Dokumentation

1. Projektstart und Planung

Zuerst habe ich mir überlegt, wie man das Projekt umsetzen kann. Es waren zwei Entitäten gegeben, Project und Person. Da ich nur Erfahrung mit SQL habe, habe ich mich dafür entschieden und nicht für Hibernate beziehungsweise ORM. Die GUI habe ich mit JavaFX gestaltet, da ich das ebenfalls in der Umschulung gelernt habe. Mit .fxml-Dateien bin ich nicht erfahren. Die Grundstruktur habe ich nach dem MVC-Pattern aufgebaut, damit der Code sauber, modular und leicht wartbar bleibt, wie ich es ebenfalls in der Umschulung gelernt und angewendet habe.

2. Model-Klassen

Jede Klasse repräsentiert genau eine Entität aus der Aufgabenstellung und enthält Konstruktoren mit Getter- und Setter-Methoden für einen einfachen Zugriff. Ich habe mich für separate DAO-Klassen (Data Access Object) entschieden, damit eine Trennung zwischen den Daten und dem Datenbankzugriff stattfindet. Dadurch kann man beispielsweise die Datenquelle ändern, ohne die GUI anpassen zu müssen.

3. Datenbankverbindung

In der Klasse Database.java erstelle ich die Tabellen für Project und Person, falls diese noch nicht vorhanden sind. Somit vermeide ich Code-Duplikate bei der Datenbankverbindung.

4. View-Klassen

Das Layout habe ich teilweise mit Hilfe von KI generieren lassen, um die Oberfläche visuell verständlicher zu gestalten. Jede View-Klasse hat ihre eigene JavaFX-Oberfläche mit einer klaren Trennung der GUI-Elemente. Es gibt eine MainView für die Projektverwaltung und eine PersonView für die Personenverwaltung.

5. Controller-Klassen

Die Controller-Klassen dienen als Vermittler zwischen View und Model. Sie nehmen Benutzereingaben entgegen, validieren diese und rufen die entsprechenden DAO-Methoden auf. Dadurch ist die GUI unabhängig vom Datenzugriff, was dem Single Responsibility Prinzip entspricht. Jede Klasse hat genau eine Aufgabe.

6. Validierung und Benutzerfreundlichkeit

Fehlervermeidung ist wichtig für produktive Software. Deshalb habe ich eine Pflichtfeldprüfung für die projectid implementiert, wie in der Aufgabe gefordert. Eine Sicherheitsabfrage vor dem Löschen, die den Benutzer vor unbeabsichtigtem Löschen schützt, habe ich noch nicht umgesetzt.

7. Warum kein ORM und keine Frameworks?

Da ich nur die Grundlagen von Java, JavaFX, MVC und SQL besitze, konnte ich das

Projekt nur so umsetzen, wie ich es in der Umschulung gelernt habe. Ich wollte keine KI zur Umsetzung des Codes verwenden, abgesehen vom Layout und einzelnen Kommentaren, um das Projekt verständlicher zu machen.

8. To-Do's

Bei den Aufgaben 6 und 7 hatte ich Schwierigkeiten, da ich nur Ansätze hatte, aber nicht genau wusste, wie ich sie umsetzen kann.
Meine Gedanken zu Aufgabe 6 sind:

- Einen Button erstellen, der das Projekt löscht
- Einen Event-Handler für den Löschvorgang implementieren
- Den Button in der Oberfläche hinzufügen
- Im ProjectController eine Methode erstellen, die ein Projekt anhand der projectid löscht
- Im ProjectDAO eine Methode hinzufügen, die das Projekt aus der Datenbank entfernt

Zu Aufgabe 7 habe ich noch keine Lösung finden können.