Rendu du travail de diplôme

Vous trouverez ci-dessous les éléments importants :

Repository du projet

Pour avoir accès au code source, rendez vous sur mon repository git ici présent : L'entièreté du code est placée dans le répertoire `code`. https://github.com/ACKERMANNGUE/ACKERMANNGUE-AG_Dipl_Tech_2021_VoitureAssistee

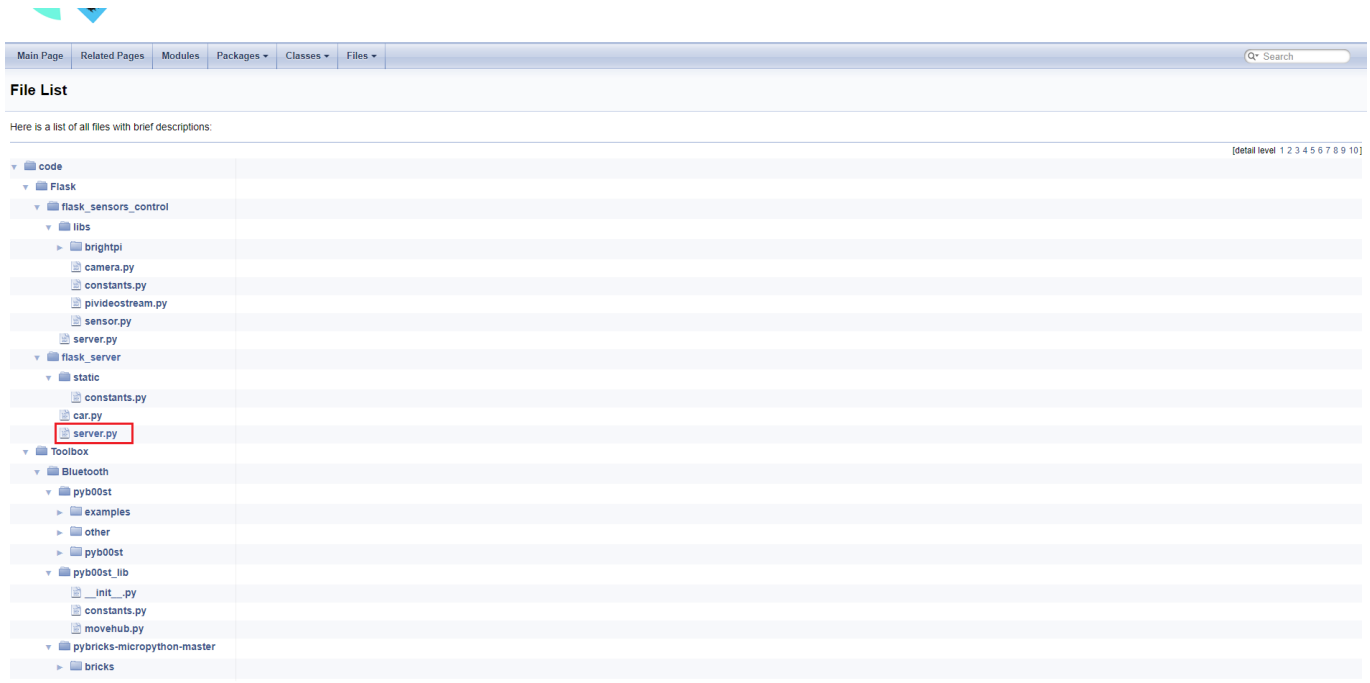
Documentation du projet

Pour avoir accès à la documentation du projet, veuillez vous rendre sur ce lien : https://ackermanngue-ag-dipl-tech-2021-voitureassistee.readthedocs.io/fr/latest/index.html

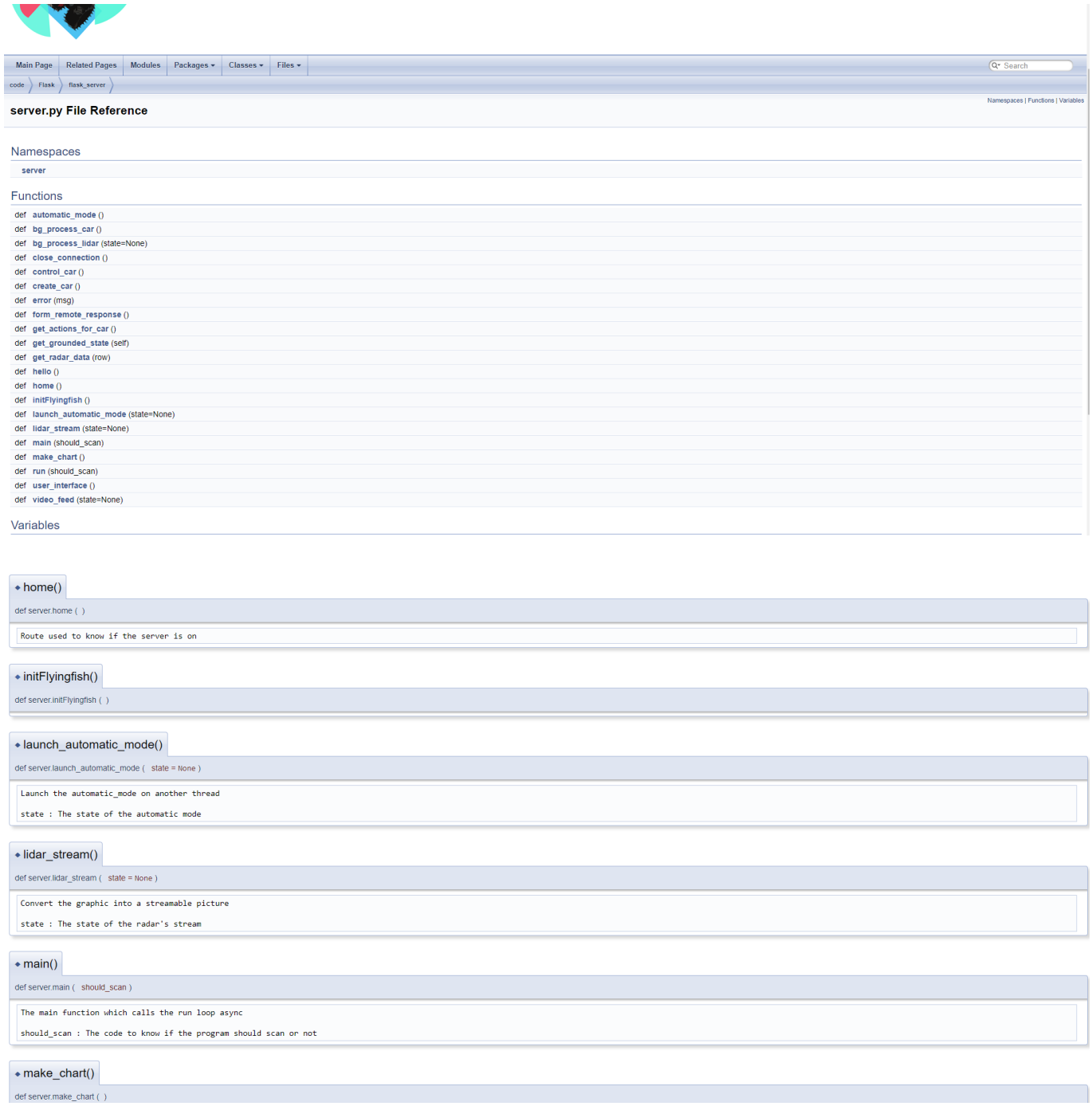
Documentation du code du projet

Pour avoir accès à la documentation du code générée avec Doxygen il faut télécharger le repository du projet à cette adresse : https://github.com/ACKERMANNGUE/ACKERMANNGUE-AG_Dipl_Tech_2021_VoitureAssistee.

Une fois le repository téléchargé, allez dans le répertoire `\docs\Doxygen\html`, puis ouvrez le fichier html nommé : `files.html`. Une fois ouvert, vous pourrez avoir accès au code commenté que j'ai écrit. Par exemple, si vous souhaitez voir le code sur serveur principal, cliquez sur le fichier `server.py` présent dans `\code\Flask\flask_server\`. Une page comme celle-ci apparaîtra :



Depuis cette page, vous aurez accès aux divers méthodes présentes dans le fichier. Si vous cliquez sur une des méthodes, vous serez redirigé vers la description de cette dernière:



Main Page | Related Pages | Modules | Packages | Classes | Files | Search

code | Flask | flask_server

server.py File Reference

Namespaces | Functions | Variables

Namespaces

server

Functions

def automatic_mode ()
def bg_process_car ()
def bg_process_lidar (state=None)
def close_connection ()
def control_car ()
def create_car ()
def error (msg)
def form_remote_response ()
def get_actions_for_car ()
def get_grounded_state (self)
def get_radar_data (row)
def hello ()
def home ()
def initFlyingfish ()
def launch_automatic_mode (state=None)
def lidar_stream (state=None)
def main (should_scan)
def make_chart ()
def run (should_scan)
def user_interface ()
def video_feed (state=None)

Variables

◆ home()

def server.home ( )

Route used to know if the server is on

◆ initFlyingfish()

def server.initFlyingfish ( )

◆ launch_automatic_mode()

def server.launch_automatic_mode (state = None)

Launch the automatic_mode on another thread

state : The state of the automatic mode

◆ lidar_stream()

def server.lidar_stream (state = None)

Convert the graphic into a streamable picture

state : The state of the radar's stream

◆ main()

def server.main (should_scan)

The main function which calls the run loop async

should_scan : The code to know if the program should scan or not

◆ make_chart()

def server.make_chart ( )

Pour voir les divers mini projets de tests élaborés qui m'ont servis à implémenter les divers éléments, allez dans le répertoire `Toolbox`. Ce répertoire contient aussi des codes d'exemples trouvés sur internet.