

## *Projektdokumentation*

### Inhalt

|  |   |
|--|---|
| 1. Projektübersicht .....                      | 2 |
| 2. Risikomanagement .....                      | 2 |
| 3. Zeitplanung .....                           | 4 |
| 3.1 Phasen .....                               | 4 |
| 3.1.1 Analysephase (PSP 1.0) .....             | 5 |
| 3.1.2 Entwurfsphase (PSP 2.0) .....            | 5 |
| 3.1.3 Implementierungsphase (PSP 3.0) .....    | 5 |
| 3.1.3.1 Projektaufbau .....                    | 5 |
| 3.1.3.2 Datenbanklogik .....                   | 5 |
| 3.1.3.3 GUI-Entwicklung .....                  | 6 |
| 3.1.3.4 CRUD-Funktionen vervollständigen ..... | 6 |
| 3.1.3.5 GUI-Feinschliff .....                  | 6 |
| 3.1.4 Testphase (PSP 4.0) .....                | 6 |
| 3.1.5 Abschlussphase (PSP 5.0) .....           | 6 |
| 3.2 Meilensteine .....                         | 7 |
| 3.3 Gantt-Diagramm .....                       | 7 |
| 4. Benutzeranleitung .....                     | 8 |
| Abbildungsverzeichnis .....                    | 9 |
| Tabellenverzeichnis .....                      | 9 |

## 1. Projektübersicht

Das Ziel des Projekts ist die Erstellung der Anwendung „Medianizer“. Diese soll es ermöglichen Medien (CDs und Filme) zu verwalten. Durch eine übersichtliche Benutzeroberfläche wird eine intuitive Nutzung gewährleistet. Nach dem Start der Anwendung wird ein Eingabefenster geöffnet, in dem der Nutzer die Medieninformationen eintragen kann. Gibt es entsprechende Einträge, werden diese in einer Tabelle ausgegeben, deren Einträge bearbeitet oder gelöscht werden können. Gibt es keine Einträge wird ein Fenster aufgerufen, welches das Erstellen eines Eintrags ermöglicht. Auf diese Weise werden alle grundlegenden Datenbankfunktionen (Erstellen, Lesen, Aktualisieren und Löschen) in der Anwendung nutzbar.

Die Anwendung wird in Java programmiert, verwendet JavaFX für die grafische Benutzeroberfläche und speichert die Daten in einer SQLite-Datenbank, die lokal mitgeliefert wird und keine zusätzliche Installation erfordert. Die Projektstruktur, sowie Abhängigkeiten werden mit Apache Maven verwaltet, um eine einfache Reproduzierbarkeit und Wartbarkeit zu gewährleisten. Die Wartbarkeit wird außerdem durch die Nutzung des GitHub Repositories (s. Link im Glossar) unterstützt.

Für die Nutzung des Programms ist kein Internetzugang erforderlich. Allgemein soll die Anwendung möglichst nutzerfreundlich sein und den Anwender unterstützen, indem Meldungen sowohl bei Erfolgen, als auch bei Fehlern angezeigt werden und Eingaben validiert werden.

## 2. Risikomanagement

Während der Projektplanung wurde eine Risikoanalyse durchgeführt, um potenzielle Probleme frühzeitig zu erkennen und passende Maßnahmen festzulegen. Dabei wurden technische (T), organisatorische (O) und personelle (P) Risiken berücksichtigt. Ziel war es, die Eintrittswahrscheinlichkeit und die Auswirkungen auf den Erfolg des Projekts realistisch einzuschätzen. Durch regelmäßige Kontrolle und Pufferzeiten in der Zeitplanung soll das Gesamtrisiko auf ein Minimum reduziert werden.

Besonders kritisch wurde das Risiko einer Zeitüberschreitung eingeschätzt, da in verschiedenen Teilbereichen (z.B. Verwendung von GitHub) keine Vorkenntnisse gegeben sind und der geplante Abgabetermin eingehalten werden soll. Dieses Risiko wurde durch tägliche GitHub-Commits, sowie eine konsequente Priorisierung der Hauptfunktionen ebenso gesenkt, wie durch die sorgfältige Zeitplanung. Die Zeitplanung berücksichtigt lediglich Werktage. Technische Risiken, wie beispielsweise fehlerhafte Datenbankverbindungen, werden durch frühe Tests und eine stabile Projektstruktur reduziert.

Durch die im Folgenden aufgeführten Maßnahmen kann das Gesamtrisiko als niedrig bis mittel eingestuft werden.

| Nr. | Risiko                                       | Kategorie | Wahrscheinlichkeit | Auswirkung | Bewertung   | Gegenmaßnahmen  |
|-----|--|-----------|--------------------|------------|-------------|---|
| 1   | GUI in JavaFX funktioniert nicht wie geplant | T         | Mittel             | Hoch       | Mittel-Hoch | - Frühes Testen jedes Fensters<br>- Controller trennen                  |
| 2   | Zeitüberschreitung durch Detailarbeit        | O         | Niedrig            | Mittel     | Niedrig     | - Zeitplan mit Puffer (Tag 9-10)<br>- Feature-Priorisierung beibehalten |
| 3   | SQLite-Verbindung fehlerhaft                 | T         | Niedrig            | Mittel     | Niedrig     | - Früh testen (Tag 1-2)   |
| 4   | Fehlerhafte Datenmanipulation                | T         | Mittel             | Mittel     | Mittel      | - Unit-Tests für DB-Funktionen  |
| 5   | Unsaubere GitHub-Nutzung                     | O         | Niedrig            | Niedrig    | Niedrig     | - Tägliche Commits<br>- klare Branch-Namen<br>- .gitignore korrekt      |
| 6   | Unentdeckte Bugs                             | T         | Mittel             | Hoch       | Mittel-Hoch | - Tests ab Tag 6-7 einplanen<br>- Use-Cases prüfen                      |
| 7   | Überlastung                                  | P         | Niedrig            | Mittel     | Niedrig     | - Arbeitsblöcke begrenzen<br>- Fortschritte festhalten                  |
| 8   | Abhängigkeiten oder Build-Fehler             | T         | Niedrig            | Mittel     | Niedrig     | - pom.xml früh testen<br>- keine unnötigen Dependencies                 |
| 9   | Zu wenig Zeit für Dokumentation              | O         | Niedrig            | Mittel     | Niedrig     | - Tag 10 fest für Dokumentation reservieren                             |

Tabelle 1

Nachfolgend wird die Risikoanalyse um Frühwarnindikatoren ergänzt.

| Nr. | Frühwarnindikatoren                              |
|-----|--|
| 1   | Fehlermeldungen häufen sich, GUI reagiert nicht  |
| 2   | Rückstand im Zeitplan, geringere Commit-Frequenz |
| 3   | Verbindungsfehler in Logs, CRUD schlägt fehl     |
| 4   | Tests schlagen häufiger fehl, falsche DB-Werte   |
| 5   | Merge-Konflikte, fehlende Commits                |
| 6   | Unerwartete Abstürze, lange Bugliste             |
| 7   | Leistungsabfall, Fehlerhäufung                   |
| 8   | Maven-Build schlägt wiederholt fehl              |
| 9   | Textabschnitte unvollständig, fehlende Inhalte   |

Tabelle 2

Zusammenfassend kann festgehalten werden, dass die geplanten technischen und organisatorischen Maßnahmen ausreichen, um den Erfolg des Projekts auch bei kleineren Problemen sicherzustellen. Kontinuierliches Monitoring, so wie das Durchführen von Tests während der Umsetzung sorgen für eine hohe Projektsicherheit.

### 3. Zeitplanung

Die Zeitplanung unterteilt das Projekt in insgesamt zehn Arbeitstage mit klar definierten Arbeitspaketen und Meilensteinen. Jedes Arbeitspaket wurde Umfang und Dauer betreffend realistisch eingeschätzt, um die Arbeitsbelastung gleichmäßig zu gestalten. Anhand der Meilensteine kann regelmäßig überprüft werden, ob Zeitplanung und Umsetzung übereinstimmen. So kann der Fortschritt dokumentiert werden und sollte es zu Abweichungen kommen, kann auf diese adäquat reagiert werden.

#### 3.1 Phasen

Die vollständige Zeitplanung mit Arbeitspaketen und ihrer Beschreibung im Detail wird im Folgenden in fünf Phasen unterteilt.

### 3.1.1 Analysephase (PSP 1.0)

Ziel: Anforderungen und grundlegende Systemstruktur festlegen

| PSP | Aufgabe                | Start      | Ende       | Dauer | Abhängigkeit |
|-----|------------------------|------------|------------|-------|--------------|
| 1.1 | Anforderungen sammeln  | 28.10.2025 | 28.10.2025 | 1 Tag | –            |
| 1.2 | Datenmodell definieren | 28.10.2025 | 28.10.2025 | 1 Tag | 1.1          |
| 1.3 | GUI-Konzept erstellen  | 29.10.2025 | 29.10.2025 | 1 Tag | 1.2          |

Tabelle 3

### 3.1.2 Entwurfsphase (PSP 2.0)

Ziel: Architektur, UML und Systementwurf abschließen

| PSP | Aufgabe                           | Start      | Ende       | Dauer | Abhängigkeit |
|-----|-----------------------------------|------------|------------|-------|--------------|
| 2.1 | Architekturmodell (MVC) erstellen | 29.10.2025 | 29.10.2025 | 1 Tag | 1.2          |
| 2.2 | Klassendiagramme erstellen        | 30.10.2025 | 30.10.2025 | 1 Tag | 2.1          |

Tabelle 4

### 3.1.3 Implementierungsphase (PSP 3.0)

Ziel: Datenbank, Logik und GUI vollständig implementieren

#### 3.1.3.1 Projektaufbau

| PSP   | Aufgabe                                 | Start      | Ende       | Dauer | Abhängigkeit |
|-------|---|------------|------------|-------|--------------|
| 3.1.1 | GitHub-Repository anlegen               | 28.10.2025 | 28.10.2025 | 1 Tag | –            |
| 3.1.2 | Maven konfigurieren                     | 28.10.2025 | 28.10.2025 | 1 Tag | 3.1.1        |
| 3.1.3 | SQLite testen & Grundstruktur erstellen | 28.10.2025 | 28.10.2025 | 1 Tag | 3.1.2        |

Tabelle 5

#### 3.1.3.2 Datenbanklogik

| PSP   | Aufgabe                  | Start      | Ende       | Dauer | Abhängigkeit |
|-------|--------------------------|------------|------------|-------|--------------|
| 3.2.1 | CRUD-Methoden entwickeln | 29.10.2025 | 29.10.2025 | 1 Tag | 3.1.3        |

Tabelle 6

### 3.1.3.3 GUI-Entwicklung

| PSP   | Aufgabe                               | Start      | Ende       | Dauer | Abhängigkeit |
|-------|---------------------------------------|------------|------------|-------|--------------|
| 3.3.1 | GUI Schnittstelle 1 implementieren    | 30.10.2025 | 30.10.2025 | 1 Tag | 3.2.1        |
| 3.3.2 | GUI Schnittstelle 2 implementieren    | 31.10.2025 | 31.10.2025 | 1 Tag | 3.3.1        |
| 3.3.3 | GUI Schnittstelle 3 (Tabellenansicht) | 03.11.2025 | 03.11.2025 | 1 Tag | 3.3.2        |

Tabelle 7

### 3.1.3.4 CRUD-Funktionen vervollständigen

| PSP   | Aufgabe                           | Start      | Ende       | Dauer | Abhängigkeit |
|-------|-----------------------------------|------------|------------|-------|--------------|
| 3.4.1 | Bearbeiten/Löschen implementieren | 04.11.2025 | 04.11.2025 | 1 Tag | 3.3.3        |

Tabelle 8

### 3.1.3.5 GUI-Feinschliff

| PSP   | Aufgabe                                 | Start      | Ende       | Dauer | Abhängigkeit |
|-------|---|------------|------------|-------|--------------|
| 3.5.1 | UI-Validierung, Fehlermeldungen, Design | 05.11.2025 | 05.11.2025 | 1 Tag | 3.4.1        |

Tabelle 9

### 3.1.4 Testphase (PSP 4.0)

Ziel: Qualität sichern

| PSP | Aufgabe                             | Start      | Ende       | Dauer | Abhängigkeit |
|-----|-------------------------------------|------------|------------|-------|--------------|
| 4.1 | Gesamttest & Bugfixing              | 06.11.2025 | 06.11.2025 | 1 Tag | 3.5.1        |
| 4.2 | Code-Refactoring                    | 06.11.2025 | 06.11.2025 | 1 Tag | 4.1          |
| 4.3 | Optimierung & kleine Verbesserungen | 07.11.2025 | 07.11.2025 | 1 Tag | 4.2          |

Tabelle 10

### 3.1.5 Abschlussphase (PSP 5.0)

Ziel: Projektfertigstellung

| PSP | Aufgabe                        | Start      | Ende       | Dauer | Abhängigkeit |
|-----|--------------------------------|------------|------------|-------|--------------|
| 5.1 | Dokumentation vervollständigen | 10.11.2025 | 10.11.2025 | 1 Tag | 4.3          |
| 5.2 | Abgabe / Release vorbereiten   | 10.11.2025 | 10.11.2025 | 1 Tag | 5.1          |

Tabelle 11

### 3.2 Meilensteine

| Meilenstein | Beschreibung                      | Datum      | Zugehöriger PSP |
|-------------|-----------------------------------|------------|-----------------|
| <b>M1</b>   | Analyse abgeschlossen             | 29.10.2025 | 1.0             |
| <b>M2</b>   | Entwurf abgeschlossen             | 30.10.2025 | 2.0             |
| <b>M3</b>   | Basisfunktionalität implementiert | 04.11.2025 | 3.0             |
| <b>M4</b>   | Testphase abgeschlossen           | 07.11.2025 | 4.0             |
| <b>M5</b>   | Projektabschluss                  | 10.11.2025 | 5.0             |

Tabelle 12

### 3.3 Gantt-Diagramm

|                                  | 28.10. | 29.10. | 30.10. | 31.10. | 03.11. | 04.11. | 05.11. | 06.11. | 07.11. | 10.11. |
|----------------------------------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| <b>1.0 Analysephase</b>          |        |        |        |        |        |        |        |        |        |        |
| 1.1                              |        |        |        |        |        |        |        |        |        |        |
| 1.2                              |        |        |        |        |        |        |        |        |        |        |
| 1.3                              |        |        |        |        |        |        |        |        |        |        |
| <b>2.0 Entwurfsphase</b>         |        |        |        |        |        |        |        |        |        |        |
| 2.1                              |        |        |        |        |        |        |        |        |        |        |
| 2.2                              |        |        |        |        |        |        |        |        |        |        |
| <b>3.0 Implementierungsphase</b> |        |        |        |        |        |        |        |        |        |        |
| 3.1.1                            |        |        |        |        |        |        |        |        |        |        |
| 3.1.2                            |        |        |        |        |        |        |        |        |        |        |
| 3.1.3                            |        |        |        |        |        |        |        |        |        |        |
| 3.2.1                            |        |        |        |        |        |        |        |        |        |        |
| 3.3.1                            |        |        |        |        |        |        |        |        |        |        |
| 3.3.2                            |        |        |        |        |        |        |        |        |        |        |
| 3.3.3                            |        |        |        |        |        |        |        |        |        |        |
| 3.4.1                            |        |        |        |        |        |        |        |        |        |        |
| 3.5.1                            |        |        |        |        |        |        |        |        |        |        |
| <b>4.0 Testphase</b>             |        |        |        |        |        |        |        |        |        |        |
| 4.1                              |        |        |        |        |        |        |        |        |        |        |
| 4.2                              |        |        |        |        |        |        |        |        |        |        |
| 4.3                              |        |        |        |        |        |        |        |        |        |        |
| <b>5.0 Abschlussphase</b>        |        |        |        |        |        |        |        |        |        |        |
| 5.1                              |        |        |        |        |        |        |        |        |        |        |
| 5.2                              |        |        |        |        |        |        |        |        |        |        |

Abbildung 1

Durch diese strukturierte Planung und Aufteilung in überschaubare Arbeitspakete ist eine termingerechte Fertigstellung des Projekts bis zum 10.11.2025 realistisch. Da täglich das Erreichen der Meilensteine geprüft wird, ist das Risiko von Verzögerungen minimiert.

## 4. Benutzeranleitung

Bitte beachten Sie folgende Systemvoraussetzungen:

- Betriebssystem: Windows 10 oder höher
- Speicherplatz: ca. 150 MB
- Keine zusätzliche Java- oder JavaFX-Installation notwendig

Die Anwendung Medianizer ist als ausführbare Desktop-App verpackt und kann ohne separate Java-Installation gestartet werden. Nach dem Entpacken des Zip-Ordners kann das Programm durch einen Doppelklick auf die Datei Medianizer.bat im Unterordner bin des Ordners MedianizerApp gestartet werden. Alle benötigten Laufzeitkomponenten (Java-Laufzeitumgebung und Bibliotheken) sind bereits enthalten. Beim ersten Start wird automatisch eine SQLite-Datenbank im Ordner data angelegt, in der die Medien-Einträge gespeichert werden. Zum Beenden der Anwendung schließen Sie das Fenster.

Nach dem Start der Anwendung öffnet sich die erste Benutzerschnittstelle.

### 1. Erste Benutzerschnittstelle

Wählen Sie zunächst im Drop-Down-Menü auf der linken Seite ihr gewünschtes Medium. Dadurch schalten sich die Textfelder frei (Titel bei Film, Titel und Interpret bei CD). Sie können nun Ihre Suchkriterien eingeben und mit OK bestätigen. Es folgt ein Fenster je nachdem ob Dateneinträge gefunden wurden oder nicht. Wurden keine Datenbankeinträge gefunden, öffnet sich die zweite Benutzerschnittstelle, andernfalls die dritte Benutzerschnittstelle

### 1. Zweite Benutzerschnittstelle

Diese Schnittstelle übernimmt Ihre Eingaben, muss aber noch vervollständigt werden, wenn der Eintrag gespeichert werden soll. Es muss immer ein Titel angegeben sein, bei CDs zusätzlich der Interpret und in jedem Fall die Anzahl der Ihnen vorliegenden Exemplare. Durch einen Klick auf Speichern fügen Sie den Eintrag Ihrer Datenbank hinzu bzw. bearbeiten einen bestehenden. Soll der Eintrag nicht gespeichert werden, kann die Schnittstelle einfach geschlossen werden.

### 2. Dritte Benutzerschnittstelle

Diese Schnittstelle zeigt in einer Tabelle die Datenbankeinträge zu Ihrer Suche. Sie können nun per Klick eine Zeile der Tabelle auswählen und diese dann über die Buttons oberhalb der Tabelle bearbeiten oder löschen. Der Button Bearbeiten führt Sie zur zweiten Benutzerschnittstelle. Der Button löschen, löscht den gewählten Eintrag aus der Datenbank und die Tabelle aktualisiert sich.



## Abbildungsverzeichnis Projektdokumentation

|             |   |
|-------------|---|
| ABBILDUNG 1 | 7 |
|-------------|---|

## Tabellenverzeichnis Projektdokumentation

|            |   |
|------------|---|
| TABELLE 1  | 3 |
| TABELLE 2  | 4 |
| TABELLE 3  | 5 |
| TABELLE 4  | 5 |
| TABELLE 5  | 5 |
| TABELLE 6  | 5 |
| TABELLE 7  | 6 |
| TABELLE 8  | 6 |
| TABELLE 9  | 6 |
| TABELLE 10 | 6 |
| TABELLE 11 | 6 |
| TABELLE 12 | 7 |