ANTRAG FÜR DIE BETRIEBLICHE PROJEKTARBEIT

Fachinformatiker/-in für Anwendungsentwicklung

Antragsteller

Name	Kai Delor
Ausbildungsbetrieb	Rotkreuz-Institut BBW
Ausbildungsberuf	Fachinformatiker/-in für Anwendungsentwicklung
Prüfungsausschuss	IHK [Region]
Ausbildungsjahr	[Jahr]
Antragsdatum	xx. Monat Jahr

---

1. PROJEKTBEZEICHNUNG

Entwicklung eines intelligenten Bewässerungs-Zeitplan-Management-Systems (GießPlan) zur fairen Aufgabenverteilung in der beruflichen Rehabilitation

1.1. Kurzform der Aufgabenstellung

Für das Rotkreuz-Institut BBW soll ein webbasiertes System entwickelt werden, das die wöchentliche Planung von Bewässerungsaufgaben für Teilnehmer in beruflichen Rehabilitationsmaßnahmen automatisiert. Die derzeit manuelle Planung mittels einer laminierten 6-Wochen-Folie, auf die Namen handschriftlich eingetragen werden, soll durch ein intelligentes digitales System ersetzt werden, welches mittels fortgeschrittener Fairness-Algorithmen eine gerechte Aufgabenverteilung über unterschiedliche Teilnahmedauern hinweg gewährleistet. Dabei sollen neue Teilnehmer automatisch mit erfahrenen Mentoren gepaart und historische Fairness-Daten über Jahresgrenzen hinweg getrackt werden.

1.2. Ist-Analyse

Das Rotkreuz-Institut BBW betreut jährlich ca. 50 Teilnehmer in beruflichen Rehabilitationsmaßnahmen, von denen 5-20 Personen gleichzeitig aktiv sind. Diese Teilnehmer

übernehmen wöchentlich die Bewässerung der Pflanzen im Gebäude. Die Organisation erfolgt derzeit durch 2-3 Programm-Koordinatoren mittels einer laminierten Folie.

Aktuelles System:

Die Planung erfolgt auf einer laminierten Folie, die einen 6-Wochen-Zeitraum abdeckt. Die Namen der eingeteilten Teilnehmer werden wöchentlich handschriftlich mit abwischbaren Stiften eingetragen. Nach 6 Wochen wird die Folie gelöscht und neu beschriftet.

Derzeit auftretende Probleme:

Die manuelle Planung mit der Folie ist zeitaufwendig (30-60 Minuten pro 6 Wochen) und fehleranfällig, insbesondere bei der hohen Fluktuation von über 50% jährlich. Das System bietet keinen Überblick über vergangene Einteilungen - nach dem Löschen der Folie sind die historischen Daten verloren. Neue Teilnehmer werden nicht systematisch mit erfahrenen Teilnehmern gepaart, was zu Unsicherheiten bei der Aufgabenausführung führt.

Die Aufgabenverteilung berücksichtigt nicht die individuelle Anwesenheitsdauer der Teilnehmer - wer länger anwesend ist, erhält proportional nicht mehr Aufgaben als Kurzzeit-Teilnehmer, was zu Ungerechtigkeiten führt. Es existieren keine nachvollziehbaren Fairness-Metriken. Bei Rückfragen, warum bestimmte Teilnehmer häufiger eingeteilt wurden als andere, kann keine objektive Begründung gegeben werden.

Die Folie ermöglicht keine digitale Auswertung oder Archivierung. Änderungen an der Planung (z.B. bei Krankheit oder Urlaub) müssen mit Radieren und Neuschreiben vorgenommen werden, was zu unleserlichen Stellen führen kann. Abwesenheitszeiten werden nicht systematisch berücksichtigt, und die Erstellung von Auswertungen für Berichte ist nicht möglich.

Es besteht keine Möglichkeit, historische Daten über Programmperioden hinweg zu analysieren oder Teilnehmer zu vergleichen, die zu unterschiedlichen Zeiten anwesend waren. Die physische Folie ist nur an einem Ort verfügbar und kann nicht remote eingesehen werden.

---

2. ZIELSETZUNG ENTWICKELN / SOLL-KONZEPT

2.1. Was soll am Ende des Projektes erreicht sein?

Am Ende des Projektes soll eine vollständig funktionsfähige webbasierte Anwendung bereitstehen, die folgende Kernfunktionen erfüllt:

Das System soll automatisch faire Bewässerungspläne für 1-52 Wochen generieren, wobei die Fairness über zeitproportionale Algorithmen sichergestellt wird. Teilnehmer mit längerer Anwesenheitsdauer sollen proportional mehr Aufgaben erhalten als Kurzzeit-Teilnehmer. Die

Fairness-Metriken Gini-Koeffizient (< 0.25) und Variationskoeffizient (< 0.30) müssen eingehalten werden.

Neue Teilnehmer (weniger als 4 Wochen anwesend) sollen automatisch mit erfahrenen Mentoren gepaart werden. Pro Woche werden 2 Hauptpersonen und 2 Ersatzpersonen zugeteilt, wobei bei aktivierter Mentor-Anforderung mindestens eine Person ein erfahrener Mentor sein muss.

Das System muss die vollständige Lebenszyklusverwaltung von Teilnehmern ermöglichen: Erfassung von Ankunftsdatum, Abgangsdatum, Abgangsgrund und Mehrfachteilnahmen bei Wiedereintritt. Historische Fairness-Daten müssen über Jahresgrenzen und Programmperioden hinweg gespeichert und berücksichtigt werden.

Koordinatoren sollen manuelle Anpassungen vornehmen können (Personen ersetzen, tauschen, Kommentare hinzufügen), wobei die Fairness-Metriken automatisch aktualisiert werden. Das System muss Datenexport in JSON, CSV und Excel-Format ermöglichen.

Die gesamte Anwendung muss ohne Server-Infrastruktur funktionieren und Daten lokal über die File System Access API speichern. Die Performance muss bei 10 Personen unter 100ms und bei 100 Personen unter 5 Sekunden liegen. Die Testabdeckung muss mindestens 80% betragen.

2.2. Welche Anforderungen müssen erfüllt sein?

Folgende Anforderungen sollen durch das System erfüllt werden:

Funktionale Anforderungen:

- Personenverwaltung mit vollständigem Lebenszyklus (Ankunft, Abgang, Wiedereintritt)
- Automatische Zeitplan-Generierung für 1-52 Wochen
- Implementierung von drei Fairness-Algorithmen (Bayesian Random Walk, Penalized Priority, Gumbel-Softmax)
- Automatisches Mentor-Mentee-Pairing
- Manuelle Anpassungsmöglichkeiten (Ersetzen, Tauschen, Kommentare)
- Fairness-Metriken-Berechnung und -Visualisierung
- Datenexport (JSON, CSV, Excel)
- Lokale Datenspeicherung über File System Access API

Nicht-funktionale Anforderungen:

- Moderne Webtechnologien (React 19, TypeScript 5.7, Vite 6)
- Responsive Design für Desktop und Tablet
- Performance: < 100ms für 10 Personen, < 5s für 100 Personen

- Testabdeckung > 80% durch automatisierte Unit- und Integration-Tests
- Benutzerfreundliche UI mit intuitivem Tab-basiertem Layout
- Datenschutz durch ausschließlich lokale Speicherung
- Keine Server-Abhängigkeit (vollständig client-seitig)
- Wartbarkeit durch klare Architektur und Code-Dokumentation

Qualitätsanforderungen:

- Der Code muss testgetrieben entwickelt werden (TDD)
- Alle öffentlichen APIs müssen dokumentiert sein (JSDoc)
- Der Code muss TypeScript-Strict-Mode erfüllen
- Performance-Tests für Stress-Szenarien (100+ Personen, 52 Wochen)
- Benutzer-Tests mit tatsächlichen Koordinatoren

2.3. Welche Einschränkungen müssen berücksichtigt werden?

Technische Einschränkungen:

Die File System Access API wird nur von modernen Browsern unterstützt (Chrome/Edge 102+, Safari 15.2+). Firefox unterstützt die API derzeit nicht vollständig, weshalb alternative Datenspeicherung über Downloads erforderlich ist.

Die gesamte Anwendung muss client-seitig funktionieren, da keine Server-Infrastruktur bereitgestellt werden kann. Dies schränkt die Möglichkeiten für Backend-Logik, Datenbank-Nutzung und Benutzer-Authentifizierung ein.

Organisatorische Einschränkungen:

Das Projekt muss innerhalb von 70 Stunden gemäß IHK-Vorgaben abgeschlossen sein. Die Entwicklung erfolgt durch einen einzelnen Auszubildenden ohne dediziertes Entwicklungsteam.

Benutzer-Tests können nur mit 2-3 Koordinatoren durchgeführt werden, eine umfangreiche Nutzergruppe steht nicht zur Verfügung.

Fachliche Einschränkungen:

Die Fairness-Algorithmen können mathematisch nicht garantieren, dass in jeder einzelnen Woche perfekte Fairness herrscht. Fairness wird als langfristiges Ziel über mehrere Wochen betrachtet.

Bei sehr wenigen Teilnehmern (< 4 Personen) oder hoher Fluktuation können Wochen entstehen, in denen nicht alle Positionen besetzt werden können. Das System muss diese Lücken erkennen und markieren.

Die Mentor-Anforderung kann nur erfüllt werden, wenn genügend erfahrene Teilnehmer (> 4 Wochen Anwesenheit) vorhanden sind. Ist dies nicht der Fall, werden Constraint-Violations gemeldet.

---

3. PROJEKTSTRUKTURPLAN ENTWICKELN

3.1. Was ist zur Erfüllung der Zielsetzung erforderlich?

Das Projekt wird agil in iterativen Zyklen entwickelt, wobei kontinuierlich Feedback von den Programm-Koordinatoren eingeholt wird. Dies ermöglicht schnelle Anpassungen an Änderungswünsche und reduziert das Risiko von Fehlentwicklungen.

Die Entwicklung erfolgt testgetrieben (Test-Driven Development) mit automatisierten Unit- und Integration-Tests. Jede neue Funktion wird zunächst durch Tests spezifiziert, dann implementiert und schließlich gegen die Tests validiert. Dies gewährleistet hohe Code-Qualität und erleichtert spätere Wartung.

Das gesamte Projekt wird in Git versioniert mit aussagekräftigen Commit-Messages. Die Entwicklung folgt einem Branch-basierten Workflow mit Feature-Branches für neue Funktionen.

Die Dokumentation wird parallel zur Entwicklung erstellt und umfasst API-Dokumentation (JSDoc), Benutzerhandbuch, Architektur-Dokumentation und IHK-Projektdokumentation.

Regelmäßige Code-Reviews werden durchgeführt, um Code-Qualität sicherzustellen und Best Practices zu gewährleisten. Performance-Benchmarks werden kontinuierlich überwacht.

3.2. Aufgaben auflisten

Analyse

- Durchführung einer Ist-Analyse des aktuellen Planungsprozesses
- Ermittlung von Use-Cases durch Interviews mit Koordinatoren
- Durchführung einer Wirtschaftlichkeitsanalyse und Amortisationsrechnung
- Erstellung eines Lastenheftes mit funktionalen und nicht-funktionalen Anforderungen

Entwurf

- Entwurf der Systemarchitektur (Schichtenmodell, Komponenten)
- Erstellung eines Datenmodells (TypeScript Interfaces, Types)
- Spezifikation der Fairness-Algorithmen (Bayesian, Priority, Softmax)
- Entwurf der Benutzeroberfläche (Wireframes, Mockups)
- Auswahl und Begründung des Technologie-Stacks

- Erstellung eines Pflichtenheftes

Implementierung

- Implementierung des Datenmodells (TypeScript Types)
- Implementierung der File Storage (File System Access API)
- Implementierung der Fairness-Algorithmen mit Tests
  - Bayesian State Tracking
  - Penalized Priority Calculation
  - Gumbel-Softmax Selection
  - Fairness Constraints Checking
- Implementierung der Schedule Engine mit Tests
- Implementierung des Person Managers mit Tests
- Implementierung der UI-Komponenten
  - People Tab (Personenverwaltung)
  - Schedule Tab (Zeitplan-Generierung)
  - Manual Tab (Manuelle Anpassungen)
  - Data Tab (Import/Export)
- Integration aller Komponenten
- Bugfixing und Optimierung

Testing

- Erstellung von Unit-Tests für alle Algorithmen
- Erstellung von Integration-Tests für Workflows
- Durchführung von Performance-Tests und Stress-Tests
- Durchführung von Benutzer-Tests mit Koordinatoren
- Fehleranalyse und Behebung

Dokumentation

- Erstellung der API-Dokumentation (JSDoc)
- Erstellung des Benutzerhandbuches
- Erstellung der Architektur-Dokumentation
- Erstellung der IHK-Projektdokumentation
- Erstellung von README und Deployment-Guide

Abnahme und Einführung

- Produktiv-Deployment der Anwendung
- Einweisung der Koordinatoren

- Übergabe an den Betrieb
- Vorbereitung der Abschlusspräsentation

3.3. Grafische und tabellarische Darstellung

Phase	Dauer in Stunden
Analyse	8
Entwurf	10
Implementierung	35
Testing	10
Dokumentation	5
Abnahme und Deployment	2
Summe	**70**

---

4. PROJEKTPHASEN MIT ZEITPLANUNG IN STUNDEN

Analyse (8 h)

- Ist-Analyse durchführen (Dokumentation aktueller Excel-Prozess) **2 h**
- Wirtschaftlichkeitsprüfung und Amortisationsrechnung durchführen **1 h**
- Use-Cases ermitteln durch Interviews mit Koordinatoren **2 h**
- Anforderungen aufnehmen (funktional und nicht-funktional) **2 h**
- Lastenheft erstellen **1 h**

Entwurf (10 h)

- Systemarchitektur entwerfen (Schichtenmodell, Komponentendiagramm) **3 h**
- Datenmodell entwickeln (ER-Diagramm, TypeScript Interfaces) **2 h**
- Fairness-Algorithmen spezifizieren (Pseudocode, Ablaufdiagramme) **2 h**
- UI-Konzept erstellen (Wireframes, Mockups) **2 h**
- Technologie-Stack auswählen und begründen **1 h**

Implementierung (35 h)

- Projekt-Setup (Vite, TypeScript, Dependencies konfigurieren) **2 h**
- Datenmodell implementieren (TypeScript Interfaces und Types) **3 h**

- File Storage entwickeln (File System Access API Integration) **3 h**
- Fairness-Algorithmen implementieren mit Tests **12 h**
  - Bayesian State Tracking (bayesianState.ts + Tests) **4 h**
  - Penalized Priority (penalizedPriority.ts + Tests) **3 h**
  - Softmax Selection (softmaxSelection.ts + Tests) **3 h**
  - Fairness Constraints (fairnessConstraints.ts + Tests) **2 h**
- Schedule Engine implementieren (scheduleEngine.ts Kernlogik) **4 h**
- Person Manager implementieren (personManager.ts CRUD) **3 h**
- UI-Komponenten entwickeln **7 h**
  - People Tab (Personenverwaltung) **2 h**
  - Schedule Tab (Zeitplan-Generierung) **2 h**
  - Manual Tab (Manuelle Anpassungen) **2 h**
  - Data Tab (Import/Export) **1 h**
- Integration aller Komponenten und Bugfixing **1 h**

Testing (10 h)

- Unit-Tests schreiben (70+ Tests für Algorithmen) **4 h**
- Integration-Tests erstellen (20+ Tests für Workflows) **2 h**
- Performance-Tests und Stress-Tests durchführen **2 h**
- Benutzer-Tests mit Koordinatoren **1 h**
- Fehleranalyse und Bugfixes **1 h**

Dokumentation (5 h)

- API-Dokumentation erstellen (JSDoc für alle Public APIs) **1 h**
- Benutzerhandbuch erstellen (USER_GUIDE.md mit Screenshots) **1 h**
- Architektur-Dokumentation erstellen (ARCHITECTURE.md) **1 h**
- IHK-Projektdokumentation fertigstellen **2 h**

Abnahme und Deployment (2 h)

- Produktiv-Deployment durchführen (Build, Hosting) **0.5 h**
- Übergabe an Betrieb (Einweisung Koordinatoren) **0.5 h**
- Präsentationsvorbereitung (Folien, Demo) **1 h**

---

5. EINSATZGEBIET / FACHBEREICH

Das Projekt wird im Berufsbildungswerk des Rotkreuz-Instituts BBW durchgeführt, speziell im Bereich der beruflichen Rehabilitation. Das System wird von den Programm-Koordinatoren eingesetzt, die für die Organisation und Betreuung der Rehabilitationsteilnehmer verantwortlich sind.

Meine aktuelle Tätigkeit im Ausbildungsbetrieb umfasst die Entwicklung webbasierter Anwendungen zur Unterstützung interner Prozesse. Schwerpunkte sind:

- Entwicklung mit modernen JavaScript/TypeScript-Frameworks (React)
- Implementierung von Geschäftslogik und Algorithmen
- Datenbankdesign und Datenmodellierung
- Testing und Qualitätssicherung
- Technische Dokumentation

---

6. PROJEKTUMFELD

Umfeld beim Kunden und im Betrieb

Auftraggeber: Rotkreuz-Institut BBW
Fachbereich: Berufliche Rehabilitation, Programm-Koordination

Interne Ansprechpartner:

- Programm-Koordinatoren (2-3 Personen): Hauptnutzer des Systems, liefern fachliche Anforderungen
- IT-Abteilung: Unterstützung bei technischen Fragen und Deployment
- Ausbildungsbetreuer: Projektverantwortlicher, fachliche und organisatorische Begleitung

Betroffene Nutzergruppen:

- Primär: Programm-Koordinatoren (wöchentliche Planung)
- Sekundär: Rehabilitationsteilnehmer (5-20 aktiv, ~50 pro Jahr)

Technisches Umfeld:

- Nutzung auf lokalen Desktop-PCs (Windows/macOS/Linux)
- Moderne Browser (Chrome 102+, Edge 102+)
- Keine Server-Infrastruktur erforderlich
- Lokale Datenspeicherung über File System Access API

Prozessschnittstellen:

- Einstieg: Import bestehender Teilnehmerdaten aus Excel
- Ausstieg: Export von Zeitplänen und Berichten (JSON, CSV, Excel)
- Integration: Standalone-Anwendung ohne externe Systemanbindungen

---

7. DURCHFÜHRUNGSZEITRAUM

Projektstart: Ab Genehmigung durch den IHK-Prüfungsausschuss
Projektende: 30. April 2026 (spätestens)
Geplante Projektdauer: 6 Wochen (70 Stunden)

---

8. PROJEKTVERANTWORTLICHE/-R IM AUSBILDUNGSBETRIEB

Name: [Betreuer Vorname Nachname]
Position: [Position/Rolle]
Abteilung: [Abteilung]
Telefonnummer: [Telefonnummer]
E-Mail: [E-Mail-Adresse]

Ausbildungsverantwortlicher: [Falls abweichend]
Telefonnummer: [Telefonnummer]
E-Mail: [E-Mail-Adresse]

ANLAGEN

- Anforderungskatalog
- Zeit- und Kostenplanung (detailliert)
- Use-Case-Diagramm
- Wirtschaftlichkeitsanalyse

Datum: xx. Monat Jahr

Unterschrift Auszubildender: _

Unterschrift Ausbildungsbetrieb: _

Eingabe:

- Benutzereingaben über Web-UI
- Lokale JSON-Dateien (Import)

Ausgabe:

- JSON-Dateien (lokaler Speicher)
- CSV-Export (Excel-kompatibel)
- UI-Anzeige im Browser

Keine externen APIs oder Dienste

4.3 Datenschutz

- Alle Daten lokal gespeichert
- Keine Server-Kommunikation
- Keine Cloud-Dienste
- Keine personenbezogenen Daten außer Namen
- DSGVO-konform durch lokale Verarbeitung

---

5. Projektphasen mit Zeitplanung

Phase	Tätigkeiten	Zeit (h)
1. Analysephase	Ist-Analyse, Anforderungsaufnahme, Wirtschaftlichkeitsbetrachtung, Pflichtenheft	8
2. Entwurfsphase	Systemarchitektur, Datenmodell, Algorithmen-Design, UI-Mockups, Technologieauswahl	10
3. Implementierung	Frontend (React/TypeScript), Fairness-Engine, Business Logic, File Storage	35
4. Testphase	Unit-Tests, Integration-Tests, Performance-Tests, Benutzer-Tests	10
5. Dokumentation	Code-Kommentare, API-Dokumentation, Benutzerhandbuch, Projektdokumentation	5
6. Abschluss	Deployment, Übergabe, Präsentationsvorbereitung	2
Gesamt		**70**

Zeitlicher Rahmen

- **Projektbeginn**: [Startdatum]
- **Projektende**: [Enddatum]
- **Projektdauer**: 6 Wochen
- **Gesamtaufwand**: 70 Stunden (gemäß IHK-Vorgaben)

---

6. Dokumentation

6.1 Projektdokumentation

Die Projektdokumentation wird gemäß IHK-Vorgaben erstellt und umfasst:

1. **Analysephase**
   - Ist-/Soll-Analyse
   - Anforderungskatalog (funktional/nicht-funktional)
   - Use-Case-Beschreibungen
   - Wirtschaftlichkeitsbetrachtung
2. **Entwurfsphase**
   - Systemarchitektur (Schichtenmodell)
   - Datenmodell (ER-Diagramm, TypeScript-Interfaces)
   - Algorithmen-Spezifikation
   - UI-Design-Konzept
3. **Implementierungsphase**
   - Technologie-Stack-Begründung
   - Implementierungs-Highlights
   - Herausforderungen und Lösungen
   - Code-Qualitäts-Metriken
4. **Testphase**
   - Test-Strategie und -Coverage
   - Unit-/Integration-/Performance-Tests
   - Benutzer-Feedback
   - Fehlerbehebung
5. **Projektabschluss**
   - Soll-/Ist-Vergleich
   - Lessons Learned
   - Fazit und Ausblick

6.2 Technische Dokumentation

- API-Referenz (alle öffentlichen Funktionen/Interfaces)
- Architektur-Guide (Schichten, Module, Algorithmen)
- Testing-Guide (Test-Strategie, Beispiele)
- User-Guide (Benutzerhandbuch mit Screenshots)

6.3 Entwicklungsdokumentation

- Git-Repository mit 150+ Commits
- README.md mit Installationsanleitung
- Changelog mit Feature-Historie
- Contributing Guidelines

---

7. Präsentation

Die Präsentation für die mündliche Prüfung wird:

- **Dauer**: 15 Minuten
- **Inhalte**:
  - Projektübersicht und Zielsetzung (2 Min)
  - Ist-/Soll-Analyse (2 Min)
  - Systemarchitektur (3 Min)
  - Fairness-Algorithmen (3 Min)
  - Live-Demo (3 Min)
  - Ergebnisse und Ausblick (2 Min)
- **Medien**: PowerPoint/LibreOffice, Live-Demo
- **Fachgespräch**: 15 Minuten

---

8. Anlagen

1. Detaillierte Zeitplanung (Gantt-Chart)
2. Anforderungskatalog
3. Systemarchitektur-Diagramme
4. Datenmodell (ER-Diagramm, TypeScript-Interfaces)
5. UI-Mockups/Screenshots
6. Code-Beispiele (Algorithmen)

9. Erklärung des Prüflings

Hiermit versichere ich, dass ich den Projektantrag selbstständig erstellt habe und dass das geplante Projekt im Rahmen meiner Ausbildung im Betrieb durchgeführt werden kann.

Die angegebenen Zeitansätze entsprechen realistischen Schätzungen auf Basis vergleichbarer Projekte und meiner bisherigen Erfahrungen.

Ich verpflichte mich, das Projekt gemäß diesem Antrag durchzuführen und die Projektdokumentation fristgerecht einzureichen.

Ort, Datum: ____

Unterschrift Auszubildender: ____

---

10. Bestätigung des Ausbildungsbetriebs

Der Ausbildungsbetrieb bestätigt:

- Das Projekt kann im Rahmen der Ausbildung durchgeführt werden
- Die erforderlichen Ressourcen (Hardware, Software, Zeit) stehen zur Verfügung
- Ein fachlicher Betreuer steht für Rückfragen zur Verfügung
- Das Projekt entspricht dem Anforderungsprofil der Abschlussprüfung

---

Ort, Datum: ____

Unterschrift Ausbildungsleiter: ____

Stempel Ausbildungsbetrieb:

Monat Jahr