | Find another thing to do |
| We order an important component but we never have it | 14 | Order as early as possible. Think about other features that not need this component | Find an alternative |
| Code is lost | 15 | Do not forget to git push when there is a big addition for the project. | When we do something a second time, it is always better so don’t worry |
| Almost all the team is sick for an important meeting/presentation | 16 | Warn as early as possible | Postpone it |
| Inadequate communication within the team or with stakeholders can result in misunderstandings and missed requirements | 17 | Maintain clear lines of communication, document project details, and establish regular check-ins | Address communication issues and adjust project plans or documentation to align with revised requirements |
| Stakeholders may request additional features or modifications beyond the initially defined scope, leading to project delays | 18 | Clearly define the project scope and obtain stakeholder approval. Establish a change control process. | Assess the impact of requested changes,negotiate with stakeholders, and adjust the project plan as needed. |

| Risk Matrix | Very high proba | High probability | Small probability | Unlikely |
| --- | --- | --- | --- | --- |
| Critical |  |  | 14 | 11;15 |
| Major |  | 17 | 5;12 | 16 |
| Minor |  | 1;2;6;7;13;8;18 | 9 |  |
| Negligible | 3 | 10 | 4 |  |