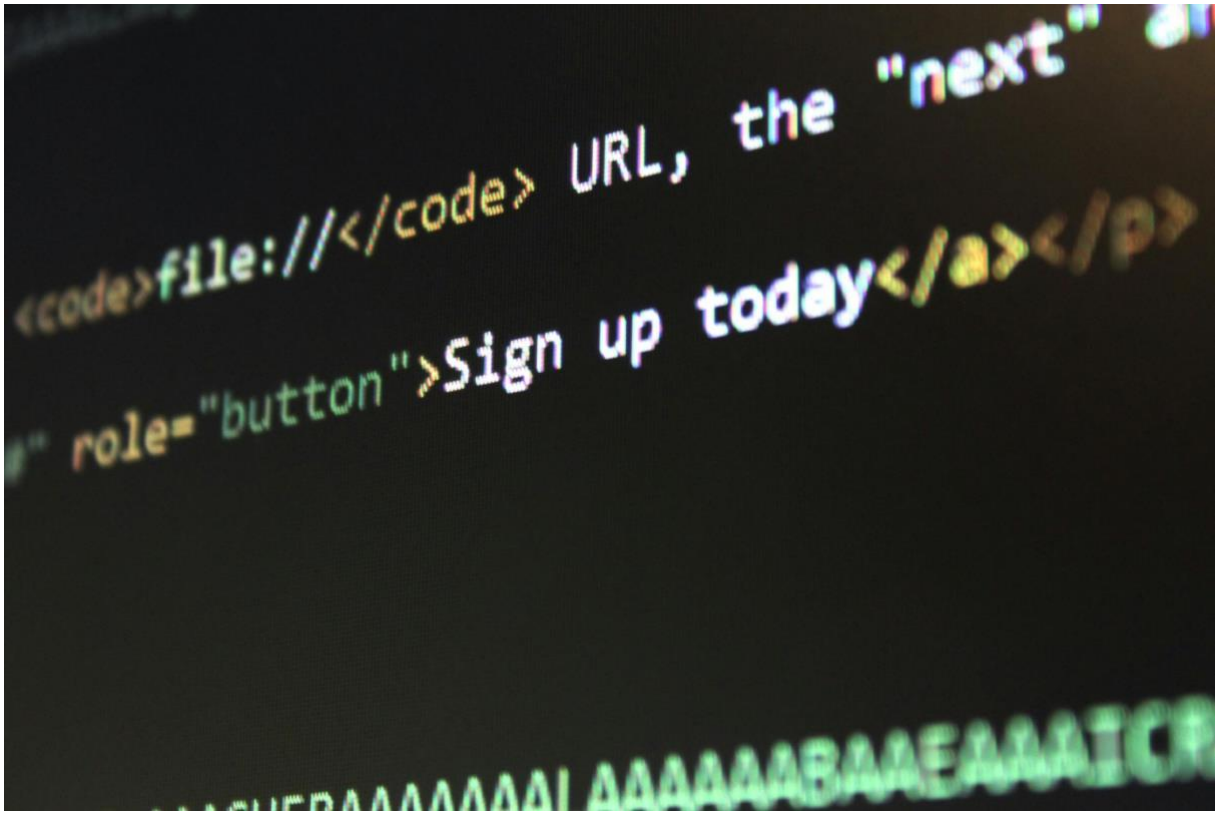

IT Projekt «Time Recording»

System Specification

Kernel Panic

Semester:	6
Autoren:	Ece Kaya
	Furkan Aydin
	Philipp Dunkel
Dozierender:	Lukas Frey
Ort, Datum:	Olten, 28. März 2025

Inhaltsverzeichnis

1 Datenmodelle

1.1 Entity Relationship Diagram (ERD)

Das **ERD zeigt die Tabellenstruktur der Datenbank** mit **Primärschlüsseln (PK), Fremdschlüsseln (FK) und Relationen**.
Es stellt sicher, dass alle Daten konsistent und effizient gespeichert werden.

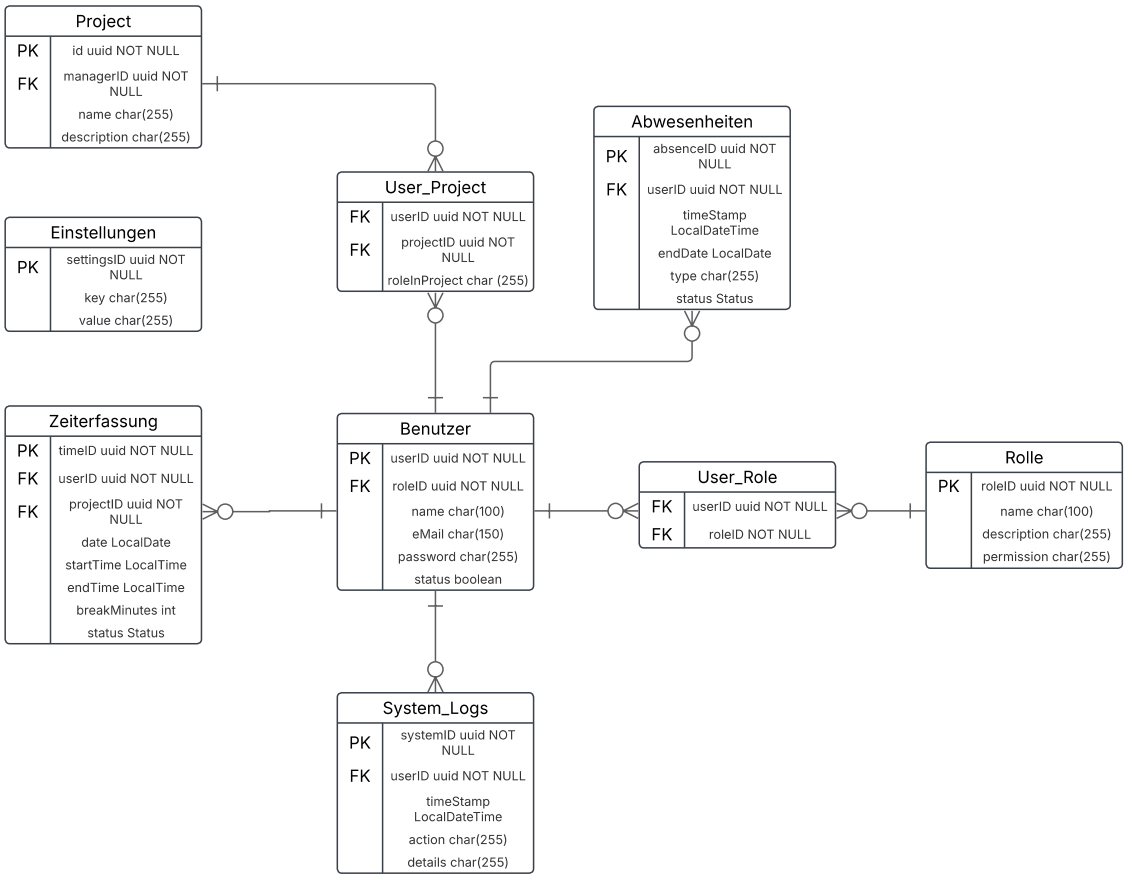

Abbildung 1: Entity Relationship Diagramm (eigene Darstellung)

1.2 Klassendiagram

Das UML-Klassendiagramm zeigt **die objektorientierte Struktur des Systems**, insbesondere **die Klassen, Attribute und Methoden**.
Hier werden keine **Datenbankstrukturen, sondern Software-Klassen und deren Beziehungen** dargestellt.

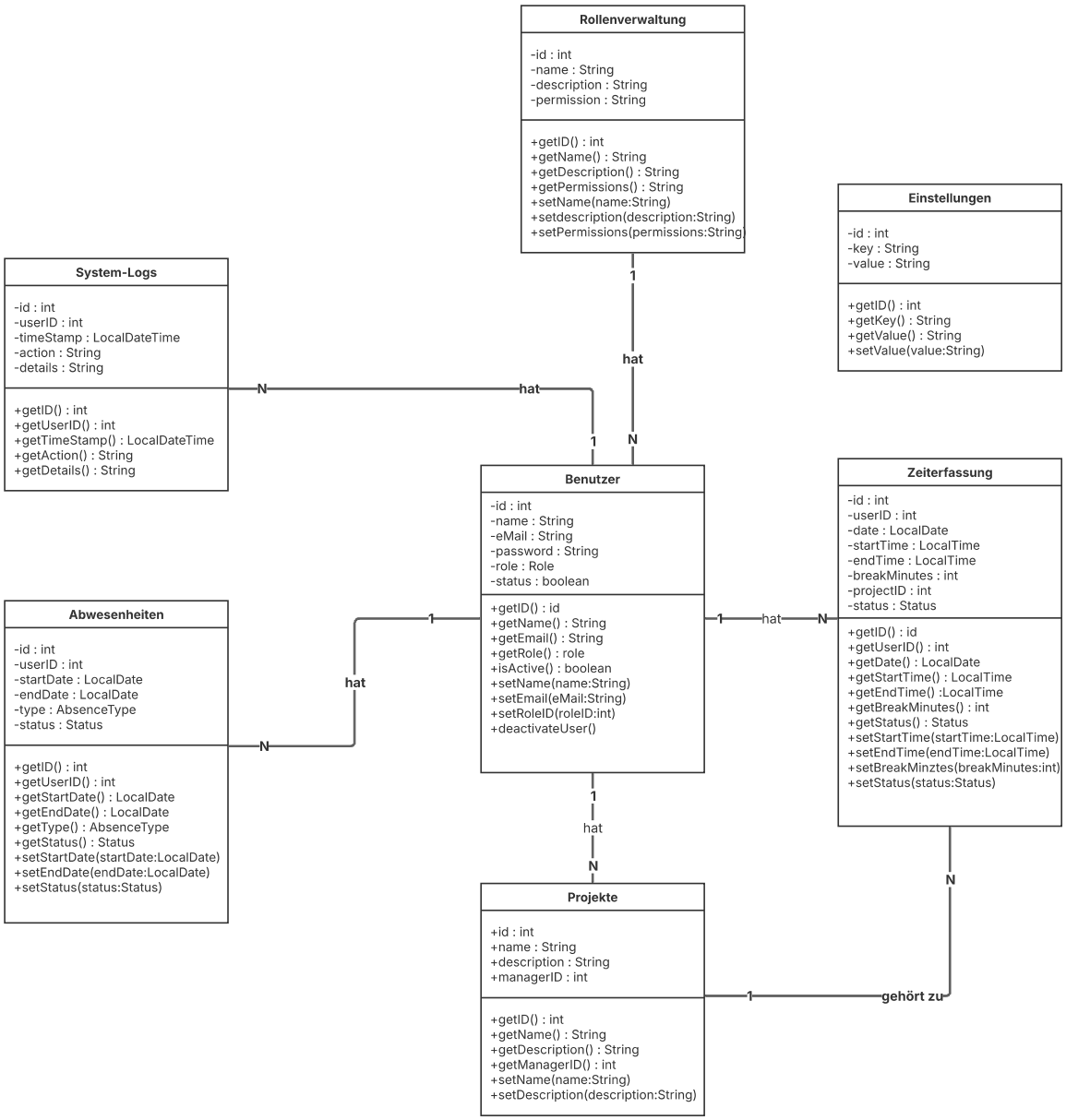

Abbildung 2: Klassendiagramm (eigene Darstellung)

2 Kommunikationsprotokoll

2.1 Übersicht Kommunikation

Die Kommunikation im **Time Recording System** erfolgt über **HTTP (Hypertext Transfer Protocol)** für die Client-Server-Interaktion und **JDBC (Java Database Connectivity)** zur Verbindung mit der Datenbank. HTTP ermöglicht eine flexible und sichere Kommunikation zwischen Frontend und Backend, während JDBC eine stabile und performante Verbindung zur Datenbank sicherstellt. Eine grafische Darstellung ist in **Abbildung 3** ersichtlich.

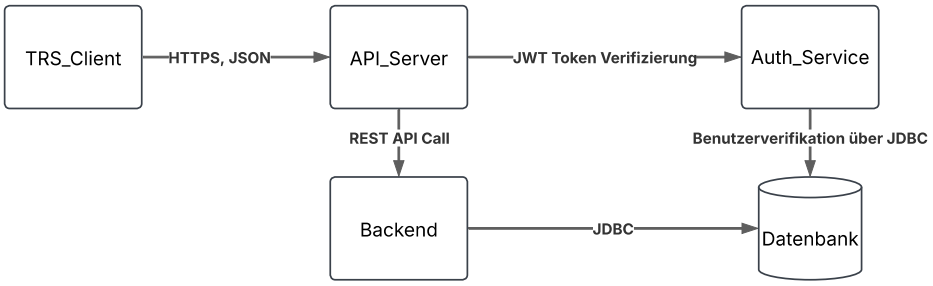

Abbildung 3: Kommunikationsprotokoll (eigene Darstellung)

2.2 Server API

2.2.1 User

2.2.1.1 Registrieren eines neuen Benutzers

Beschreibung	Admin legt neuen Benutzer (Mitarbeiter oder weiteren Admin) an.	
Methode	**POST**	
Pfad	**/api/users**	
Berechtigung	**Admin**	

Anfrage Format	<pre>{
	  "name": "Furkan Dunkel",
	  "email": "f.dunkel@kernel.ch",
	  "role": "EMPLOYEE",
	  "plannedHoursPerDay": 8
	}</pre>	

Header	Authorization: Bearer <token>	

Antwort Payload	200	<pre>{
		  "id": 21,
		  "name": "Furkan Dunkel",
		  "email": "f.dunkel@kernel.ch",</pre>

		"role": "EMPLOYEE" }

	400	{
		"error": "Error"
		}

	403	{
		"error": "Unauthorized – Admin rights required"
		}

Tabelle 1: Registrieren von Benutzern

2.2.1.2 Passwort ändern

Beschreibung	Der eingeloggte Benutzer ändert sein Passwort	
Methode	**POST**	
Pfad	**/api/users/change-password**	
Berechtigung	**Alle User**	

Anfrage Format	{
	"oldPassword": "test123",
	"newPassword": "newsecure123"
	}	

Header	Authorization: Bearer <token>	

Antwort Payload	200	{
		"message": "Password updated successfully"
		}

	400	{
		"error": "Something went wrong"
		}

Tabelle 2: Passwortänderung

2.2.1.3 Link zum Zurücksetzen des Passworts

Beschreibung	Versendet einen Link zum Zurücksetzen des Passworts per E-Mail.
Methode	**POST**
Pfad	**/api/users/reset-password**
Berechtigung	**Alle User**

Anfrage Format	{
	"email": "anna@example.com"
	}

Header	Authorization: Bearer <token>	

Antwort Payload	200	<pre>{
		  "message": "Reset-Link gesendet"
		}</pre>

	400	<pre>{
		  "error": "Something went wrong"
		}</pre>

2.2.1.4 Benutzer bearbeiten

Beschreibung	Admin bearbeitet Benutzerinformationen.	
Methode	**PUT**	
Pfad	**/api/users/{id}**	
Berechtigung	**Admin**	

Anfrage Format	<pre>{
	  "name": "Furkan Kaya",
	  "role": "MANAGER",
	  "plannedHoursPerDay": 7.5
	}</pre>	

Header	Authorization: Bearer <token>	

Antwort Payload	200	<pre>{
		  "message": "User updated successfully"
		}</pre>

	400	<pre>{
		  "error": "Something went wrong"
		}</pre>

2.2.1.5 Benutzer deaktivieren

Beschreibung	Admin deaktiviert ein Benutzerkonto.	
Methode	**PATCH**	
Pfad	**/api/users/{id}/deactivate**	
Berechtigung	**Admin**	
Anfrage Format		
Header	Authorization: Bearer <token>	

Antwort Payload	<pre>{
	  "message": "User deactivated"</pre>	

	}

2.2.1.6 Login eines Benutzers

Beschreibung	Login eines Benutzers (Admin oder Mitarbeiter). Gibt bei Erfolg ein JWT zurück.	
Methode	**POST**	
Berechtigung	**Alle User**	
Pfad	**/api/auth/login**	

Anfrage Format	<pre>{
	  "email": "f.dunkel@kernel.ch",
	  "password": "secure123"
	}</pre>	

Header	Authorization: Bearer <token>	

Antwort Payload	200	<pre>{
		  "token": "eyJhbGciOi...",
		  "user": {
		  "id": 12,
		  "name": "Furkan Dunkel",
		  "role": "EMPLOYEE"
		  }
		}</pre>

	400	<pre>{
		  "error": "Login failed"
		}</pre>

Tabelle 3: Login eines Benutzers

2.2.1.7 Logout eines Benutzers

Beschreibung	Loggt den aktuell angemeldeten Benutzer aus. Invalide Token ggf. auf Serverseite löschen.
Methode	**POST**
Pfad	**/api/auth/logout**
Berechtigung	**Alle Benutzer**

Anfrage Format	<pre>{
	  "token": "eyJhbGciOi..."
	}</pre>

Header	Authorization: Bearer <token>

Antwort Payload	<pre>{
	  "message": "Logout was successful"</pre>

	<pre>"logout": true
	}</pre>

Tabelle 4: Logout eines Benutzers

2.2.1.8 Liste von allen Benutzern

Beschreibung	Gibt eine Liste aller registrierten Benutzer zurück – inklusive Rollen und Status. Wird z. B. in der Admin-Oberfläche zur Verwaltung verwendet.	
Methode	**GET**	
Pfad	**/api/users**	
Berechtigung	**Admin**	
Anfrage Format	{}	
Header	Authorization: Bearer <admin-token>›	

Antwort Payload	200	<pre>{
		"users": [
		{
		"id": 1,
		"name": "Furkan Dunkel",
		"email": "f.dunkel@kernel.ch",
		"role": "EMPLOYEE",
		"active": true,
		"plannedHoursPerDay": 8
		},
		{
		"id": 2,
		"name": "Ece Kaya",
		"email": "e.kaya@kernel.ch",
		"role": "ADMIN",
		"active": true,
		"plannedHoursPerDay": 8
		}
		]
		}</pre>

	403	<pre>{
		"error": "Access denied – Admin privileges required"
		}</pre>

Tabelle 5: Liste von allen Benutzern

2.2.2 Zeiterfassung

2.2.2.1 Zeit erfassen

Beschreibung	Benutzer trägt eine Arbeitszeit ein. Automatische Berechnung der Ist-Zeit und Abweichung zur Sollzeit.	
Methode	**POST**	
Pfad	**/api/time-entries**	
Berechtigung	**Alle User**	

Anfrage Format	<pre>{
	  "date": "2025-03-21",
	  "startTimes": ["08:15", "13:00"],
	  "endTimes": ["11:45", "17:00"],
	  "breaks": [{ "start": "12:00", "end": "13:00" }]
	}</pre>	

Header	Authorization: Bearer <token>	

Antwort Payload	200	<pre>{
		  "id": 40,
		  "actualHours": "07:30",
		  "plannedHours": "08:00",
		  "difference": "-00:30"
		}</pre>

	400	<pre>{
		  "error": "Something went wrong"
		}</pre>

	403	<pre>{
		  "error": "User is not authorized"
		}</pre>

Tabelle 6: Zeiterfassung

2.2.2.2 Zeiterfassung bearbeiten

Beschreibung	Arbeitszeiteintrag bearbeiten.
Methode	**PUT**
Pfad	**/api/time-entries/{id}**
Berechtigung	**Alle User**

Anfrage Format	<pre>{
	  "startTimes": ["08:00"],
	  "endTimes": ["12:00"],
	  "breaks": []
	}</pre>

Header	Authorization: Bearer <token>

Antwort Payload	{
	"message": "Entry updated"
	}

Tabelle 7: Zeiterfassung bearbeiten

2.2.2.3 Zeiterfassung löschen

Beschreibung	Löschen eines Arbeitszeiteintrags.
Methode	**DELETE**
Pfad	**/api/time-entries/{id}**
Berechtigung	**Admin**
Anfrage Format	{}
Header	Authorization: Bearer <token>

Antwort Payload	{
	"message": "Entry deleted"
	}

Tabelle 8: Zeiterfassung gelöscht

2.2.2.4 Eigene Zeiteinträge anzeigen (User)

Beschreibung	Eigene Zeiteinträge anzeigen lassen.
Methode	**GET**
Pfad	**/api/time-entries**
Berechtigung	**Alle User**
Header	Authorization: Bearer <token>

Antwort Payload	{
	"entries": [
	{
	"id": 42,
	"date": "2025-03-21",
	"startTimes": ["08:00", "13:00"],
	"endTimes": ["12:00", "16:30"],
	"breaks": [{ "start": "12:00", "end": "13:00"}]
	"actualHours": "07:30",
	"plannedHours": "08:00",
	"difference": "-00:30"
	"project": {
	"id": 6,

	<pre>"name": "Support"
	}
	}
	{
	"id": 43,
	"date": "2025-03-22",
	"startTimes": ["09:00"],
	"endTimes": ["12:00"],
	"breaks": []
	"actualHours": "03:00",
	"plannedHours": "08:00",
	"difference": "-05:00"
	}
	]
	}</pre>

Tabelle 9: Lesen aller Zeiteinträge

2.2.2.5 Zeiteinträge anzeigen (Admin)

Beschreibung	Admin sieht alle Zeiteinträge eines bestimmten Benutzers.
Methode	**GET**
Pfad	**/api/users/{userId}/time-entries**
Berechtigung	**Admin**
Header	Authorization: Bearer <token>

Antwort Payload	<pre>{
	"entries": [
	{
	"id": 42,
	"user": "Furkan Aydin",
	"userId": 7,
	"date": "2025-03-21",
	"startTimes": ["08:00"],
	"endTimes": ["16:30"],
	"actualHours": "07:30",
	"plannedHours": "08:00",
	"difference": "-00:30"
	"project": {
	"id": 6,
	"name": "Support"
	}
	}
	{
	"id": 43,
	"user": "Furkan Aydin",
	"userId": 7,</pre>

	<pre>"date": "2025-03-22",
	"startTimes": ["09:00"],
	"endTimes": ["12:00"],
	"actualHours": "03:00",
	"plannedHours": "08:00",
	"difference": "-05:00"
	}
	]
	}</pre>

Tabelle 10: Lesen aller Zeiteinträge (Admin)

2.2.3 Reporting

Beschreibung	Monatsbericht eines Users: Planzeit, Ist-Zeit und Differenz.
Methode	**GET**
Pfad	**/api/reports/user/{id}?month=YYYY-MM**
Berechtigung	**Alle User**
Header	Authorization: Bearer <token>

Antwort Payload	<pre>{
	"user": "Furkan Dun",
	"month": "2025-03",
	"totalPlanned": "176:00",
	"totalActual": "171:30",
	"difference": "-04:30"}
	
	}</pre>

Tabelle 11: Monatsübersicht Zeiterfassung

2.2.4 Sicherheit & Protokoll

Beschreibung	Admin sieht alle Änderungen an Zeiteinträgen oder Benutzerkonten.
Methode	**GET**
Pfad	**/api/logs**
Berechtigung	**Admin**
Header	Authorization: Bearer <token>

Antwort Payload	<pre>{
	"logs": [
	{
	"timestamp": "2025-03-20T12:00:00Z",</pre>

	<pre>{
	  "user": "admin",
	  "action": "Edited time entry ID 42"
	  }
	  ]
	}</pre>

Tabelle 12: Änderungen anzeigen

2.2.5 Erinnerungen für Mitarbeitende

Beschreibung	Erinnerungen oder Hinweise für Mitarbeitende (z.B. fehlende Einträge).
Methode	**GET**
Pfad	**/api/notifications**
Berechtigung	**Alle User**
Header	Authorization: Bearer <token>

Antwort Payload	<pre>{
	  "notifications": [
	  {
	  "type": "MISSING_ENTRY",
	  "date": "2025-03-20",
	  "message": "Kein Zeiteintrag vorhanden"
	  }
	  ]
	}</pre>

Tabelle 13: Erinnerungen

2.2.6 Projekte

2.2.6.1 Projekt erstellen

Beschreibung	Admin erstellt ein neues Projekt.	
Methode	**POST**	
Pfad	**/api/projects**	
Berechtigung	**Admin**	

Anfrage Format	<pre>{
	  "name": "Intranet Redesign",
	  "description": "Phase 1: Planung"
	}</pre>	

Header	Authorization: Bearer <token>	

Antwort Payload	200	<pre>{
		  "id": 5,
		  "message": "Projekt erfolgreich erstellt"</pre>

		}

	400	{
		"error": "Something went wrong"
		}

Tabelle 14: Neues Projekt erstellen

2.2.6.2 Projektübersicht

Beschreibung	Gibt eine Liste aller Projekte zurück.
Methode	**GET**
Pfad	**/api/projects**
Berechtigung	**Admin**
Header	Authorization: Bearer <token>

Antwort Payload	{
	"projects": [
	{
	"id": 5,
	"name": "Intranet Redesign",
	"description": "Phase 1: Planung"
	}
	]
	}

Tabelle 15: Projektübersicht

2.2.6.3 Zeiterfassung zuweisen

Beschreibung	Weist eine Zeiterfassung einem Projekt zu.	
Methode	**POST**	
Pfad	**/api/time-entries/{id}/assign-project**	
Berechtigung	**Admin**	

Anfrage Format	{
	"projectId": 5
	}	

Header	Authorization: Bearer <token>	

Antwort Payload	200	{
		"message": "Zeiteintrag zu Projekt erfolgreich zugewiesen"
		}

Tabelle 16: Zeiterfassung zuweisen

2.2.7 Start-/Stopp-Funktion

2.2.7.1 Start-Zeiterfassung

Beschreibung	Startet eine neue Zeiterfassung (z.B. Beginn der Arbeit)	
Methode	**POST**	
Pfad	**/api/time-entries/start**	
Berechtigung	**Alle User**	

Anfrage Payload	<pre>{
	  "projectId": 5
	}</pre>	

Header	Authorization: Bearer <token>	

Antwort Payload	200	<pre>{
		  "entryId": 101,
		  "startTime": "2025-03-28T08:12:00",
		  "message": "Zeiterfassung gestartet"
		}</pre>

	400	<pre>{
		  "error": "Something went wrong"
		}</pre>

Tabelle 17: Start-Zeiterfassung

2.2.7.2 Stopp-Zeiterfassung

Beschreibung	Beendet eine laufende Zeiterfassung.	
Methode	**POST**	
Pfad	**/api/time-entries/{entryId}/stop**	
Berechtigung	**Alle User**	
Anfrage Format	<pre>{}</pre>	
Header	Authorization: Bearer <token>	

Antwort Payload	200	<pre>{
		  "message": "Zeiterfassung gestoppt",
		  "endTime": "2025-03-28T11:15:00",
		  "actualHours": "03:15"
		}</pre>

	400	<pre>{
		  "error": "Something went wrong"
		}</pre>

Tabelle 18: Stopp-Zeiterfassung

2.2.8 Abwesenheiten

2.2.8.1 Abwesenheit eintragen

Beschreibung	Mitarbeitende tragen geplante Abwesenheiten ein.	
Methode	**POST**	
Pfad	**/api/absences**	
Berechtigung	**Alle User**	

Anfrage Format	<pre>{
	"startDate": "2025-04-01",
	"endDate": "2025-04-05",
	"reason": "Urlaub"
	}</pre>	

Header	Authorization: Bearer <token>	

Antwort Payload	200	<pre>{
		"id": 8,
		"message": "Abwesenheit eingetragen" "
		}</pre>

	400	<pre>{
		"error": "Something went wrong"
		}</pre>

Tabelle 19: Abwesenheit eintragen

2.2.8.2 Übersicht Abwesenheiten (User)

Beschreibung	Zeigt eigene Abwesenheiten an.
Methode	**GET**
Pfad	**/api/absences**
Berechtigung	**Alle User**
Header	Authorization: Bearer <token>

Antwort Payload	<pre>{
	"absences": [
	{
	"id": 8,
	"startDate": "2025-04-01",
	"endDate": "2025-04-05",
	"reason": "Urlaub",
	"status": "genehmigt"
	}
	]
	}</pre>

Tabelle 20: Abwesenheiten anzeigen User

2.2.8.3 Übersicht Abwesenheiten (Admin)

Beschreibung	Admin kann Abwesenheiten eines bestimmten Users einsehen.
Methode	**GET**
Pfad	**/api/users/{userId}/absences**
Berechtigung	**Admin**
Header	Authorization: Bearer <token>

Antwort Payload	{
	"absences": [
	{
	"userId": 7,
	"name": "Furkan Aydin",
	"id": 8,
	"startDate": "2025-04-01",
	"endDate": "2025-04-05",
	"reason": "Urlaub"
	}
	]
	}

Tabelle 21: Abwesenheiten anzeigen Admin

2.2.8.4 Abwesenheit genehmigen

Beschreibung	Admin genehmigt eine Abwesenheit
Methode	**PATCH**
Pfad	**/api/absences/{id}/approve**
Berechtigung	**Admin**
Header	Authorization: Bearer <token>

Anwort Payload	{
	"approved": true
	"message": "Absences approved"
	}

Tabelle 22: Abwesenheit genehmigen

3 Arbeitsplanung

Titel	Beschreibung	Zuständig
UI/UX-Design	Gestaltung der Benutzeroberfläche (Zeiterfassung, Admin-Panel)	Ece
API-Entwicklung	Implementierung der REST-Endpunkte zur Zeiterfassung	Furkan
Authentifizierungssystem	Implementierung der JWT-basierten Authentifizierung und Benutzerrollen	Philipp
Datenbank-Design	Modellierung der relationalen Datenbank für Benutzer & Zeiteinträge	Furkan
Frontend-Backend-Integration	Verbindung des UI mit den API-Endpunkten (REST-Requests)	Ece
Benutzerregistrierung	Registrierung neuer Benutzer und Rollenvergabe	Furkan
Benutzerprofil Bearbeiten	Benutzer können ihre Daten aktualisieren	Ece
Passwort ändern	Implementierung der Funktion zur Passwortänderung	Furkan
Zeiterfassungslogik	Logik für Start/Stopp, Bearbeiten & Löschen von Zeiteinträgen	Furkan
Zeiteinträge bearbeiten	Bearbeiten von vorhandenen Zeiterfassungen	Ece
Abwesenheitsverwaltung	Eintragung und Verwaltung von Urlaub und Krankheitstagen	Philipp

Projekte & Zeiterfassung	Zuordnung von Arbeitszeiten zu Projekten	Philipp
Berichte & Statistiken	Entwicklung von Berichts- und Analysetools für Admins	Philipp
Rollen- und Rechteverwaltung	Umsetzung der Rollensteuerung für Benutzer	Ece
UI-Komponenten	Entwicklung von wiederverwendbaren UI-Elementen für Zeiterfassung	Philipp
Testing	Erstellung und Durchführung von Testfällen für API & UI	Team
Deployment	Einrichtung der Server-Infrastruktur (Docker, CI/CD)	Team
Projektdokumentation	Erstellung einer vollständigen Projektdokumentation	Team
Spring Boot-Setup	Projektstruktur, Abhängigkeiten, erste Endpunkte, Security-Konfiguration	Team

Tabelle 23: Arbeitsplanung

4 Verzeichnisse

4.1 Abbildungsverzeichnis

4.2 Tabellenverzeichnis