

Détecteur de chute pour vélo

Sommaire

- Analyse du besoin
- Planification
- Choix du matériel
- Développement du projet
 - Schéma de câblage
 - Algorigrammes des différents programme [Python et html...]
- Conclusion

Analyse du besoin

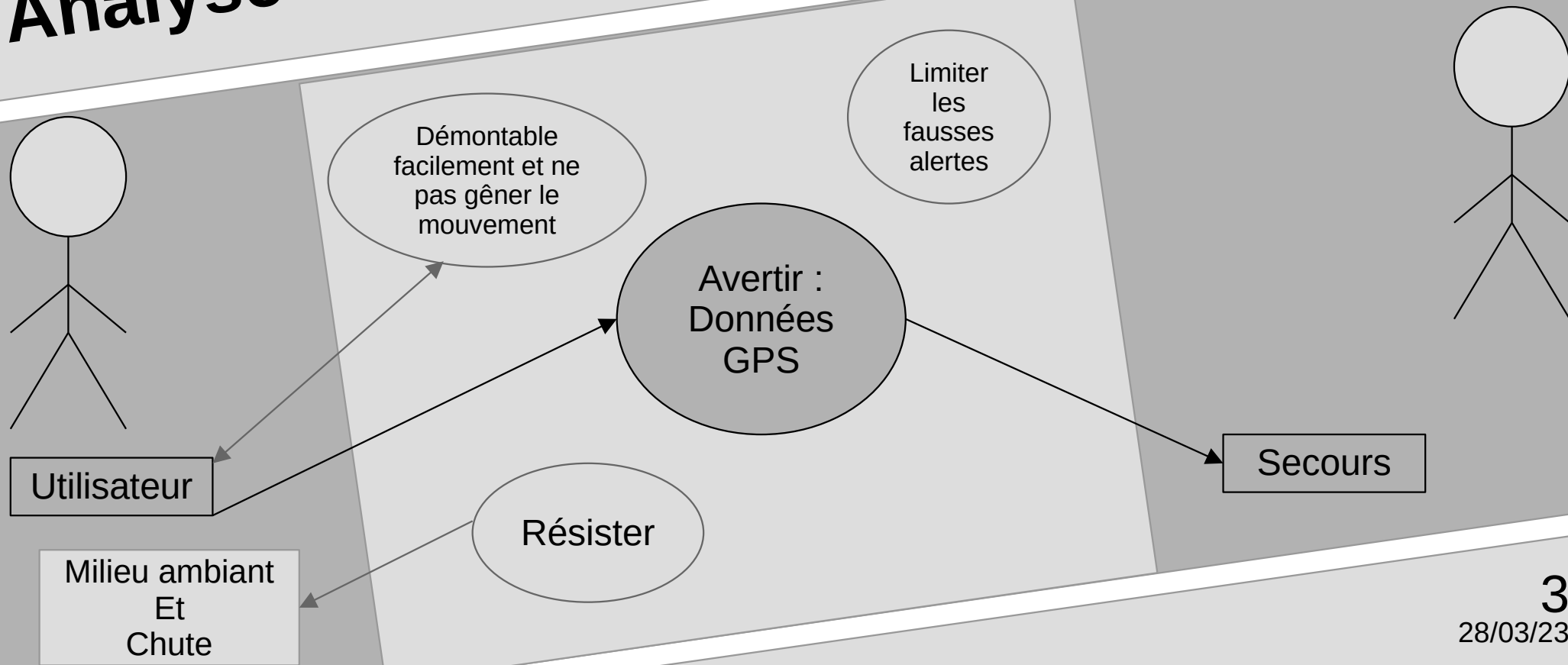
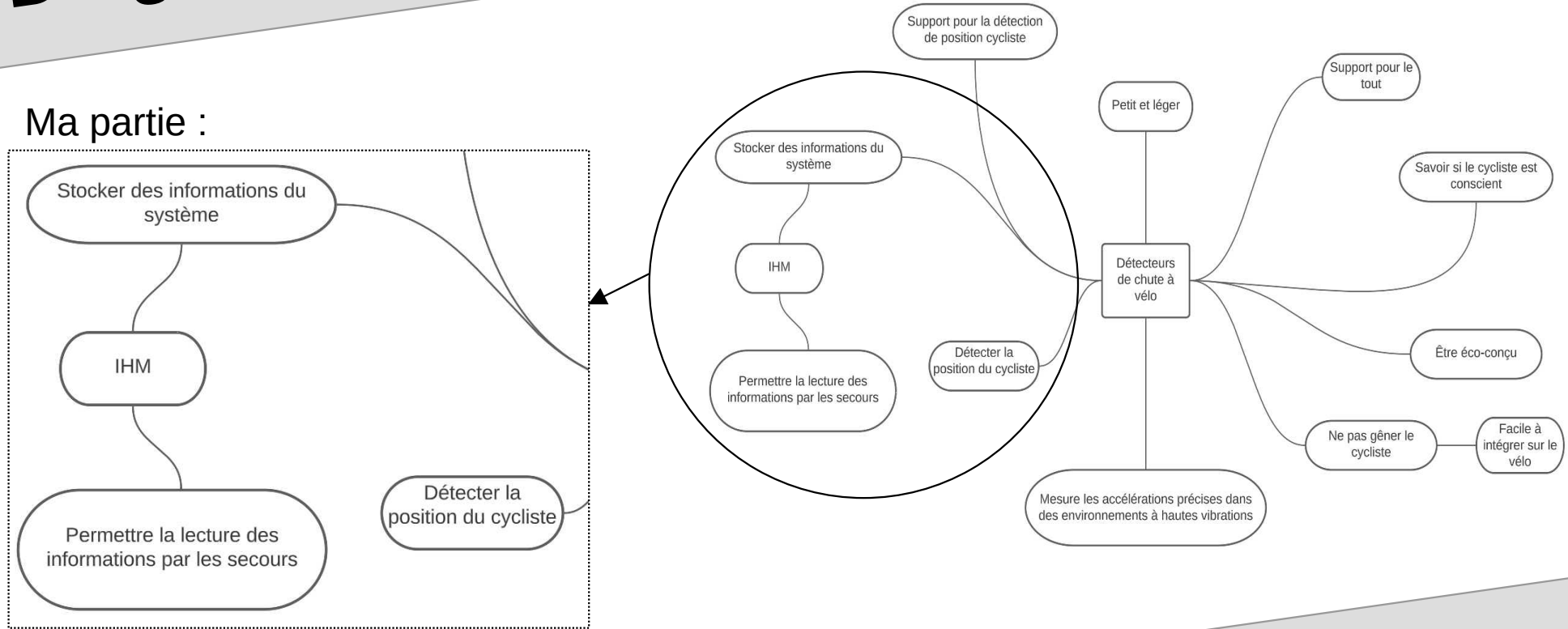


Diagramme des exigences

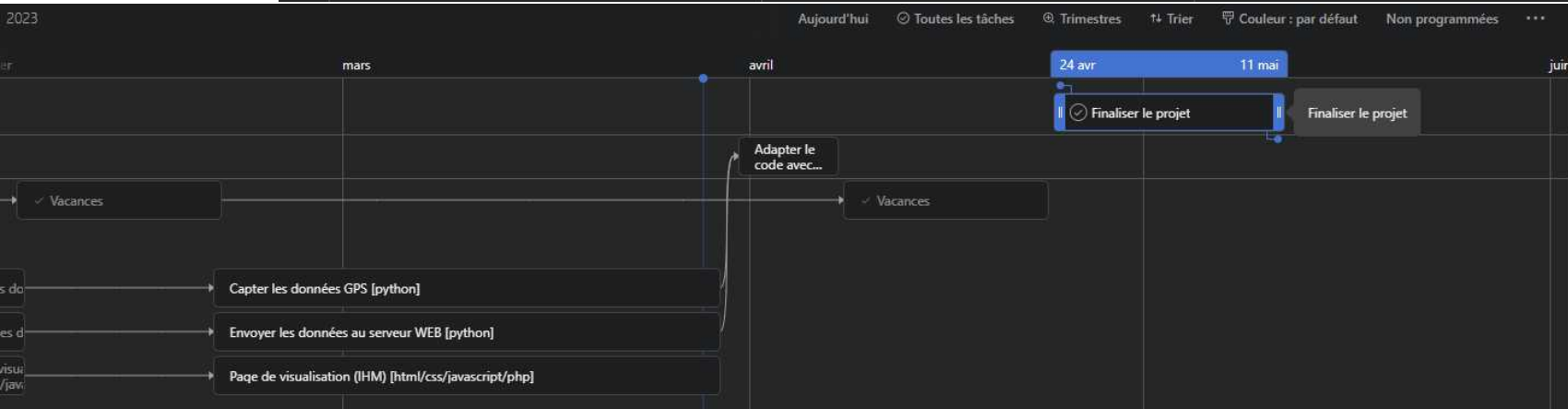
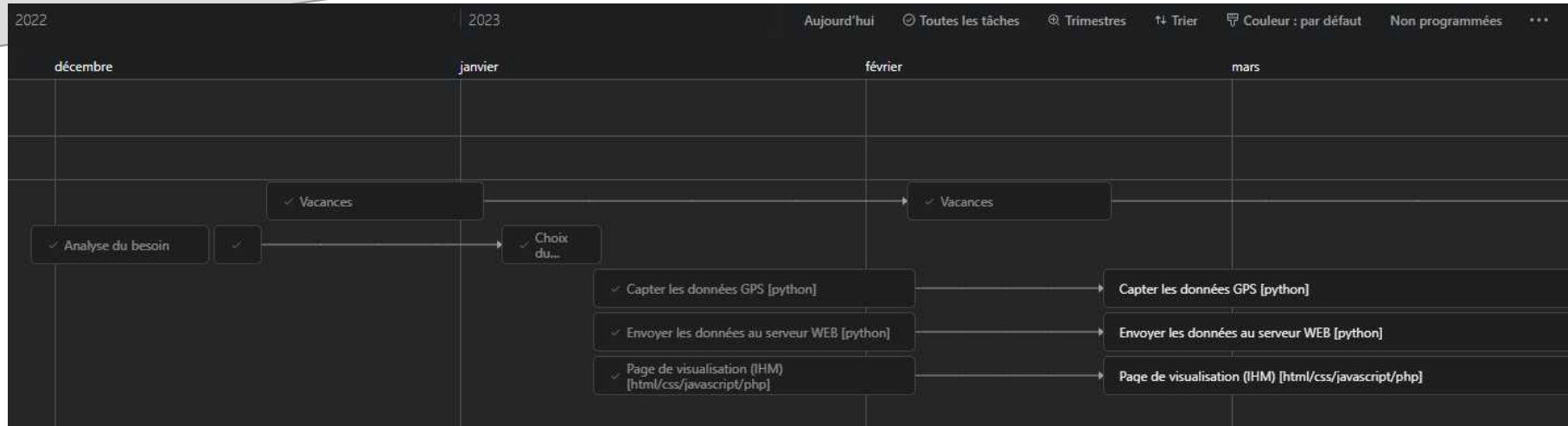
Ma partie :



Cahier des charges

Capteur GPS	
Doit être compatible	Raspberry Pi 3
Doit être léger et petit	10g et 40x20x10mm/15x15x5mm
Doit être précis	2,5m (théorie) ; 100m (testé en intérieur)

Planification



Choix du matériel

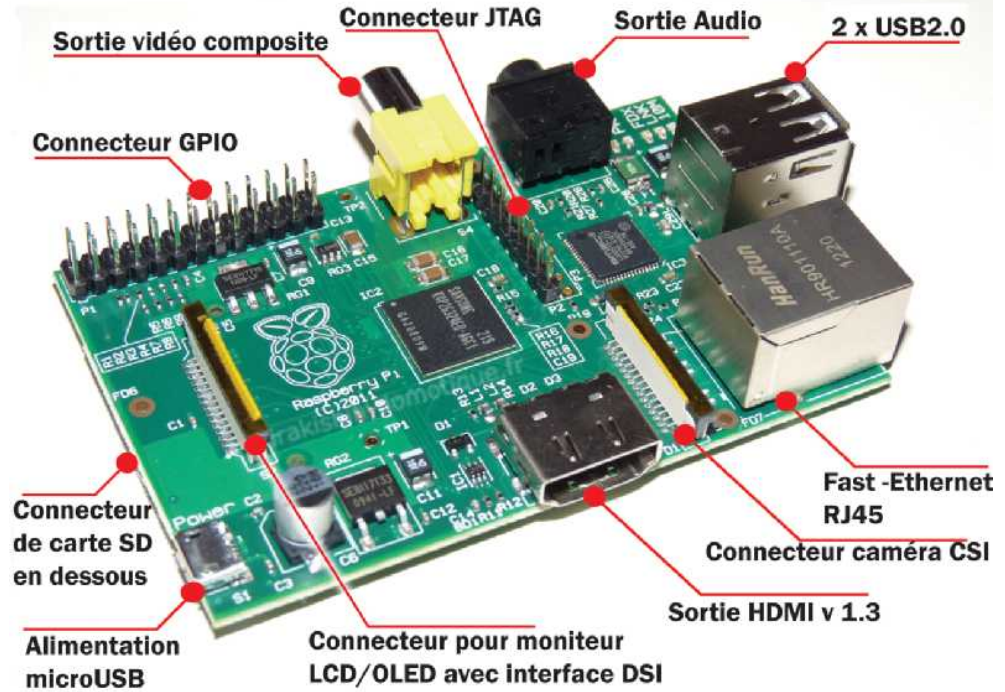
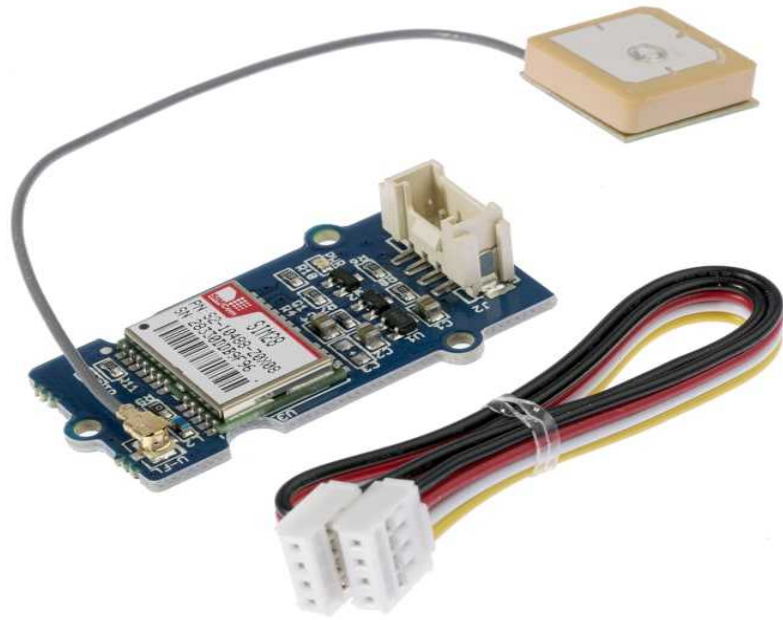
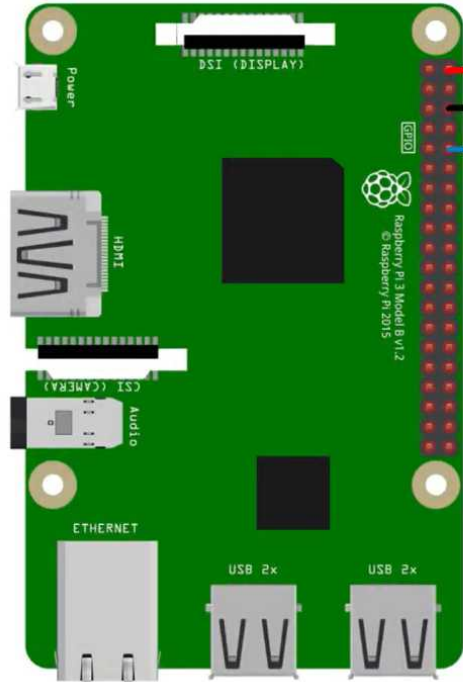


Schéma de câblage

Les broches :
2 ; 6 et 10



Neo 6M Module

Pi 5v --> Neo 6M VCC
pi GND --> Neo 6M GND
pi RX --> Neo 6M TX

@Sparklers

Raspberry Pi 26 Pin Header

3V3 Power	1	2	5V Power
GPIO2 SDA1 I2C	3	4	5V Power
GPIO3 SCL1 I2C	5	6	Ground
GPIO4 1-wire	7	8	GPIO14 UART0_TXD
Ground	9	10	GPIO15 UART0_RXD
GPIO17	11	12	GPIO18 PCM_CLK
GPIO27	13	14	Ground
GPIO22	15	16	GPIO23

Détails :
2 : 5V
6 : GND
10 : GPIO15
Soit RXD

Algorithme [Py]

```
def read(self): # Lire les données du GPS
    # Nous lisons toutes les données, une par une. elles ne sont pas envoyées en même temps
    while True:
        GPS.inp = ser.readline()
        # print(GPS.inp[0:6])
        if GPS.inp[0:6] == b'$GPGGA': # nous n'avons besoin que de cette données ($GPGGA)
            GPS.inp = GPS.inp.decode("utf-8") # conversion du binaire au utf-8
            # print(GPS.inp)
            break
        time.sleep(0.1)
    # Séparation des données
    GPS.GGA = GPS.inp.split(",")
    return [GPS.GGA]
```

Fonction de récupération des données GPS

Tant que : Les données cherchées ne sont pas trouvées

Faire : Lire les données GPS

Si : Les données correspondent à celles recherchées

Faire : Convertir les données GPS en utf-8 (De façon à pouvoir les lire)
et arrêter la boucle

Sinon faire : Attendre 0,1s

Séparer les données et les mettre dans un tableau

Retour du tout

Algorithme [Py]

```
def sendDataToServer(variables): # Requête au serveur
    res = request.urlopen(f"http://lien/de/la/page/add.php{variables}").read()
    # res est de type bytes, il faut le convertir
    page = res.decode("utf8")
    return page
```

Fonction d'envoi des données GPS récupérées au préalable

Envoie de la requête contenant les variables GPS

Et réception de la page retournée

Conversion de la page retourné en utf-8

Retour de la valeur de la page (true ou false)

Algorithme [php/sql]

Fonction de récupération de la requête envoyer par le Raspberry Pi 3

Appel de la base de données

Si : toutes les données sont reçues

Faire : Préparation de la requête sql

Envoie des données à la base de données et retourner true

Sinon faire : retourner false

```
<?php
require('../admin/log-db.php');

if(isset($_GET['time'], $_GET['alt'], $_GET['lat'], $_GET['long'], $_GET['alert'])) {
    $variables = [$_GET['time'], $_GET['long'], $_GET['lat'], $_GET['alt'], $_GET['alert']];

    $input = $bdd->prepare('INSERT INTO `data` (`time`, `longitude`, `latitude`, `altitude`, `alert`, `archive`) VALUES (?, ?, ?, ?, ?, 0);');
    $input->execute($variables);

    echo 'true';
} else {
    echo 'false';
}
?>
```

Algorithme [ajax.js]

```
let div = [document.getElementsByClassName("contents")].map(n => n)[0];
for (let i = 0; i < div.length; i++) {
  setTimeout(() => {
    div[0].remove();
  }, 1);
}

$.ajax({
  type: 'POST',
  url: './pages/data/avec.php',
  success: function (data) {
    $('#content-1').append(data);
  }
});

$.ajax({
  type: 'POST',
  url: './pages/data/sans.php',
  success: function (data) {
    $('#content-2').append(data);
  }
});
```

Retour des données de la base de données au secours sur une page Web

Conclusion

Ce qu'il reste à faire

The screenshot displays a task management interface with two columns: 'À faire' (To do) and 'En cours' (In progress). Each column has a header with a plus sign and three dots. The 'À faire' column contains a task 'Finaliser le projet' with a checkmark icon and a due date of '24 avr – 11 mai'. The 'En cours' column contains a task 'Adapter le code avec celui des autres membres du projet' with a clock icon and a due date of '31 mar – 7 avr'. Both tasks have a circular profile icon next to the due date. At the bottom of each column is a button labeled '+ Ajouter une tâche'.

À faire	En cours
<input checked="" type="checkbox"/> Finaliser le projet 24 avr – 11 mai	<input type="checkbox"/> Adapter le code avec celui des autres membres du projet 31 mar – 7 avr
+ Ajouter une tâche	+ Ajouter une tâche

“

”