ON-DEMAND TRAFFIC LIGHT CONTROL

ES Professional Nanodegree Project

Mahmoud Naegh Abdelkader

SYSTEM DESCRIPTION

This is a traffic light system with an on-demand feature that allows the pedestrians to cross whenever they want.

There are two modes:

- Normal mode where the traffic light operates as usual, switching between red, yellow (blinks), and green lights every 5 seconds.
- Pedestrian mode where the traffic light will turn red so the pedestrian can cross.
  If the signal is already red nothing will happen.
  If the signal is yellow also nothing will happen, the yellow will continue blinking.
  If the signal is green the signal will switch to yellow blinking for 5 seconds then red so the pedestrian can walk.

SYSTEM DESIGN

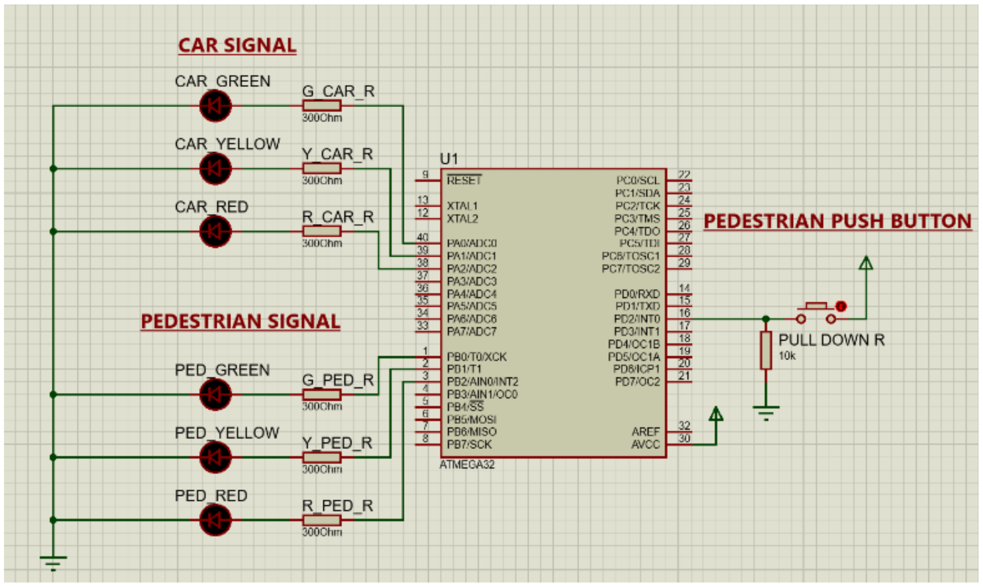
The system is based on the AVR ATmega32 microcontroller, and contains 3 LEDs for the car’s signal:

GREEN, YELLOW, RED → connected on PORTA pins 0,1,2 respectively.

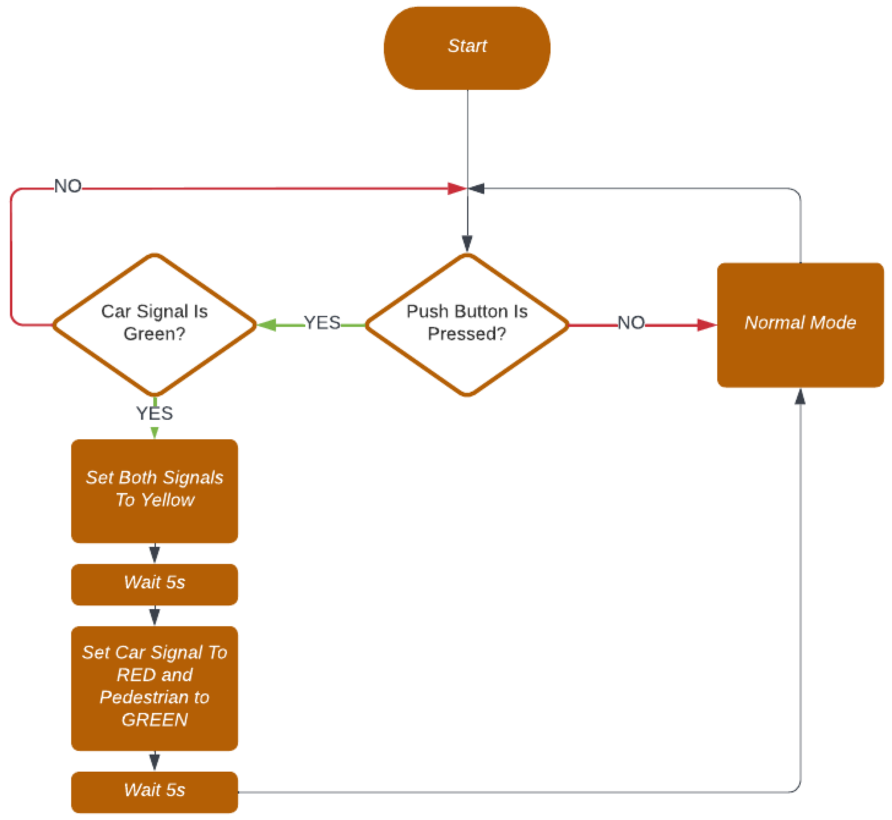
3 LEDs for the pedestrian’s signal:

GREEN, YELLOW, RED → connected on PORTB pins 0,1,2 respectively.

One push button → connected on INT0 (PORTD pin 2)



SYSTEM FLOW CHART



System Constraints

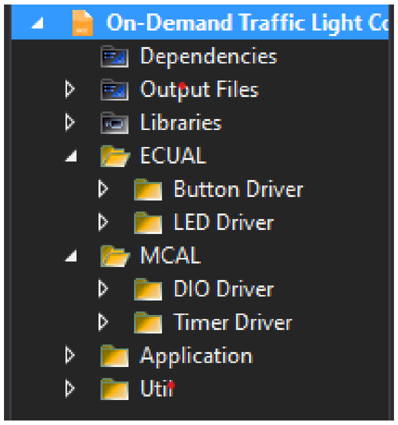
Double press on the push button will only read the first press.

A long press on the push button does nothing.

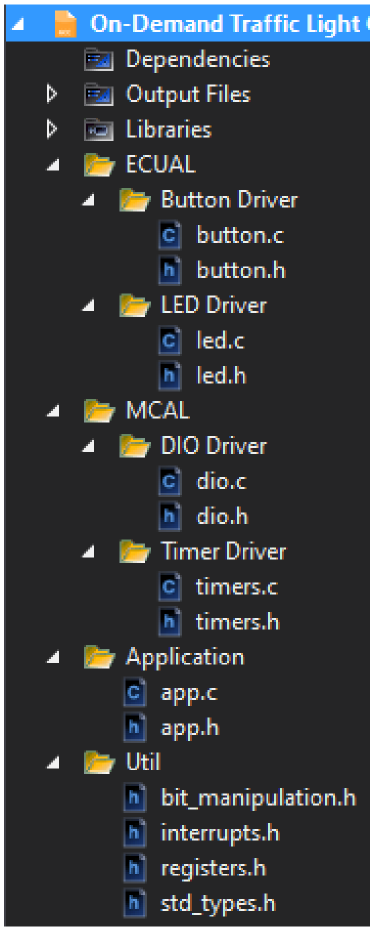
Pressing the button while the Car Signal is RED or YELLOW does nothing.

DEVELOPMENT ENVIRONMENT PREPARATION

Layer Folder Structure

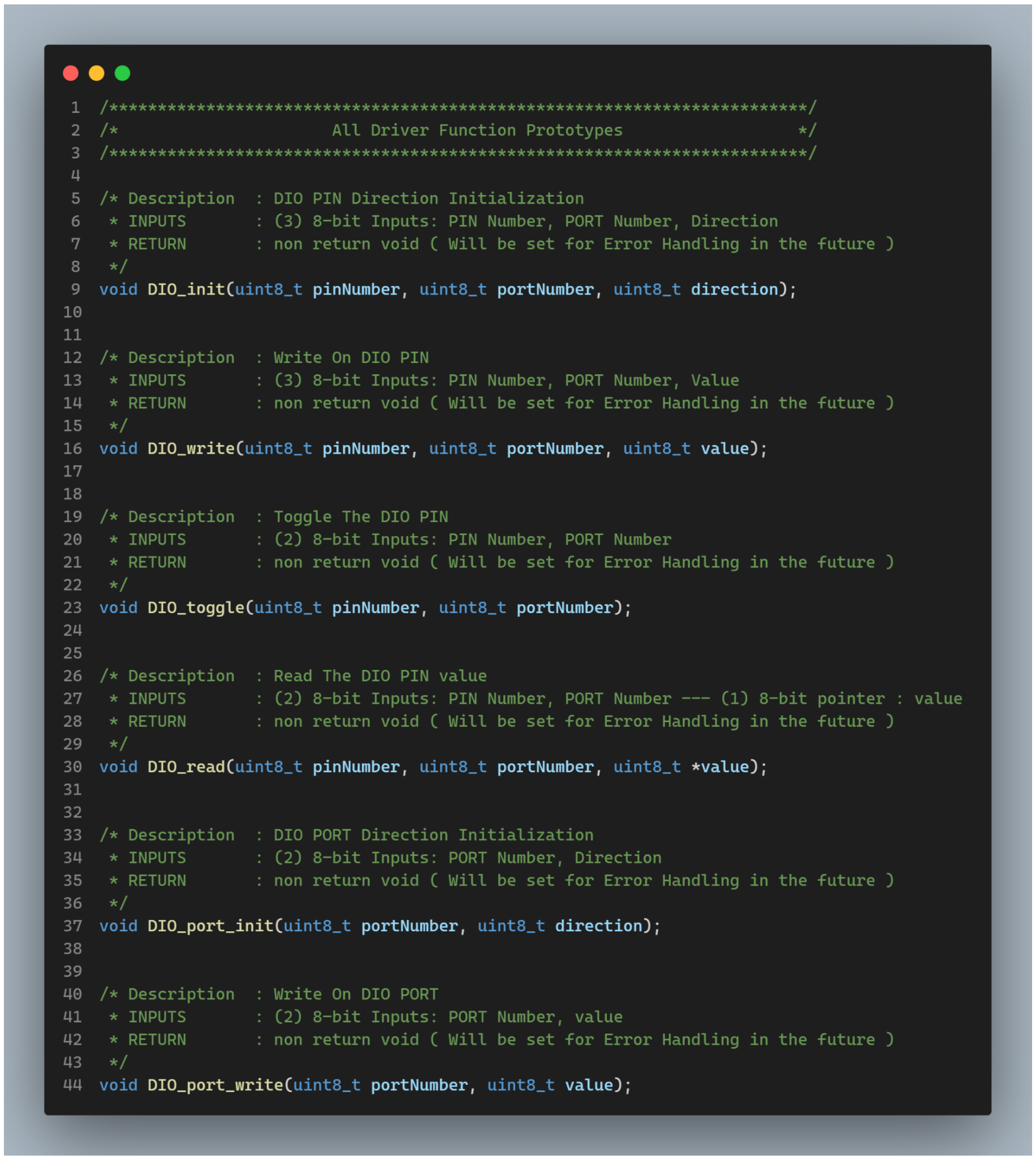


Create .c and .h file for each Folder



(MCAL) DIO DRIVER

Fill in dio.h file with functions' prototypes and macros

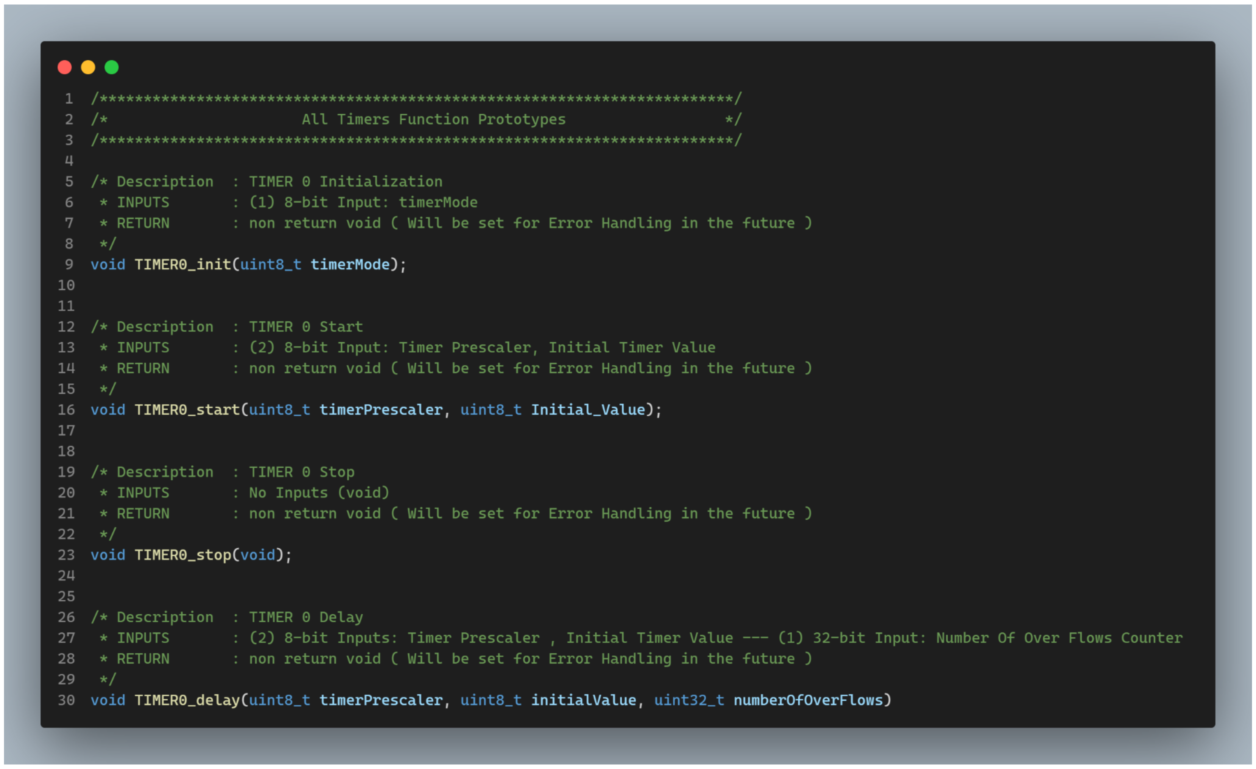


```c
/*******************************************************************************/
/*                      All Driver Function Prototypes                    */
/*******************************************************************************/

/* Description  : DIO PIN Direction Initialization
 * INPUTS       : (3) 8-bit Inputs: PIN Number, PORT Number, Direction
 * RETURN       : non return void ( Will be set for Error Handling in the future )
 */
void DIO_init(uint8_t pinNumber, uint8_t portNumber, uint8_t direction);


/* Description  : Write On DIO PIN
 * INPUTS       : (3) 8-bit Inputs: PIN Number, PORT Number, Value
 * RETURN       : non return void ( Will be set for Error Handling in the future )
 */
void DIO_write(uint8_t pinNumber, uint8_t portNumber, uint8_t value);


/* Description  : Toggle The DIO PIN
 * INPUTS       : (2) 8-bit Inputs: PIN Number, PORT Number
 * RETURN       : non return void ( Will be set for Error Handling in the future )
 */
void DIO_toggle(uint8_t pinNumber, uint8_t portNumber);


/* Description  : Read The DIO PIN value
 * INPUTS       : (2) 8-bit Inputs: PIN Number, PORT Number --- (1) 8-bit pointer : value
 * RETURN       : non return void ( Will be set for Error Handling in the future )
 */
void DIO_read(uint8_t pinNumber, uint8_t portNumber, uint8_t *value);


/* Description  : DIO PORT Direction Initialization
 * INPUTS       : (2) 8-bit Inputs: PORT Number, Direction
 * RETURN       : non return void ( Will be set for Error Handling in the future )
 */
void DIO_port_init(uint8_t portNumber, uint8_t direction);


/* Description  : Write On DIO PORT
 * INPUTS       : (2) 8-bit Inputs: PORT Number, value
 * RETURN       : non return void ( Will be set for Error Handling in the future )
 */
void DIO_port_write(uint8_t portNumber, uint8_t value);
```

Functions Implementation:
https://github.com/MahmoudNageh/On-Demand_Traffic_Light/tree/main/On-Demand%20Traffic%20Light%20Control/On-Demand%20Traffic%20Light%20Control/MCAL/DIO%20Driver

(MCAL) TIMERS DRIVER

Fill in timers.h file with functions' prototypes and macros



```c
/*******************************************************************************/
/*                        All Timers Function Prototypes                     */
/*******************************************************************************/

/* Description  : TIMER 0 Initialization
 * INPUTS       : (1) 8-bit Input: timerMode
 * RETURN       : non return void ( Will be set for Error Handling in the future )
 */
void TIMER0_init(uint8_t timerMode);


/* Description  : TIMER 0 Start
 * INPUTS       : (2) 8-bit Input: Timer Prescaler, Initial Timer Value
 * RETURN       : non return void ( Will be set for Error Handling in the future )
 */
void TIMER0_start(uint8_t timerPrescaler, uint8_t Initial_Value);


/* Description  : TIMER 0 Stop
 * INPUTS       : No Inputs (void)
 * RETURN       : non return void ( Will be set for Error Handling in the future )
 */
void TIMER0_stop(void);


/* Description  : TIMER 0 Delay
 * INPUTS       : (2) 8-bit Inputs: Timer Prescaler , Initial Timer Value --- (1) 32-bit Input: Number Of Over Flows Counter
 * RETURN       : non return void ( Will be set for Error Handling in the future )
 */
void TIMER0_delay(uint8_t timerPrescaler, uint8_t initialValue, uint32_t numberOfOverFlows)
```

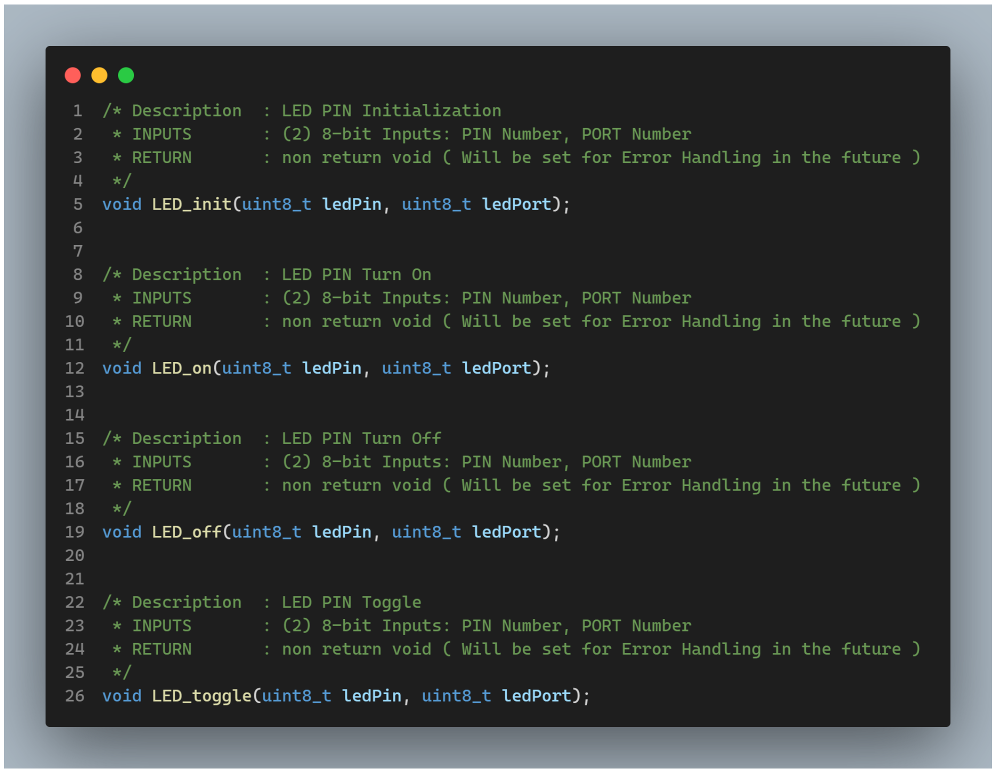
Functions Implementation:
https://github.com/MahmoudNageh/On-Demand_Traffic_Light/tree/main/On-Demand%20Traffic%20Light%20Control/On-Demand%20Traffic%20Light%20Control/MCAL/Timer%20Driver

MCAL test cases:

https://youtu.be/giG1-qiVvHI

(ECUAL) LED DRIVER

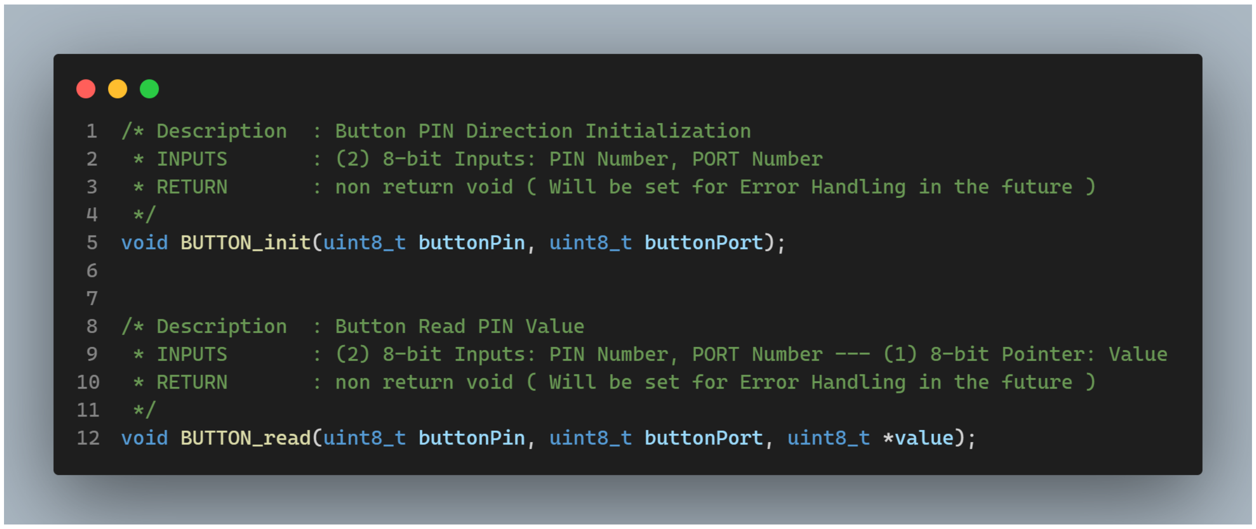
Fill in led.h file with functions' prototypes and macros



```c
/* Description  : LED PIN Initialization
 * INPUTS       : (2) 8-bit Inputs: PIN Number, PORT Number
 * RETURN       : non return void ( Will be set for Error Handling in the future )
 */
void LED_init(uint8_t ledPin, uint8_t ledPort);


/* Description  : LED PIN Turn On
 * INPUTS       : (2) 8-bit Inputs: PIN Number, PORT Number
 * RETURN       : non return void ( Will be set for Error Handling in the future )
 */
void LED_on(uint8_t ledPin, uint8_t ledPort);


/* Description  : LED PIN Turn Off
 * INPUTS       : (2) 8-bit Inputs: PIN Number, PORT Number
 * RETURN       : non return void ( Will be set for Error Handling in the future )
 */
void LED_off(uint8_t ledPin, uint8_t ledPort);


/* Description  : LED PIN Toggle
 * INPUTS       : (2) 8-bit Inputs: PIN Number, PORT Number
 * RETURN       : non return void ( Will be set for Error Handling in the future )
 */
void LED_toggle(uint8_t ledPin, uint8_t ledPort);
```

(ECUAL) BUTTON DRIVER

Fill in button.h file with functions' prototypes and macros



```c
/* Description  : Button PIN Direction Initialization
 * INPUTS       : (2) 8-bit Inputs: PIN Number, PORT Number
 * RETURN       : non return void ( Will be set for Error Handling in the future )
 */
void BUTTON_init(uint8_t buttonPin, uint8_t buttonPort);


/* Description  : Button Read PIN Value
 * INPUTS       : (2) 8-bit Inputs: PIN Number, PORT Number --- (1) 8-bit Pointer: Value
 * RETURN       : non return void ( Will be set for Error Handling in the future )
 */
void BUTTON_read(uint8_t buttonPin, uint8_t buttonPort, uint8_t *value);
```

Functions Implementation:
https://github.com/MahmoudNageh/On-Demand_Traffic_Light/tree/main/On-Demand%20Traffic%20Light%20Control/On-Demand%20Traffic%20Light%20Control/ECUAL

Implementation:
https://youtu.be/VEih-Yrq9a0

TESTING THE APPLICATION

User Story 1: PASSED
https://youtu.be/6FQ2Ic2A3Ak

User Story 2,3 : PASSED
https://youtu.be/CgoQjFSWqJ4

User Story 4, 5: PASSED

https://youtu.be/J4VTGMa45P4

CONCLUSION

The On-Demand Traffic Light was successfully implemented and tested with the various user stories and PASSED all of them.