ANTRAG FÜR DIE BETRIEBLICHE PROJEKTARBEIT

NADOOIT

1. PROJEKTBEZEICHNUNG

Entwicklung einer Anwendung zur Zeiterfassung in Remote-Arbeitsmodellen unter Beachtung von DSGVO und Softwarequalitätsstandards für den Zeitraum 01.03.2025 -25.04.2025.

1.1. KURZFORM DER AUFGABENERSTELLUNG

Für die Backhaus IT soll die Erfassung von Arbeitszeiten, insbesondere für remote arbeitende Mitarbeiter, durch eine lokale Softwarelösung erleichtert werden. Ziel ist es, die manuelle Zeiterfassung zu ersetzen und eine benutzerfreundliche Anwendung zu entwickeln, die Arbeitszeiten effizient dokumentiert, speichert und auswertet. Hierzu soll eine plattformunabhängige Anwendung in Python entwickelt werden, die direkt auf den Geräten der Mitarbeiter genutzt werden kann.

1.2. IST-ANALYSE

Die Mitarbeiter der Backhaus IT arbeiten größtenteils im Homeoffice und erfassen ihre Arbeitszeiten individuell. Derzeit geschieht dies manuell, zum Beispiel auf Papier, in Tabellen oder über verschiedene externe Tools. Dadurch treten immer wieder Probleme auf: Die erfassten Zeiten sind oft fehlerhaft und nicht einheitlich, was es schwierig macht, sie nachzuverfolgen und auszuwerten.
Außerdem müssen Arbeitszeiten, Pausen und Überstunden aktuell noch per Hand berechnet werden, was zeitaufwändig und anfällig für Fehler ist. Eine zentrale, benutzerfreundliche Softwarelösung fehlt bisher. Dies erhöht den Aufwand für die Mitarbeiter und erschwert die einheitliche Erfassung der Arbeitszeiten

2. ZIELSETZUNG ENTWICKELN / SOLL-KONZEPT

2.1 WAS SOLL AM ENDE DES PROJEKTES ERREICHT SEIN?

Ziel des Projekts ist die Entwicklung einer benutzerfreundlichen Zeiterfassungssoftware für die Backhaus IT. Die Anwendung wird lokal auf den Geräten der Mitarbeiter installiert und plattformunabhängig entwickelt, sodass sie problemlos auf Windows, MacOS und Linux genutzt werden kann.

Die Software soll den Arbeitsalltag vereinfachen, indem Arbeitszeiten, Pausen und Überstunden schnell und fehlerfrei erfasst werden können. Durch eine intuitive

Benutzeroberfläche wird sichergestellt, dass die Bedienung auch ohne lange Einarbeitung möglich ist. Arbeitszeiten und Pausenzeiten werden automatisch berechnet, wodurch Fehler und manuelle Nacharbeit vermieden werden. Zusätzlich werden die Daten in einem einheitlichen Format gespeichert, sodass Berichte und Statistiken, wie etwa Monatsübersichten, leicht erstellt werden können. Die Software folgt dabei einer festen Geschäftslogik, denn bei jeder Erfassung überprüft das System automatisch, ob die eingegebenen Zeiten vollständig und korrekt sind. Falls eine Zeit fehlt oder nicht plausibel erscheint (z. B. eine zu kurze Pause oder eine Überlappung von Arbeitszeiten), wird der Nutzer darauf hingewiesen. Basierend auf den gespeicherten Daten berechnet die Software die Netto-Arbeitszeit unter Berücksichtigung gesetzlicher Vorgaben oder betrieblicher Regeln. Zudem werden alle Änderungen an den erfassten Zeiten in einer Protokollhistorie gespeichert. Das bedeutet, dass jede nachträgliche Korrektur – sei es durch den Nutzer selbst oder durch eine berechtigte Person – genau festgehalten wird, damit jederzeit nachvollziehbar bleibt, wann und warum eine Anpassung vorgenommen wurde.

Ein zentrales Thema ist der Datenschutz. Alle Daten werden ausschließlich lokal auf den Geräten der Mitarbeiter gespeichert, sodass keine Informationen an externe Server weitergeleitet werden. Damit erfüllt die Software die Vorgaben der Datenschutz-Grundverordnung (DSGVO). Gleichzeitig werden alle Änderungen – wie Korrekturen von Arbeitszeiten – genau protokolliert. So bleibt jederzeit nachvollziehbar, wann und von wem Anpassungen vorgenommen wurden.

Technisch basiert die Software auf Python und nutzt Frameworks wie Toga und Briefcase für die plattformübergreifende Entwicklung. Die Daten werden in einer SQLite-Datenbank gespeichert, die leicht in der Anwendung integriert werden kann und keine zusätzlichen Server benötigt. Das macht die Software ressourcenschonend und einfach in der Wartung. Außerdem wird die Anwendung so gestaltet, dass sie in Zukunft leicht erweitert werden kann, etwa durch Funktionen wie Projektzuweisungen oder Schnittstellen zu anderen Tools.

Durch diese Software wird die manuelle Zeiterfassung vollständig abgelöst, wodurch der administrative Aufwand für Mitarbeiter und Management reduziert und die Datenqualität verbessert wird.

2.2 WELCHE ANFORDERUNGEN MÜSSEN ERFÜLLT SEIN?

Die Software beinhaltet eine Geschäftslogik, die alle erfassten Arbeitszeiten automatisch überprüft und berechnet. Da keine festen Arbeitszeiten vorgegeben sind, wird der früheste mögliche Arbeitsbeginn auf 6 Uhr und das späteste Ende auf 23 Uhr begrenzt. Die Berechnung der Arbeitszeit berücksichtigt dabei die gesetzlichen Vorgaben, wie beispielsweise Pausenzeiten (mindestens 30 Minuten bei mehr als 6 Stunden Arbeit) sowie den zeitlichen Abstand zwischen Feierabend und dem nächsten Arbeitsbeginn (mindestens 11 Stunden Ruhezeit).
Eingaben werden geprüft, um sicherzustellen, dass diese Regeln eingehalten werden. So wird etwa validiert, ob Pausen korrekt eingeplant sind und ob die Gesamtdauer der Arbeitszeit plausibel ist. Überstunden werden erkannt, wenn die tatsächliche Arbeitszeit die geplante Tagesarbeitszeit übersteigt.

2.3 WELCHE EINSCHRÄNKUNGEN MÜSSEN BERÜCKSICHTIGT WERDEN?

Plattformunabhängigkeit: Die Software muss auf gängigen Betriebssystemen (Windows, MacOS, Linux) laufen.

Ressourcenoptimierung: Sie darf keine hohen Systemressourcen beanspruchen und muss auf allen Geräten effizient laufen.

Keine externen Integrationen: Die Software funktioniert unabhängig und ohne Schnittstellen zu anderen Abteilungen oder Systemen.

Datenschutz: Die Software muss lokal speichern und den Anforderungen der DSGVO entsprechen.

3. PROJEKTSTRUKTURPLAN ENTWICKELN

3.1 WAS IST ZUR ERFÜLLUNG DER ZIELSETZUNG ERFORDERLICH?

Das Projekt wird in einem agilen Ansatz umgesetzt, um flexibel auf Anforderungen und Änderungen reagieren zu können. Durch regelmäßiges Feedback der zukünftigen Nutzer können notwendige Anpassungen frühzeitig erkannt und umgesetzt werden. Gleichzeitig wird dadurch die spätere Einführung und Schulung der Software erleichtert. Die Entwicklung erfolgt testgetrieben (Test-Driven Development). Das bedeutet, dass jede neue Funktion zunächst als Aufgabe (Issues) definiert wird. Diese wird anschließend programmiert und mit passenden Tests überprüft, bevor sie in die finale Software integriert wird. Dieses Vorgehen stellt sicher, dass die Software von Anfang an stabil und fehlerfrei entwickelt wird. Außerdem legen wir Code-Qualitätsstandards (z. B. Namenskonventionen für Variablen & Methoden) fest.

4. PROJEKTPHASEN MIT ZEITPLANUNG IN STUNDEN

Analyse	**10 h**
Ist-Analyse der bestehenden Zeiterfassungssysteme und Prozesse	3 h
Aufnahme der Anforderungen durch Befragung der Mitarbeiter	3 h
Erstellung eines Lastenheftes mit klaren Anforderungen	2 h
Prüfung relevanter DSGVO-Richtlinien für die Datenverarbeitung	2 h
Entwurf	**15 h**
Erstellung eines groben Prototyps der Benutzeroberfläche	3 h
Planung der Integration von Frameworks (Toga und Briefcase) für Frontend und Backend sowie Erstellung des SQLite-Datenmodells	2h
Konzeption des Datenmodells für die Zeiterfassung	3 h
Planung des Datenmodells und der Geschäftslogik (inkl. gesetzlicher Vorgaben wie Pausenzeiten und Ruhezeiten)	3 h
Abstimmung des Designs mit dem Chef in Form von Feedback	2 h
Erstellung eines Pflichtenheftes mit detaillierten Umsetzungsplänen	2 h
Implementierung	**35 h**
Entwicklung der Grundfunktionalität der Zeiterfassung (Erfassung und Speicherung von Zeitdaten)	6 h
Erstellung der Eingabemaske und Verarbeitung der Eingaben	3 h
Datenbankintegration mit SQLite und Test der Datenstruktur	4 h
Entwicklung der Benutzeroberfläche mit Toga	4 h
Aufbau einer intuitiven und übersichtlichen GUI	3 h
Implementierung grundlegender Interaktionslogiken	4 h
Umsetzung der Geschäftslogik für Zeitberechnungen, Fehlerprüfungen und Validierung gesetzlicher Vorgaben	4 h
Erstellung von Berichts- und Exportfunktionen (z. B. PDF und Excel)	3 h
Test und Debugging der einzelnen Module (Funktionstest und Integrationstest)	4 h
Abnahme und Einführung	**10 h**
Bereitstellung der Software in einer Testumgebung	2 h
Durchführung eines Probelaufs mit ausgewählten Nutzern	3 h
Anpassung der Software auf Basis des Nutzer-Feedbacks	3 h
Einführung der Software und Schulung der Nutzer	2 h
Finale Prüfung der Software auf DSGVO-Konformität vor dem Release	1 h
Erstellen der Dokumentationen	**9 h**
Erstellung einer Projektdokumentation für die Beschreibung der Planung und Umsetzung	4 h
Erstellung eines Handouts zur Bedienung der Software	2 h
Erstellung einer Entwicklerdokumentation mit Hinweisen zur Wartung und Erweiterung	2 h
Erstellung eines Testprotokolls zur Dokumentation der durchgeführten Tests	2h