2. Projektplanung

2.1 Projektphasen

Die Zeitplanung sieht einen Umfang von 80 Stunden vor. Die Durchführung des Projekts begann am 31.03.2025 und endete am 11.04.2025. Die Arbeiten fanden während der regulären Arbeitszeit statt.

Die Einteilung der Projektphasen mit der geplanten Dauer setzten sich wie folgt zusammen:

|  |  |  |
| --- | --- | --- |
| Projektphase | Stunden pro Phase | Stunden pro Unterpunkt |
| Analyse | 8 | - |
| - Erstellen eines Pflichtenhefts | - | 5 |
| - Wirtschaftlichkeitsanalyse | - | 3 |
| Entwurf | 12 | - |
| - Planung der GUI | - | 5 |
| - Planung der Datenmodelle | - | 7 |
| Entwicklung | 40 | - |
| - Entwicklung der GUI | - | 18 |
| - Implementierung der Funktionalität | - | 22 |
| Tests und Abnahme | 4 | - |
| Dokumentation | 16 | - |
| GESAMT | 80 |  |

2.2 Enwicklungsprozess  
  
Um die Aufgabenverwaltung einfacher zu gestalten und mich an den Prinzipien des agilen Softwareentwicklungsprozesses zu orientieren habe ich ein Kanban-Board auf Github genutzt. Die Aufteilung in viele kleinere Aufgaben, zusammen mit der Einteilung in die Gruppen „To-Do“, „In-Progress“ und „Done“ waren ein guter Weg um das Projekt übersichtlich zu halten und auf eventuelle Änderungen reagieren zu können.

2.3 Ressourcenplanung

Die Entwicklung der Anwendung fand in den Räumlichkeiten der cdemy GmbH statt. Gearbeitet wurde an einem PC mit Windows 11 Betriebssystem. Zum Schreiben des Programmcodes, zum Debugging und Compilen wurde die IDE Visual-Studio-Code verwendet. Das Projekt wird auf Github gehostet und mit Git lokal verwaltet.

2.3.1 Programmiersprachen

Zur Umsetzung des Projekts wurde die Programmiersprache Dart gewählt, in Kombination mit dem plattformübergreifenden UI-Framework Flutter.  
  
Dart wurde ursprünglich von Google entwickelt und eignet sich besonders für die Erstellung moderner Anwendungen sowohl im Web- als auch im Desktop- und Mobile-Umfeld.  
  
Dart ist plattformunabhängig und kann auf Windows, macOS, Linux, Webbrowsern und mobilen Endgeräten ausgeführt werden, was langfristige Flexibilität ermöglicht. Dank Hot Reload können Änderungen am Code sofort sichtbar gemacht werden, ohne dass die Anwendung neu gestartet werden muss. Dart bietet eine objektorientierte Syntax mit Features wie Null-Sicherheit, asynchroner Programmierung, Extension-Methods und Mixins, die eine strukturierte und wartbare Codebasis unterstützen. Flutter, das Framework für die Benutzeroberfläche der Anwendung, basiert vollständig auf Dart. Dadurch wird eine gute Verzahnung zwischen Anwendungslogik und UI ermöglicht.  
  
Die Wahl von Dart für dieses Projekt wurde aus mehreren Gründen getroffen:  
- Die enge Verzahnung mit Flutter erlaubt die plattformübergreifende Entwicklung effizienter Desktop-Anwendungen, was eine spätere Portierung der Anwendung auf andere System wie macOS grundsätzlich ermöglicht.

- Die morderne Sprachstruktur von Dart unterstützt eine saubere, wartbare Projektarchitektur.  
- Ich selbst habe bereits Erfahrungen im Umgang mit Dart sammeln können und musste mich nicht in die Möglichkeiten und Eigenheiten der Sprache einarbeiten.