BÖLÜM 1: PROJEYE GİRİŞ VE TEMEL AMAÇLAR

Siber Güvenlik Farkındalık Oyunu Projesi

Bu sunum, siber güvenlik farkındalık oyunu projesinin tüm aşamalarını, kullanılacak teknolojileri ve ekip çalışma sürecini detaylandırmaktadır. Her bir geliştirme aşaması, teknik kavramlar ve uygulama detayları ile birlikte açıklanmıştır.

Slayt 2: Projenin Özü ve Vizyonu

Nedir?

Bu proje, kullanıcıları siber güvenlik alanında eğitmeyi amaçlayan bir yazılım projesidir.

Temel Hedef

Kullanıcıları "Phishing" ve "Şifre Güvenliği" gibi günümüzün en yaygın ve kritik siber tehditleri hakkında bilinçlendirmektir.

"Phishing (Oltalama)"

Kullanıcıların parolaları, kredi kartı bilgileri gibi hassas verilerini çalmak amacıyla gönderilen sahte e-postalar, mesajlar veya web siteleri aracılığıyla yapılan bir dolandırıcılık türüdür.

"Şifre Güvenliği"

Kişisel hesapları korumak için tahmin edilmesi zor, karmaşık ve benzersiz parolalar oluşturma ve yönetme pratiğidir.

Teknoloji: Proje, **"Java"** programlama dili kullanılarak geliştirilecektir.

Yaklaşım: Etkileşimli Deneyim

Proje, **"etkileşimli"** bir deneyim sunmayı hedefler. Bu, kullanıcının pasif bir okuyucu olmak yerine, oyun içindeki senaryolara aktif olarak katılması, soruları yanıtlaması ve sonuçlarına göre anında geri bildirim alması demektir. Bu yöntem, öğrenmeyi daha kalıcı ve ilgi çekici hale getirir.

BÖLÜM 2: PLANLAMA VE İÇERİK GELİŞTİRME AŞAMASI

Slayt 3: Aşama 1 - Konsept ve Senaryo Geliştirme

Bu aşama, projenin fikirsel temelinin atıldığı, eğitici içeriğin planlandığı evredir.

"Oyun Senaryosunu Belirleyelim"

Anlamı: Oyunun hikayesini ve akışını oluşturmak. Kullanıcının oyun içinde karşılaşacağı durumları, zorlukları ve görevleri tasarlamak.

Amaç: Senaryolar, kullanıcıyı sadece test etmemeli, aynı zamanda onları "gerçekten düşündürecek" ve onlara "yeni bilgiler katacak" nitelikte olmalıdır. Örneğin, gerçek bir oltalama e-postasını taklit eden bir senaryo oluşturmak.

"İçerik Geliştirelim"

Anlamı: Oyunun eğitim materyallerini, yani bilgilendirici metinleri hazırlamak.

Önemi: Bu içerikler, oyunun bir eğlence aracı olmasının ötesine geçerek "eğitim amacını desteklemesini" sağlar. Kullanıcı bir soruyu yanlış cevapladığında, neden yanlış olduğunu ve doğrusunun ne olması gerektiğini açıklayan metinler sunulmalıdır.

BÖLÜM 3: MİMARİ VE KOD TASARIMI AŞAMASI

Slayt 4: Aşama 2 - Yazılımın İskeletini Oluşturma

Bu aşama, yazılımın nasıl bir yapıya sahip olacağını, kodun nasıl organize edileceğini ve hangi mühendislik prensiplerinin kullanılacağını belirler. Sağlam bir mimari, projenin esnek, yönetilebilir ve uzun ömürlü olmasını sağlar.

Slayt 5: Teknik Prensip 1 - Nesne Yönelimli Programlama (OOP)

"Nesne Yönelimli Yapıyı Kuralım"

Yazılımı, gerçek dünyadaki gibi birbiriyle etkileşen "nesneler" topluluğu olarak tasarlama yöntemidir. Örneğin, bir "Kullanıcı" nesnesi, bir "Soru" nesnesi, bir "Skor" nesnesi olabilir. Bu yaklaşım kodu daha **"modüler"** (parçalara ayrılmış) ve **"anlaşılır"** hale getirir.

Temel Kavramlar:

"Kalıtım (Inheritance)"

Bir sınıfın özelliklerini başka bir sınıfa miras bırakmasıdır. Örneğin, "Çoktan Seçmeli Soru" ve "Doğru/Yanlış Sorusu" sınıfları, ortak özelliklerini (soru metni gibi) genel bir "Soru" sınıfından miras alabilir. Bu, kod tekrarını önler.

"Polimorfizm (Polymorphism)"

"Çok biçimlilik" demektir. Farklı nesnelerin aynı mesaja (metot çağrısına) farklı şekillerde tepki vermesidir. Örneğin, bir skoruHesapla() komutu, Phishing modülü için farklı, Şifre Güvenliği modülü için farklı çalışabilir.

Slayt 6: Teknik Prensip 2 - State Pattern (Durum Tasarım Deseni)

"State Pattern'i Uygulayalım"

Bu, projenin "en önemli gereksinimlerinden biridir". Bir nesnenin, iç durumu değiştikçe davranışlarının da tamamen değişmesini sağlayan bir tasarım desenidir.

State Pattern Nedir?

State Pattern (Durum Tasarım Deseni), bir nesnenin iç durumu değiştiğinde davranışının da değişmesini sağlayan davranışsal bir tasarım desenidir. Bu desen, bir nesnenin farklı durumlarda farklı davranışlar sergilemesini organize bir şekilde yönetir.

Temel Prensipler:

- Her durum ayrı bir sınıf olarak tanımlanır
- Nesne, mevcut durumunu temsil eden bir durum nesnesine sahiptir
- Durum değiştiğinde, nesne farklı bir durum nesnesine geçer
- Her durum nesnesi, o duruma özgü davranışları içerir

Gerçek Hayat Örneği:

Bir ATM makinesi düşünün: Para çekme, para yatırma, bakiye sorgulama gibi farklı durumlarda farklı işlemler yapar. Her durumda kullanıcı etkileşimleri ve sistem tepkileri farklıdır.

Neden Kullanılıyor? Oyunumuz farklı aşamalardan oluşuyor: Giriş Ekranı, Phishing Testi, Şifre Güvenliği Testi, Sonuç Ekranı. Her aşamanın kendine özgü kuralları ve kullanıcı etkileşimleri vardır. State Pattern, bu aşamalar arasındaki geçişleri ve her aşamanın kendi mantığını karmaşık if-else yapıları kullanmadan, temiz ve yönetilebilir bir şekilde kodlamamızı sağlar.

Nasıl Çalışacak?

01	02	03
Her oyun aşaması (Phishing, Şifre Güvenliği vb.) ayrı bir **"Durum (State)"** sınıfı olarak kodlanacaktır. Bunun için bir State arayüzü ve onu uygulayan PhishingState, PasswordSecurityState gibi sınıflar oluşturulacaktır.	Oyunun ana yöneticisi olan **"GameContext"** nesnesi, o an hangi durumda (State) olduğumuzu bilecek ve yönetecek.	Kullanıcı bir eylem yaptığında, GameContext bu eylemi mevcut State nesnesine iletecek. Mevcut State nesnesi de duruma uygun tepkiyi verecek ve gerekirse GameContext'i bir sonraki duruma geçirecektir.

BÖLÜM 4: GELİŞTİRME VE KODLAMA AŞAMASI

Slayt 7: Aşama 3 - Kodun Hayata Geçirilmesi

Bu aşama, tasarlanan mimarinin ve planlanan modüllerin fiilen kodlandığı evredir.

- **"Geliştirme Ortamı: IntelliJ IDEA":** Kodların yazılacağı, düzenleneceği ve test edileceği profesyonel yazılım programıdır. Geliştiricilere kod yazarken yardımcı olan birçok özellik sunar.

"Versiyon Kontrolünü Kullanalım"

Anlamı: Kod üzerinde yapılan tüm değişikliklerin tarihçesini kaydeden bir sistemdir.

Araçlar: "Git" (versiyon kontrol sisteminin kendisi) ve **"GitHub"** veya **"GitLab"** (kodu bulutta saklayan ve paylaşmayı sağlayan platformlar) kullanılacaktır.

Faydaları:

- **Ekip İçi İş Birliği:** Birden fazla geliştiricinin aynı proje üzerinde sorunsuzca çalışmasını sağlar.
- **Kod Güvenliği:** Kodun merkezi bir yerde yedeğinin tutulmasını sağlar. Bilgisayarda oluşabilecek "dual boot gibi risklere" (işletim sistemi sorunları, disk arızaları vb.) karşı tam koruma sağlar.

Slayt 8: Geliştirilecek Ana Parçalar (Modüller)

"Modülleri Geliştirelim": Projenin fonksiyonel kısımlarının ayrı ayrı kodlanmasıdır.

"Phishing Modülü"

Phishing senaryosunu, soruları ve cevap kontrol mekanizmasını içeren kod parçasıdır.

"Şifre Güvenliği Modülü"

Kullanıcıdan şifre oluşturmasını isteyen ve bu şifrenin ne kadar güvenli olduğunu (güçlülük kontrolü) analiz eden mantığı içerir.

"Kullanıcı Sınıfı"

Oyunu oynayan kişinin adını, oyun boyunca topladığı puanı (**"skorunu"**) ve genel başarı düzeyini (**"öğrenme seviyesini"**) saklayan bir veri yapısıdır.

BÖLÜM 5: ANALİZ VE RAPORLAMA AŞAMASI

Slayt 9: Aşama 4 - Performans Ölçümü ve Geri Bildirim

Bu aşama, kullanıcının oyundaki performansını veriye dayalı olarak ölçmeyi ve ona anlamlı geri bildirimler sunmayı hedefe almaktadır.

"Veri Toplamalıyız"

Anlamı: Kullanıcının etkileşimlerini kaydetmek.

Ne Kaydedilecek? Her soruya verilen **"doğru/yanlış"** cevaplar ve bu **"cevapları verme süreleri"**. Süre, kullanıcının bir konuda ne kadar emin olduğunu anlamak için önemli bir veridir.

"Analiz Mekanizması Geliştirelim"

Anlamı: Toplanan ham verileri işleyerek anlamlı sonuçlar üreten kod mantığı.

Ne Hesaplanacak? Kullanıcının genel **"doğru yanıt oranı"** ve **"konu başına başarısı"** (Phishing'de ne kadar başarılı, Şifre Güvenliğinde ne kadar başarılı olduğu) hesaplanacaktır.

"Raporu Oluşturalım"

Anlamı: Analiz sonuçlarını kullanıcıya anlaşılır bir şekilde sunan bir ekran tasarlamak.

İçerik: Rapor, rakamlardan daha fazlasını içermelidir. Kullanıcının **"güçlü ve zayıf yönlerini belirten özet bilgiler"** sunmalıdır. Örneğin: "Oltalama e-postalarını tanımada çok başarılısınız, ancak güvenli şifre oluşturma pratiğinizi geliştirmeniz gerekiyor."

Örnek Rapor Çıktısı:

Kullanıcı: Ahmet Yılmaz

Test Tarihi: 15 Ekim 2024

Toplam Süre: 12 dakika

Genel Performans:

- Toplam Soru: 20
- Doğru Cevap: 16 (80%)
- Yanlış Cevap: 4 (20%)

Modül Bazında Başarı:

- Phishing Testi: 9/10 (90%) - Mükemmel
- Şifre Güvenliği: 7/10 (70%) - Geliştirilmeli

Kişiselleştirilmiş Geri Bildirim:

> "Tebrikler! Oltalama e-postalarını tanımada çok başarılısınız. Şüpheli linkleri ve sahte gönderenler konusunda güçlü bir sezginiz var. Ancak güvenli şifre oluşturma konusunda pratik yapmanızı öneririz. Özellikle özel karakterler ve sayı kombinasyonları kullanımını geliştirin."

Öğrenilen Teknik Kavramlar:

State Pattern

Bir nesnenin durumu değiştikçe davranışının da değişmesini sağlayan tasarım deseni.

OOP (Nesne Yönelimli Programlama)

Yazılımı gerçek dünya nesneleri gibi tasarlama yöntemi.

Git

Kod değişikliklerini takip eden versiyon kontrol sistemi.

Modüler Yapı

Yazılımı küçük, bağımsız parçalara ayırma yaklaşımı.

Öneriler:

- Şifre yöneticisi kullanmayı deneyin
- İki faktörlü doğrulama aktif edin
- Düzenli güvenlik güncellemelerini takip edin

BÖLÜM 6: TEST VE İYİLEŞTİRME AŞAMASI

Slayt 10: Aşama 5 - Kalite Kontrol

Bu son aşama, geliştirilen yazılımın hatasız, güvenilir ve beklentilere uygun çalıştığından emin olmak için yapılır.

"Kapsamlı Testler Yapmalıyız"

Anlamı: Uygulamanın tüm özelliklerini farklı senaryolar altında deneyerek olası hataları bulma sürecidir.

Neler Test Edilecek?

- Her bir oyun aşamasının doğru çalıştığı.
- Durumlar arası geçişlerin (örneğin bir testten diğerine geçişin) sorunsuz olduğu.
- Raporlama verilerinin ve skorların doğru hesaplandığı.