Raumbuchung Campus Technik

Diplomarbeit: Pflichtenheft

Auftraggeber	Kurt Munter HFTM Grenchen Campus Technik, 2540 Grenchen
Institution	HFTM Grenchen
Begleitperson	Juan Ferreiro
Autor	Mehmet Oezdag
Klassifizierung	Nicht klassifiziert
Status	Pflichtenheft

1. Management Summary

Die vorliegende Diplomarbeit dokumentiert die Entwicklung einer Web-Applikation zur zentralen Raumbuchung am Campus Technik in Grenchen. Das Sekretariat verwaltet aktuell die Reservationen von Schulzimmern, Parkplätzen und dem Filmstudio über zwei separate Systeme: WebUntis für die Schulzimmer und Microsoft Outlook für Parkplätze und das Filmstudio.

Die aktuelle Situation erfordert manuelle Koordination zwischen den Systemen und verursacht einen geschätzten Arbeitsaufwand von 2-3 Stunden pro Woche. Externe Mieter (z.B. Rodania Altersheim) müssen telefonisch oder per E-Mail anfragen, was zu Verzögerungen und potentiellen Doppelbuchungen führt.

Das Ziel des Projekts ist die Entwicklung einer Web-Applikation, die beide Datenquellen in einer zentralen Kalenderansicht vereint. Externe Benutzer können sich registrieren, die Verfügbarkeit einsehen und Buchungsanfragen stellen. Das System sendet automatisch E-Mails an das Sekretariat, welches die finale Buchung im jeweiligen Quellsystem vornimmt.

Die technische Lösung basiert auf einer modernen Web-Architektur mit React/Vue.js Frontend und Node.js/Python Backend. Die Entwicklung erfolgt nach HERMES-Methodik mit einer Mock-First Teststrategie, um unabhängig von den produktiven APIs entwickeln zu können.

Die Wirtschaftlichkeitsbetrachtung zeigt ein jährliches Einsparpotenzial von CHF 17'500 (Mieteinnahmen CHF 15'000 + Admin-Einsparung CHF 3'000 - Hosting CHF 500). Die Amortisation der Entwicklungskosten erfolgt innerhalb des ersten Betriebsjahres.

Inhaltsverzeichnis

1. Management Summary 2
2. Einleitung & Projekthintergrund 4
3. Initialisierung & Studie 5
4. Lösungsanforderungen 8
5. Lösungsarchitektur 12
6. Realisierung & Qualitätssicherung 15
7. Projektmanagement & Planung 18
8. Unterschriften 21
9. Tabellenverzeichnis 22

2. Einleitung & Projekthintergrund

2.1 Unternehmenskontext: HFTM & Campus Technik

Die Höhere Fachschule Technik Mittelland (HFTM) betreibt den Campus Technik in Grenchen. Der Campus verfügt über verschiedene Räumlichkeiten, die sowohl für den Schulbetrieb als auch für externe Vermietung genutzt werden: Schulzimmer, Besprechungsräume, ein Filmstudio sowie Besucherparkplätze.

2.2 Problemstellung und Handlungsbedarf

Die Verwaltung der Ressourcen erfolgt aktuell über zwei getrennte Systeme:

- WebUntis: Verwaltung der Schulzimmer und Stundenpläne
- Microsoft Outlook: Verwaltung der Parkplätze und des Filmstudios

Dieses Vorgehen birgt signifikante Nachteile:

- Keine zentrale Übersicht über alle verfügbaren Ressourcen
- Manuelle Koordination zwischen den Systemen erforderlich
- Externe Mieter haben keinen Self-Service Zugang
- Hoher administrativer Aufwand (2-3 Stunden/Woche)

2.3 Zielsetzung

Entwicklung einer Web-Applikation «Raumbuchung Campus Technik», die eine zentrale Kalenderansicht mit Daten aus beiden Quellsystemen bietet und externen Benutzern ermöglicht, Buchungsanfragen selbstständig zu stellen.

3. Initialisierung & Studie

3.1 IST-Prozess Analyse

Der aktuelle Prozess für eine externe Buchungsanfrage:

1. Externer Mieter kontaktiert Sekretariat (Telefon/E-Mail)
2. Sekretariat prüft Verfügbarkeit in WebUntis ODER Outlook
3. Sekretariat bestätigt/ablehnt per E-Mail
4. Bei Zusage: Manuelle Eintragung im jeweiligen System

3.2 Stärken- und Schwächenanalyse (SWOT)

Stärken des aktuellen Systems:

Nr.	Stärke	Begründung
St01	Etablierte Systeme	WebUntis und Outlook sind im Betrieb bewährt und stabil
St02	Keine Investition	Bestehende Infrastruktur, keine zusätzlichen Kosten
St03	Persönlicher Kontakt	Direkte Kommunikation mit Mietern möglich

Tabelle 1: SWOT Stärken

Schwächen des aktuellen Systems:

Nr.	Schwäche	Auswirkung	Verbesserung
Sw01	Zwei getrennte Systeme	Keine Gesamtübersicht	Zentrale Kalenderansicht
Sw02	Kein Self-Service	Hoher Admin-Aufwand	Web-App für Externe
Sw03	Manuelle Prozesse	Fehleranfällig	Automatisierung

Tabelle 2: SWOT Schwächen

3.3 Wirtschaftlichkeitsbetrachtung

Einsparpotenzial (Wiederkehrend pro Jahr):

Position	Beschreibung	Betrag
Mieteinnahmen	Erhöhte Auslastung durch bessere Sichtbarkeit	+CHF 15'000
Admin-Einsparung	Reduktion manueller Aufwand (2h/Woche × CHF 40 × 40 Wochen)	+CHF 3'200
Hosting-Kosten	Cloud-Infrastruktur (Server, Domain)	-CHF 500
Netto-Nutzen	Jährlicher wirtschaftlicher Vorteil	CHF 17'700

Tabelle 3: Wirtschaftlichkeitsbetrachtung

3.4 Zieldefinition (SMART-Ziele)

Ziel	S (Spezifisch)	M (Messbar)	A (Attraktiv)	R (Realistisch)	T (Termin)
M1: Konzept	Pflichtenheft mit Architektur und Anforderungen	Dokument abgenommen	Klare Projektbasis	HERMES-konform	21.02.26
M2: Prototyp	Lauffähige Web-App mit Kalender und Mock-Daten	Demo funktioniert	Visualisierung der Lösung	6 Wochen Entwicklung	29.04.26
M3: Abgabe	Vollständige Dokumentation und Validierung	DA-Bericht eingereicht	Diplomabschluss	Alle Tests bestanden	13.05.26

Tabelle 4: SMART-Ziele

4. Lösungsanforderungen

4.1 Funktionale Anforderungen

ID	Anforderung	Beschreibung	Priorität
FA01	WebUntis Integration	Abruf der Raumbelegung via JSON-RPC API (getRooms, getTimetable)	Muss
FA02	Outlook Integration	Abruf der Ressourcen via Microsoft Graph API mit OAuth2	Muss
FA03	Kalenderansicht	Tages-, Wochen- und Monatsansicht aller Ressourcen	Muss
FA04	Buchungsformular	Erfassung: Ressource, Datum, Zeit, Kontakt, Zweck	Muss
FA05	E-Mail Service	Automatischer Versand der Anfrage an Sekretariat	Muss
FA06	Benutzerregistrierung	Registrierung mit E-Mail-Verifizierung	Muss
FA07	Login/Logout	Sichere Authentifizierung mit Session-Management	Muss
FA08	Kategoriefilter	Filterung nach Schulzimmer, Parkplatz, Studio	Kann
FA09	Admin-Dashboard	Übersicht aller Buchungsanfragen für Sekretariat	Kann

Tabelle 5: Funktionale Anforderungen

4.2 Nicht-funktionale Anforderungen

ID	Kategorie	Anforderung	Messgrösse
NFA01	Performance	Schnelle Ladezeiten	Seitenaufbau < 3 Sekunden
NFA02	Portabilität	Responsive Design	Desktop, Tablet, Mobile
NFA03	Zuverlässigkeit	Hohe Verfügbarkeit	Uptime > 99%
NFA04	Sicherheit	Sichere Authentifizierung	HTTPS, Passwort-Hash
NFA05	Wartbarkeit	Modularer Code	Dokumentierte API

Tabelle 6: Nicht-funktionale Anforderungen

4.3 User Stories

ID	Rolle	Ich möchte...	Damit...	Akzeptanzkriterium
US01	Externer Mieter	die Verfügbarkeit online sehen	ich selbstständig planen kann	Kalender zeigt freie Slots
US02	Externer Mieter	eine Buchungsanfrage stellen	ich nicht anrufen muss	Formular sendet E-Mail
US03	Sekretariat	alle Anfragen auf einen Blick sehen	ich effizient arbeiten kann	Dashboard mit Übersicht
US04	Sekretariat	wissen, woher die Anfrage kommt	ich den Mieter kontaktieren kann	Kontaktdaten in Anfrage

Tabelle 7: User Stories

4.4 Detaillierte Anforderungen

FA01: WebUntis Integration

Identifikationsnummer	FA01
Priorität	Muss
Beschreibung	Das System muss die Raumbelegung der Schulzimmer aus WebUntis via JSON-RPC API abrufen. Methoden: authenticate(), getRooms(), getTimetable(roomId, startDate, endDate).
Output	Liste aller Schulzimmer mit aktueller Belegung im Kalenderformat
Akzeptanzkriterien	API-Verbindung erfolgreich; Daten werden korrekt im Kalender angezeigt; Antwortzeit < 3 Sekunden
Testkriterien	Unit-Test mit Mock-Daten; Integrationstest mit echter API nach Erhalt der Zugangsdaten

Tabelle 8: Detaillierte Anforderung FA01

FA02: Outlook Integration

Identifikationsnummer	FA02
Priorität	Muss
Beschreibung	Das System muss Ressourcen (Parkplätze, Filmstudio) aus Microsoft Outlook via Graph API abrufen. OAuth2 für Authentifizierung. Endpoints: /calendars, /events.
Output	Liste aller Outlook-Ressourcen mit Verfügbarkeit
Akzeptanzkriterien	OAuth2-Flow funktioniert; Kalenderdaten werden korrekt abgerufen und angezeigt
Testkriterien	Unit-Test mit Mock-Daten; Integrationstest mit Graph API Sandbox

Tabelle 9: Detaillierte Anforderung FA02

FA04: Buchungsformular

Identifikationsnummer	FA04
Priorität	Muss
Beschreibung	Web-Formular zur Erfassung von Buchungsanfragen mit Feldern: Ressource, Datum, Startzeit, Endzeit, Name, E-Mail, Organisation, Verwendungszweck.
Output	Buchungsanfrage in Datenbank gespeichert; E-Mail an Sekretariat gesendet
Akzeptanzkriterien	Validierung aller Pflichtfelder; Bestätigungsmeldung an User; E-Mail enthält alle Details
Testkriterien	Formularvalidierung testen; E-Mail-Versand prüfen; Datenbank-Eintrag verifizieren

Tabelle 10: Detaillierte Anforderung FA04

4.5 Datenmodell

Das System verwendet folgende Datenbank-Tabellen:

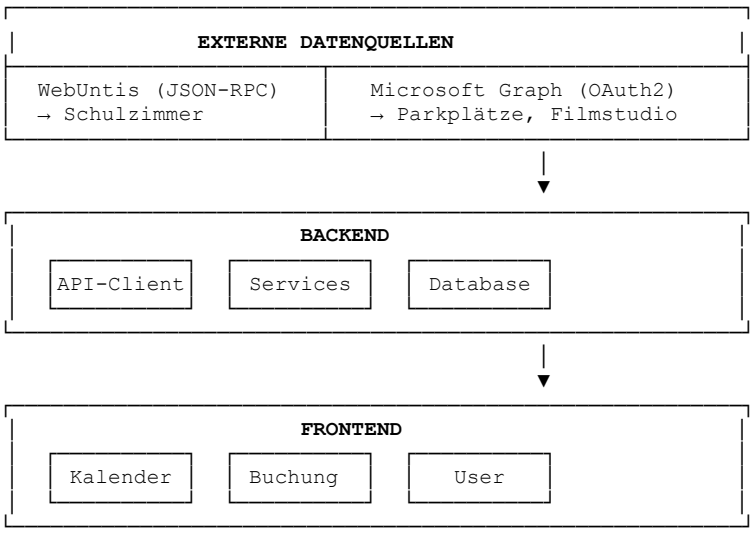
Tabelle	Felder	Beschreibung
users	id, email, password_hash, name, organization, verified, created_at	Registrierte Benutzer
bookings	id, user_id, resource_id, resource_source, date, start_time, end_time, purpose, status	Buchungsanfragen
resources	id, name, type, source, capacity	Ressourcen-Stammdaten (Cache)

Tabelle 11: Datenmodell

5. Lösungsarchitektur

5.1 Systemübersicht

Die Applikation ist als dreischichtige Web-Architektur konzipiert. Sie integriert zwei externe Datenquellen (WebUntis und Microsoft Outlook) in eine zentrale Benutzeroberfläche.



5.2 Technologie-Stack

Komponente	Technologie	Begründung
Backend	Node.js / Python	Gute API-Integration, async Support, breite Community
Frontend	React / Vue.js	Modular, komponentenbasiert, SPA-fähig
Datenbank	PostgreSQL / SQLite	Zuverlässig, SQL-Standard, einfaches Deployment
Kalender UI	FullCalendar.js	Feature-reich, responsive, gute Dokumentation
CSS Framework	Tailwind CSS	Utility-first, responsive, schnelle Entwicklung
E-Mail	SMTP / Nodemailer	Standard-Protokoll, flexibel konfigurierbar

Tabelle 12: Technologie-Stack

5.3 Ressourcen-Mapping

Wichtig: Jede Ressource gehört zu genau EINEM Quellsystem:

Ressourcentyp	Datenquelle	API	Beispiele
Schulzimmer	WebUntis	JSON-RPC	A116, A117, B201, Aula
Besprechungsräume	WebUntis	JSON-RPC	Meeting Room 1, 2
Parkplätze	Outlook	Graph API	P01, P02, P03
Filmstudio	Outlook	Graph API	STUDIO

Tabelle 13: Ressourcen-Mapping

5.4 Schnittstellen

Nr.	Schnittstelle	Protokoll	Beschreibung
S01	WebUntis API	JSON-RPC/HTTPS	Authentifizierung, Raum- und Stundenplanabfrage
S02	Microsoft Graph API	REST/HTTPS	OAuth2 Flow, Kalenderabfrage für Outlook-Ressourcen
S03	SMTP Server	SMTP/TLS	E-Mail-Versand für Buchungsanfragen
S04	Frontend API	REST/JSON	Interne API zwischen Frontend und Backend

Tabelle 14: Schnittstellen

5.5 Abgrenzung (Out of Scope)

- Direkte Buchung in WebUntis/Outlook (nur Anfragen werden erstellt)
- Zahlungsabwicklung / Online-Payment
- Native Mobile Apps (iOS/Android) - nur responsive Web
- Rechnungsstellung / Fakturierung

6. Realisierung & Qualitätssicherung

6.1 Teststrategie: Mock-First

Gemäss Sitzung mit der Begleitperson Juan Ferreiro am 28.01.2026 wird eine Mock-First Strategie angewendet. Dies ermöglicht die Entwicklung und das Testen unabhängig von den produktiven APIs (WebUntis, Microsoft Graph).

Vorteile:

• Entwicklung ohne API-Zugang (Risikominimierung)
• Reproduzierbare Testszenarien
• Schnelle Iteration und Debugging
• Keine Abhängigkeit von externen Systemen

6.2 Testfallbeschreibungen

TC-01: WebUntis Daten abrufen

ID / Bezeichnung	TC-01: WebUntis Mock-Daten abrufen
Referenz	FA01 (WebUntis Integration)
Ziel	Nachweis, dass die API-Anbindung Räume korrekt abruft
Voraussetzung	Mock-Server läuft; mock_webuntis_rooms.json vorhanden
Ablauf	1. API-Client initialisieren 2. getRooms() aufrufen 3. Ergebnis prüfen
Erwartetes Ergebnis	4 Räume werden zurückgegeben (A116, A117, B201, AULA)

Tabelle 15: Testfall TC-01

TC-02: Buchungsanfrage erstellen

ID / Bezeichnung	TC-02: Buchungsanfrage End-to-End
Referenz	FA04, FA05 (Buchungsformular, E-Mail)
Ziel	Nachweis, dass eine Buchungsanfrage korrekt verarbeitet wird
Voraussetzung	Benutzer eingeloggt; Raum A116 verfügbar
Ablauf	1. Kalenderansicht öffnen 2. Freien Slot auswählen 3. Formular ausfüllen 4. Absenden
Erwartetes Ergebnis	Bestätigung angezeigt; E-Mail an Sekretariat; DB-Eintrag erstellt

Tabelle 16: Testfall TC-02

TC-03: Benutzerregistrierung

ID / Bezeichnung	TC-03: Registrierung mit E-Mail-Verifizierung
Referenz	FA06 (Benutzerregistrierung)
Ziel	Nachweis, dass die Registrierung inkl. Verifizierung funktioniert
Voraussetzung	Gültige E-Mail-Adresse verfügbar
Ablauf	1. Registrierungsformular ausfüllen 2. Absenden 3. E-Mail empfangen 4. Verifizierungslink klicken
Erwartetes Ergebnis	Account ist aktiv; Login möglich

Tabelle 17: Testfall TC-03

6.3 Mock-Daten Spezifikation

Datei	Inhalt	Anzahl Datensätze
mock_webuntis_rooms.json	Schulzimmer mit Kapazität und Equipment	4 Räume
mock_webuntis_timetable.json	Beispiel-Stundenplan mit Belegungen	10 Einträge
mock_outlook_resources.json	Parkplätze und Filmstudio	4 Ressourcen
mock_outlook_events.json	Beispiel-Kalendereinträge	5 Events
mock_bookings.json	Beispiel-Buchungsanfragen	3 Buchungen

Tabelle 18: Mock-Daten Übersicht

7. Projektmanagement & Planung

7.1 Vorgehensmodell

Das Projekt wird nach HERMES-Methodik durchgeführt. Die Phasen Initialisierung und Konzept sind abgeschlossen. Die Realisierung erfolgt vom 01.04.2026 bis 13.05.2026 in einem iterativen Ansatz.

7.2 Terminplanung & Meilensteine

Phase	Aktivitäten	Termin
1. Initialisierung	Projektskizze, Projektantrag	✓ 18.11.25 / 13.12.25
2. Konzept	Pflichtenheft, Architektur, Wireframes	21.02.26
3. Realisierung I	Backend-Entwicklung, API-Integration	22.04.26
4. Realisierung II	Frontend-Entwicklung, UI	29.04.26
5. Testing	Unit-Tests, Integrationstests, E2E	06.05.26
6. Abschluss	Dokumentation, Validierung, Abgabe	13.05.26

Tabelle 19: Terminplanung

7.3 Risikomanagement

Nr.	Risiko	Eintrittswahrsch.	Auswirkung	Massnahme
R1	API-Zugang verzögert	Mittel	Hoch	Mock-First Strategie
R2	Technische Komplexität	Mittel	Mittel	Proof of Concept früh
R3	Zeitdruck	Hoch	Hoch	Scope-Priorisierung (Muss/Kann)
R4	OAuth2 Integration	Mittel	Mittel	Microsoft Dokumentation studieren

Tabelle 20: Risikomanagement

7.4 Ressourcenplanung

Für die Durchführung wird ein Arbeitsaufwand von ca. 150 Stunden veranschlagt:

Phase	Aktivität	Aufwand (h)
Konzept	Pflichtenheft, Architektur	20
Entwicklung	Backend (API-Clients, Services)	40
Entwicklung	Frontend (UI, Kalender)	35
Testing	Unit-, Integrations-, E2E-Tests	25
Dokumentation	DA-Bericht, Benutzerhandbuch	20
Diverses	Meetings, Präsentation	10
	Total	150

Tabelle 21: Ressourcenplanung

7.5 Qualitätsmanagement

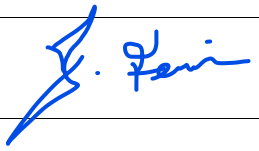
Zur Sicherstellung der Projektziele werden folgende qualitätssichernde Massnahmen eingeplant:

- Code Reviews: Regelmässige Überprüfung des Codes
- Unit Tests: Automatisierte Tests für kritische Funktionen
- Integrationstests: Prüfung des Zusammenspiels der Komponenten
- Validierung: Endabnahme mit dem Auftraggeber

8. Unterschriften

Mit ihrer Unterschrift bestätigen die unten genannten Personen, dass sie dieses Pflichtenheft gelesen haben und mit dem darin beschriebenen Projektumfang und den Anforderungen einverstanden sind.

Ort, Datum: Grenchen, 21.02.2026

Auftraggeber:	(Kurt Munter, HFTM)
Begleitperson:	(Juan Ferreiro) 
Projektleiter:	(Mehmet Oezdag) 

9. Tabellenverzeichnis

Tabelle 1: SWOT Stärken 5
Tabelle 2: SWOT Schwächen 5
Tabelle 3: Wirtschaftlichkeitsbetrachtung 6
Tabelle 4: SMART-Ziele 7
Tabelle 5: Funktionale Anforderungen 8
Tabelle 6: Nicht-funktionale Anforderungen 8
Tabelle 7: User Stories 9
Tabelle 8: Detaillierte Anforderung FA01 10
Tabelle 9: Detaillierte Anforderung FA02 10
Tabelle 10: Detaillierte Anforderung FA04 10
Tabelle 11: Datenmodell 11
Tabelle 12: Technologie-Stack 12
Tabelle 13: Ressourcen-Mapping 13
Tabelle 14: Schnittstellen 14
Tabelle 15: Testfall TC-01 16
Tabelle 16: Testfall TC-02 16
Tabelle 17: Testfall TC-03 16
Tabelle 18: Mock-Daten Übersicht 17
Tabelle 19: Terminplanung 18
Tabelle 20: Risikomanagement 18
Tabelle 21: Ressourcenplanung 19

10. Sitzungsprotokolle

Datum	Teilnehmer	Themen / Beschlüsse
06.12.2025	K. Munter, M. Oezdag	Projektidee vorgestellt; API-Konzept besprochen; Anforderungen definiert
28.01.2026	J. Ferreiro, M. Oezdag	HERMES-Methodik; Mock-First Teststrategie beschlossen; Pflichtenheft-Review

Tabelle 22: Sitzungsprotokolle