Projemizdeki amac , Arduino’nun mikrodenetleyicisi icerisinde kaydedilen UID degerlerinden biri eger RFID iceren kart icerisindeki UID’lerden biriyse bu karta onceden atanmis bakiye bir azalacaktir ve Arduino’nun yedinci pinine bagli olan yesil LED yanacaktir. Eger bakiye sifirsa gecisi izin verilmeyecektir ve Arduino’nun yedinci pinine bagli olan sari LED yanacaktir. Eger kart , EEPROM’a kayitli degilse gecersiz kart olarak sayilacaktir ve Arduino’nun yedinci pinine bagli olan kirmizi LED yanacaktir. Kartimizi okuttuktan sonra onemli olan iki tane veri UID ve bakiyesidir. Bunun icin bu iki tane veriyi tek bir nesne altinda toplamamiz gerekir. CardInformation adinda bir yapi olusturarak bakiyeyi integer olarak , UID ise 32 karakterlik char dizisi olarak bir yapi altinda tutuyoruz. Projemizde iki farkli olayi gosterebilmek icin baslangic olarak writeInitialValues metodu icerisinde iki tane bu yapidan ornek olusturup bunlari EEPROM icerisine yazdirdik. String ile calisirken diger programlama dillerinde farkli olarak direkt esitleme C++ uzerinde isaretleyicilerle saglanir cunku string , referans tipinde bir veri tipidir. Strcpy metodu ile elimizde olan UID leri , olusturdugumuz yapilarin char dizilerine isaretliyoruz.

Normalde EEPROM kutuphanesini kullanarak ilkel veri tipleri olan char , integer gibi veri tiplerini kendi kutuphanesindeki metotlarla kaydedip tekrardan bilgileri okuyabiliriz. Fakat okuyacagimiz bilgi daha karmasik yapida bir veri tipi oldugu icin bu bilgiyi isaretleyiciler ile tasimak gerekir. En asagida yer alan writeAnythingtoEEPROM ile metot icerisindeki parametrede yazilan herhangi bir degiskeni EEPROM uzerine yazabilmekte , readAnythingFromEEPROM ile metot icerisinde parametrede yazilan parametre icerisine okunan degeri kopyalayabilmektedir. Template komutu ise herhangi bir degiskeni parametre olarak verebilecegimizi gosterir. Projeyi ilk defa calistirdiktan sonra tekrardan writeInitialValues metodunu yorum satirina alirsak baslangic degerlerini EEPROM uzerine kaydetmek yerine degisen degerlerle devam edecektir. Loop fonksiyonun basinda , fonksiyonun sonunda islemlerden sonra LED’lerden birini aktif edecegimiz icin hepsini inaktif durumuna getiriyoruz. Basit kart okuma projemizde oldugu gibi herhangi bir RFID iceren kart okunana dek return komutu araciligiyla tekrardan kosullarin oldugu komuta yonelerek fonksiyonun ilerlemesini engelliyoruz. Eger kosul saglanirsa artik karttaki UID degerini okuyabiliriz demektir. UID degerlerlerinin tek karakterlerini on altilik sayi sistemine donusturup hepsini string veri tipine donusturup birlestirdikten sonra elimizde duzgun bir UID degeri olmus oluyor. EEPROM uzerinde kaydedilmis bir deger olup olmadigina isCardExists metodu icerisinde kontrol edecegiz.

Oncesinde EEPROM uzerinden okudugumuz degerleri CardInformation yapisindan olusan statik diziye kopyaladik. Gelen bilgileri dongu icerisine atarak eger dongudeki yapilardan birinde ayni UID varsa geri donus degeri olarak dizide hangi indexte ise o degeri yoksa -1 gonderecektir. Metot uzerinden gelen bilgiyi kosulladiktan sonra bir sonraki kosullayacagimiz islem acaba okutulan kartta bakiyenin sifir olup olmadigidir. Eger sifirsa altinci portta bulunan sari LED yanacaktir ve yetkisiz gecis cunku bakiyeniz sifirdir seklinde bir mesaji seri monitore yollayacaktir. Eger sifir degilse yedinci portta olan yesil LED yanacaktir ve aldigimiz index degeri sayesinde yapimizdeki bakiye degerini bir azaltip tekrardan EEPROM uzerine yazacaktir. Seri monitore ise yetkili giris yazip bakiyesini yazacaktir. isCardExists metodundan geriye -1 donerse else kismi uygulanacaktir. Else kisminda ise besinci portta olan kirmizi LED yanacaktir ve seri monitore gecersiz kart mesaji gonderecektir. Her mesaj gonderildikten sonra loop fonksiyonuna tekrardan donup karti tekrardan okuyacagi icin araya uc saniyelik bir gecikme koymamiz gerekir. Projemiz ileride Arduino yerine NodeMCU ESP8266 veya ESP32 kullanilarak bu UID degerlerini API uzerinden alarak daha dinamik hale getirilebilir.