

**KOCAELİ SAĞLIK VE TEKNOLOJİ ÜNİVERSİTESİ**

**MÜHENDİSLİK VE DOĞA BİLİMLERİ FAKÜLTESİ**

**BİLGİSAYAR/YAZILIM MÜHENDİSLİĞİ**

**PROGRAMLAMA LAB II / MANTIK KAPILARI**

**PROJE 3**

**ÖĞRENCİ ADI: Yiğit Samet Kalkan**

**ÖĞRENCİ NUMARASI: 220502042**

**GITHUB: github.com/yigitskalkan**

**ÖĞRENCİ ADI: Emirhan Dursun**

**ÖĞRENCİ NUMARASI: 220502039**

**GITHUB: github.com/renola41**

**DERS SORUMLUSU:**

**PROF. DR ÖĞR. ÜYESİ Nevcihan DURU**

**TARİH:02.06.2024**

**1 GİRİŞ**

**1.1 Projenin Amacı**

Bu projenin amacı, kullanıcıların basit mantık devrelerini tasarlayabilmeleri, simüle edebilmeleri ve test edebilmeleri için bir dijital platform geliştirmektir. Geliştirilecek olan bu platform, kullanıcılara mantık kapıları, giriş ve çıkış elemanları, bağlantı elemanları ve kontrol tuşları gibi araçları kullanarak dijital devreler oluşturma imkanı sunar. Bu sayede, kullanıcılar dijital devre tasarımı ve mantık kapıları hakkında pratik yapabilir ve bilgi sahibi olabilirler. Proje, eğitim amacıyla kullanılabilecek interaktif bir araç olarak tasarlanmıştır.

**2 GEREKSİNİM ANALİZİ**

**2.1 Arayüz Gereksinimleri**

Arayüz gereksinimleri, kullanıcının platformla nasıl etkileşime gireceğini ve hangi araçları kullanabileceğini belirler. Bu gereksinimler aşağıdaki gibi tanımlanmıştır:

**1. Ana Pencere ve Araç Çubuğu:**

* Mantık kapılarını, giriş ve çıkış elemanlarını, bağlantı elemanlarını ve kontrol tuşlarını içeren bir araç çubuğu.
* Araç çubuğunda her bir mantık kapısı ve devre elemanı için ayrı ayrı düğmeler.
* Tasarım alanında devre elemanlarının ve bağlantıların yerleştirilebileceği bir alan.

**2. Mantık Kapıları:**

* VE, VEYA, DEĞİL, XOR gibi temel mantık kapıları.
* Her bir kapı için belirli sayıda giriş bağlantısı.

**3.Giriş ve Çıkış Elemanları:**

* Giriş pinleri: Başlangıç değerleri ayarlanabilir.
* Çıkış pinleri ve LED'ler: Devrenin çıkış değerlerini gösterir.

**4. Bağlantı Elemanları:**

* Devre elemanları arasında bağlantı kurmak için kablolar.
* Bağlantı düğümleri: Bağlantı hatlarının birleştirildiği noktalar.

**5.Kontrol Tuşları:**

* Çalıştır, durdur ve reset tuşları: Simülasyonu başlatmak, durdurmak ve resetlemek için.

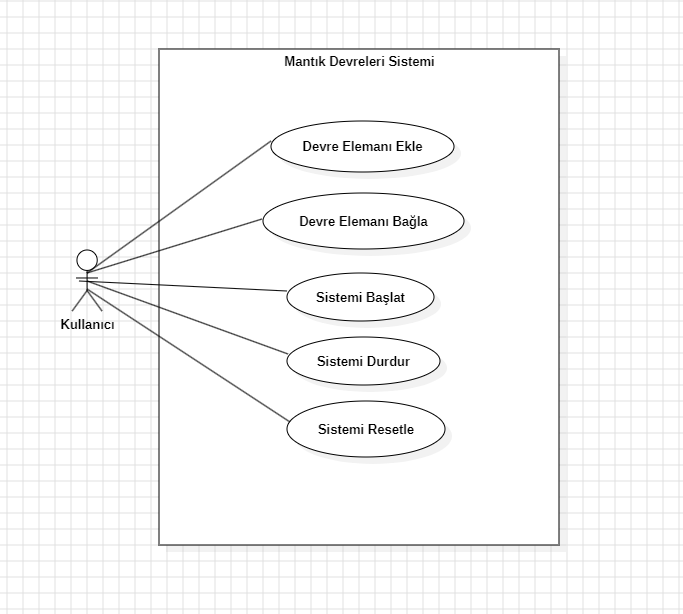
**6.Özellik Tablosu:**

* Mantık kapıları, giriş/çıkış elemanları ve bağlantı elemanları için etiket ve renk gibi özelliklerin görüntülenmesi ve düzenlenmesi.
* Bu özelliklerin, elemanların üzerine sağ tıklanarak erişilebilmesi

**2.2 Fonksiyonel Gereksinimler**

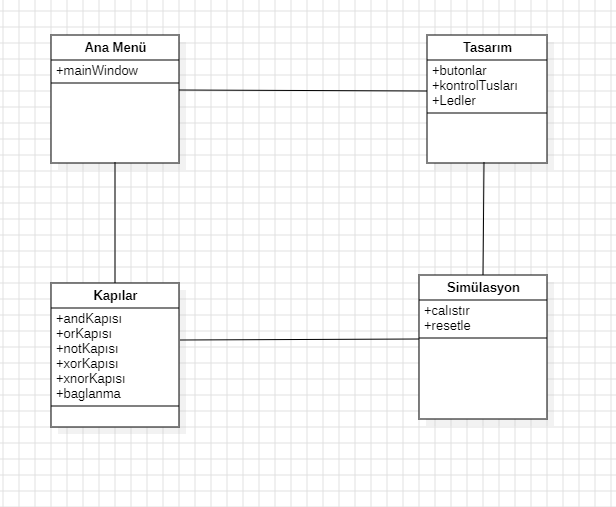
Fonksiyonel gereksinimler, platformun nasıl çalışacağını ve hangi işlemleri gerçekleştireceğini belirler. Bu gereksinimler aşağıdaki gibi tanımlanmıştır:

1. **Devre Elemanlarının Eklenmesi:**
   * Kullanıcı, araç çubuğundaki düğmelere tıklayarak tasarım alanına mantık kapıları, giriş ve çıkış elemanları ekleyebilir.
   * Aynı elemandan birden fazla eklenebilir.
2. **Bağlantıların Kurulması:**
   * Devre elemanları arasında kablo bağlantıları kurularak giriş ve çıkışlar bağlanabilir.
   * Bağlantı hatları, gerektiğinde bağlantı düğümleri kullanılarak birleştirilebilir.
3. **Özelliklerin Görüntülenmesi ve Düzenlenmesi:**
   * Her bir devre elemanının özellikleri (etiket, renk, vb.) sağ tıklanarak görüntülenebilir ve düzenlenebilir.
   * Giriş elemanlarının başlangıç değerleri ayarlanabilir.
4. **Simülasyon İşlemleri:**
   * Çalıştır tuşuna basılarak devre simülasyonu başlatılabilir.
   * Simülasyon sırasında kullanıcı giriş elemanlarının değerlerini değiştirebilir.
   * Devre simülasyonu çalışırken çıkış pinleri veya LED'ler çıkış değerlerini gösterir.
   * Durdur tuşuna basılarak simülasyon durdurulabilir.
   * Reset tuşuna basılarak devre başlangıç durumuna getirilebilir.
5. **Devre Tasarımının Saklanması ve Yüklenmesi:**
   * Kullanıcı, tasarladığı devreyi kaydedebilir ve daha sonra yükleyebilir (ek gereksinim).
   * **2.3 Use-Case Diyagramı**



**3. TASARIM**

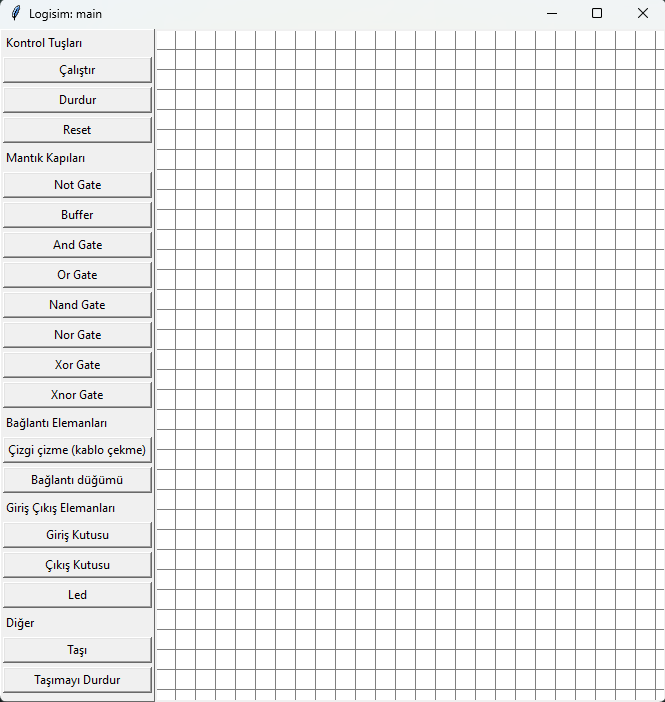
* 1. **Mimari Tasarım**



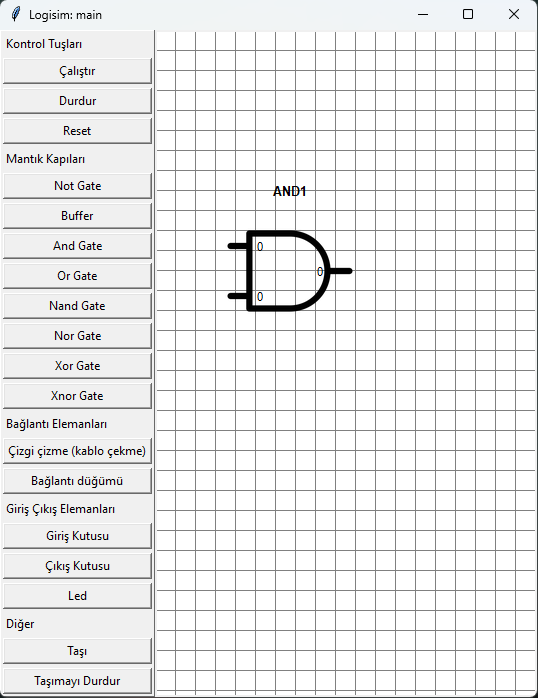
**3.2 Kullanılan Teknolojiler:**

* Yazılım dili için Python.
* Arayüz tasarımı ve devre çalıştırılması için tkinter kütüphanesi
* Sistem Kontrolü için sys kütüphanesi
* Visual Studio Code

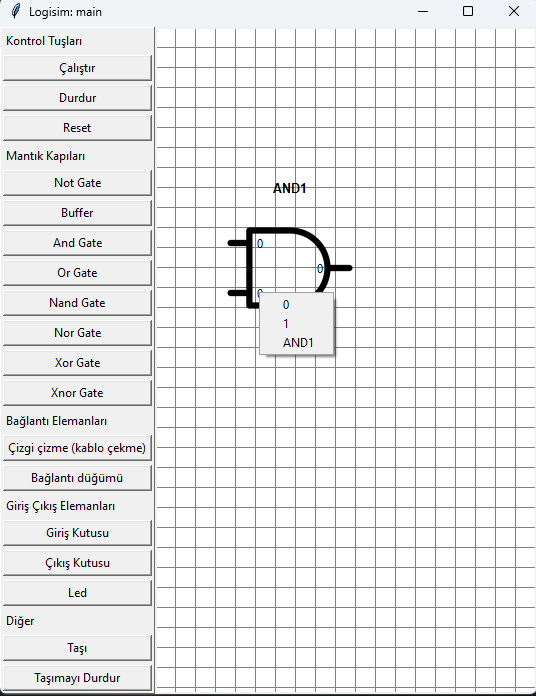
**3.3 Kullanıcı Arayüzü Tasarımı**



Arayüzümüz bu şekilde. Kullanıcıyı sol kısımda belli butonlar karşılamakta. Bu butonlardakullanıcı sistemi çalıştırabilir, Mantık kapılarından istediğini menüye getirebilir, bu kapılarla bağlantı kurabilir ve sonuç alabilir. Örneğin AND kapısını simülasyonumuza ekleyelim:



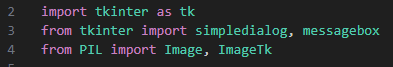
AND bloğuna tıkladığımızda default olarak böyle bir görsel çıkmakta. Buradan sol üst ve sol alt tarafta görebileceğiniz şekilde değerleri girebilir ve çıktısını görebiliriz. Örnek olarak gösterecek olursak:



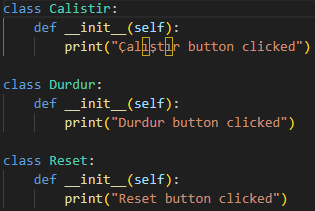
Burada örnek olarak sol alttaki 0’a tıkladığımızda girebileceğimiz değerler çıkmakta. İki taraf için de 1’i seçme durumumuzda çıktımız da 1 olacaktır.

**4.UYGULAMA**

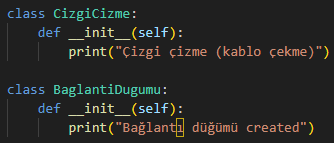
**4.1 Kod Blokların Açıklamaları**



* **tkinter**: Python'un standart GUI kütüphanesidir. Kullanıcı arayüzü oluşturmak için kullanılır. **tkinter** modülü, pencere, düğme, menü gibi birçok GUI bileşenini içerir.
* **simpledialog** ve **messagebox**: **tkinter** modülünün alt bileşenleridir. Basit diyalog pencereleri ve mesaj kutuları oluşturmak için kullanılır.
* **PIL (Python Imaging Library)**: **PIL**, resim işleme için kullanılır. **Image** ve **ImageTk** modülleri resim yükleme ve Tkinter ile kullanmak için dönüştürme işlevlerini sağlar.



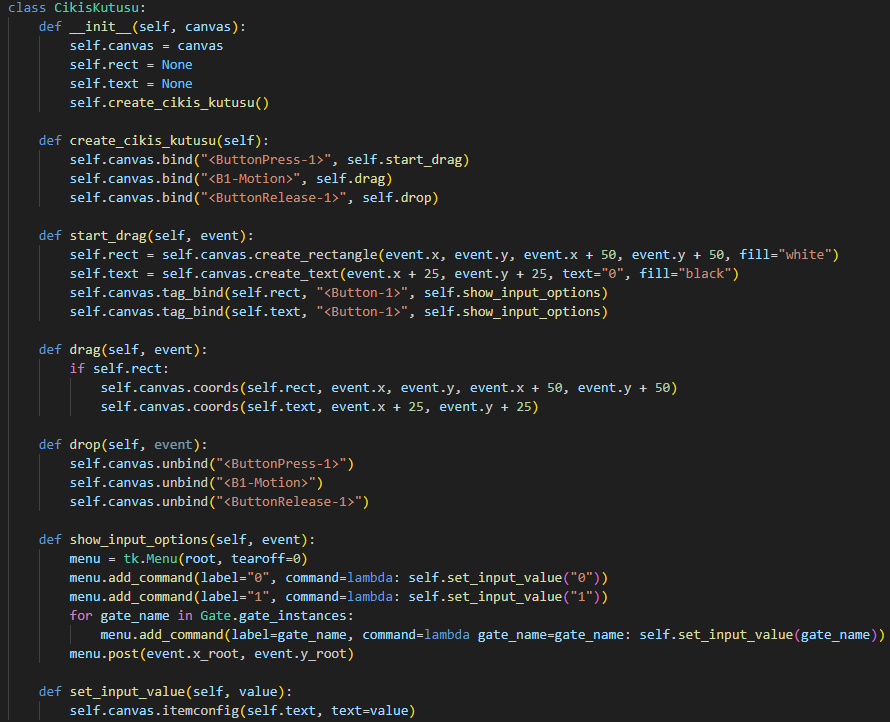
* **Calistir, Durdur, Reset Sınıfları**: Bu sınıflar, uygulamanın çalıştırma, durdurma ve sıfırlama işlevlerini temsil eder. **\_\_init\_\_** metotları, butonların tıklama olaylarını terminale yazdırır. Gerçek işlevsellik eklenmediği için placeholder (yer tutucu) olarak işlev görürler.



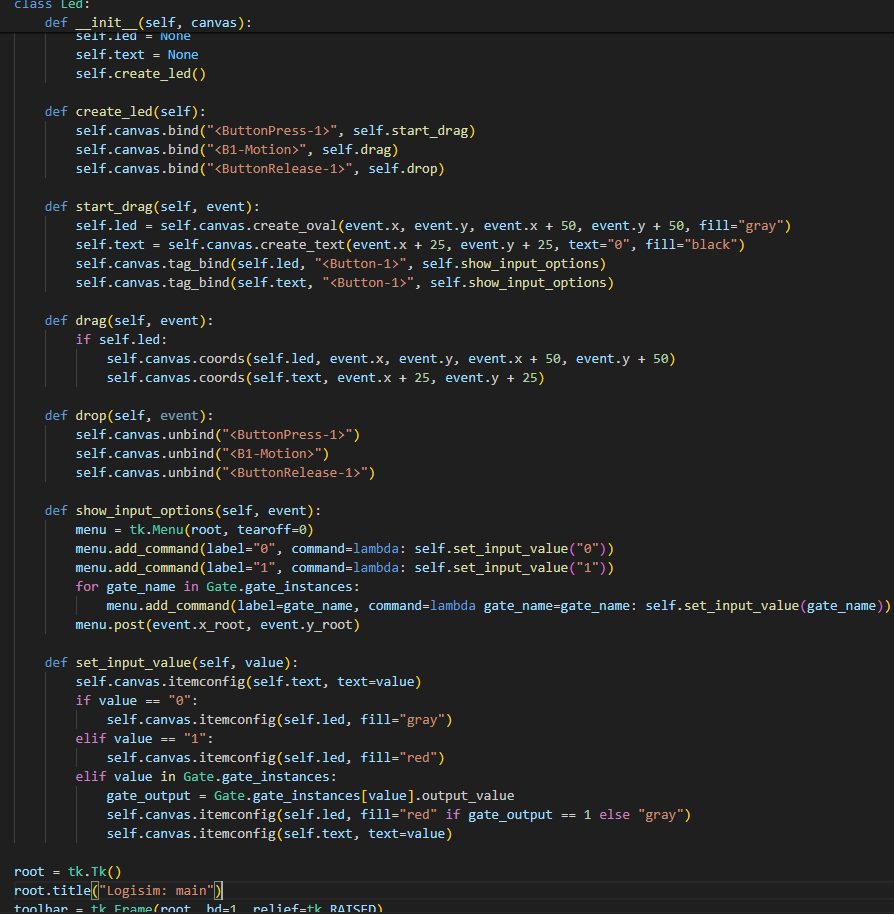
* **CizgiCizme ve BaglantiDugumu Sınıfları**: Bu sınıflar, devre elemanları arasında bağlantılar oluşturmak için kullanılır. **\_\_init\_\_** metotları, çizgi çizme ve bağlantı düğümü oluşturma işlemlerini başlatır ve terminale mesaj yazdırır. Henüz detaylı işlevsellik eklenmemiştir.



* **GirisKutusu Sınıfı**: Bu sınıf, kullanıcıların devreye giriş kutusu eklemelerine ve etkileşimde bulunmalarına olanak tanır.
  + **init Metodu**: Giriş kutusunu oluşturur ve gerekli değişkenleri tanımlar. **create\_giris\_kutusu** metodunu çağırarak fare olaylarını bağlar.
  + **create\_giris\_kutusu Metodu**: Fare olaylarını (tıklama, sürükleme, bırakma) bağlar.
  + **start\_drag Metodu**: Kullanıcı fareye bastığında, giriş kutusunu oluşturur. Dikdörtgen ve içinde bir metin elemanı çizer.
  + **drag Metodu**: Fare sürükleme olayında, dikdörtgenin ve metin elemanının konumunu günceller.
  + **drop Metodu**: Fareyi bıraktığında, olayları çözerek sürükleme işlemini durdurur.
  + **toggle\_value Metodu**: Kullanıcı dikdörtgeni veya metin elemanını tıkladığında, giriş kutusunun değerini "0" ve "1" arasında değiştirir.

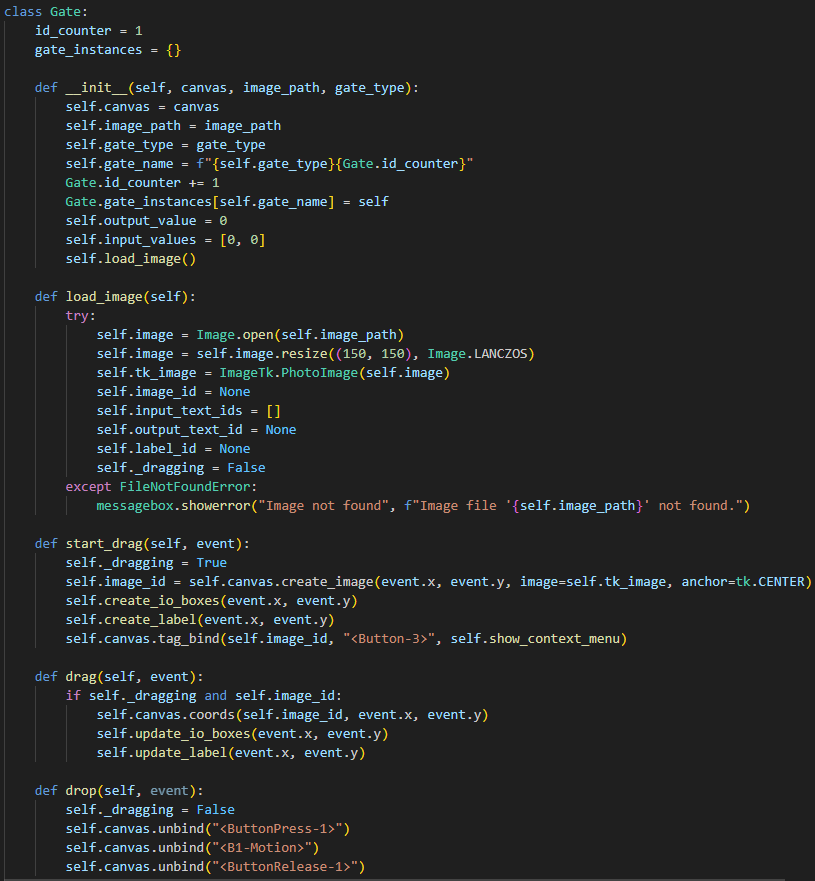


* **CikisKutusu Sınıfı**: Bu sınıf, kullanıcıların devreye çıkış kutusu eklemelerine ve etkileşimde bulunmalarına olanak tanır.
  + **init Metodu**: Çıkış kutusunu oluşturur ve gerekli değişkenleri tanımlar. **create\_cikis\_kutusu** metodunu çağırarak fare olaylarını bağlar.
  + **create\_cikis\_kutusu Metodu**: Fare olaylarını (tıklama, sürükleme, bırakma) bağlar.
  + **start\_drag Metodu**: Kullanıcı fareye bastığında, çıkış kutusunu oluşturur. Dikdörtgen ve içinde bir metin elemanı çizer.
  + **drag Metodu**: Fare sürükleme olayında, dikdörtgenin ve metin elemanının konumunu günceller.
  + **drop Metodu**: Fareyi bıraktığında, olayları çözerek sürükleme işlemini durdurur.
  + **show\_input\_options Metodu**: Kullanıcı dikdörtgeni veya metin elemanını sağ tıkladığında, bir menü gösterir. Menüden bir değer seçilebilir.
  + **set\_input\_value Metodu**: Menüden seçilen değeri çıkış kutusuna atar ve metin elemanını günceller.

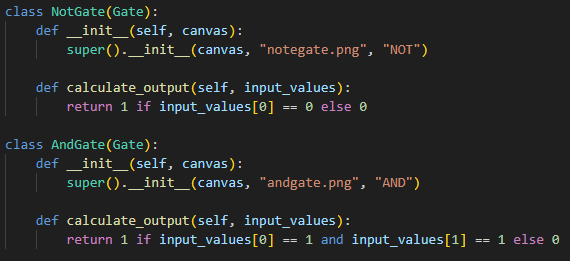


* **Led Sınıfı**: Bu sınıf, kullanıcıların devreye LED eklemelerine ve etkileşimde bulunmalarına olanak tanır.
  + **init Metodu**: LED'i oluşturur ve gerekli değişkenleri tanımlar. **create\_led** metodunu çağırarak fare olaylarını bağlar.
  + **create\_led Metodu**: Fare olaylarını (tıklama, sürükleme, bırakma) bağlar.
  + **start\_drag Metodu**: Kullanıcı fareye bastığında, LED'i oluşturur. Dikdörtgen ve içinde bir metin elemanı çizer.
  + **drag Metodu**: Fare sürükleme olayında, LED'in ve metin elemanının konumunu günceller.
  + **drop Metodu**: Fareyi bıraktığında, olayları çözerek sürükleme işlemini durdurur.
  + **show\_input\_options Metodu**: Kullanıcı LED'i sağ tıkladığında, bir menü gösterir. Menüden bir değer seçilebilir.
  + **set\_input\_value Metodu**: Menüden seçilen değeri LED'e atar ve LED'in rengini değiştirir. Bağlı bir mantık kapısının çıkış değeri ise bu değere göre LED'in rengini günceller.

**GATE SINIFI:**



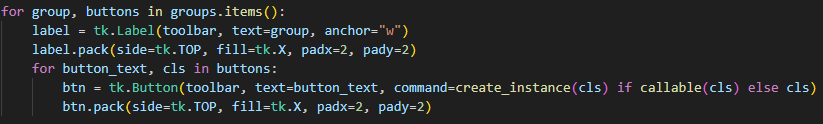
* **id\_counter:** Her yeni kapıya benzersiz bir kimlik atamak için kullanılan bir sayaçtır.
* **gate\_instances:** Tüm kapıların örneklerini saklayan bir sözlüktür. Bu, kapıların adlarıyla erişilmesine olanak tanır.
* **load\_image Fonksiyonu:** Bu metod, kapının resmini yükler ve gerekli boyutlandırmaları yapar
* **start\_drag Metodu :** Bu metod, kapı sürüklenmeye başlandığında çağrılır.
* **event:** Fare olayını temsil eder.
* **\_dragging:** Kapının şu anda sürüklenip sürüklenmediğini belirten bir bayrak.
* **create\_io\_boxes():** Giriş ve çıkış kutularını oluşturur**.**
* **create\_label():** Kapının adını içeren etiketi oluşturur.
* **show\_context\_menu():** Sağ tıklama menüsünü gösterir.
* **drop Metodu:** Kapı bırakıldığında çağrılır ve sürükleme işlemini sonlandırır.



* **NotGate, AndGate vb. Sınıfları:** Bu sınıflar, mantıksal kapıların özelleştirilmiş sürümlerini temsil eder. Her biri **Gate** sınıfını miras alır ve giriş değerlerine göre çıkış değerini hesaplamak için **calculate\_output** metodunu uygular. Diğer mantık kapılarının tanımları da bu şekilde devam eder.



* **create\_gate\_instance ve create\_instance Fonksiyonları:** Bu fonksiyonlar, mantık kapılarının veya diğer elemanların oluşturulmasını sağlar. Fare olaylarını bu kapılara bağlar.
* **enable\_move\_mode ve disable\_move\_mode Fonksiyonları:** Bu fonksiyonlar, elemanların sürüklenmesini sağlar. Sürükleme modunu açar veya kapatır.



Bu döngü, araç çubuğunda gruplandırılmış düğmelerin oluşturulmasını sağlar. Her grup için bir etiket ve düğmeler oluşturulur. Düğmelerin komutları**, create\_instance** fonksiyonu aracılığıyla belirlenir. Bu şekilde, her düğmeye tıklandığında ilgili öğe oluşturulur.