T.C
KOCAELİ SAĞLIK VE TEKNOLOJİ ÜNİVERSİTESİ
MÜHENDİSLİK VE DOĞA BİLİMLERİ FAKÜLTESİ
BİLGİSAYAR/YAZILIM MÜHENDİSLİĞİ

PROJE KONUSU: Basit mantık devrelerini tasarlamak için bir platform geliştirmek

ÖĞRENCİ ADI: Zeynep KEDİKLİ-Nurdan BULUT
ÖĞRENCİ NUMARASI: 220501012-220502021

DERS SORUMLUSU:
PROF. DR./DR. ÖĞR. ÜYESİ Ercan ÖLÇER

TARİH:02.06.2024

1 GİRİŞ

1.1 Projenin amacı

- Bu projenin amacı, basit mantık devrelerini tasarlamak için bir platform geliştirmektir. Bu platform, kullanıcıların farklı mantık kapılarını ekleyebilecekleri, bu kapıları bağlayabilecekleri ve ardından tasarladıkları devreyi simüle edebilecekleri bir ortam sunar. Temel olarak, kullanıcıların sayısal tasarım konseptlerini öğrenmelerini ve uygulamalarını sağlamak için bir araç sunmak amaçlanmıştır.

1.2 Projede beklenenler

- **Platform Geliştirme:**

Kullanıcıların basit mantık devrelerini tasarlamak için kullanabilecekleri bir platformun geliştirilmesi.

- **Araçlar Bölümü:**

Platformun araçlar bölümünde, farklı mantık kapılarını eklemek için kullanılacak araçların bulunması.

- **Özellik Tabloları:**

Her bir aracın özelliklerini gösteren özellik tablolarının oluşturulması ve bu tabloların kullanıcılar tarafından görüntülenebilir ve düzenlenebilir olması.

- **Simülasyon Yeteneği:**

Tasarlanan devrenin simüle edilebilmesi için çalıştır, reset ve durdur tuşlarının platforma eklenmesi.

- **Giriş/Çıkış Elemanları:**

Giriş kutuları, çıkış kutuları ve LED gibi elemanların kullanıcılar tarafından devreye eklenebilmesi.

- **Bağlantı Elemanları:**

Çizgi çizme ve bağlantı düğümleri gibi elemanların kullanıcılar tarafından kullanılabilmesi.

- **Kullanıcı Etkileşimi:**

Kullanıcıların simülasyon sırasında her bir elemanın giriş değerlerini değiştirebilmesi ve çıkışın görsel olarak takip edilebilmesi.

- **Performans ve Kullanıcı Dostu Arayüz:**

Platformun performanslı çalışması ve kullanıcı dostu bir arayüz sunması.

2 GEREKSİNİM ANALİZİ

2.1 Arayüz gereksinimleri

- **Kullanıcı Dostu Arayüz:**
Platformun kullanıcı dostu bir arayüzle donatılması, kullanıcıların kolayca araçlara erişebilmesi ve tasarım yapabilmesi için önemlidir.

- **Araçlar Menüsü:**
Kullanıcıların araçları seçebilecekleri ve tasarım alanına ekleyebilecekleri bir araçlar menüsü olmalıdır. Bu menü, mantık kapıları, giriş/çıkış elemanları ve bağlantı elemanları gibi araçları içermelidir.

- **Tasarım Alanı:**
Kullanıcıların devrelerini tasarlayacakları ana çalışma alanı, net ve düzenli bir şekilde sunulmalıdır. Bu alanda kullanıcılar, araçları sürükleyip bırakarak tasarım yapabilmelidir.

- **Özellikler ve Değerler:**
Her bir elemanın özelliklerini ve değerlerini göstermek için bir panel veya açılır menü bulunmalıdır. Bu özellikler, elemanın tipi, giriş/çıkış sayısı, başlangıç değeri, rengi vb. olabilir.

- **Simülasyon Kontrolleri:**
Çalıştır, reset ve durdur gibi simülasyon kontrolleri, kullanıcıların tasarımlarını simüle etmelerini sağlamak için görünür olmalıdır. Bu kontroller, tasarım alanının yanında veya üstünde yer alabilir.

- **Giriş Değerlerinin Değiştirilmesi:**
Kullanıcıların her bir elemanın giriş değerlerini kolayca değiştirebilmeleri için bir mekanizma sağlanmalıdır. Bu, kullanıcıların tasarımın davranışını test etmelerine olanak tanır.

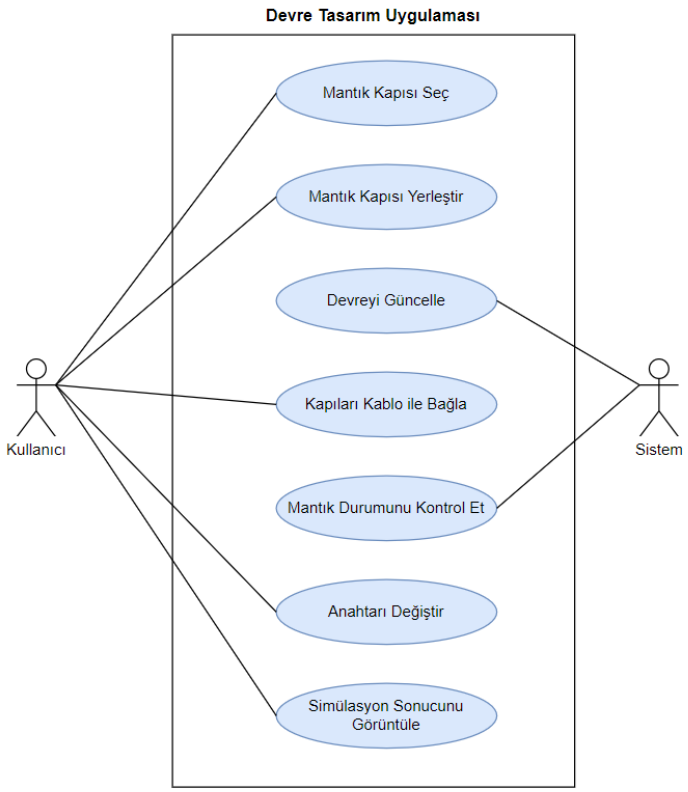
- **Çıkış Göstergeleri:**
Tasarımın çıkış değerlerini göstermek için LED'ler veya çıkış kutuları gibi görsel göstergeler bulunmalıdır. Bu, kullanıcıların tasarımın işlevselliğini anlamasına ve takip etmesine yardımcı olur.

- **Kullanıcı Geri Bildirimi:**
Kullanıcıların yaptıkları değişiklikler hakkında geri bildirim alabilecekleri ve hataları düzeltebilecekleri bir mekanizma sağlanmalıdır.

2.2 Fonksiyonel gereksinimler

- **Mantık Kapıları Ekleme:** Kullanıcıların platformda bulunan mantık kapılarından seçim yaparak tasarım alanına ekleyebilmesi.
- **Bağlantı Kurma:** Kullanıcıların mantık kapılarını birbirine bağlayarak devreler oluşturabilmesi.
- **Simülasyon Yürütme:** Kullanıcıların tasarımı simüle ederek çıkışların doğruluğunu kontrol edebilmesi.
- **Giriş Değerlerini Değiştirme:** Kullanıcıların simülasyon sırasında giriş değerlerini değiştirerek tasarımın dinamik davranışını gözlemleyebilmesi.
- **Simülasyon Kontrolleri:** Kullanıcıların simülasyonu başlatma, duraklatma, yeniden başlatma gibi kontrolleri kullanarak tasarımı yönetebilmesi.
- **Çıkış Göstergeleri:** Kullanıcıların tasarımın çıkış değerlerini doğrudan görebileceği göstergelerin sağlanması.

2.3 Use-Case diyagramı



3 TASARIM

3.1 Mimari tasarım

- **Sınıf Yapısı**:

Programın ana bileşenleri, farklı mantık kapılarını, soketleri, anahtarları ve ampulleri temsil eden sınıflardan oluşur. Her bir bileşen, Pygame'in Sprite sınıfından türetilmiştir, bu da onları ekranda kolayca görselleştirmeyi sağlar.

- **Miras Alma (Kalıtım)**:

Her mantık kapısı, `MantikKapisi` sınıfından miras alır. Bu, her mantık kapısının temel özelliklerini (örneğin, konum ve giriş/çıkış soketleri) paylaşmasını sağlar. Farklı mantık kapıları için özelleştirilmiş mantık işlemleri bu alt sınıflarda gerçekleştirilir.

- **Soketler ve Bağlantılar**:

Soketler, bileşenler arasındaki bağlantı noktalarını temsil eder. Her mantık kapısı, giriş ve çıkış soketlerine sahiptir. Bu soketler, birbirleriyle bağlanarak mantık kapılarının birbirine bağlanmasını sağlar.

- **Mantık İşlemleri**:

Her mantık kapısı sınıfı, `mantikIslemi` adlı bir yöntem içerir. Bu yöntem, o mantık kapısının spesifik mantık işlemini gerçekleştirir. Örneğin, VE kapısı, girişlerinden her ikisinde de akım varsa çıkışa akım gönderir.

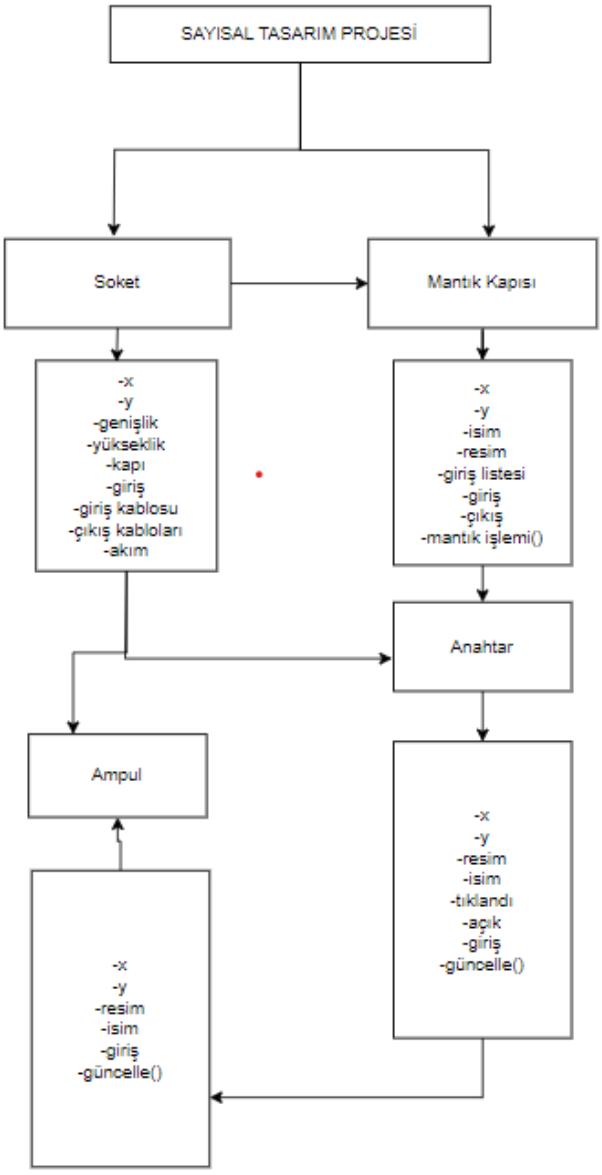
- **Anahtarlar ve Ampuller**:

Anahtarlar, mantık devresinde akımı açıp kapatmak için kullanılır. Ampuller ise devrenin çıkışını görsel olarak temsil eder. Her ikisi de mantık kapılarına bağlanabilir.

- **Polimorfizm (Çok Biçimlilik)**:

Mantık kapısı sınıflarının çoğu, `mantikIslemi` yöntemini geçersiz kılar ve kendi mantık işlemlerini uygular. Bu, her mantık kapısının kendi özel mantık işlevini gerçekleştirebilmesini sağlar.

- **Modül Diyagramı:**



3.2 Kullanılacak teknolojiler

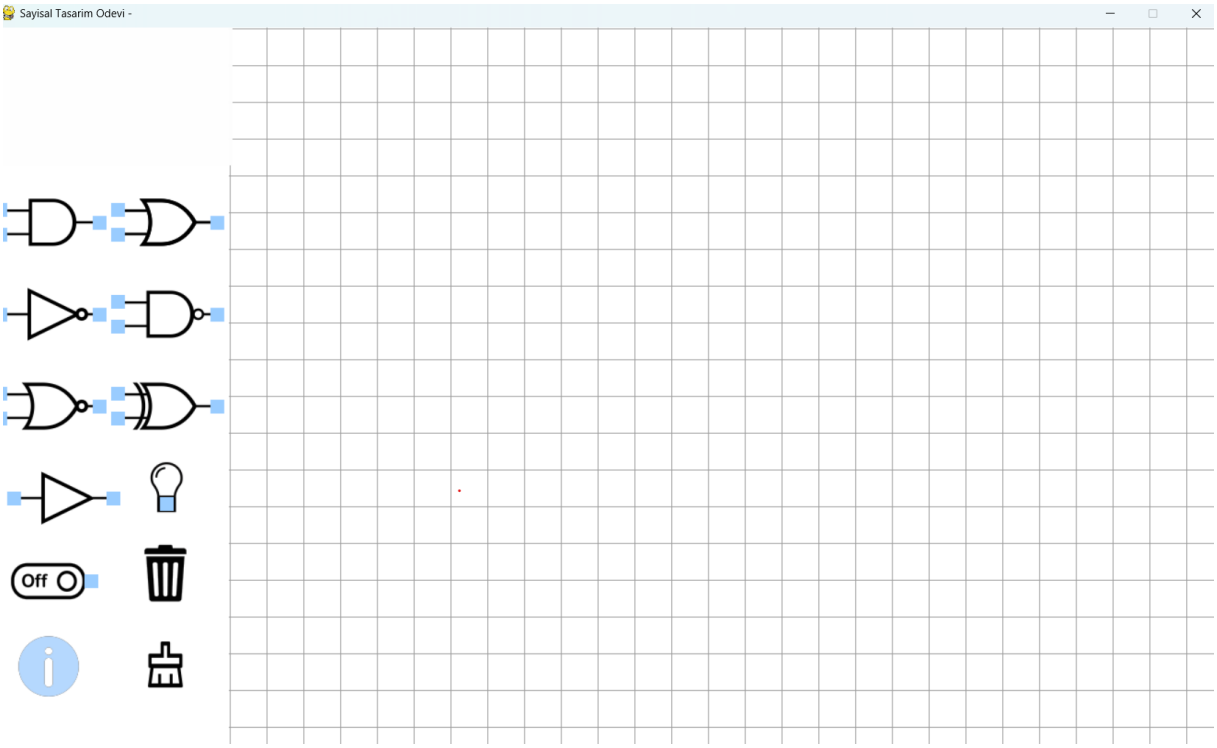
- **Python Programlama Dili:** Yazılımın temelini oluşturan dil Python'dur..

- **Pygame Kütüphanesi:** Pygame, Python dilinde oyun ve grafik uygulamaları geliştirmek için kullanılan bir kütüphanedir. Bu kütüphane, çeşitli grafik nesnelerini oluşturma, oyun döngüsü yönetimi, fare ve klavye etkileşimlerini işleme gibi işlevler sağlar.

3.3 Arayüz tasarımı

Pygame, Python'da oyun geliştirmek için sıklıkla kullanılan bir kütüphanedir ve kullanıcı arayüzü tasarımı için temel çizim ve etkileşim işlevselliği sağlar.

Kodda `Soket`, `MantikKapisi`, `Anahtar` ve `Ampul` gibi sınıflar bulunuyor. Bu sınıfların çoğu, oyun ekranında görüntülenebilecek nesneleri temsil eder. Örneğin, `Soket` sınıfı, oyun ekranında soketlerin ve bağlantıların görsel temsili için kullanılır. Benzer şekilde, `Anahtar` ve `Ampul` sınıfları da görsel nesneleri temsil eder.



4 UYGULAMA
4.1 Kodlanan bileşenlerin açıklamaları

- **Soket Sınıfı**: Soket sınıfı, mantık kapıları arasındaki giriş ve çıkış bağlantılarını temsil eder. Soketler, bileşenlerin giriş ve çıkışlarını sağlar ve bağlantılara izin verir. Her mantık kapısının giriş soketleri ve bir çıkış soketi vardır, her biri Kablo sınıfının bir nesnesine bağlanabilir.

    - __init__Metodu: Bu metod, Soket sınıfının başlatıcısıdır. Soketin boyutlarını ve konumunu belirler, soketin ait olduğu mantık kapısını ve soketin giriş mi çıkış mı olduğunu belirler. Ayrıca, soketin başka bir sokete bağlı olup olmadığını, sokete bağlı kabloları ve soketten akım geçip geçmediğini takip eder.

- **MantikKapisi Sınıfı**: MantikKapisi sınıfı, tüm mantık kapısı türleri için üst sınıf olarak işlev görür. Bu sınıf, mantık kapılarının ortak özelliklerini ve davranışlarını tanımlar ve diğer mantık kapısı sınıfları bu sınıftan türetilir.

    - __init__ Metodu: Bu metod, MantikKapisi sınıfının başlatıcısıdır. Mantık kapısının görselini, adını ve konumunu belirler. Eğer mantık kapısı "DEGILKapisi" veya "BUFFERKapisi" ise, tek bir giriş soketi oluşturur; aksi takdirde iki giriş soketi oluşturur. Ayrıca, çıkış soketini de oluşturur.

    - mantikIslemi Metodu: Bu metod, mantık kapısının gerçekleştireceği mantıksal işlemi tanımlar. Varsayılan olarak OR kapısı mantığını uygular; yani, girişlerden herhangi birinde akım varsa çıkışta akım olur.

- **VEKapisi Sınıfı**: VEKapisi sınıfı, MantikKapisi sınıfından türetilen ve AND mantık kapısını temsil eden bir sınıftır. Bu sınıf, AND kapısının mantıksal işlemini gerçekleştirir.

    - mantikIslemi Metodu: Bu metod, AND kapısının mantıksal işlemini tanımlar. Girişlerin ikisinde de akım varsa (self.girisA.akim and self.girisB.akim), çıkışta akım olur (self.cikis.akim = True). Aksi takdirde, çıkışta akım olmaz (self.cikis.akim = False).

- **VEYAKapisi Sınıfı**: VEYAKapisi sınıfı, MantikKapisi sınıfından türetilen ve OR mantık kapısını temsil eden bir sınıftır. Bu sınıf, OR kapısının mantıksal işlemini gerçekleştirir.

    - mantikIslemi Metodu: Bu metod, OR kapısının mantıksal işlemini tanımlar. Girişlerden herhangi birinde akım varsa (self.girisA.akim or self.girisB.akim), çıkışta akım olur (self.cikis.akim = True). Aksi takdirde, çıkışta akım olmaz (self.cikis.akim = False).

- **DEGILKapisi Sınıfı**: DEGILKapisi sınıfı, MantikKapisi sınıfından türetilen ve NOT mantık kapısını temsil eden bir sınıftır. Bu sınıf, NOT kapısının mantıksal işlemini gerçekleştirir.

    - mantikIslemi Metodu: Bu metod, NOT kapısının mantıksal işlemini tanımlar. Girişte akım varsa (self.giris.akim), çıkışta akım olmaz (self.cikis.akim = False). Girişte akım yoksa (self.giris.akim), çıkışta akım olur (self.cikis.akim = True).

- **VEDEGILKapisi Sınıfı**: VEDEGILKapisi sınıfı, MantikKapisi sınıfından türetilen ve mantık kapısını temsil eden bir sınıftır. Bu sınıf, NAND kapısının mantıksal işlemini gerçekleştirir.

    - mantikIslemi Metodu: Bu metod, NAND kapısının mantıksal işlemini tanımlar. Girişlerin ikisinde de akım yoksa (not (self.girisA.akim and self.girisB.akim)), çıkışta akım olur (self.cikis.akim = True). Aksi takdirde, çıkışta akım olmaz (self.cikis.akim = False).

- **VEYADEGILKapisi Sınıfı**: VEYADEGILKapisi sınıfı, MantikKapisi sınıfından türetilen ve NOR mantık kapısını temsil eden bir sınıftır. Bu sınıf, NOR kapısının mantıksal işlemini gerçekleştirir.

    - mantikIslemi Metodu: Bu metod, NOR kapısının mantıksal işlemini tanımlar. Girişlerin ikisinde de akım yoksa (not self.girisA.akim and not self.girisB.akim), çıkışta akım olur (self.cikis.akim = True). Aksi takdirde, çıkışta akım olmaz (self.cikis.akim = False).

- **OZELVEYAKapisi Sınıfı**: OZELVEYAKapisi sınıfı, MantikKapisi sınıfından türetilen ve mantık kapısını temsil eden bir sınıftır. Bu sınıf, XOR kapısının mantıksal işlemini gerçekleştirir.

    - mantikIslemi Metodu: Bu metod, XOR kapısının mantıksal işlemini tanımlar. Girişlerden yalnızca birinde akım varsa ((self.girisA.akim or self.girisB.akim) and (not self.girisA.akim or not self.girisB.akim)), çıkışta akım olur (self.cikis.akim = True). Aksi takdirde, çıkışta akım olmaz (self.cikis.akim = False).

- **BUFFERKapisi Sınıfı**: BUFFERKapisi sınıfı, MantikKapisi sınıfından türetilen ve BUFFER mantık kapısını temsil eden bir sınıftır. Bu sınıf, BUFFER kapısının mantıksal işlemini gerçekleştirir.

    - mantikIslemi Metodu: Bu metod, BUFFER kapısının mantıksal işlemini tanımlar. Girişte akım varsa (self.giris.akim), çıkışta da akım olur (self.cikis.akim = True). Girişte akım yoksa (self.giris.akim), çıkışta da akım olmaz (self.cikis.akim = False).

- **Anahtar Sınıfı**: Anahtar sınıfı, mantık devresinde akımı açıp kapatmaya yarayan anahtarı temsil eder. Bu sınıf, anahtarın durumunu (açık/kapalı) ve çıkış akımını yönetir.

    - guncelle Metodu: Bu metod, anahtarın durumuna (açık/kapalı) göre görselini ve çıkış akımını günceller. Eğer anahtar açıksa (self.acik), görseli açık anahtar resmi (acikResim) olur ve çıkışta akım olur (self.cikis.akim = True). Eğer anahtar kapalıysa (self.acik), görseli kapalı anahtar resmi (kapaliResim) olur ve çıkışta akım olmaz (self.cikis.akim = False).

- **Ampul Sınıfı**: Ampul sınıfı, mantık devresinde akımı göstermek için kullanılan ampulü temsil eder. Bu sınıf, ampulün durumunu (yanıyor/sönük) ve giriş akımını yönetir.

    - guncelle Metodu: Bu metod, ampulün durumuna (yanıyor/sönük) göre görselini günceller. Eğer girişte akım varsa (self.giris.akim), ampulün görseli yanıyor resmi (acikResim) olur. Eğer girişte akım yoksa (self.giris.akim), ampulün görseli sönük resmi (kapaliResim) olur.

- **CopKutusu Sınıfı**: CopKutusu sınıfı, mantık devresinde bileşenlerin ve kabloların silinmesini sağlayan çöp kutusunu temsil eder. Bu sınıf, çöp kutusunun özelliklerini ve durumunu yönetir.

- **BilgiMenusu Sınıfı**: BilgiMenusu sınıfı, mantık kapısı simülatöründe kullanıcıya nasıl kullanılacağını ve her bileşenin nasıl çalıştığını açıklayan bilgi menüsünü temsil eder.

- **Kablo Sınıfı**: Kablo sınıfı, farklı mantık kapısı bileşenlerini birbirine bağlayan kabloları temsil eder. Bu sınıf, kablonun başlangıç ve bitiş noktalarını, rengini ve genişliğini yönetir.

    - ciz Metodu: Bu metod, kabloyu ekrana çizer. pygame.draw.line fonksiyonunu kullanarak kablonun rengini, başlangıç ve bitiş noktalarını ve genişliğini belirler.

- **FareImleci Sınıfı**: FareImleci sınıfı, kullanıcının fare imlecini temsil eder. Bu sınıf, fare ile bileşenleri sürükleyip bırakma işlemlerini gerçekleştirir ve sürüklenen mantık kapılarının, soketlerin ve kabloların pozisyonlarını günceller.

    - guncelle Metodu: Bu metod, farenin ve taşıdığı nesnelerin pozisyonunu günceller. Fare hareket ettiğinde, sürüklenen bileşenler, soketler ve bağlı kabloların pozisyonlarını günceller. Kullanıcı fareyi kullanarak bileşenleri sürükleyip bırakabilir ve soketler arasında kablo bağlantıları oluşturabilir.

- **YanMenu Sınıfı**: FareImleci sınıfı, kullanıcının fare imlecini temsil eder. Bu sınıf, fare ile bileşenleri sürükleyip bırakma işlemlerini gerçekleştirir ve sürüklenen mantık kapılarının, soketlerin ve kabloların pozisyonlarını günceller.

- ornekOlustur Metodu: Bu metod, yan menüdeki bileşenlerin ilk örneklerini oluşturur ve ilgili sprite gruplarına ekler. Bileşenlerin konumlarını ve türlerini belirler.
- tiklandigindaOrnekOlustur Metodu: Bu metod, yan menüden bir bileşen sürüklendiğinde yeni örnekler oluşturur ve ilgili sprite gruplarına ekler. Bu, yan menüden sürüklenen bileşenlerin yeniden oluşturulmasını sağlar.

- spriteCiz Metodu: Bu metod, mantık kapılarını, soketleri ve kabloları çizer. Ayrıca, anahtarın ve ampulün durumlarını günceller.

- **Main Fonksiyonu**: Bu fonksiyon, ana oyun döngüsünü içerir ve program çalıştırıldığında çağrılır. Pygame başlatılır, ekran ve saat ayarları yapılır. Yan menü, fare imleci, bilgi menüsü ve çöp kutusu oluşturulur. Ana döngü içinde, kullanıcı etkileşimleri (fare tıklamaları ve hareketleri) işlenir, bileşenler ve kablolar güncellenir ve ekran sürekli olarak yeniden çizilir. Çöp kutusu ile bileşenlerin silinmesi ve akım iletimi gibi işlemler de burada gerçekleştirilir. Oyun penceresi kapatıldığında pygame sonlandırılır.

4.2 Görev dağılımı

- Projede pair programming yapılmışır.
- Raporun 1, 2, 3.1 ve 3.2 maddeleri Zeynep kedikli tarafından yazılmıştır.
- Raporun 3.3, 3.4, 4 ve 5. Maddeleri Nurdan Bulut trafından yapılmışıtr

4.3 Karşılaşılan zorluklar ve çözüm yöntemleri

- Pygame kütüphanesi özelliklerini kavrama
- Mantık kapıları, kablolar ve diğer bileşenler için kullanıcı dostu bir arayüz oluşturmak.
- Sürükle ve bırak işlevselliğinin sağlanması ve bileşenlerin doğru bir şekilde bağlanması karmaşıklığı.
- Kısıtlı zaman.
- Birçok problem pygame kütüphanesi özellikleri sayesinde çözüldü.
- Proje planını yaparken de çeşitli zorluklar yaşadık.

4.4 Proje isterlerine göre eksik yönler

- Giriş – Çıkış kutusu bulunmuyor.
- Bağlantı düğümü bulunmuyor.
- Özellik tablosundaki değerler tasarım alanına yerleştirilen elemanlar üzerinde sağ fare tuşu ile tıklanarak görüntülenemiyor.
- Çalıştır, reset ve durdur tuşları bulunmuyor.

5 TEST VE DOĞRULAMA

5.1 Yazılımın test süreci

- Yazılım manuel olarak test edilmiştir. Tüm ister ve özelliklerin manuel testleri yapılmıştır.

5.2 Yazılımın doğrulanması

- Manuel test yapılırken devre çiziminde çöpe sürüklenmiş öğelerle bazı durumlarda kendiliğinden bağlantı kurmaktadır, bu problem dışında tasarlanan devrelerin sorunsuz olarak çalıştığı görülmüştür.

NURDAN BULUT
https://github.com/nurdanbulut

ZEYNEP KEDİKLİ
https://github.com/Zeynepkedikli