

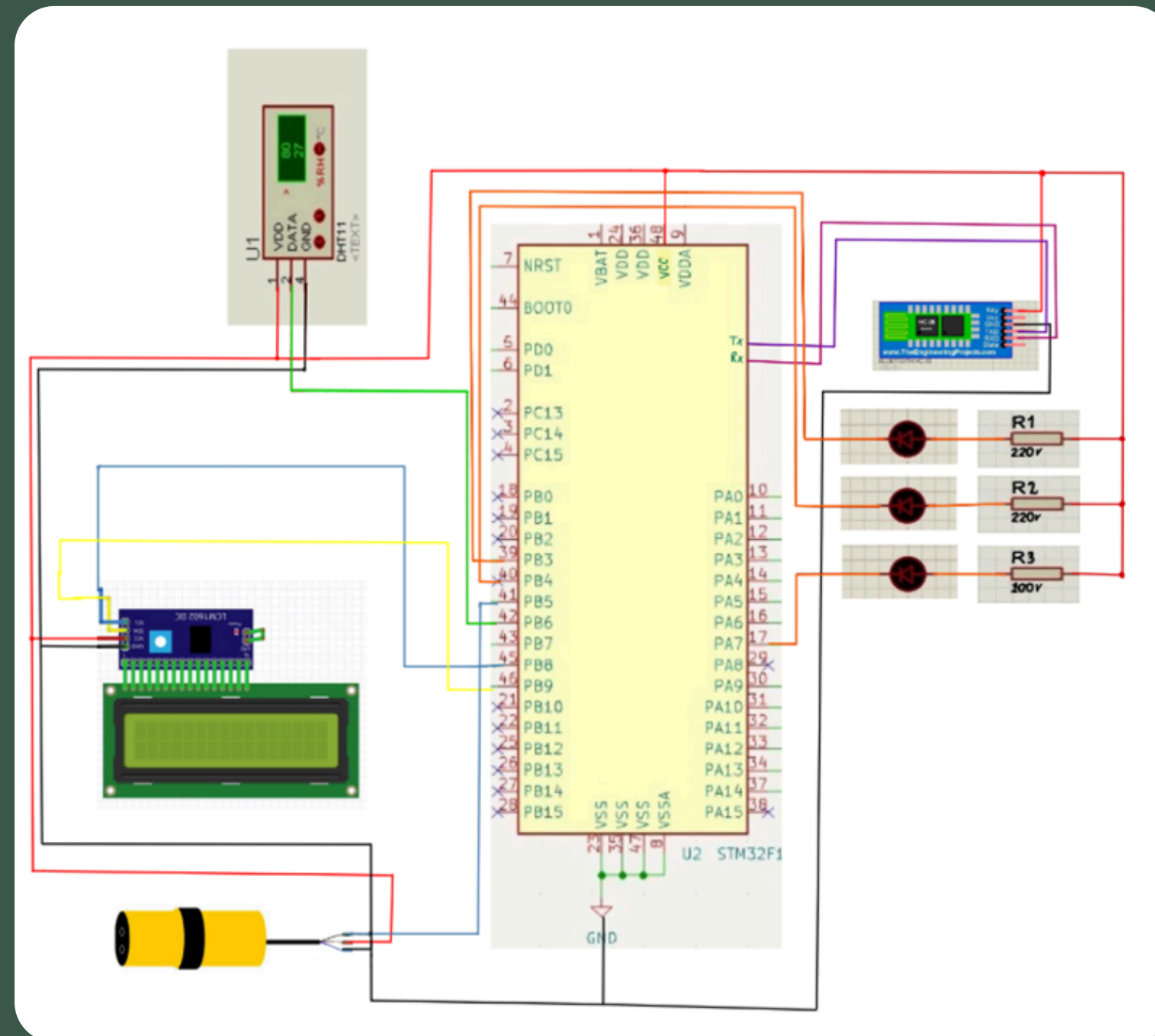
# Sistema de medición de humedad y temperatura con sensor infrarrojo

Programación de microcontroladores

Docente: Ing. Rubén Dario Mansilla

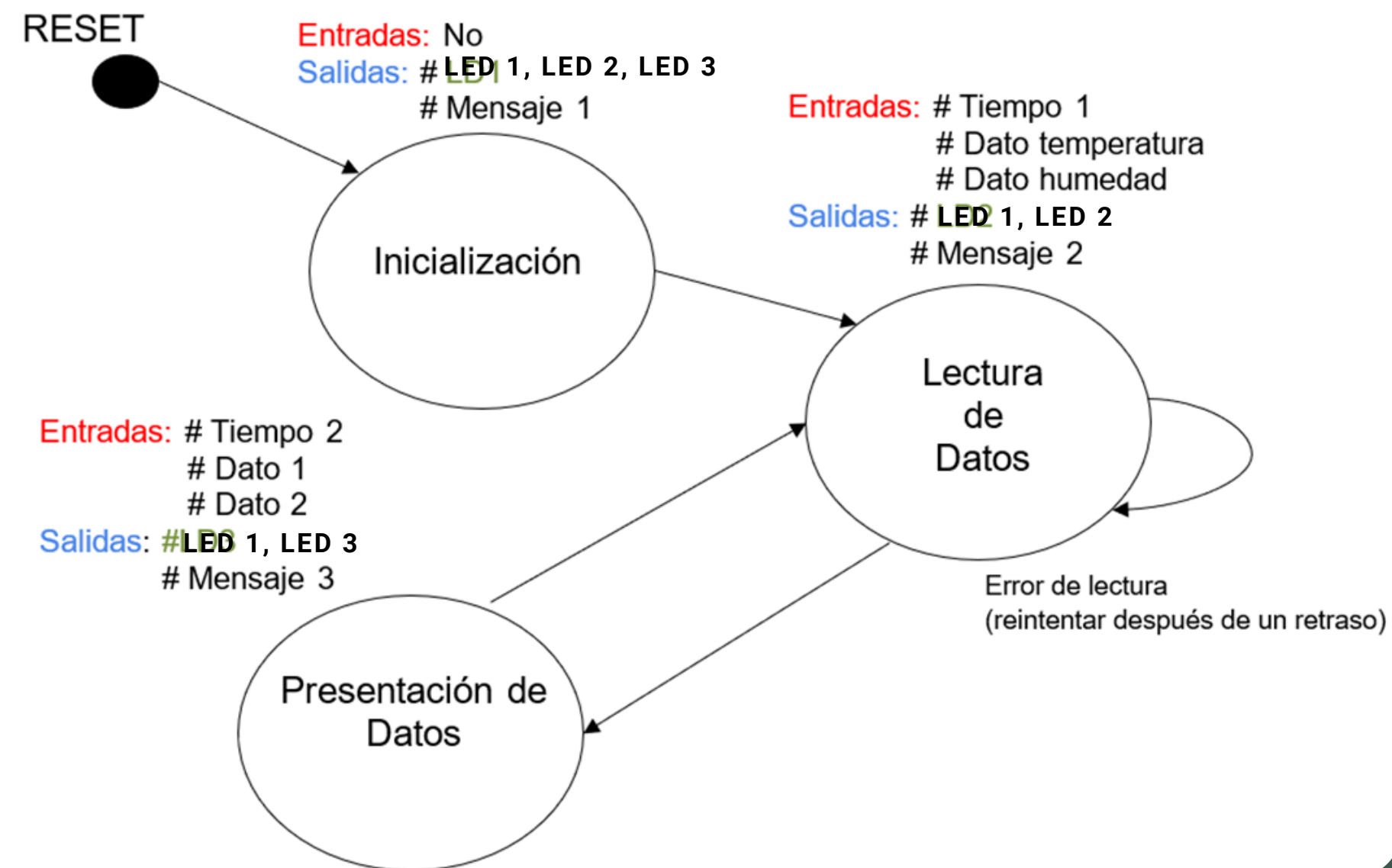
Alumnos: Acosta Marcelo  
Carrasco Sebastián  
Soria Aldana  
Vallejo Manuel

# EL HARDWARE DEL DISPOSITIVO Y SU CONEXIÓN FÍSICA

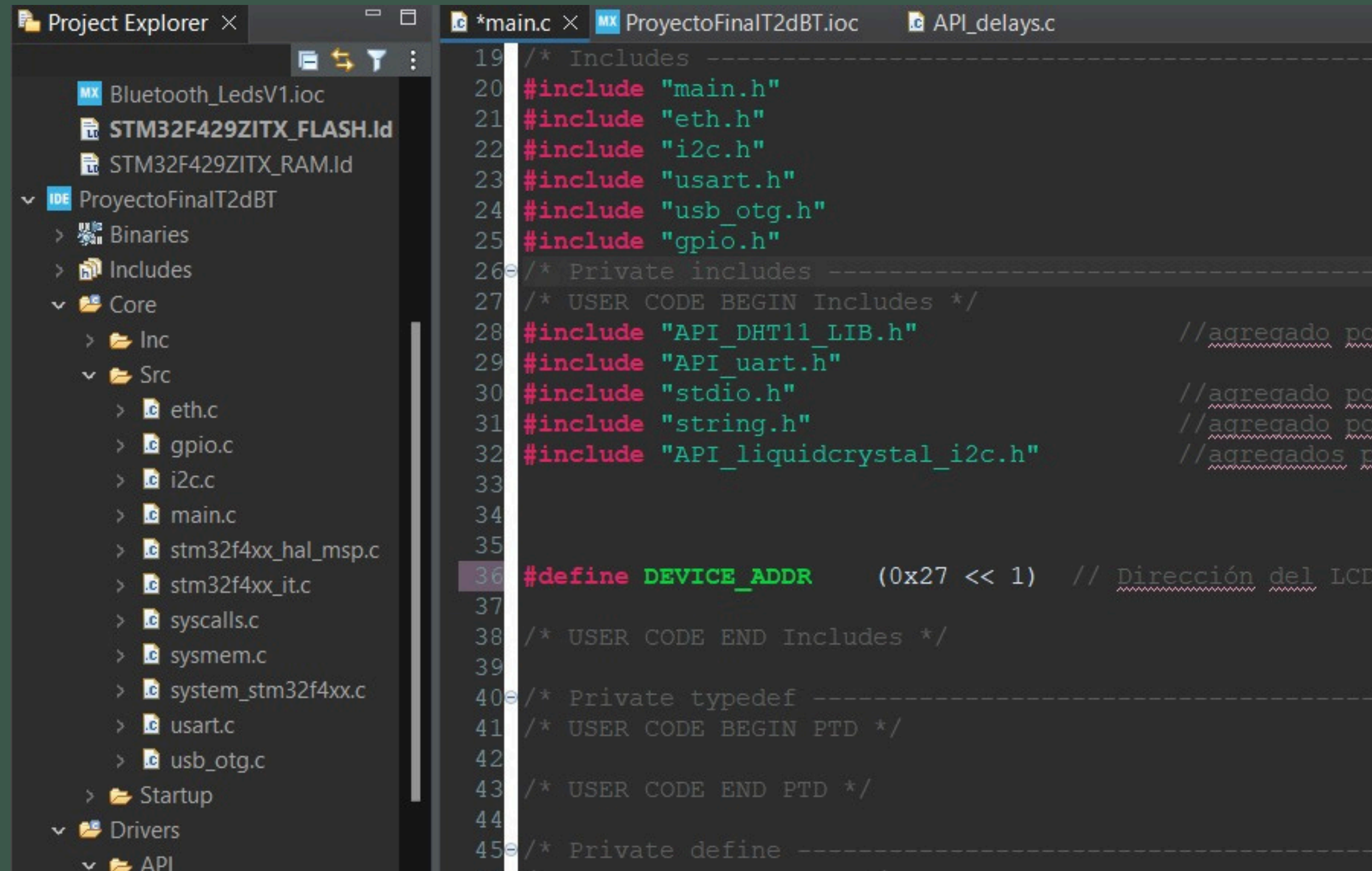


# Consideraciones sobre el software

Diagrama de estados de la aplicación



# Consideraciones sobre el software

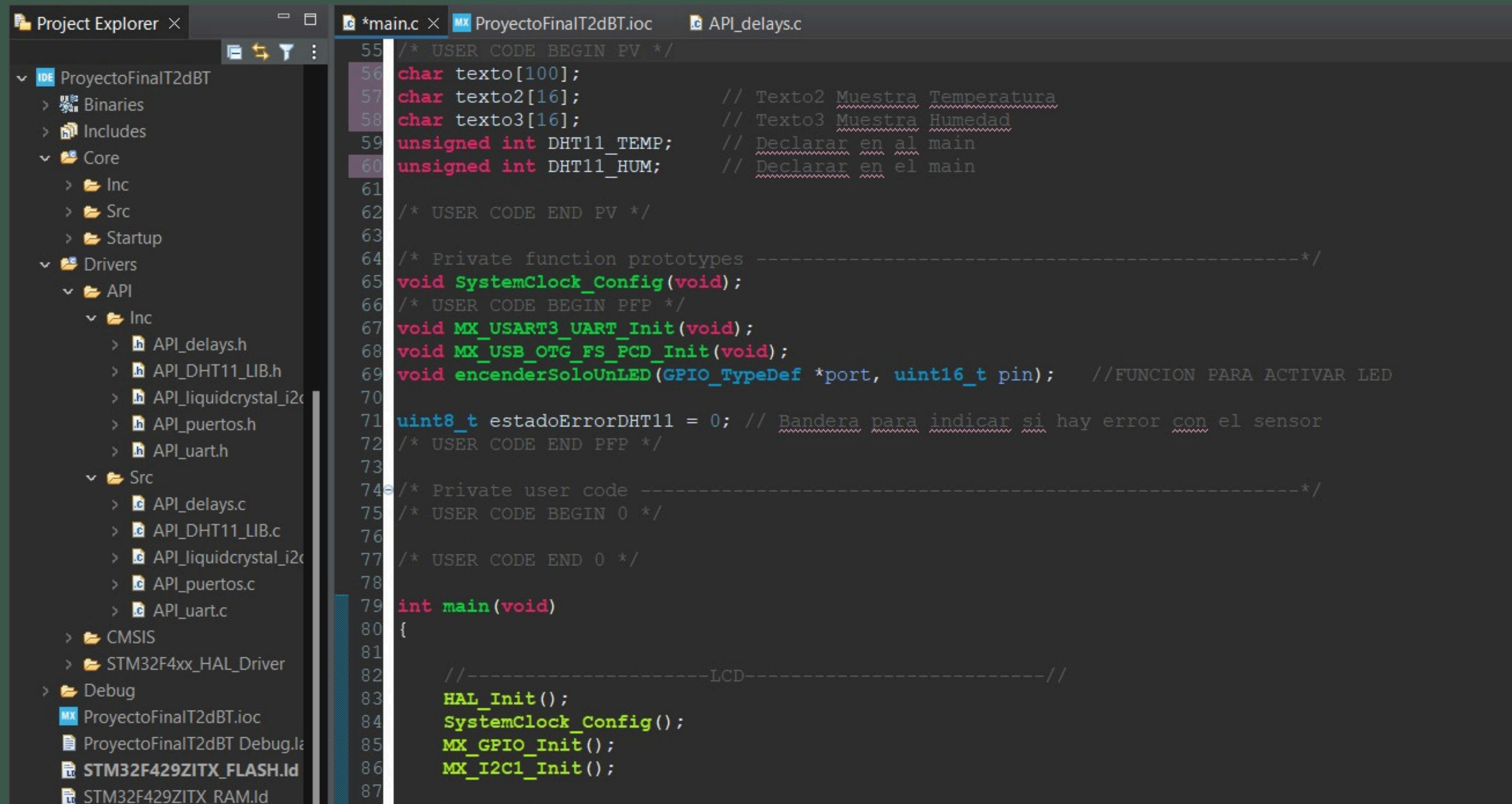


The image shows a screenshot of an IDE interface. On the left is the 'Project Explorer' panel, which displays a project structure for 'ProyectoFinalT2dBT'. The project is organized into several folders: 'Binaries', 'Includes', 'Core', and 'Drivers'. The 'Core' folder is expanded, showing subfolders 'Inc' and 'Src'. The 'Src' folder contains several C source files, including 'eth.c', 'gpio.c', 'i2c.c', 'main.c', 'stm32f4xx\_hal\_msp.c', 'stm32f4xx\_it.c', 'syscalls.c', 'systemem.c', 'system\_stm32f4xx.c', 'usart.c', and 'usb\_otg.c'. The 'main.c' file is selected. On the right is the main editor window, which displays the content of 'main.c'. The code is written in C and includes several header files and a macro definition. The code is as follows:

```
19 /* Includes -----
20 #include "main.h"
21 #include "eth.h"
22 #include "i2c.h"
23 #include "usart.h"
24 #include "usb_otg.h"
25 #include "gpio.h"
26 /* Private includes -----
27 /* USER CODE BEGIN Includes */
28 #include "API_DHT11_LIB.h" //agregado por
29 #include "API_uart.h"
30 #include "stdio.h" //agregado por
31 #include "string.h" //agregado por
32 #include "API_liquidcrystal_i2c.h" //agregados po
33
34
35
36 #define DEVICE_ADDR (0x27 << 1) // Dirección del LCD
37
38 /* USER CODE END Includes */
39
40 /* Private typedef -----
41 /* USER CODE BEGIN PTD */
42
43 /* USER CODE END PTD */
44
45 /* Private define -----
```

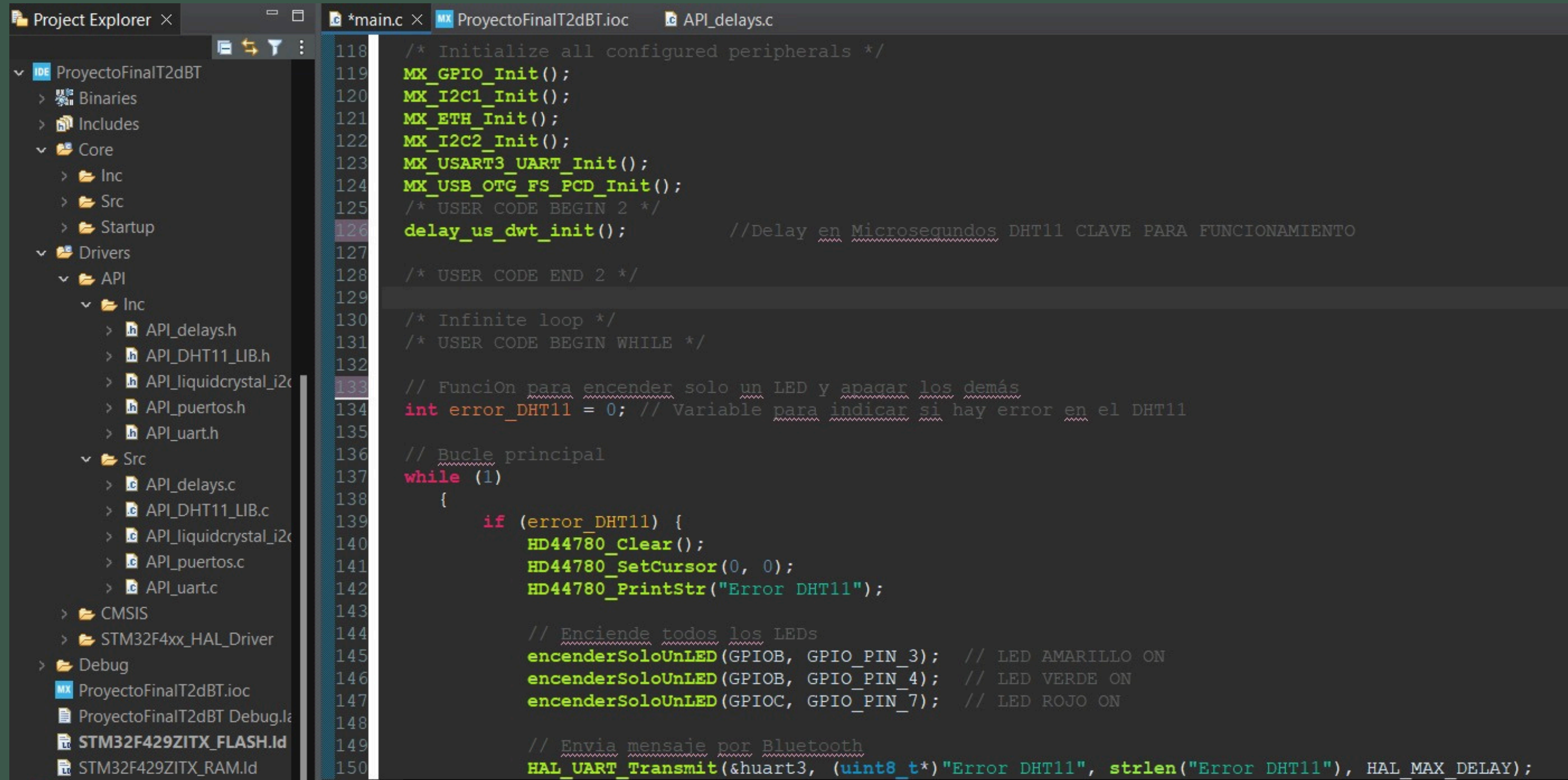


# Consideraciones sobre el software



```
55 /* USER CODE BEGIN PV */
56 char texto[100];
57 char texto2[16];           // Texto2 Muestra Temperatura
58 char texto3[16];           // Texto3 Muestra Humedad
59 unsigned int DHT11_TEMP;    // Declarar en el main
60 unsigned int DHT11_HUM;     // Declarar en el main
61
62 /* USER CODE END PV */
63
64 /* Private function prototypes -----*/
65 void SystemClock_Config(void);
66 /* USER CODE BEGIN PFP */
67 void MX_USART3_UART_Init(void);
68 void MX_USB_OTG_FS_PCD_Init(void);
69 void encenderSoloUnLED(GPIO_TypeDef *port, uint16_t pin); //FUNCION PARA ACTIVAR LED
70
71 uint8_t estadoErrorDHT11 = 0; // Bandera para indicar si hay error con el sensor
72 /* USER CODE END PFP */
73
74 /* Private user code -----*/
75 /* USER CODE BEGIN 0 */
76
77 /* USER CODE END 0 */
78
79 int main(void)
80 {
81
82     //-----LCD-----//
83     HAL_Init();
84     SystemClock_Config();
85     MX_GPIO_Init();
86     MX_I2C1_Init();
87
```

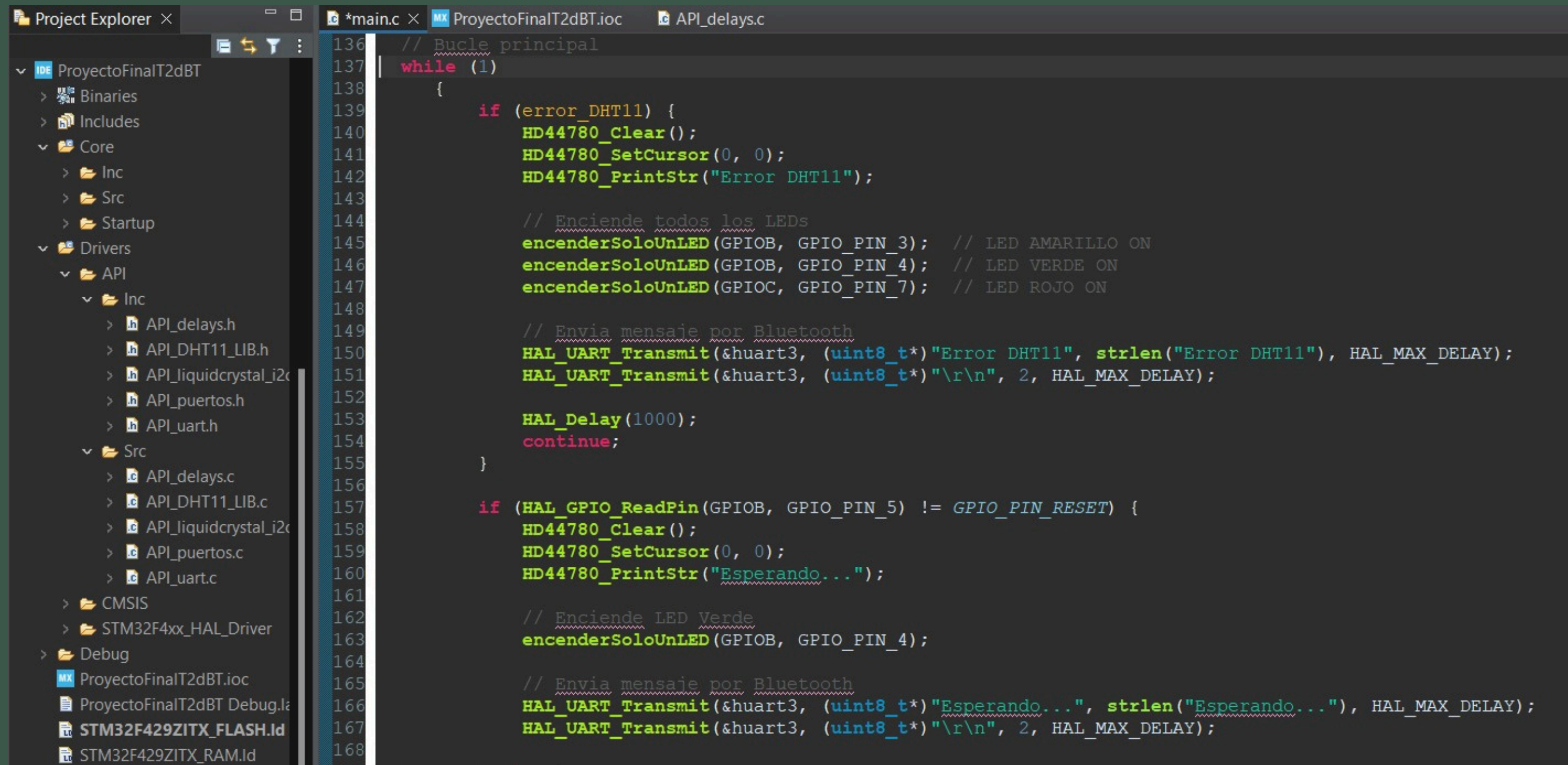
# Consideraciones sobre el software



```
118  /* Initialize all configured peripherals */
119  MX_GPIO_Init();
120  MX_I2C1_Init();
121  MX_ETH_Init();
122  MX_I2C2_Init();
123  MX_USART3_UART_Init();
124  MX_USB_OTG_FS_PCD_Init();
125  /* USER CODE BEGIN 2 */
126  delay_us_dwt_init();           //Delay en Microsegundos DHT11 CLAVE PARA FUNCIONAMIENTO
127
128  /* USER CODE END 2 */
129
130  /* Infinite loop */
131  /* USER CODE BEGIN WHILE */
132
133  // Funcion para encender solo un LED y apagar los demás
134  int error_DHT11 = 0; // Variable para indicar si hay error en el DHT11
135
136  // Bucle principal
137  while (1)
138  {
139      if (error_DHT11) {
140          HD44780_Clear();
141          HD44780_SetCursor(0, 0);
142          HD44780_PrintStr("Error DHT11");
143
144          // Enciende todos los LEDs
145          encenderSoloUnLED(GPIOB, GPIO_PIN_3); // LED AMARILLO ON
146          encenderSoloUnLED(GPIOB, GPIO_PIN_4); // LED VERDE ON
147          encenderSoloUnLED(GPIOC, GPIO_PIN_7); // LED ROJO ON
148
149          // Envia mensaje por Bluetooth
150          HAL_UART_Transmit(&huart3, (uint8_t*)"Error DHT11", strlen("Error DHT11"), HAL_MAX_DELAY);
```



# Consideraciones sobre el software



```
136 // Bucle principal
137 while (1)
138 {
139     if (error_DHT11) {
140         HD44780_Clear();
141         HD44780_SetCursor(0, 0);
142         HD44780_PrintStr("Error DHT11");
143
144         // Enciende todos los LEDs
145         encenderSoloUnLED(GPIOB, GPIO_PIN_3); // LED AMARILLO ON
146         encenderSoloUnLED(GPIOB, GPIO_PIN_4); // LED VERDE ON
147         encenderSoloUnLED(GPIOC, GPIO_PIN_7); // LED ROJO ON
148
149         // Envia mensaje por Bluetooth
150         HAL_UART_Transmit(&huart3, (uint8_t*)"Error DHT11", strlen("Error DHT11"), HAL_MAX_DELAY);
151         HAL_UART_Transmit(&huart3, (uint8_t*)"\r\n", 2, HAL_MAX_DELAY);
152
153         HAL_Delay(1000);
154         continue;
155     }
156
157     if (HAL_GPIO_ReadPin(GPIOB, GPIO_PIN_5) != GPIO_PIN_RESET) {
158         HD44780_Clear();
159         HD44780_SetCursor(0, 0);
160         HD44780_PrintStr("Esperando...");
161
162         // Enciende LED Verde
163         encenderSoloUnLED(GPIOB, GPIO_PIN_4);
164
165         // Envia mensaje por Bluetooth
166         HAL_UART_Transmit(&huart3, (uint8_t*)"Esperando...", strlen("Esperando..."), HAL_MAX_DELAY);
167         HAL_UART_Transmit(&huart3, (uint8_t*)"\r\n", 2, HAL_MAX_DELAY);
168     }
```



# Consideraciones sobre el software

```
188 //Enviamos datos combinados por Bluetooth utilizando USART3
189 HAL_UART_Transmit(&huart3, (uint8_t*)mensaje_combinado, strlen(mensaje_combinado), HAL_MAX_DELAY);
190 HAL_UART_Transmit(&huart3, (uint8_t*)"\r\n", 2, HAL_MAX_DELAY);
191
192 encenderSoloUnLED(GPIOB, GPIO_PIN_3); // Enciende LED Amarillo
193 } else {
194 HD44780_Clear();
195 HD44780_SetCursor(0, 0);
196 HD44780_PrintStr("Error DHT11");
197
198 //Enciende todos los LEDs
199 encenderSoloUnLED(GPIOB, GPIO_PIN_3); // LED Amarillo
200 encenderSoloUnLED(GPIOB, GPIO_PIN_4); // LED VERDE ON
201 encenderSoloUnLED(GPIOC, GPIO_PIN_7); // LED ROJO ON
202
203 //Enviar mensaje por Bluetooth
204 HAL_UART_Transmit(&huart3, (uint8_t*)"Error DHT11", strlen("Error DHT11"), HAL_MAX_DELAY);
205 HAL_UART_Transmit(&huart3, (uint8_t*)"\r\n", 2, HAL_MAX_DELAY);
206
207 error_DHT11 = 1;
208 }
209 }
210 HAL_Delay(1000); //Esperar antes de la siguiente lectura
211 }
212 }
213
214 void encenderSoloUnLED(GPIO_TypeDef *port, uint16_t pin) {
215 HAL_GPIO_WritePin(GPIOC, GPIO_PIN_7, GPIO_PIN_RESET); // LED ROJO OFF
216 HAL_GPIO_WritePin(GPIOB, GPIO_PIN_4, GPIO_PIN_RESET); // LED VERDE OFF
217 HAL_GPIO_WritePin(GPIOB, GPIO_PIN_3, GPIO_PIN_RESET); // LED Amarillo OFF
218 HAL_GPIO_WritePin(port, pin, GPIO_PIN_SET); // Enciende el LED seleccionado
219 }
220
```





**MUCHAS GRACIAS!!**