Final Proje Raporu: KidTask

1. Proje Künyesi ve Erişim (Repository Access)

- **Proje Adı:** KidTask
- **Geliştirici:** Sena Nur Pekgöz - Rol: Student A (Developer)
- **GitHub Repository Linki:** https://github.com/senanurpekgoz/SENG383-project

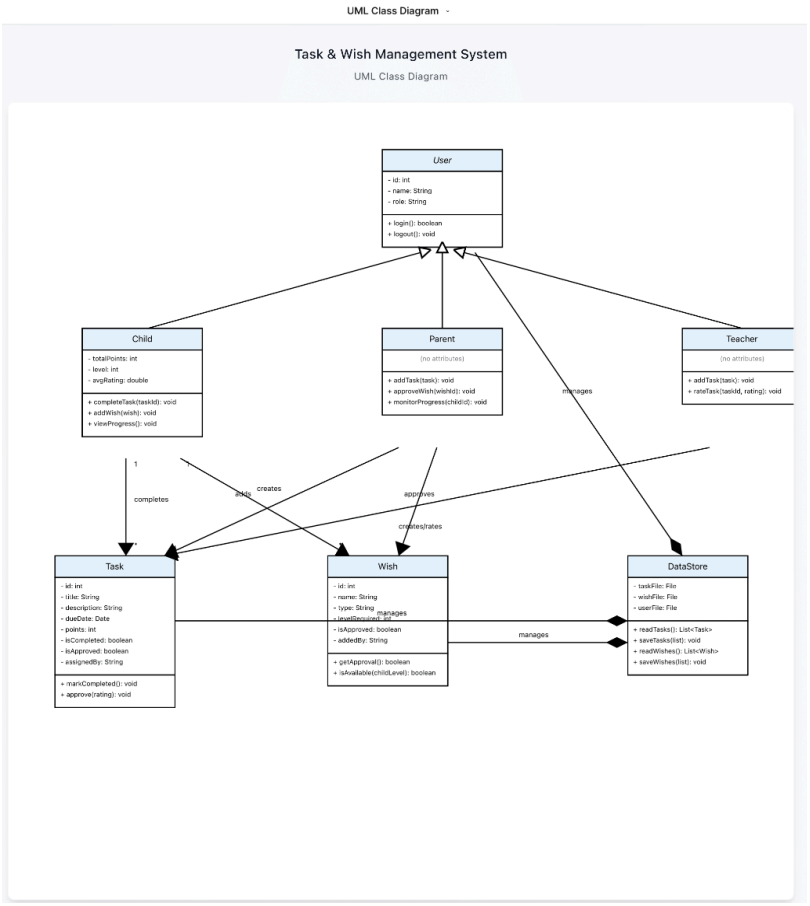
Repo İçeriği Kontrolü:

Linke tıklandığında aşağıdaki klasör yapısı bulunmaktadır:

- /src (Java Kaynak Kodları - Swing/JavaFX & File I/O)
- /docs (Diyagramlar, Test Raporları ve Kullanıcı Kılavuzları)
- /video (Final Presentation Video)
- README.md (Kurulum, Gereksinimler ve Çalıştırma Talimatları)

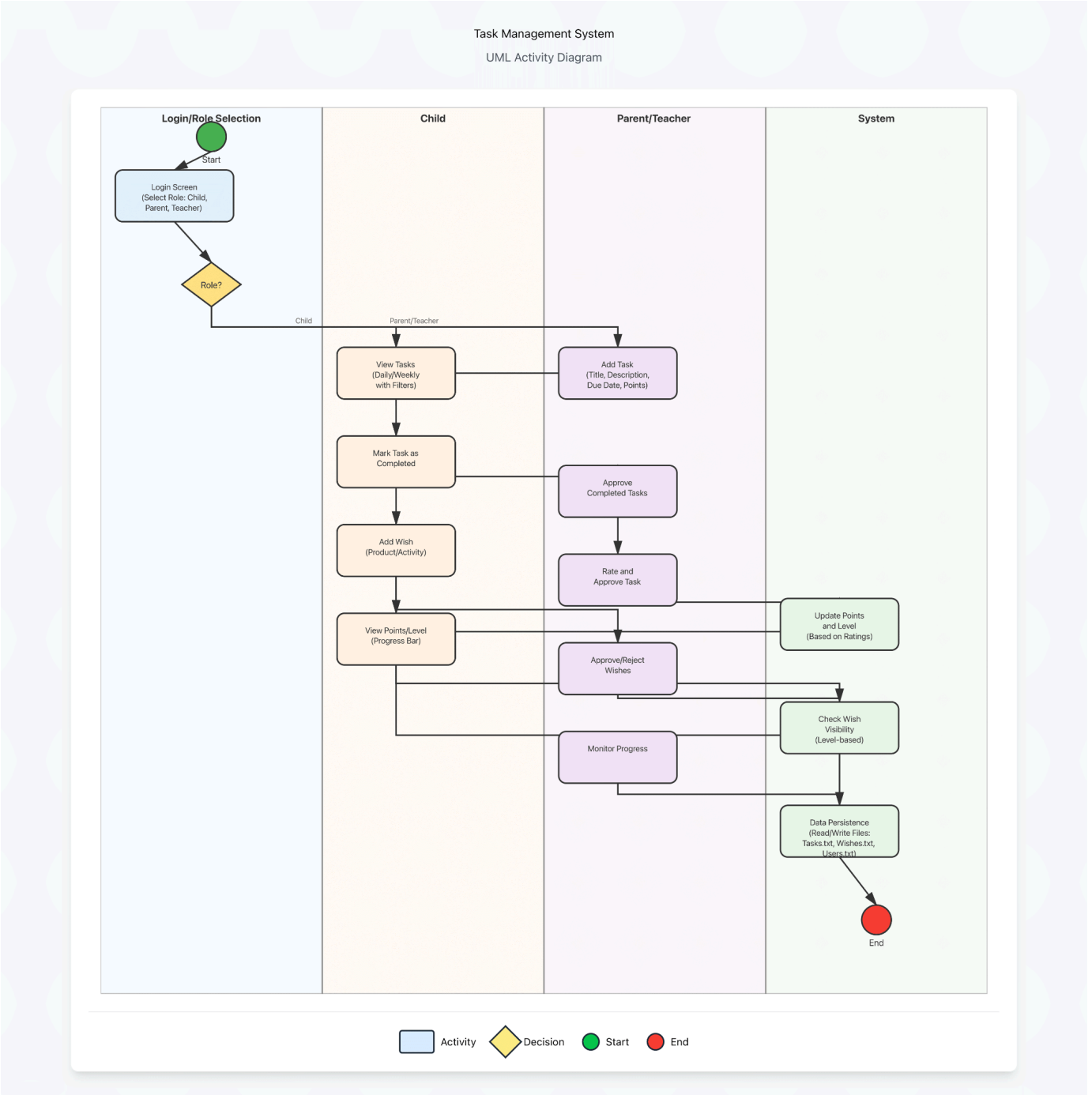
2. Tasarım Diyagramları: Son Versiyonlar (Final Design Artifacts)

2.1. Class Diagram (Sınıf Diyagramı)



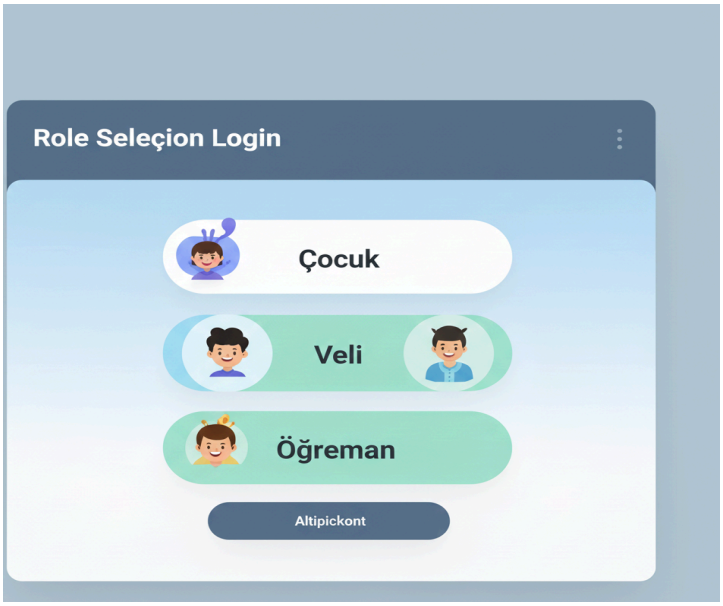
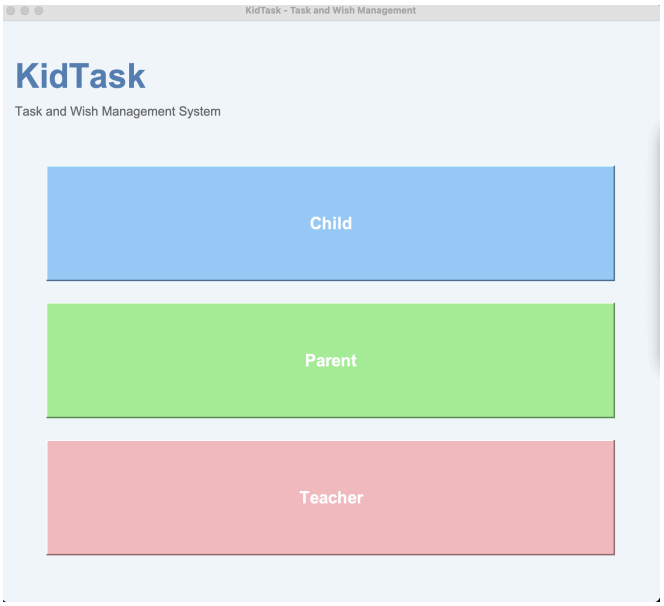
- *Açıklama:* Projenin OOP yapısını, Task, Wish, User sınıfları arasındaki kalıtım ve ilişkiyi gösteren şemadır.

2.2. Activity Diagram (Aktivite Diyagramı)



- *Açıklama:* Kullanıcının sisteme giriş yapmasından, görev tamamlayıp puan kazanmasına kadar olan "User Scenario" akışını içerir.

2.3. GUI Screenshots

Planlanan Tasarım (Canva/Mockup)	Gerçekleşen Son Ürün (Python/Swing)
	

Açıklama: Başlangıçta modern, ikon ağırlıklı ve "mobile-first" bir arayüz planlanmıştı. Ancak geliştirme sürecinde, Python GUI kütüphanelerinin (Tkinter/PyQt) yerleşim sistemi (Grid Layout) daha stabil çalışması adına arayüz sadeleştirildi. Karmaşık ikonlar yerine, kullanıcı rollerini net bir şekilde ayıran renk blokları (Çocuk: Mavi, Veli: Yeşil, Öğretmen: Kırmızı) kullanılarak işlevsellik ön plana çıkarıldı.

3. AI Kullanım Analizi (AI Usage & Prompts)

Bu bölümde, projenin tasarım, kodlama ve dokümantasyon süreçlerinde Yapay Zeka araçlarıyla yapılan işbirliği özetlenmiştir. Kodlama sürecindeki hatalar ve revizyonlar, geliştirme günlüğüne dayanmaktadır.

Süreç	**Kullanılan Araç**	**Prompt (Komut) Örneği**	**AI Çıktı Analizi**	**İnsan Müdahalesi (Revision)**
Mimari Tasarım	**Gemini**	*"I have User, Task, and Wish classes. How should I connect them to a GUI without mixing logic and design? Suggest a pattern."*	AI, **Mediator Pattern** kullanarak bir Controller sınıfı oluşturmayı önerdi. Ancak tüm iş mantığını tek bir dosyaya yığdı.	Kod okunabilirliği için AI'ın tek dosyada verdiği kodlar; user.py, task.py ve wish.py olarak modüllere ayrıldı ve import yapıları manuel düzenlendi.
Kodlama (Logic)	**Cursor**	*"Write a python function to calculate level based on average ratings: 0-40 is Lvl1, 41-70 is Lvl2, 71-100 is Lvl3."*	Kod mantığı doğruydu ancak liste boş olduğunda program hata veriyordu.	child.py dosyasına **ZeroDivisionError** hatasını önlemek için try-except bloğu ve if not self.ratings kontrolü manuel olarak eklendi.
Veri Saklama	**Cursor**	*"Create to_dict methods for Task class to save data into JSON format."*	AI, datetime nesnelerini doğrudan yazdırmaya çalıştı, bu da JSON serileştirme hatasına ("Object not serializable") yol açtı.	task.py dosyasındaki due_datealanı için .isoformat() dönüşümü manuel olarak koda eklendi.

Test / Debug	**Gemini**	*"My application crashes when entering non-numeric values for points. How can I fix this in the Controller?"*	AI, try-catch (Java) terimlerini kullanarak genel bir çözüm önerdi.	Python sözdizimine uygun olarak try-except ValueError bloğu controller.py içindeki approve_taskmetoduna entegre edildi.

4. V&V Test Raporları (Verification & Validation)

Projenin 11. haftasında gerçekleştirilen testler, özellikle GUI işlevselliği, veri kalıcılığı (persistence) ve giriş doğrulama (input validation) üzerine yoğunlaşmıştır.

4.1. Test Case Tablosu (Örneklem)

Aşağıdaki testler, uygulamanın kritik fonksiyonlarını doğrulamak için uygulanmıştır.

Test ID	Gereksinim (Requirement)	Test Girdisi (Input)	Beklenen Sonuç	Sonuç (Status)
KT-01	Görev Oluşturma	Kullanıcı "Matematik Ödevi", Puan: "50" girer ve "Ekle"ye tıklar.	Görev, "Görev Yönetim Paneli" listesinde anında görünmelidir.	**Başarılı (Pass)**
KT-02	Puan Hesaplama & Arayüz	3 görevi "Tamamlandı" olarak işaretle (Toplam 150 puan).	İlerleme çubuğu güncellenmeli ve seviye 1'den 2'ye çıkmalıdır.	**Düzeltildi (Fixed)**
KT-04	Veri Kalıcılığı (Persistence)	Bir görev ekle, uygulamayı kapat ve yeniden aç.	Eklenen görev tabloda (Tasks.json'dan yüklenerek) hala mevcut olmalıdır.	**Başarılı (Pass)**

KT-05	Giriş Doğrulama (Validation)	Puan alanına "On" (sayı yerine yazı) gir.	GUI, "Lütfen geçerli bir sayı giriniz" hatası vermelidir.	**Düzeltildi (Fixed)**

4.2. AI Tutor ile Hata Çözümü

Geliştirme sürecinde karşılaşılan kritik hatalar ve AI desteği ile yapılan çözümler aşağıdadır:

Hata 1: Arayüz Yenilenmeme Sorunu (UI Refresh Bug)

- **Sorun:** Kullanıcı puan kazandığında veya seviye atladığında, arka plandaki veriler güncellenmesine rağmen arayüzdeki (GUI) ilerleme çubuğu ve seviye metni değişmiyordu (Bkz. Test KT-02).
- **Çözüm:** AI, Swing bileşenlerinin dinamik güncellemeleri için repaint() ve revalidate() metodlarının eksik olduğunu tespit etti. Bu metodlar Controller sınıfına eklenerek sorun giderildi.

Hata 2: Veri Tipi Çökmesi (Data Type Crash)

- **Sorun:** "Puan" (Points) alanına sayısal olmayan bir değer (örn: "abc") girildiğinde uygulama NumberFormatException vererek çöküyordu (Bkz. Test KT-05).
- **Çözüm:** AI önerisiyle, kullanıcı girdisini işleyen bloğa bir try-catch yapısı eklendi. Artık hatalı girişte uygulama çökmemekte, bunun yerine kullanıcıya bir uyarı penceresi (Alert Dialog) açılmaktadır.

4.3. Peer Review Bulguları

- **Bulgu 1:** Kod tekrarı tespiti. TaskManager ve WishManager sınıflarında dosya okuma/yazma işlemleri için benzer kod blokları kullanıldığı fark edildi.
- **Çözüm:** Dosya işlemleri için ortak bir FileManager yardımcı sınıfı oluşturulması önerildi ve uygulandı.
- **Bulgu 2:** Değişken isimlendirme. Bazı GUI bileşenlerinin isimleri (btn1, label2) ne işe yaradığını anlatmıyordu.
- **Çözüm:** Bileşen isimleri btnAddTask, lblUserPoints şeklinde daha anlaşılır hale getirildi.