Autoren: Brun, Lenny und Ulu, Ege Tuna  
Klasse: Applikationsentwicklung Basel  
Schule: IBZ-Basel  
Datum: 06.12.2024

Modul 295  
Ski-Service Management

Backend für Applikationen realisieren

Table of Contents

[1. Einleitung 2](#_Toc184374545)

[1.1 Ausgangslage 2](#_Toc184374546)

[1.2 Zielsetzung 2](#_Toc184374547)

[1.3 Aufbau der Dokumentation 2](#_Toc184374548)

[2. Projektplanung 2](#_Toc184374549)

[2.1 Organisation 2](#_Toc184374550)

[2.2 Zeitplan 3](#_Toc184374551)

[3. Anforderungen 3](#_Toc184374552)

[3.1 Allgemeine Anforderungen 3](#_Toc184374553)

[3.2 Zusätzliche Anforderungen 3](#_Toc184374554)

[3.3 Randbedingungen 4](#_Toc184374555)

[4. Projektumsetzung 4](#_Toc184374556)

[4.1 Datenbankdesign 4](#_Toc184374557)

[4.1.1 Tabellenstruktur 4](#_Toc184374558)

[4.2 Web-API-Entwicklung 4](#_Toc184374559)

[4.2.1 Authentifizierung 4](#_Toc184374560)

[4.2.2 API-Endpunkte 4](#_Toc184374561)

[4.3 Implementierung der Anforderungen 5](#_Toc184374562)

[4.3.1 Login-Funktion 5](#_Toc184374563)

[4.3.2 Service-Auftragsmanagement 5](#_Toc184374564)

[4.3.3 Statusänderung und Löschen 6](#_Toc184374565)

[5. Test und Validierung 7](#_Toc184374566)

[5.1 Testplan 7](#_Toc184374567)

[5.2 Testergebnisse 9](#_Toc184374568)

[5.3 Fehlerbehebung 10](#_Toc184374569)

[6. Projektmanagement 11](#_Toc184374570)

[6.1 IPERKA-Schritte 11](#_Toc184374571)

[7. Fazit und Ausblick 13](#_Toc184374572)

[7.1 Lessons Learned 13](#_Toc184374573)

[7.2 Verbesserungspotenziale 13](#_Toc184374574)

[7.3 Ausblick 13](#_Toc184374575)

# 1. Einleitung

## 1.1 Ausgangslage

Die Firma Jetstream-Service führt als KMU Ski-Servicearbeiten durch und möchte ihre internen Prozesse digitalisieren. Ziel ist eine web- und datenbankbasierte Lösung zur Verwaltung der Ski-Service-Aufträge.

## 1.2 Zielsetzung

Das Projektziel ist die Entwicklung eines Backend-Systems, das alle Anforderungen an das Auftragsmanagement erfüllt, einschließlich Authentifizierung, Fehlerprotokollierung und API-Dokumentation.

## 1.3 Aufbau der Dokumentation

Diese Dokumentation beschreibt die Projektplanung, die technischen Anforderungen, die Umsetzungsschritte, die Testprozesse und abschließende Erkenntnisse.

# 2. Projektplanung

## 2.1 Organisation

Das Projekt wurde als Partnerarbeit mit einem Zeitbedarf von 12 Stunden durchgeführt, die schlussendlich aber überschritten geworden sind.

|  |  |
| --- | --- |
| Aufgabe | Bearbeiter |
| Datenbankdesign | Lenny |
| Web-API-Entwicklung | Lenny |
| Test und Validierung | Beide |
| Dokumentation | Ege |

## 2.2 Zeitplan

A table with text on it

Description automatically generated

# 3. Anforderungen

## 3.1 Allgemeine Anforderungen

* Login mit Benutzername und Passwort.
* Anzeige und Filterung von Serviceaufträgen nach Priorität.
* Statusänderung und Löschung von Aufträgen.
* Fehlerprotokollierung aller API-Aufrufe.

## 3.2 Zusätzliche Anforderungen

* Kommentare zu Aufträgen hinzufügen.
* Personalisierte Auftragslisten für eingeloggte Mitarbeiter.

## 3.3 Randbedingungen

* Datenbank: MS-SQL oder MySQL.
* Postman zur Web-API-Testung.
* Datenbankzugriff über einen OR-Mapper.

# 4. Projektumsetzung

## 4.1 Datenbankdesign

### 4.1.1 Tabellenstruktur

Die Tabellen wurden in der 3. Normalform (3NF) erstellt.

|  |  |
| --- | --- |
| Tabelle | Beschreibung |
| Mitarbeiter | Speichert Login-Daten der Mitarbeiter. |
| Aufträge | Enthält Informationen zu Serviceaufträgen. |
| Dienstleistungen | Liste der angebotenen Dienstleistungen |

## 4.2 Web-API-Entwicklung

### 4.2.1 Authentifizierung

Die Authentifizierung wurde mit JWT (JSON Web Tokens) implementiert. Mitarbeiter können sich mit Benutzername und Passwort einloggen.

### 4.2.2 API-Endpunkte

|  |  |
| --- | --- |
| Endpunkte | Funktion |
| POST /login | Mitarbeiter-Login |
| GET /orders | Anzeige der Serviceaufträge |
| PUT /orders/{id} | Statusänderung eines Auftrags |
| DELETE /orders/{id} | Löschen eines Auftrags |

## 4.3 Implementierung der Anforderungen

### 4.3.1 Login-Funktion

Die Implementierung der Login-Funktion wurde mit Fokus auf Sicherheit und Benutzerfreundlichkeit durchgeführt. Die folgenden Techniken und Prozesse wurden implementiert:

1. **Passwort-Hashing**:
   * Zum Schutz der Benutzerpasswörter wird das Passwort-Hashing-Verfahren bcrypt verwendet.
   * Beim Erstellen eines neuen Mitarbeiterkontos wird das Passwort mit einem Salt versehen und anschließend gehasht, um die Sicherheit zu erhöhen.
   * In der Datenbank werden nur die gehashten Passwörter gespeichert, sodass selbst bei einem Datenbankleck keine Klartextpasswörter preisgegeben werden.
2. **Token-basierte Authentifizierung**:
   * Die Authentifizierung erfolgt über JSON Web Tokens (JWT).
   * Nach erfolgreichem Login wird ein Token generiert, der die Benutzerinformationen und eine Gültigkeitsdauer enthält.
   * Der Token wird bei jedem API-Aufruf im Authorization-Header gesendet und überprüft, um sicherzustellen, dass nur autorisierte Benutzer auf geschützte Funktionen zugreifen können.

A screenshot of a login screen

Description automatically generatedA screenshot of a login form

Description automatically generated

### 4.3.2 Service-Auftragsmanagement

Das Service-Auftragsmanagement bildet das Herzstück der Anwendung. Die folgenden Funktionen wurden implementiert:

1. **Anzeige der Aufträge**:
   * Alle bestehenden Serviceaufträge können über einen GET-API-Endpunkt abgerufen werden.
   * Eine Paginierungsfunktion wurde hinzugefügt, um die Anzeige großer Datensätze zu erleichtern.
   * Die Daten werden in einem JSON-Format zurückgegeben und enthalten alle relevanten Informationen wie Kundenname, Dienstleistung, Priorität, Status und Kontaktinformationen.
2. **Filter- und Suchfunktionen**:
   * Mitarbeiter können Aufträge nach verschiedenen Kriterien filtern, z. B. Priorität, Status oder Dienstleistung.
   * Eine Volltextsuche wurde implementiert, die es ermöglicht, Aufträge anhand von Kundenname oder E-Mail zu durchsuchen.
3. **Bearbeitung von Aufträgen**:
   * Mitarbeiter können bestehende Aufträge ändern. Dies umfasst die Aktualisierung von Kundeninformationen, die Dienstleistung oder die Priorität des Auftrags.
   * Alle Änderungen werden in der Datenbank protokolliert, um eine Nachverfolgbarkeit sicherzustellen.

### 4.3.3 Statusänderung und Löschen

Die Verwaltung des Status eines Serviceauftrags sowie die Möglichkeit zum Löschen von Aufträgen sind essenzielle Funktionen des Systems:

1. **Statusänderung**:
   * Mitarbeiter können den Status eines Auftrags ändern. Die verfügbaren Stati sind: **Offen**, **In Arbeit** und **Abgeschlossen**.
   * Statusänderungen werden sofort in der Datenbank gespeichert und können von anderen Mitarbeitern in Echtzeit eingesehen werden.
   * Ein Audit-Log wurde hinzugefügt, um Änderungen am Status zu dokumentieren, einschließlich des Datums, der Uhrzeit und des Mitarbeiters, der die Änderung vorgenommen hat.
2. **Löschung von Aufträgen**:
   * Aufträge können aus der Liste gelöscht werden, z. B. bei Stornierungen oder fehlerhaften Einträgen.
   * Anstatt den Auftrag physisch aus der Datenbank zu entfernen, wird ein Lösch-Flag gesetzt. Dies gewährleistet, dass keine Daten verloren gehen und gelöschte Aufträge weiterhin für Berichte oder Auswertungen verfügbar sind.

# 5. Test und Validierung

Die Testphase war entscheidend, um sicherzustellen, dass alle implementierten Funktionen den Anforderungen entsprechen und die Anwendung stabil läuft. Die Tests wurden in einer kontrollierten Umgebung mit definierten Szenarien durchgeführt, die verschiedene Anwendungsfälle abdeckten.

## 5.1 Testplan

Ein umfassender Testplan wurde erstellt, um die wichtigsten Funktionen des Systems zu überprüfen. Die Tests wurden manuell und automatisiert durchgeführt, um eine hohe Abdeckung und Zuverlässigkeit zu gewährleisten.

|  |  |  |
| --- | --- | --- |
| Testfall | Beschreibung | Ergebnis |
| Login mit falschem Passwort | Es wurde getestet, ob ein Konto nach drei aufeinanderfolgenden fehlerhaften Anmeldeversuchen automatisch gesperrt wird. | Bestanden |
| Login mit korrektem Passwort | Es wurde überprüft, ob ein Benutzer mit korrekten Anmeldedaten erfolgreich auf das System zugreifen kann. | Bestanden |
| Token-Ablaufzeit | Nach Ablauf der JWT-Token-Gültigkeit wird der Zugriff verweigert, und eine erneute Authentifizierung ist erforderlich. | Bestanden |
| Auftragsanzeige | Alle bestehenden Aufträge werden korrekt in der Liste angezeigt, einschließlich Filter- und Paginierungsoptionen. | Bestanden |
| Statusänderung eines Auftrags | Ein Auftrag wird erfolgreich von „Offen“ auf „In Arbeit“ geändert und die Änderung wird in der Datenbank gespeichert. | Bestanden |
| Löschen eines Auftrags | Ein Auftrag wird erfolgreich als gelöscht markiert, ohne dass er aus der Datenbank entfernt wird. | Bestanden |
| Filterung nach Priorität | Aufträge mit einer bestimmten Priorität (z. B. "Hoch") werden korrekt gefiltert und angezeigt. | Bestanden |
| Fehlende Berechtigungen | Ein Benutzer ohne gültigen Token wird korrekt daran gehindert, geschützte API-Endpunkte aufzurufen. | Bestanden |
| Fehlerbehandlung | Bei ungültigen Eingaben oder nicht vorhandenen Ressourcen wird eine klare und hilfreiche Fehlermeldung ausgegeben. | Bestanden |

## 5.2 Testergebnisse

Die Testergebnisse zeigen, dass alle Funktionen wie erwartet funktionieren. Nachfolgend finden Sie detailliertere Informationen zu einigen Testfällen:

1. **Login mit falschem Passwort**:
   * **Ziel**: Sicherstellen, dass ein Benutzerkonto nach drei fehlerhaften Anmeldeversuchen gesperrt wird.
   * **Durchführung**: Ein Benutzer versuchte sich dreimal hintereinander mit einem falschen Passwort einzuloggen.
   * **Ergebnis**: Das System sperrte das Konto automatisch und gab eine entsprechende Fehlermeldung zurück.
2. **Auftragsanzeige**:
   * **Ziel**: Überprüfung, ob alle Aufträge in der Datenbank korrekt abgerufen und angezeigt werden.
   * **Durchführung**: API-Aufruf zur Abfrage der Aufträge, einschließlich Paginierung und Filterung nach Priorität.
   * **Ergebnis**: Alle Aufträge wurden korrekt zurückgegeben, und die Filterfunktion funktionierte wie vorgesehen.
3. **Statusänderung eines Auftrags**:
   * **Ziel**: Sicherstellen, dass ein Auftrag von einem Mitarbeiter auf „In Arbeit“ gesetzt werden kann.
   * **Durchführung**: Ein Auftrag wurde über die API von „Offen“ auf „In Arbeit“ geändert.
   * **Ergebnis**: Der neue Status wurde in der Datenbank gespeichert und konnte bei einem erneuten Abruf der Aufträge eingesehen werden.
4. **Fehlende Berechtigungen**:
   * **Ziel**: Überprüfung, ob ein nicht authentifizierter Benutzer keinen Zugriff auf geschützte API-Endpunkte erhält.
   * **Durchführung**: Ein Benutzer versuchte, ohne gültigen JWT-Token auf geschützte Endpunkte zuzugreifen.
   * **Ergebnis**: Der Zugriff wurde verweigert, und eine Fehlermeldung wurde zurückgegeben.

## 5.3 Fehlerbehebung

Während der Tests wurden einige kleinere Fehler identifiziert und behoben:

1. **Fehlerhafte Statusänderung bei ungültigen IDs**:
   * **Problem**: Wenn eine ungültige Auftrags-ID angegeben wurde, gab die API keine Fehlermeldung zurück.
   * **Lösung**: Ein Validierungsmechanismus wurde hinzugefügt, um sicherzustellen, dass nur bestehende Aufträge geändert werden können.
2. **Probleme mit der Filterfunktion**:
   * **Problem**: Die Filterfunktion für Aufträge nach Priorität lieferte bei einigen Anfragen keine Ergebnisse.
   * **Lösung**: Die SQL-Abfrage wurde optimiert, um alle Prioritäten korrekt zu berücksichtigen.
3. **Token-Ablaufzeit nicht überprüft**:
   * **Problem**: Abgelaufene Tokens wurden nicht immer korrekt zurückgewiesen.
   * **Lösung**: Die Token-Validierung wurde angepasst, um das Ablaufdatum strikt zu überprüfen.

# 6. Projektmanagement

Das Projektmanagement folgte den IPERKA-Schritten, einem strukturierten Ansatz zur Durchführung von Projekten, der die Phasen **Informieren**, **Planen**, **Entscheiden**, **Realisieren**, **Kontrollieren** und **Auswerten** umfasst. Diese Methodik sorgte für einen klaren Ablauf und eine zielgerichtete Umsetzung der Projektanforderungen.

## 6.1 IPERKA-Schritte

**Informieren**

In der Informationsphase wurden alle notwendigen Anforderungen und Rahmenbedingungen des Projekts gesammelt.

* Die Aufgabenstellung wurde gründlich analysiert, und die allgemeinen sowie zusätzlichen Anforderungen wurden dokumentiert.
* Quellen wie die Projektbeschreibung, die Anleitungen der Lehrkräfte und bestehende technische Dokumentationen wurden verwendet, um die Ziele klar zu definieren.
* Relevante Technologien und Tools wie JWT für die Authentifizierung, MS-SQL für die Datenbank und Postman für die API-Tests wurden recherchiert und auf ihre Eignung geprüft.

**Planen**

Auf Basis der gesammelten Informationen wurde ein detaillierter Plan erstellt, um das Projekt effizient umzusetzen.

* Die Struktur der Datenbank wurde entworfen, einschließlich eines ER-Modells und der Definition der Tabellenstruktur in der 3. Normalform.
* Das API-Design wurde festgelegt, einschließlich der Endpunkte, der benötigten Funktionen und der Fehlerbehandlung.
* Ein Zeitplan wurde erstellt, der die einzelnen Arbeitsschritte wie Implementierung, Tests und Dokumentation in logische Abschnitte gliederte.

**Entscheiden**

Die Entscheidungsphase beinhaltete die Auswahl der Technologien und Methoden, die für das Projekt am besten geeignet waren.

* **Technologien**: MS-SQL wurde aufgrund seiner Zuverlässigkeit und Unterstützung für relationale Datenbanken gewählt. Für die API-Entwicklung wurde C# mit ASP.NET Core verwendet, da es eine robuste Grundlage für Webanwendungen bietet.
* **Entwicklungsmethoden**: Der OR-Mapper Entity Framework wurde für den Datenbankzugriff gewählt, um die Arbeit mit relationalen Daten zu erleichtern.
* **Testtools**: Postman wurde für die API-Tests eingesetzt, da es benutzerfreundlich ist und umfangreiche Funktionen für API-Tests bietet.

**Realisieren**

Die Realisierung umfasste die eigentliche Entwicklung der Web-API und die Implementierung der Datenbank.

* Die Datenbank wurde gemäß dem zuvor entworfenen Modell erstellt und mit Testdaten gefüllt.
* Die API wurde entwickelt, um die definierten Anforderungen zu erfüllen, einschließlich der Authentifizierung, der Verwaltung von Serviceaufträgen und der Protokollierung von API-Aufrufen.
* Besondere Aufmerksamkeit wurde der Fehlerbehandlung und der Sicherheit gewidmet, einschließlich Passwort-Hashing und Token-basiertem Zugriff.
* Optionale Erweiterungen, wie das Hinzufügen von Kommentaren zu Aufträgen und personalisierte Auftragslisten, wurden ebenfalls umgesetzt, um den Funktionsumfang zu erweitern.

**Kontrollieren**

In der Kontrollphase wurde die Anwendung ausführlich getestet, um sicherzustellen, dass sie den Anforderungen entspricht und stabil läuft.

* Funktionstests: Alle Funktionen, einschließlich Login, Auftragsanzeige und Statusänderung, wurden geprüft.
* Sicherheitstests: Die Sicherheitsmaßnahmen, wie das Sperren von Konten nach mehreren Fehlversuchen und die Überprüfung von Tokens, wurden getestet.
* Belastungstests: Die API wurde unter simulierten Lastbedingungen getestet, um sicherzustellen, dass sie auch bei einer hohen Anzahl von gleichzeitigen Zugriffen stabil bleibt.
* Identifizierte Fehler wurden behoben, und die Tests wurden wiederholt, um die Korrekturen zu verifizieren.

**Auswerten**

In der Auswertungsphase wurden die Ergebnisse reflektiert und dokumentiert.

* **Lessons Learned**: Die Bedeutung einer detaillierten Planung und der frühzeitigen Identifikation von potenziellen Problemen wurde hervorgehoben.
* **Projektreflexion**: Das Projekt wurde als erfolgreich bewertet, da alle definierten Anforderungen erfüllt und die optionalen Erweiterungen umgesetzt wurden.
* **Dokumentation**: Eine umfassende Projektdokumentation, einschließlich des Designs, der Tests und der Ergebnisse, wurde erstellt, um die Arbeit transparent darzustellen und zukünftigen Projekten als Referenz zu dienen.

# 7. Fazit und Ausblick

## 7.1 Lessons Learned

Die Arbeit hat gezeigt, wie wichtig Planung und Testen für die Entwicklung eines stabilen Backends sind.

## 7.2 Verbesserungspotenziale

Die Protokollierung könnte durch ein Dashboard ergänzt werden, um Auswertungen zu erleichtern.

## 7.3 Ausblick

Eine Erweiterung um eine Benutzeroberfläche könnte in zukünftigen Projekten erfolgen.