Projektdokumentation

SE-GL Abschlussprojekt

Matthias

LUTZ UND GRUB

Inhaltsverzeichnis

Einleitung

Projekt Informationen

Umschüler: Matthias

GitHub-Link: https://github.com/Lord-Mandos

Ausbildungsstätte/Ausbildungsfirma/Firma: LuG

Projektbezeichnung: Konsolenbasiertes Aufgaben- & Projektmanagement-Tool („Kaban“)

Zulassender Dozent: Jochen

Bewertender Dozent: Michael

Konkrete Projektbeschreibung

„Kaban“ ist eine Konsolenanwendung zur strukturierten Verwaltung von Aufgaben innerhalb kleiner Projekte.

Das Tool soll Anwendern ermöglichen, Aufgaben zu planen, zu organisieren und den Fortschritt zu kontrollieren.

Die Software orientiert sich an einfachen Projektmanagement-Methoden wie Kanban und To-Do-Listen.

Ziel ist es, eine klare, intuitive Konsolenoberfläche zu schaffen, mit der ein Benutzer Aufgaben:

- *erstellen,*
- *bearbeiten,*
- *löschen*
- *und in verschiedene Statusbereiche verschieben kann*

Zur Verbesserung der Sicherheit wird die Benutzerverwaltung mit *PBKDF2-Passwort-Hashing* umgesetzt.

Die Oberfläche wird mit *Spectre.Console* gestaltet, um eine moderne und strukturierte Darstellung im Terminal zu ermöglichen (z. B. Tabellen, Panels, farbige Hervorhebungen).

Alle Daten werden lokal in JSON-Dateien gespeichert und beim Programmstart wieder geladen.

Projektantrag / Planung

Pflichtfunktionen

Benutzerverwaltung

- *Benutzer registrieren*
- *Anmeldung über Benutzername + Passwort*
- *Speicherung von:*
  - *Passwort-Hash (PBKDF2)*
  - *Salt*
  - *Iterationsanzahl*
- *Kein Klartextspeichern*

Aufgabenmanagement

- *Aufgaben erstellen*
- *Aufgaben anzeigen*
- *Aufgaben bearbeiten*
- *Aufgaben löschen*
- *Aufgaben suchen (ID, Titel, Kategorie)*

Kanban-Board

- *Statuswechsel zwischen Backlog, In Arbeit, Erledigt*
- *Darstellung mit Spectre.Console (Tabellen/Layouts)*

Datenspeicherung

- *JSON-Dateien für Benutzer- und Aufgabendaten*
- *Automatisches Speichern nach Änderungen*
- *Laden beim Programmstart*

Spectre.Console UI

- *Tabellen für Kanban-Board*
- *Panels, Markup und Farben*
- *Auswahlmenüs (SelectionPrompt)*

Optionale Features

- *Statistiken (z. B. erledigte Aufgaben je Zeitraum)*
- *Deadline-Verwaltung*
- *CSV-/TXT-Export*
- *Prioritätsmarkierungen (Farben)*
- *Benutzerrollen (Admin/User)*

Geplante Architektur

Klassen (geplant)

- **User**
  - *ID*
  - *Username*
  - *PasswordHash*
  - *Salt*
  - *Tasks*
- **Task**
  - *ID*
  - *Title*
  - *Description*
  - *Category*
  - *Priority*
  - *Status*
  - *CreatedAt*
  - *CompletedAt*
- **AuthService**
  - *PasswordHashWithPBKDF2()*
  - *VerifyPassword()*
- **TaskService**
  - *AddTask()*
  - *EditTask()*
  - *DeleteTask()*
  - *MoveTask()*
  - *SearchTasks()*
- **StorageManager**
  - *Save()*
  - *Load()*
- **MenuSystem**
  - *LoginMenu()*
  - *MainMenu()*
  - *TaskMenu()*
  - *BoardMenu()*
- **UIRenderer (für Spectre.Console)**
  - *RenderBoard()*
  - *RenderTaskList()*
  - *RenderMessage()*

Einsatz von Nicht-Standard-Bibliotheken

Für die Umsetzung des Projekts werden folgende nicht zum üblichen Standardumfang gehörenden Bibliotheken verwendet:

System.Security.Cryptography –(PBKDF2)

Wird verwendet, um Passwörter sicher mit PBKDF2 zu hashen.

Dies dient der sicheren Authentifizierung und verhindert die Speicherung von Klartextpasswörtern.

Spectre.Console

Diese Bibliothek wird zur Gestaltung einer modernen und strukturierten Konsolenoberfläche eingesetzt.

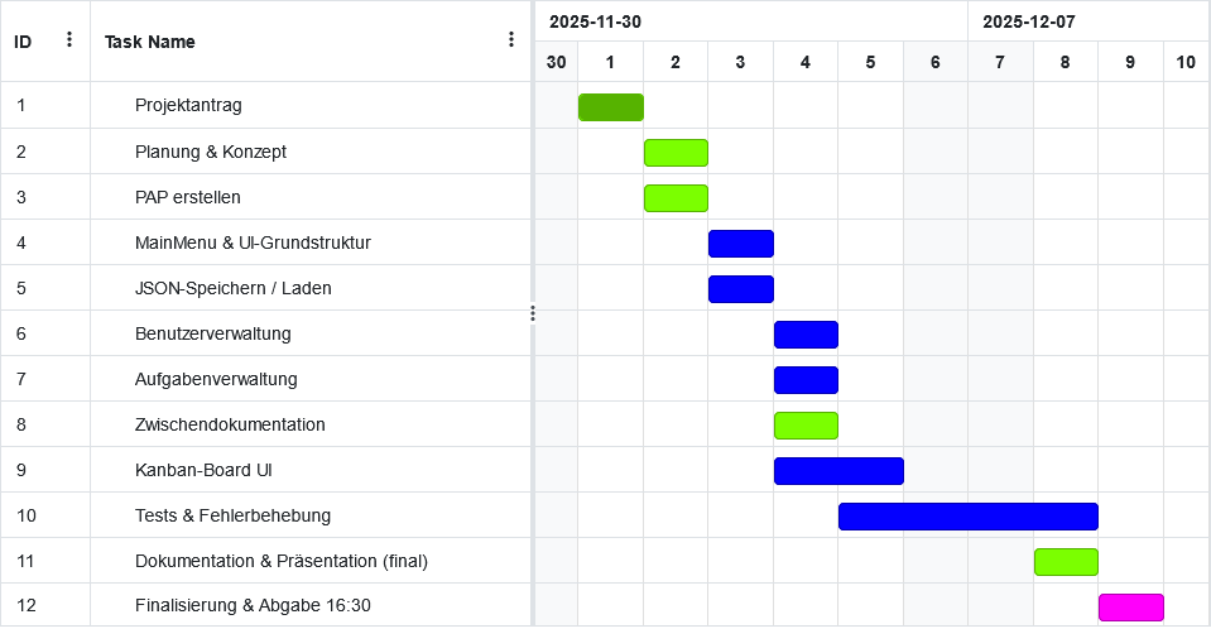
Damit können Tabellen, Panels, Markup-Text, Auswahlmenüs und farbliche Hervorhebungen realisiert werden, um das Kanban-Board und die Navigation übersichtlich darzustellen.

Inhalt der Projektdokumentation

1. *Einleitung*
2. *Projektantrag / Planung*
3. *Zeitplanung / Projektablauf*
4. *Implementierung und Umsetzung*
5. *Probleme, Schwierigkeiten und Überarbeitungen*
6. *Screenshots (Code & Debugging)*
7. *Diagramme (Flowchart, Nassi-Schneiderman)*
8. *Quellenangaben*
9. *Anlagen*

Zeitplanung / Projektablauf

Gantt Diagramm



Erläuterung zum Gantt Diagramm

Nach Genehmigung des Projektantrages wird das Projekt geplant und ein PAP erstellt, um eine stabile Basis zu schaffen.

Die UI-Grundstruktur und das JSON-Speichersystem wurden früh eingeplant, damit alle späteren Module darauf aufbauen können.

Darauf werden Benutzerverwaltung und Aufgabenverwaltung implementiert, womit die Grundfunktionen des Programmes funktionsfähig sind.

Durch die Zwischendokumentation am 04.12. wird die Enddokumentation deutlich entlastet.

Die Kanban-Board UI wird zu Schluss eingeplant da sie auf allen anderen Funktionen aufbaut.

Tests und Fehlerbehebung wurden bewusst über mehrere Tage verteilt und bilden den Übergang zur finalen Dokumentation.

Der letzte Tag ist ausschließlich für finale Anpassungen, Aufräumen und die Abgabe reserviert.

Geplante vs. tatsächlicher Umsetzung

Hier dokumentiere ich die dynamische Entwickelung meines Projektes.

Gründe für Änderungen meines Plans können technische Probleme, Prioritätsänderung oder unrealistische Vorplanung sein.

Implementierung und Umsetzung

Probleme, Schwierigkeiten und Überarbeitungen

Screenshots (Code & Debugging)

Diagramme (Flowchart, Nassi-Schneiderman)

Quellenangaben

Anlagen