**1. User Class:**

* **Çıkarılanlar**:
  + attempts (Özellik): Artık kullanıcı sınav giriş sayısı, remaining\_attempts ve exam\_status gibi yeni özelliklerle daha dinamik bir şekilde yönetilmektedir.
  + save\_user\_data() ve get\_attempts() gibi metodlar, yeni sistemde daha kapsamlı record\_exam\_score() ve update\_exam\_status() gibi metodlarla değiştirilmiştir. Bu sayede sınav durumu daha iyi yönetilebilmektedir.

**2. Question Class:**

* **Çıkarılanlar**:
  + calculate\_score(self, user\_answers) (Metod): Bu metod, soru bazında puan hesaplamayı içeriyordu ancak bu fonksiyonun sorularla ilgili daha spesifik bir şekilde işlenmesi gerektiği düşünüldü ve genel sonuç hesaplamaları daha çok Exam ve Result sınıflarına taşındı.

**3. Exam Class:**

* **Çıkarılanlar**:
  + display\_question(self, question) ve get\_user\_input(self, question) gibi metodlar, GUI veya komut satırı arayüzü için dışarıda (QUI sınıfında) yönetilecek şekilde dışarıda tutulmuştur.
  + Bu metodlar, kullanıcı etkileşimiyle ilgili olmaktan çok, Exam sınıfının sınav mantığı ve hesaplamaları yönetmesi gerektiği için çıkarılmıştır.

**4. Timer Class:**

* **Çıkarılanlar**:
  + check\_time(self) ve stop\_timer(self) gibi metodlar, sadece temel zaman yönetimi sağlamak için bırakıldı. Timer sınıfı yalnızca sınav süresi ile ilgili temel işlevleri yerine getirecek şekilde basitleştirildi.

**5. DataHandler Class:**

* **Çıkarılanlar**:
  + Önceden belirtilen validate\_json(self, filename) gibi metodlar veri doğrulama için daha az gereklidir çünkü verinin doğru şekilde yönetilmesi, uygulamanın başka kısımlarında (örneğin, Exam ve User sınıflarında) daha doğrudan yapılacaktır.

**6. Result Class:**

* **Çıkarılanlar**:
  + Eski yapıda calculate\_section\_scores(self) ve calculate\_overall\_score(self) gibi metodlar vardı, ancak bunlar Exam sınıfına taşındı. Sonuçların daha kapsamlı bir şekilde hesaplanması için Result sınıfı, sadece verileri göstermeye ve sonuçları sunmaya odaklanmıştır.

**7. QuestionBank Class:**

* **Çıkarılanlar**:
  + get\_random\_questions(self, section, num\_questions) gibi metodlar, mevcut soruları rastgele seçme işlemi temel işlevsellikten dışarıya taşındı. Question sınıfı, soruların özelliklerini tanımlar ve Exam sınıfı ise bu soruları alıp kullanır.

**---------------------------------------------------------------------------**

**1. User Class:**

* **Yeni Eklendikler**:
  + exam\_status: Kullanıcının sınav geçme durumu.
  + last\_attempt\_date: Kullanıcının son sınav tarihi.
  + total\_score: Kullanıcının toplam sınav puanı.
  + Yeni metodlar eklendi: check\_attempts(), get\_exam\_status(), update\_exam\_status(), record\_exam\_score().

**2. Question Class:**

* **Yeni Eklendikler**:
  + question\_type: Sorunun türü (True/False, Single/Multiple Choice).
  + section: Sorunun ait olduğu bölüm.
  + points: Sorunun puan değeri.
  + Yeni metodlar eklendi: validate\_answer\_format(), assign\_points().

**3. Exam Class:**

* **Yeni Eklendikler**:
  + remaining\_attempts: Kullanıcının kalan sınav hakkı.
  + exam\_status: Sınavın geçme durumu.
  + section\_scores: Her bölümün skorları.
  + total\_score: Genel puan.
  + Yeni metodlar eklendi: check\_remaining\_attempts(), calculate\_section\_scores(), calculate\_overall\_score().

**4. Timer Class:**

* **Değişiklik Yok**: Zamanlayıcı ve süre yönetimi mevcut haliyle yeterli.

**5. DataHandler Class:**

* **Değişiklik Yok**: Veritabanı işlemleri ve güvenlik için gerekli tüm metodlar burada bulunuyor.

**6. Result Class:**

* **Yeni Eklendikler**:
  + section\_scores: Bölüm bazında başarı skorları.
  + overall\_score: Genel başarı puanı.
  + Yeni metodlar eklendi: calculate\_section\_scores(), calculate\_overall\_score(), display\_results().

**7. QuestionBank Class:**

* **Değişiklik Yok**: Soru yönetimi için mevcut yapıyı korudum.

**Genel Değişiklikler:**

* **Yeni metodlar ve özellikler eklendi**: Her sınıfın daha kapsamlı bir işlevselliğe sahip olması sağlandı. Örneğin, exam\_status ve total\_score gibi bilgiler, kullanıcının sınavla ilgili daha fazla bilgi tutmasını sağlar.
* **Eklenti özellikler**: Soruların türleri ve puanlandırma gibi özellikler, sınavın daha ayrıntılı kontrolünü sağlar.
* **Metodlar Arasındaki İlişkiler**: check\_exam\_status(), record\_exam\_score(), gibi metodlar sınav sürecinde daha fazla etkileşim sağlayacak şekilde geliştirildi.

Bu değişiklikler, uygulamanın gereksinimlerine ve akışına uygun olarak sınıfların işlevselliğini artırmıştır.

-----------------------------------------------------------

### Dosyalarin son hali :

### 1. ****User Class****

* **Imports**: json (veri kaydetme ve okuma için)
* **Instance'lar**: Exam, Result, DataHandler

#### Attributes:

* name: Kullanıcının adı.
* surname: Kullanıcının soyadı.
* student\_no: Kullanıcının öğrenci numarası.
* attempts: Kullanıcının sınav giriş sayısı.
* exam\_status: Kullanıcının sınav geçme durumu (Geçti/Kaldı).
* last\_attempt\_date: Kullanıcının son sınav giriş tarihi.
* total\_score: Kullanıcının toplam sınav puanı.

#### Methods:

* login(self, name, surname): Kullanıcı bilgilerini alır ve oturum açar.
* update\_attempts(self): Kullanıcının sınav giriş sayısını günceller.
* get\_attempts(self): Kullanıcının sınav giriş sayısını döndürür.
* save\_user\_data(self): Kullanıcı verilerini JSON dosyasına kaydeder.
* check\_attempts(self): Kullanıcının daha fazla sınav hakkı olup olmadığını kontrol eder.
* get\_exam\_status(self): Kullanıcının sınav geçme durumunu döndürür.
* update\_exam\_status(self, status): Kullanıcının sınav geçme durumunu günceller.
* record\_exam\_score(self, score): Kullanıcının sınav puanını kaydeder.

### 2. ****Question Class****

* **Imports**: random (rastgele sorular seçmek için)
* **Instance'lar**: DataHandler

#### Attributes:

* question\_text: Sorunun metni.
* options: Sorunun seçenekleri (list).
* correct\_answer: Doğru cevabın belirtilmesi.
* question\_type: Sorunun türü (True/False, Single-Choice, Multiple-Choice).
* section: Sorunun hangi bölümde yer aldığı bilgisi.
* points: Sorunun puan değeri.

#### Methods:

* check\_answer(self, user\_answer): Kullanıcının cevabını kontrol eder.
* get\_random\_questions(self): Rastgele sorular getirir.
* calculate\_score(self, user\_answers): Kullanıcının cevaplarına göre puan hesaplar.
* validate\_answer\_format(self, user\_answer): Kullanıcının cevabını, soru tipine göre doğrular.
* assign\_points(self, points): Her soruya bir puan atar.

### 3. ****Exam Class****

* **Imports**: User, Timer, Result, Question, DataHandler
* **Instance'lar**: User, Timer, Result, QuestionBank

#### Attributes:

* user: User sınıfından bir örnek (Kullanıcı bilgileri).
* timer: Timer sınıfından bir örnek (Zamanlayıcı).
* result: Result sınıfından bir örnek (Sonuç hesaplamaları).
* remaining\_attempts: Kullanıcının kalan sınav hakkı.
* exam\_status: Sınavın geçme durumu (Geçti/Kaldı).
* section\_scores: Bölüm bazında skorlar.
* total\_score: Genel puan.

#### Methods:

* start\_exam(self): Sınavı başlatır.
* display\_question(self, question): Soruyu ekranda gösterir.
* get\_user\_input(self, question): Kullanıcının cevabını alır.
* validate\_input(self, user\_input, question\_type): Kullanıcı cevabını doğrular.
* time\_out(self): Zaman dolarsa sınavı sonlandırır.
* end\_exam(self): Sınavı bitirir.
* calculate\_results(self): Sonuçları hesaplar.
* show\_results(self): Sonuçları ekranda gösterir.
* check\_remaining\_attempts(self): Kullanıcının sınav hakkı sayısını kontrol eder.
* calculate\_section\_scores(self): Bölüm bazında başarı puanlarını hesaplar.
* calculate\_overall\_score(self): Genel başarı puanını hesaplar.

### 4. ****Timer Class****

* **Imports**: time (zaman kontrolü için)

#### Attributes:

* time\_limit: Sınavın süre sınırı (dakika olarak).
* time\_left: Kalan süre (dakika olarak).

#### Methods:

* start\_timer(self): Zamanlayıcıyı başlatır.
* check\_time(self): Kalan süreyi kontrol eder.
* stop\_timer(self): Zamanlayıcıyı durdurur.

### 5. ****DataHandler Class****

* **Imports**: json, cryptography (şifreleme ve veri okuma/yazma işlemleri için)

#### Methods:

* read\_json(self, file\_name): JSON dosyasını okur.
* write\_json(self, file\_name, data): JSON dosyasına veri yazar.
* encrypt\_data(self, data): Veriyi şifreler.
* decrypt\_data(self, data): Veriyi çözer.
* validate\_json(self, file\_name): JSON dosyasının geçerliliğini kontrol eder.

### 6. ****Result Class****

* **Imports**: Exam, DataHandler
* **Instance'lar**: Exam

#### Attributes:

* section\_scores: Bölüm bazında başarı skorları.
* overall\_score: Genel başarı puanı.

#### Methods:

* calculate\_section\_scores(self, section\_results): Bölüm bazında başarı puanlarını hesaplar.
* calculate\_overall\_score(self): Genel başarı puanını hesaplar.
* display\_results(self, results): Sonuçları gösterir.

### 7. ****QuestionBank Class (Opsiyonel)****

* **Imports**: Question, random

#### Attributes:

* questionsBank: Soruların bulunduğu veri yapısı.

#### Methods:

* add\_question(self, section, question): Bir soruyu belirli bir bölüme ekler.
* get\_question(self, section, index): Belirli bir soruyu getirir.
* get\_random\_questions(self, section, num\_questions): Belirli bir bölümden rastgele soruları alır.