Pos Projekt

Teammitglieder

- Laura
- Franziska

Idee

- Tamagotchi App

Wie werden die Mindestanforderungen umgesetzt?

Frontend (POS)

- **3 Fenster:**
    1. Login/Register Fenster
    2. Auswahl Fenster
    3. Hauptfenster (Tamagotchi)
    4. EditorFenster (Hunger, Energie, Stimmung ändern)
- **Grafische Anwendung:** Darstellung des Haustiers als Bild und Statusanzeigen (ProgressBars)
- **Vererbung:** Basis-Klasse `Pet`, abgeleitete Klassen je nach Tierart
- **Interfaces:** z.B. `IFeedable`, `IPlayable`, `ISleepable`
- **Logging:** Log-Datei über Aktionen wie "gefüttert", "gespielt", "geschlafen"
- **Unit Tests:** Kleine Tests für Status-Logik (z.B. Fütterung erhöht Energie)
- **GIT:** Projektverwaltung auf GitHub

Backend (DBI)

- **MariaDB** als Datenbank - **Mind. 3 Tabellen:** - `Users` (ID,Name, Passwort) - `Pets` (ID, Name, Hunger, Energie, Stimmung, UserID) - `Actions` (ID, PetID, ActionType, Timestamp)
- **REST API:** C# Backend, Übergabe der Daten im JSON-Format
- **Swagger Dokumentation:** Dokumentation der REST-Schnittstellen
- **Benutzerrollen:** "Admin" (kann alle Tiere bearbeiten) und "User" (nur eigenes Tier betreuen)
- **Unit Tests + Logging** auch im Backend

Welche Features sind ein Muss?

- Benutzerlogin (User/Admin)
- Darstellung eines virtuellen Haustiers
- Haustieraktionen: Füttern, Spielen, Schlafen
- Haustierstatus (Hunger, Energie, Stimmung) ändern sich über Zeit
- Speichern der Daten in einer MariaDB
- Mindestens 3 Fenster
- Basisarchitektur mit Vererbung und Interfaces
- Nutzung von GitHub
- Auswahl verschiedener Tierarten bei Spielstart *

- Mehrere Haustiere pro Benutzer *
- Animierte Haustierbilder (z.B. beim Spielen bewegt sich das Bild) *

Welche Features sind Erweiterungen (nice-to-have)?

- Hintergrundmusik oder Soundeffekte bei Aktionen
- Cloud-Datenbank (z.B. Cloudflare) statt lokaler MariaDB

Wie möchten wir das Ganze grob umsetzen?

1. **Planung:**
    - Erstellung von GUI-Skizzen für Login-, Haupt- und Statusfenster
    - Klassendiagramme für Haustier, Benutzer und Aktionen
    - Aufsetzen eines GitHub-Repositories
2. **Umsetzung Frontend (WPF, C#):**
    - Aufbau der Fensterstruktur
    - Implementierung der Haustierlogik und Benutzerführung
    - Anbindung an die REST-API für Lese-/Schreibvorgänge
    - Einbauen von Logging und ersten Unit-Tests
3. **Umsetzung Backend (C#, MariaDB):**
    - Erstellung der Datenbanktabellen in MariaDB
    - Aufbau einer REST-API (C#) zur Kommunikation mit der Datenbank
    - Implementieren von Rollenverwaltung und Sicherheitsmechanismen
4. **Testen und Dokumentieren:**
    - Durchgehendes Testen aller Funktionen
    - Erstellung der Dokumentation
5. **Abschluss:**
    - Erstellung von Präsentationsunterlagen
    - Hochladen von Dokumentation und Projekt auf GitHub

Beginn WPF 30.04.2025

LoginWindow und Mainwindow - Franziska **Gemacht** Klick auf Login-Button ➔ Wenn Benutzer = admin und Passwort = 1234 ➔ Wechsel ins Hauptfenster (MainWindow) Wenn falsch ➔ Fehlermeldung ☑ Fenster "Login" soll sich öffnen ☑ Eingabe Benutzername + Passwort möglich ☑ Nach Login auf "admin/1234" öffnet sich das Hauptfenster Klick auf „Füttern“ ➔ Hunger-Leiste füllt sich Klick auf „Spielen“ ➔ Stimmung steigt Klick auf „Schlafen“ ➔ Energie steigt ☑ MainWindow zeigt Bild, Buttons und Fortschrittsbalken ☑ Aktionen verändern die Anzeige ☑ StatusButton öffnet PetEditSelection