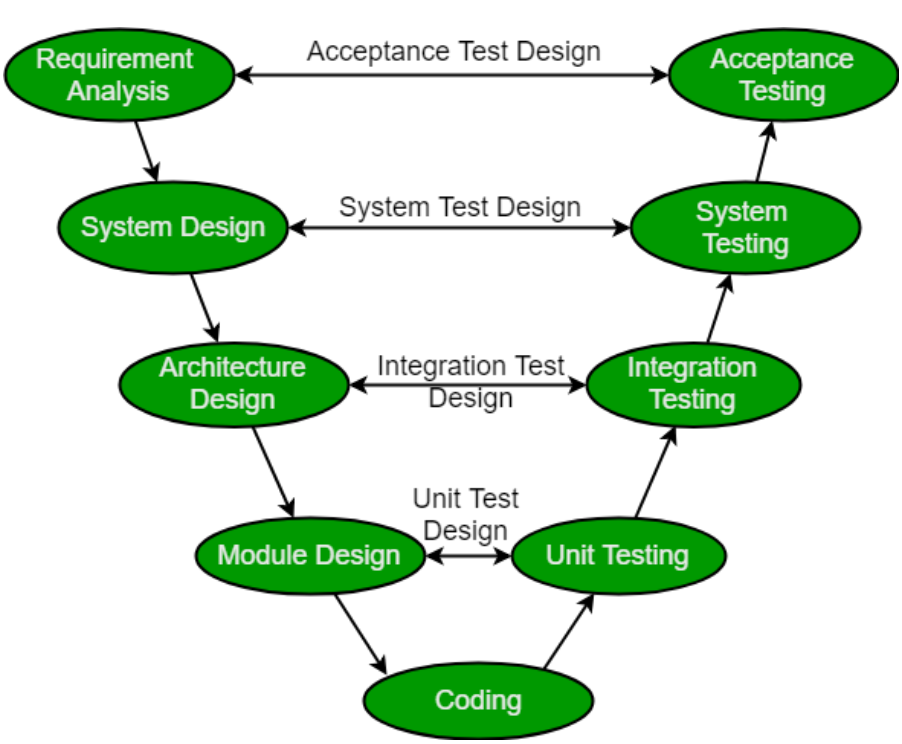
V-Model

Requirement analysis or user requirements: This step is based on analyzing the user's specifications, in our case the user wants a somnolence detection program that can alert him.



System design: In fact, this step is where we define what should the final product do. Luckily, specifications have been already defined into distinct tasks:

- Designing a system that detects if the driver is tired or sleeping while driving, using eye tracking and face expressions processing.
- Alert the driver if he is sleeping with a sound signal (not sudden or loud, so that the driver does not lose control over the vehicle)
- Developing this solution on an embedded platform
- Communication should be done using CAN-BUS and should transfer data to a dashboard

Architecture design: This step is about figuring out possibilities and techniques by which the user requirements can be implemented. In our case, to implement our system, we will need a micro-controller (Raspberry Pi), camera for image processing and a buzzer for a light sound signal. To implement the software on the platform we will need the right libraries

(OpenCV) to process the image flow in real time, and programming language such as Python or C.

Module Design: The program will have inputs and outputs and should generate a signal in case it detects any sign of fatigue, using face expressions or eyes detection. We should set a time threshold, that will be used by the program to determine if the user is sleeping or just closing his eyes, for example a two second threshold. Therefore, the program should be implemented this way:

Threshold = 2

IF Face_Detected = TRUE

AND UserEyesClosed_Threshold = TRUE

THEN SoundSignal = ON

ELSE SoundSignal = OFF

Unit testing: For this first testing processing we should test each part separately. Test if the face is detected, if the eyes are detected, if the program uses the threshold for somnolence detection, if the signal alarm works. All the distinct part or loop of the code should be tested.

Integration test: When the different components will be assembled, the coded functionalities should work together in the common environment which is represented by the embedded platform.

System testing: For this part we will inject INPUTS and see if the result corresponds to what the user is seeking for. In our case, we will test different cases: When there is no face, when there is a face and the user is awake, and the last one where he is about to sleep.

Acceptance test: This testing process concerns the user and his satisfaction with the final product. Also, if the final product did meet the predefined specifications