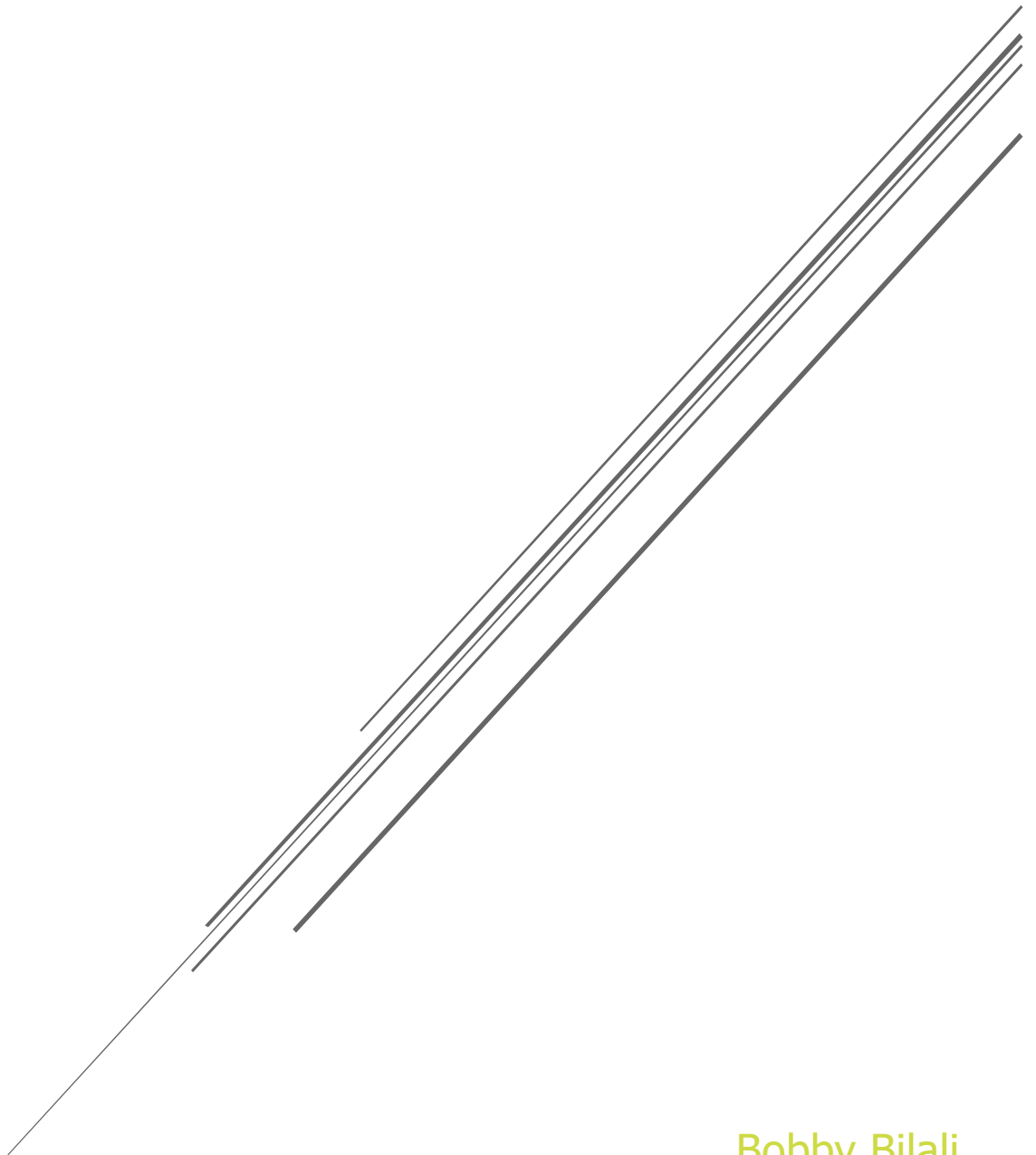


MODUL 295

Projektdokumentation



Bobby Bilali

JETSTREAM API BACKEND

Inhaltsverzeichnis

1	Versionsverzeichnis	2
2	Executive Summary: "Ski-Service Management"	2
3	Einleitung	2
4	Ausgangslage	2
5	Anforderungen	3
5.1	Zusammenfassung der Anforderungen	3
5.2	Zusätzliche Anforderungen	4
6	Informieren	4
6.1	Vorgabe/Anforderungen	4
6.2	Zusätzliche Anforderungen	4
7	Planen	4
8	Zeitplanung	5
8.1	Projektstrukturplan	5
8.2	Zeitplan/Gantt	6
8.3	Entwurf eines Designs	6
8.4	Zu verwendende Tools	6
8.5	Systemarchitektur	7
8.6	Technologien	8
8.7	Validierung und Sicherheit	8
9	Entscheiden	8
10	Realisieren	9
10.1	Backend-Entwicklung	9
10.2	Sicherheitsimplementierung	10
10.3	Testverfahren	10
10.4	Dokumentation und Wartbarkeit	10
11	Kontrollieren	10
11.1	Validierungstests durchführen	10
12	Auswerten	11
12.1	Fazit	13
13	Anhänge	14
13.1	Glossar	14
13.2	NuGet Packages Screenshot	16
13.3	Swagger Screenshots	16
13.4	Postman Collection Test Screenshot	17
13.5	xUnit Testergebnis Screenshot	17
13.6	Frontend Screenshots	18
13.7	Verweise	19

1 Versionsverzeichnis

Version	Autor	Datum	Änderung
1.0	Bobby Bilali	12.11.2023	Erstellung des Dokuments
1.1	Bobby Bilali	13.11.2023	Erstellung der Planung
1.2	Bobby Bilali	17.11.2023	Grobe Fertigstellung des Dokuments
1.3	Bobby Bilali	21.11.2023	Rechtschreibprüfung und Textkorrekturen

Tabelle 1: Versionsverzeichnis

2 Executive Summary: "Ski-Service Management"

Das Backend-System des JetstreamApi-Projekts wurde entwickelt, um die Serviceauftragsverwaltung eines Ski-Service-Unternehmens zu digitalisieren und zu optimieren. Das Projektziel ist es, eine Web- und datenbankbasierte Anwendung zu erstellen, die ein effektives Auftragsmanagement durch autorisiertes Personal ermöglicht. Die Kernfunktionen des Systems beinhalten Authentifizierung, Auftragsübersicht, Statusaktualisierung und Auftragsverwaltung.

3 Einleitung

Dieses Dokument dient als grundlegende Projektdokumentation für die Entwicklung des Backend-Systems der Jetstream-Service Webseite. Das primäre Ziel dieses Backend-Projekts besteht darin, die technische Infrastruktur zur Unterstützung der neuen Webseite zu entwickeln. Ein Hauptaugenmerk liegt auf der Implementierung eines robusten Systems für die Verwaltung von Serviceaufträgen, welches eine nahtlose Integration in die bestehende Webplattform ermöglicht.

4 Ausgangslage

Das Unternehmen Jetstream-Service, ein kleines bis mittelgroßes Unternehmen (KMU), ist im Bereich der Wintersaison Ski-Serviceleistungen tätig und strebt im Zuge der Digitalisierung danach, Serviceaufträge komplett über eine Web- und Datenbankbasierte Anwendung abzuwickeln. Die bereits bestehende Online-Anmeldung soll um zusätzliche Funktionalitäten für ein effektives Auftragsmanagement erweitert werden. Ziel ist es, dass bis zu 10 Mitarbeiter, die mit der Bearbeitung der Serviceaufträge betraut sind, über einen autorisierten und passwortgeschützten Zugang die Aufträge einsehen, bearbeiten und aktualisieren können. Dadurch soll eine effiziente und sichere Abwicklung der Serviceaufträge ermöglicht werden.

Das Projekt soll innerhalb von 10 Tagen umgesetzt werden und beinhaltet die Phasen Informieren, Planen, Entscheiden, Realisieren, Kontrollieren und Auswerten. Ein Projektstrukturplan (PSP) wird genutzt, um die Projektphasen zu strukturieren und die Fortschrittsverfolgung zu erleichtern.

Die technische Umsetzung erfolgt durch die Implementierung einer stabilen und sicheren Datenbankstruktur unter Verwendung von Microsoft SQL Server und einer effizienten Web-API mit .NET Core. Zusätzliche Anforderungen wie die Integration von Analysetools und die Automatisierung von Prozessen sollen den Mehrwert des Systems erhöhen.

Während der Realisierungsphase wird eine modulare Architektur entwickelt, die eine klare Trennung der Verantwortlichkeiten zwischen Client-Browser, Backend-Server und Datenbank bietet. Die Backend-Entwicklung fokussiert sich auf die Erstellung einer skalierbaren Architektur, die leistungsfähig und erweiterbar ist.

Die Sicherheit des Systems soll durch umfassende Tests und Sicherheitsüberprüfungen gewährleistet werden. Dazu gehören automatisierte Unit Tests, Validierungstests und Benutzerakzeptanztests (UAT), um die Zuverlässigkeit und Benutzerfreundlichkeit zu bestätigen.

Dieses Projekt legt den Grundstein für eine fortschrittliche Digitalisierung des Ski-Service-Managements und ebnet den Weg für zukünftige Entwicklungen und Expansionen des Unternehmens.

5 Anforderungen

Für das Auftragsmanagement sind folgende Kernfunktionen erforderlich:

- **Authentifizierung:** Sicheres Einloggen mit Benutzername und Passwort.
- **Auftragsübersicht:** Anzeigen von aktuellen Serviceaufträgen in einer Liste.
- **Auftragsstatus:** Möglichkeit zur Aktualisierung des Status von Serviceaufträgen (Offen, In Arbeit, Abgeschlossen).
- **Auftragsverwaltung:** Löschen von Aufträgen bei Bedarf, z.B. bei Stornierung.

Des Weiteren müssen die Informationen der Online-Anmeldung, die bereits realisiert wurden, bei Bedarf um zusätzliche Details ergänzt werden können:

- **Kundeninformationen:** Name, E-Mail, Telefonnummer.
- **Priorisierung:** Einstufung der Dringlichkeit des Auftrags.
- **Dienstleistungen:** Zuordnung einer der angebotenen Dienstleistungen zu jedem Serviceauftrag, wobei jede Dienstleistung aus einer Liste von Angeboten (Kleiner Service, Großer Service, Rennski-Service, Bindung montieren und einstellen, Fell zuschneiden, Heißwachsen) ausgewählt werden kann.

Diese Anforderungen bilden die Grundlage für die Entwicklung des Backend und stellen sicher, dass das System den operativen Bedürfnissen des Unternehmens gerecht wird.

5.1 Zusammenfassung der Anforderungen

Nr.	Beschreibung
A1	Login Dialog mit Passwort für den autorisierten Zugang der Mitarbeiter (Datenänderungen).
A2	In der Datenbank müssen die Informationen des Serviceauftrags und die Login Daten der Mitarbeiter verwaltet sein.
A3	Erfasste Serviceaufträge abrufbar sein.
A4	Die erfassten Serviceaufträge müssen selektiv nach Priorität abrufbar sein.
A5	Mitarbeiter können eine Statusänderung eines Auftrages vornehmen.
A6	Mitarbeiter können Aufträge löschen (z.B. bei Stornierungen)
A7	Die aufgerufenen API-Endpoints müssen zwecks Fehlerlokalisierung protokolliert sein (DB oder Protokolldatei).
A8	Datenbankstruktur muss normalisiert in der 3.NF sein inkl. referenzieller Integrität
A9	Für die Web-API Applikation muss ein eigener Datenbankbenutzerzugang mit eingeschränkter Berechtigung (DML) zur Verfügung gestellt werden (Benutzer root bzw. sa ist verboten).
A10	Das Web-API muss vollständig nach Open-API (Swagger) dokumentiert sein.
A11	Das Softwareprojekt ist über ein Git-Repository zu verwalten.
A12	Ganzes Projektmanagement muss nach IPERKA dokumentiert sein

Abbildung 1: Zusammenfassung Anforderungen

5.2 Zusätzliche Anforderungen

Nr.	Beschreibung
AO1	Die Mitarbeiter können zu einem Auftrag einen Freitext bzw. Kommentar hinterlegen
AO2	Ein Auftrag kann mit sämtlichen Datenfeldern geändert werden
AO3	Das Login des Mitarbeiters wird nach drei nachfolgenden Falschanmeldungen automatisch gesperrt.
AO4	Personalisierte Auftragsliste des eingeloggten Mitarbeiters. Der Mitarbeiter kann sich zusätzlich zur gesamten Auftragsliste nur die von ihm übernommenen Aufträge ansehen.
AO6	Eingeloggte Mitarbeiter können ein gesperrtes Login zurücksetzen.
AO7	Gelöscht Aufträge werden nicht aus der Datenbank entfernt, sondern nur als gelöscht markiert.

Abbildung 2: Zusätzliche Anforderungen

6 Informieren

In dieser Phase geht es um die umfassende Informationsbeschaffung, die für die Entwicklung des Backend-Systems von Jetstream-Service entscheidend ist; Die gründliche Analyse der technischen Anforderungen, die Erkundung der nötigen Systemarchitektur und die Bewertung von Sicherheitsaspekten. Ziel ist es, ein Verständnis für die technischen Herausforderungen und Bedürfnisse des Projekts zu entwickeln.

6.1 Vorgabe/Anforderungen

Die Kernelemente und Hauptziele des Backend-Projekts umfassen:

- Entwicklung einer stabilen und sicheren Datenbankstruktur für die Verwaltung von Kundeninformationen und Serviceaufträgen.
- Erstellung einer effizienten Web-API, die eine reibungslose Kommunikation zwischen dem Frontend und der Datenbank ermöglicht.
- Implementierung von Authentifizierungs- und Sicherheitsmaßnahmen zum Schutz sensibler Kundendaten.

Die Backend-Entwicklung konzentriert sich auf die Erstellung einer leistungsfähigen und skalierbaren Architektur, die in der Lage ist, den Anforderungen des Jetstream-Service gerecht zu werden. Zudem wird das Backend für die Anbindung an bestehende Systeme und zukünftige Erweiterungen vorbereitet.

6.2 Zusätzliche Anforderungen

Es müssen mindestens zwei der oberen zusätzliche Anforderungen umgesetzt werden:

- Die Mitarbeiter sollen zu einem Auftrag einen Freitext bzw. einen Kommentar hinzufügen können.
- Ein Auftrag kann mit sämtlichen Datenfelder geändert werden.

7 Planen

In der Planungsphase geht es darum, die in der Informationsphase gesammelten Daten und Erkenntnisse in einen strukturierten und umsetzbaren Plan zu überführen. Diese Phase ist entscheidend, um die Grundlage für die erfolgreiche Realisierung des Projekts zu legen.

8 Zeitplanung

Das Projekt findet innerhalb von 10 Tagen statt und wird in mehreren Phasen durchgeführt

Projektphase	Geplante Zeit
Informieren	4h
Planung, Entwurf, Entscheidung	6h
Realisierung	25h
Abnahme	4h
Dokumentation	15h
Gesamt	54h

Tabelle 2: Grobe Planung

8.1 Projektstrukturplan

Ein Projektstrukturplan (PSP) ist ein unverzichtbares Instrument im Projektmanagement, das für die Definition und Visualisierung der Struktur und des Umfangs eines Projekts verwendet wird. Durch den PSP wird das gesamte Projekt in überschaubare Abschnitte oder Arbeitspakete unterteilt, was die Planung, Überwachung und Steuerung vereinfacht.

Der PSP ermöglicht es, den Projektumfang klar zu definieren und die benötigten Ressourcen für jede Phase festzulegen.

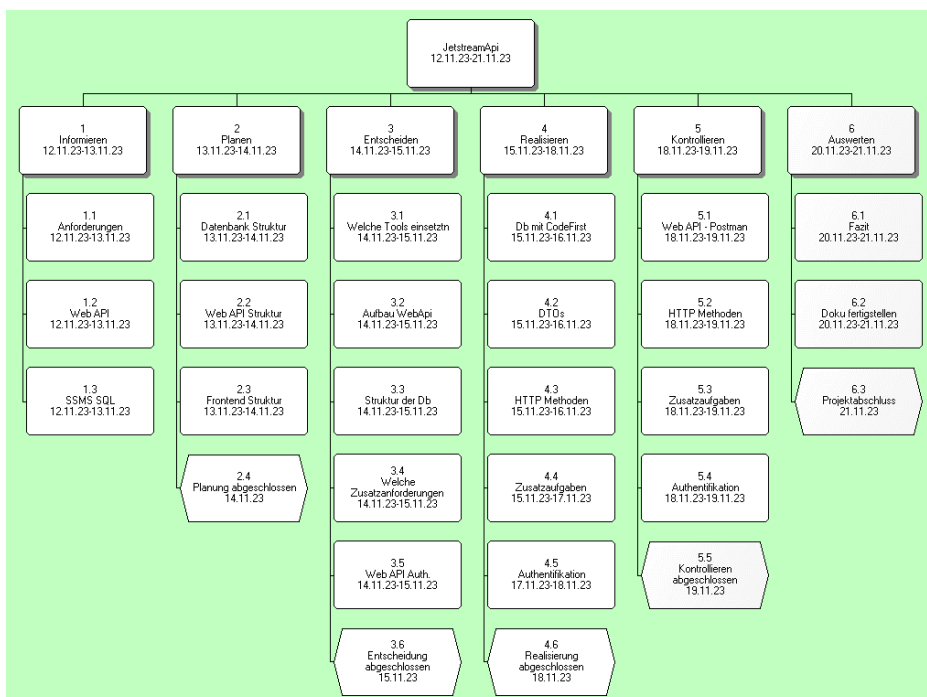


Abbildung 3: Projektstrukturplan

21.11.2023

8.2 Zeitplan/Gantt

Ein detaillierter Zeitplan, idealerweise in Form eines GANTT-Diagramms, ist für die Organisation und Übersichtlichkeit des Projekts unerlässlich. Dieser Zeitplan bildet alle Phasen, Aufgaben und Meilensteine des Backend-Entwicklungsprozesses ab. Für das JetstreamApi-Projekt wurde ein entsprechender Zeitplan erstellt, der die einzelnen Entwicklungsschritte klar strukturiert und terminiert.

8.3 Entwurf eines Designs

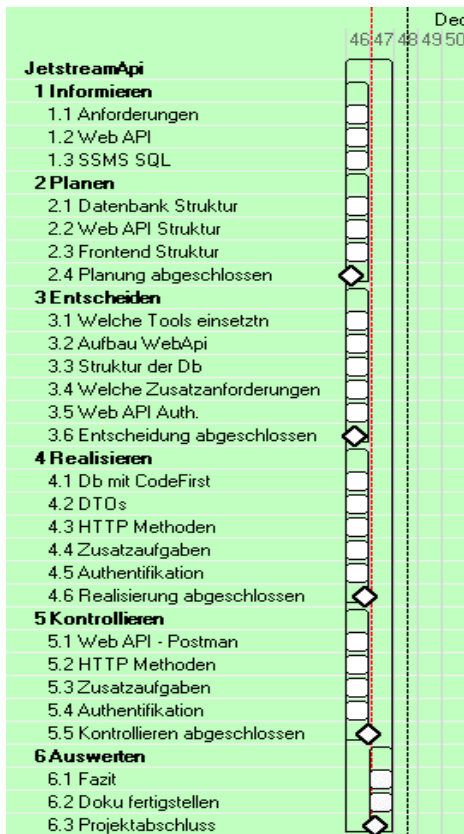


Abbildung 4: GANTT-Diagramm

8.4 Zu verwendende Tools

Für die Entwicklung und Verwaltung des Projekts werden verschiedene Tools verwendet:

GitHub: Dient als zentrales Repository für den Quellcode, ermöglichte Versionskontrolle.

Visual Studio und Visual Studio Code: Diese integrierten Entwicklungsumgebungen (IDEs) werden für die Programmierung und das Debugging des Codes eingesetzt.

Microsoft SQL Server Management Studio: Wird verwendet, um die Datenbanken zu verwalten, zu konfigurieren und zu testen.

SwaggerUI: Dient zur Dokumentation und zum Testen der RESTful APIs.

Postman: Wird eingesetzt, um die Entwicklung und das Testen der APIs zu erleichtern.

8.5 Systemarchitektur

Die Systemarchitektur des JetstreamApi-Projekts ist klar in drei Schichten aufgeteilt, die die Kommunikation und den Datenfluss zwischen dem Client-Browser, dem Backend-Server und der Datenbank verdeutlichen.

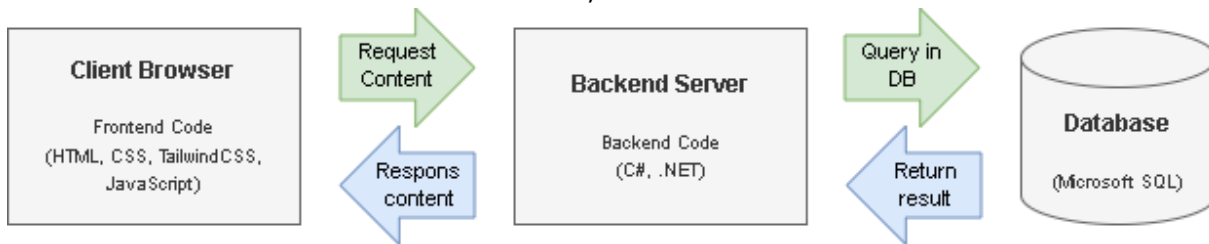


Abbildung 5: Systemarchitektur

- **Client-Browser:** Nutzer interagieren mit der Anwendung über einen Webbrowser, in dem der Frontend-Code ausgeführt wird. Dieser besteht aus HTML, CSS und JavaScript, wobei auch Frameworks wie TailwindCSS zum Einsatz kommen.
- **Backend-Server:** Wenn Inhalte angefordert werden, verarbeitet der Backend-Server diese Anfragen. Er ist mit C# und .NET programmiert und stellt die Verbindung zur Datenbank her, um erforderliche Daten abzufragen oder zu manipulieren.
- **Datenbank:** Die Datenbank, die auf Microsoft SQL Server basiert, speichert und verwaltet alle erforderlichen Daten. Sie empfängt Abfragen vom Backend-Server, führt diese aus und gibt die Ergebnisse zurück, die dann an den Client-Browser gesendet werden können.

Diese strukturierte Architektur ermöglicht eine klare Trennung der Verantwortlichkeiten und erleichtert die Wartung und Skalierung der Anwendung.

8.5.1 Datenbankstruktur

Die Datenbank ist so konzipiert, dass sie effizient Kundendaten und Serviceaufträge verwalten kann. Es wird ein relationales Datenbankschema verwendet, das die Integrität und Sicherheit der Daten gewährleistet.

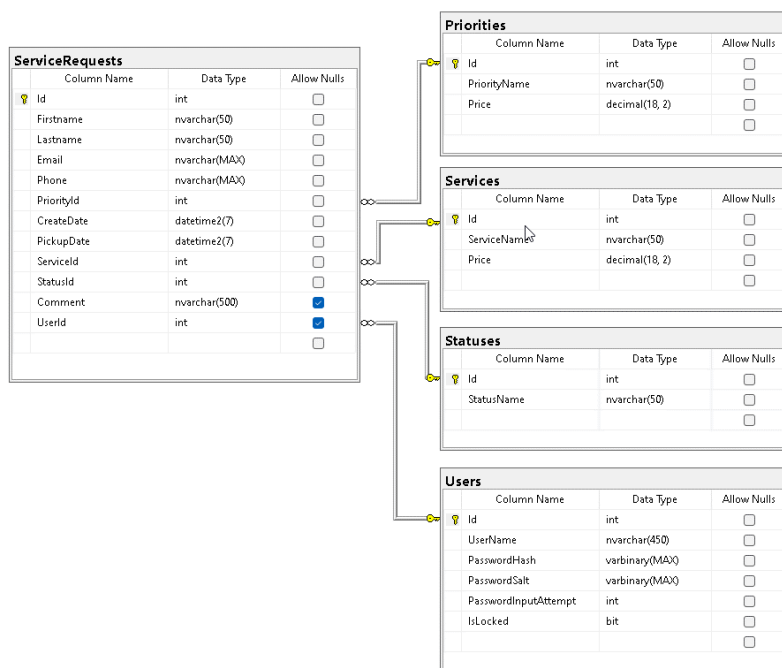


Abbildung 6: Datenbankdiagramm

21.11.2023

8.5.2 Web-API

Die Web-API ist das Herzstück des Backends und ermöglicht eine nahtlose Kommunikation zwischen dem Frontend und der Datenbank. Sie ist auf Effizienz und Sicherheit ausgelegt, um eine schnelle Verarbeitung von Anfragen und den Schutz sensibler Kundendaten zu gewährleisten.

8.6 Technologien

Die Auswahl der Technologien für die Backend-Entwicklung erfolgt mit dem Ziel, ein robustes und skalierbares System zu schaffen, das die spezifischen Anforderungen des JetstreamApi-Projekts erfüllt. Die gewählten Technologien umfassen:

- **Datenbanktechnologie: Microsoft SQL Server**
Für die Datenverwaltung wurde Microsoft SQL Server als Datenbankmanagementsystem gewählt. Es bietet eine umfangreiche Funktionalität für komplexe Abfragen und Transaktionen, hohe Verfügbarkeit, sowie Sicherheitsfeatures, die für Unternehmensanwendungen entscheidend sind.
- **Serverseitige Programmierung: .NET Core**
Als Plattform für die serverseitige Programmierung wurde .NET Core ausgewählt. .NET Core ist besonders geeignet für die Entwicklung von Webanwendungen, da es eine hohe Performance, Sicherheit und Flexibilität bei der Integration verschiedener Dienste bietet.

8.7 Validierung und Sicherheit

Ein wesentlicher Aspekt der Backend-Entwicklung ist die Implementierung von Validierungs- und Sicherheitsmechanismen, insbesondere für die Verarbeitung von Kundeninformationen. Dies umfasst:

- **Validierung:** Strenge Überprüfung der Eingabedaten, um die Korrektheit und Vollständigkeit der Kundeninformationen zu gewährleisten.
- **Sicherheit:** Umsetzung von Sicherheitsmaßnahmen wie Authentifizierung, Autorisierung und Datenverschlüsselung, um den Schutz der Kundendaten zu gewährleisten.

9 Entscheiden

In der Entscheidungsphase des Backend-Entwicklungsprozesses des JetstreamApi-Projekts werden strategische Entscheidungen getroffen, um die Ziele des Projekts zu erreichen und eine effiziente, sichere und benutzerfreundliche Anwendung zu gewährleisten. Die Entscheidungen werden auf Basis der gesammelten Informationen aus der Ausgangslage und den definierten Anforderungen getroffen und umfassten folgende Schlüsselbereiche:

Technologie-Stack:

- **Wahl der Datenbanktechnologie:** Microsoft SQL Server wurde aufgrund der Anforderungen und seiner Leistungsfähigkeit, Zuverlässigkeit und Skalierbarkeit als Datenbanktechnologie ausgewählt.
- **Serverseitige Programmiersprache:** .NET Core wurde als serverseitige Plattform ausgewählt, da es eine Anforderung ist und eine hohe Performance, Sicherheit und Kompatibilität mit verschiedenen Systemen und Diensten bietet.

Architektur-Entscheidungen:

- **Entscheidung für eine modulare Architektur:** Um Flexibilität und Skalierbarkeit zu gewährleisten, wurde eine modulare Architektur gewählt, die die unabhängige Entwicklung und Wartung von Systemkomponenten ermöglicht.
- **API-Strategie:** Die Entscheidung fiel auf die Entwicklung von RESTful-APIs, um eine standardisierte und plattformunabhängige Kommunikation zwischen Frontend und Backend sicherzustellen.

Sicherheitskonzept:

- **Authentifizierungsmechanismen:** Starke Authentifizierungsverfahren wurden implementiert, um sicherzustellen, dass nur berechtigte Nutzer Zugriff auf die Systemfunktionen erhalten.
- **Datenverschlüsselung:** Es wurde entschieden, sensible Daten zu verschlüsseln, um Datenschutz und Compliance-Anforderungen zu erfüllen.

Auswahl der Entwicklungswerkzeuge:

- **Entwicklungsumgebungen:** Visual Studio und Visual Studio Code wurden aufgrund ihrer umfangreichen Funktionen und Unterstützung für .NET Core als primäre Entwicklungswerkzeuge gewählt.
- **API-Design und Testwerkzeuge:** SwaggerUI wird für das API-Design und Postman für das Testen der APIs ausgewählt, um eine effiziente Entwicklung und Wartung der API-Schnittstellen zu gewährleisten.
- **Versionskontrolle:** GitHub wird als zentrales Tool für Versionskontrolle verwendet.

Diese Entscheidungen bilden die Basis für die weitere Umsetzung und Entwicklung des Backend-Systems und sorgen dafür, dass die Anwendung den technischen und geschäftlichen Anforderungen gerecht wird.

10 Realisieren

Die Realisierungsphase transformiert die geplanten und entschiedenen Architektur- und Anforderungsdetails in eine funktionierende Lösung. Während dieser Phase wird das Backend-System entwickelt und die zugehörigen Tests erstellt. Die erzielten Schlüsselergebnisse umfassen:

10.1 Backend-Entwicklung

Die Backend-Entwicklung ist eine zentrale Säule des Gesamtsystems. Ziel ist es, eine robuste und sichere Server-Umgebung zu schaffen, die eine reibungslose Verarbeitung und Verwaltung der Serviceanfragen ermöglicht.

Aufbau der Backend-Infrastruktur:

- Einrichtung und Konfiguration und Vorbereitung der Datenbank auf Microsoft SQL Server.
- Implementierung der Geschäftslogik in .NET Core und Entwicklung der notwendigen API-Endpunkte unter Berücksichtigung der Sicherheitsanforderungen.

Datenbankdesign und -implementierung:

- CodeFirst-Ansatz mit Entity Framework Core zur Erstellung und Verwaltung des relationalen Datenbankschemas.
- Optimierung der Datenmodelle und Einrichtung von Datenmigrationen für eine flexible Datenverwaltung.

Projektstruktur:

Organisation des Codes in einer modularen Struktur zur Förderung der Wartbarkeit und Skalierbarkeit.

- **Controllers:** 'ServiceRequestsController' und 'UsersController' behandeln HTTP-Anfragen.
- **Data:** Beinhaltet Klassen und Konfigurationen für die Datenbankzugriffsschicht, einschließlich des Datenbankkontexts zur Verwaltung der Datenbankdaten. Hier werden auch Seeds definiert, um Benutzer und Dummy-Serviceanfragen in der Datenbank zu erstellen.
- **DTOs:** Spezifizieren die Struktur für den Datenaustausch zwischen API und Clients.
- **Interfaces:** Definiert Verträge für die Services und Repository-Klassen, um die Abstraktion und lose Kopplung im Code zu fördern, was das Testen und die Wartung erleichtert.

21.11.2023

- **Middleware:** Für zentrales Fehlermanagement und andere Middleware-Funktionen.
- **Migrations:** Verwalten Änderungen am Datenbankschema.
- **Models:** Definieren die Datenbankentitäten und Geschäftsobjekte.
- **Services:** Beinhalten die Geschäftslogik und den Datenzugriff.

Die Struktur fördert die Wiederverwendbarkeit von Code, erleichtert Tests und ermöglicht ein effizientes Teamwork.

API-Entwicklung:

- Entwicklung von RESTful APIs mit .NET Core unter Einhaltung der OpenAPI-Spezifikationen.
- Sicherstellung der API-Kompatibilität mit dem Frontend und anderen Diensten, die auf das Backend zugreifen.

10.2 Sicherheitsimplementierung

Ein umfassendes Sicherheitskonzept wird implementiert, um die Integrität und Vertraulichkeit der Daten zu gewährleisten:

- Einsatz von HTTPS und Authentifizierungstoken zur Sicherstellung der sicheren Kommunikation.
- Verschlüsselungsverfahren zum Schutz sensibler Daten und regelmäßige Sicherheitsaudits.

10.3 Testverfahren

Eine Teststrategie wird etabliert, um die Zuverlässigkeit des Backends zu gewährleisten:

- Unit-Tests für die Komponentenprüfung und Integrationstests zur Überprüfung der Systeminteraktionen.
- Nutzung von Postman für End-to-End-Tests und zur Sicherstellung des korrekten Datenflusses.

10.4 Dokumentation und Wartbarkeit

Die Dokumentation und Wartung des Systems werden mit folgenden Werkzeugen unterstützt:

- SwaggerUI zur detaillierten Dokumentation der API-Endpunkte.
- GitHub zur Versionskontrolle und als Plattform für die Zusammenarbeit im Entwicklungsteam.

Die Realisierung des Backend-Systems wird iterativ durchgeführt, wobei kontinuierliches Feedback aus dem Testing und von den Stakeholdern zur stetigen Verbesserung und Anpassung des Systems beigetragen hat.

11 Kontrollieren

In der Phase werden umfangreiche Tests durchgeführt, um die Funktionalität, Sicherheit und Stabilität der entwickelten Lösung zu gewährleisten. Verschiedene Validierungstests werden etabliert, um sicherzustellen, dass die Anforderungen erfüllt werden und entsprechend den Spezifikationen funktionieren:

11.1 Validierungstests durchführen

Die Tests wurden mit xUnit und Postman durchgeführt. Der xUnit-Test konzentrierte sich ausschließlich auf die Funktionalität des User-Logins, da es zeitlich nicht möglich war, weitere Tests zu erstellen. Mir ist bewusst, dass xUnit für umfangreichere Tests genutzt werden kann und auch sollte.

1. Eingabevalidierung:

- **Namen-Validierung:** Überprüfung, dass Namen entsprechend den festgelegten Kriterien, ohne unerwünschte Leerzeichen, eingegeben werden.
- **E-Mail-Validierung:** Einsatz von regulären Ausdrücken (Regex), um die Struktur der E-Mail-Adressen zu verifizieren.
- **Telefonnummern-Validierung:** Sicherstellung, dass Telefonnummern in einem validen Format und entsprechend den internationalen Standards eingegeben werden.

2. API-Kommunikation Postman Test (Abbildung 10: Postman Collection Test):

- Überprüfung der Funktionsfähigkeit und Stabilität der API-Endpunkte.
- Testen der Datenübermittlung und -verarbeitung zwischen Frontend und Backend.
- Validierung des korrekten Speicherns und Abfragens der Daten in der SQL-Datenbank.

3. Sicherheitstests:

- Implementierung von Tests für Authentifizierungsverfahren, um die Sicherheit der Endpunkte zu gewährleisten.
- Überprüfung von Verschlüsselungsprotokollen und Datensicherheit.

4. Integrationstests:

- Testen der Interaktionen zwischen verschiedenen Backend-Komponenten und Diensten.
- Überprüfung der korrekten Zusammenarbeit zwischen dem Datenbankmanagement, Geschäftslogik und API-Schicht.

5. Benutzerakzeptanztests (UAT):

- Zusammenarbeit mit Stakeholdern, um die Backend-Funktionen in realen Anwendungsfällen zu testen.
- Anpassung des Backends auf Basis des direkten Feedbacks von Endnutzern, um die Usability zu verbessern.

Die Durchführung dieser Tests und Überprüfungen war unerlässlich, um zu garantieren, dass das System reibungslos funktioniert und den Anforderungen genügt. Nur durch diese sorgfältige Kontrolle kann sichergestellt werden, dass das Backend-System bereit für den produktiven Einsatz ist.

12 Auswerten

In der SQL-Datei befindet sich der Befehl, um einen neuen Benutzer zu erstellen. In den Dateien 'RootConnectionString.txt' und 'UserConnectionString.txt' sind die Verbindungsstrings hinterlegt, um zuerst mit dem Root-Benutzer den neuen Benutzer zu erstellen und anschließend mit dem Benutzerkonto die Verbindung zu testen.

Die Auswertungsphase dient dazu, nach Fertigstellung des Projekts eine umfassende Bewertung der verschiedenen Aspekte des entwickelten Systems vorzunehmen. Die Analyse befasst sich mit der Funktionalität, der technischen Umsetzung, der Effizienz der Entwicklung sowie den erreichten Geschäftszielen.

1. Funktionalität und Technische Umsetzung:

Die im Backend implementierten Funktionen und API-Endpunkte wurden eingehend getestet, insbesondere hinsichtlich ihrer Interaktion mit dem Frontend.

Die Datenverarbeitung und -speicherung, insbesondere die Handhabung der Serviceanfragen, wurden verifiziert und entsprechen den Anforderungen.

2. Systemleistung und Sicherheit:

Das Backend wurde auf seine Leistungsfähigkeit hin ausgewertet, mit einem besonderen Fokus auf die Antwortzeiten und die Zuverlässigkeit unter Last.

3. Effizienz der Entwicklung:

Die Entscheidungen für die verwendeten Technologien und Frameworks, wie .NET Core und Entity Framework, sowie die Projektstruktur, wurden hinsichtlich ihrer Auswirkungen auf die Entwicklungsgeschwindigkeit und -qualität bewertet.

Die Verwendung von Code-First Migrations trug zur Effizienz der Datenbankentwicklung bei.

4. Erreichte Geschäftsziele:

Die Backend-Funktionalität unterstützt nun effektiv das Auftragsmanagement und die Kundeninteraktionen, was zu einer verbesserten Betriebseffizienz führt.

Die Datenerfassung im Backend ermöglicht es, zukünftige geschäftliche Entscheidungen auf Grundlage von soliden Datenanalysen zu treffen.

5. Kunden- und Nutzerfeedback:

Rückmeldungen von Endnutzern und Stakeholdern wurden gesammelt und analysiert, um die Benutzerfreundlichkeit und Zufriedenheit mit den Backend-Funktionen zu bewerten.

Dieses Feedback wurde als Grundlage für zukünftige Verbesserungen und Funktionsupdates herangezogen.

12.1 Fazit

Das Ziel, ein robustes und effizientes Backend-System für die Verwaltung von Ski-Serviceaufträgen bei Jetstream-Service zu entwickeln, wurde erfolgreich im Rahmen einer anspruchsvollen Einzelarbeit realisiert. Diese Entwicklung markiert einen signifikanten Schritt in der Digitalisierungsinitiative des Unternehmens und optimiert die Handhabung und Verwaltung von Serviceaufträgen erheblich.

Durch die Implementierung einer modernen, datenbankbasierten Lösung unter Einsatz von .NET Core und Microsoft SQL Server konnte eine leistungsfähige, sichere und gut skalierbare Backend-Umgebung geschaffen werden. Die Entwicklung und Integration einer RESTful-API erleichterte die Interaktion mit dem bestehenden Frontend-System und sorgte für eine reibungslose Kommunikation zwischen den verschiedenen Systemkomponenten.

Die sorgfältige Integration von Sicherheitsfeatures, einschließlich Authentifizierungsprotokollen und Datenverschlüsselung, gewährleistet den Schutz sensibler Kundendaten und erfüllt die aktuellen Datenschutzstandards. Die Durchführung gründlicher Validierungstests und die Bewertung der Benutzerakzeptanz bestätigten die Effektivität und Zuverlässigkeit des Systems und trugen zur Sicherstellung einer hohen Servicequalität bei.

Bevor die Realisierung wie in der Planung beschrieben vorgegangen wurde, wurde zunächst im Frontend-Bereich gearbeitet, wobei eine quasi-fake API mit JavaScript entwickelt wurde. Diese anfängliche Fokussierung auf das Frontend brachte wertvolle Einblicke, obwohl später erkannt wurde, dass die Hauptaufgaben und Funktionen im Backend implementiert werden sollten. Diese Erfahrung führte zu einer strikteren Einhaltung der ursprünglichen Planung und einem besseren Verständnis der Bedeutung einer klaren Trennung von Frontend- und Backend-Aufgaben.

Die erfolgreiche Fertigstellung dieses Projekts als Einzelarbeit demonstriert nicht nur technische Expertise und Projektmanagement-Fähigkeiten, sondern auch ein tiefes Verständnis für die Anforderungen und Herausforderungen im Bereich des Ski-Service-Managements. Zudem war diese Arbeit äußerst bereichernd und hat viel Spaß gemacht. Der Lernprozess, der mit den Herausforderungen und der Lösungsfindung einherging, war immens und hat sowohl fachliche als auch persönliche Kompetenzen gestärkt.

Das Projekt legt eine solide Grundlage für zukünftige technologische Entwicklungen und erweitert die Kapazitäten von Jetstream-Service, effizient und effektiv auf die Bedürfnisse ihrer Kunden zu reagieren.

Insgesamt leistet dieses Backend-Projekt einen entscheidenden Beitrag zur Modernisierung und Verbesserung der Geschäftsprozesse bei Jetstream-Service und symbolisiert einen wichtigen Schritt in der digitalen Transformation des Unternehmens.

13 Anhänge

13.1 Glossar

BEGRIFF	BESCHREIBUNG / ERKLÄRUNG
API (APPLICATION PROGRAMMING INTERFACE)	Ein Satz von Routinen, Protokollen und Tools für die Erstellung von Software und Anwendungen, die eine Verbindung zwischen unterschiedlichen Softwareanwendungen ermöglichen.
AUTHENTIFIZIERUNG	Ein Prozess, der die Identität eines Benutzers verifiziert, typischerweise durch Benutzername und Passwort.
BACKEND	Der Teil der Anwendung, der auf dem Server läuft und für die Verarbeitung von Geschäftslogik, Datenbankmanagement und Server-Interaktionen zuständig ist.
CODEFIRST	Ein Ansatz bei Entity Framework, bei dem die Datenbankstruktur auf Grundlage von Code-Modellen generiert wird.
DTO (DATA TRANSFER OBJECT)	Ein Objekt, das dazu dient, Daten zwischen Subsystemen der Anwendung zu tragen, typischerweise zwischen dem Client und dem Server.
ENTITY FRAMEWORK	Ein ORM (Object-Relational Mapper), das die Datenbankzugriffsschicht einer Anwendung vereinfacht.
GANTT-DIAGRAMM	Ein Werkzeug für das Projektmanagement, das den Zeitplan von Projektaktivitäten visualisiert.
GITHUB	Eine Web-basierte Plattform für Versionskontrolle und kollaborative Softwareentwicklung.
IPERKA-MODELL	Ein Akronym für Informieren, Planen, Entscheiden, Realisieren, Kontrollieren, Auswerten - ein Prozessmodell im Projektmanagement.
JSON (JAVASCRIPT OBJECT NOTATION)	Ein leichtgewichtiges Daten-Austauschformat, das für Menschen leicht zu lesen und für Maschinen leicht zu parsen ist.
MIDDLEWARE	Software, die als Brücke zwischen einem Betriebssystem oder einer Datenbank und

	Anwendungen, insbesondere auf einem Netzwerk, dient.
MIGRATIONS	Im Kontext von Datenbanken bezieht sich dies auf den Prozess der Datenübertragung von einer Version eines Datenbankschemas zu einer anderen.
PSP (PROJEKTSTRUKTURPLAN)	Ein dokumentiertes Werkzeug, das die erforderlichen Arbeitsschritte eines Projekts in kleinere, leichter zu handhabende Teile aufgliedert.
RESTFUL API	Ein API, das die Prinzipien der Representational State Transfer (REST)-Architektur verwendet, um Interaktionen mit Webdiensten zu ermöglichen.
SWAGGERUI	Ein Werkzeug, das die Erstellung von Dokumentationen für RESTful APIs unterstützt und eine visuelle Plattform für das Testen von API-Endpunkten bietet.
VISUAL STUDIO / VISUAL STUDIO CODE	Integrierte Entwicklungsumgebungen (IDEs) von Microsoft zur Softwareentwicklung.

Tabelle 3: Glossar

13.2 NuGet Packages Screenshot

```
<Project Sdk="Microsoft.NET.Sdk.Web">

  <PropertyGroup>
    <TargetFramework>net7.0</TargetFramework>
    <Nullable>enable</Nullable>
    <ImplicitUsings>enable</ImplicitUsings>
  </PropertyGroup>

  <ItemGroup>
    <Compile Remove="Extensions\**" />
    <Content Remove="Extensions\**" />
    <EmbeddedResource Remove="Extensions\**" />
    <None Remove="Extensions\**" />
  </ItemGroup>

  <ItemGroup>
    <PackageReference Include="Microsoft.AspNetCore.Authentication.JwtBearer" Version="7.0.14" />
    <PackageReference Include="Microsoft.AspNetCore.OpenApi" Version="7.0.13" />
    <PackageReference Include="Microsoft.EntityFrameworkCore" Version="7.0.13" />
    <PackageReference Include="Microsoft.EntityFrameworkCore.Design" Version="7.0.13">
      <PrivateAssets>all</PrivateAssets>
      <IncludeAssets>runtime; build; native; contentfiles; analyzers; buildtransitive</IncludeAssets>
    </PackageReference>
    <PackageReference Include="Microsoft.EntityFrameworkCore.Proxies" Version="7.0.13" />
    <PackageReference Include="Microsoft.EntityFrameworkCore.SqlServer" Version="7.0.13" />
    <PackageReference Include="Microsoft.EntityFrameworkCore.Tools" Version="7.0.13">
      <PrivateAssets>all</PrivateAssets>
      <IncludeAssets>runtime; build; native; contentfiles; analyzers; buildtransitive</IncludeAssets>
    </PackageReference>
    <PackageReference Include="Microsoft.IdentityModel.Tokens" Version="7.0.3" />
    <PackageReference Include="Serilog.AspNetCore" Version="7.0.0" />
    <PackageReference Include="Serilog.Settings.Configuration" Version="7.0.1" />
    <PackageReference Include="Serilog.Sinks.File" Version="5.0.0" />
    <PackageReference Include="Swashbuckle.AspNetCore" Version="6.5.0" />
    <PackageReference Include="System.IdentityModel.Tokens.Jwt" Version="7.0.3" />
  </ItemGroup>

  <ItemGroup>
    <Folder Include="wwwroot\" />
  </ItemGroup>

</Project>
```

Abbildung 7: NuGet Packages Screenshot

13.3 Swagger Screenshots

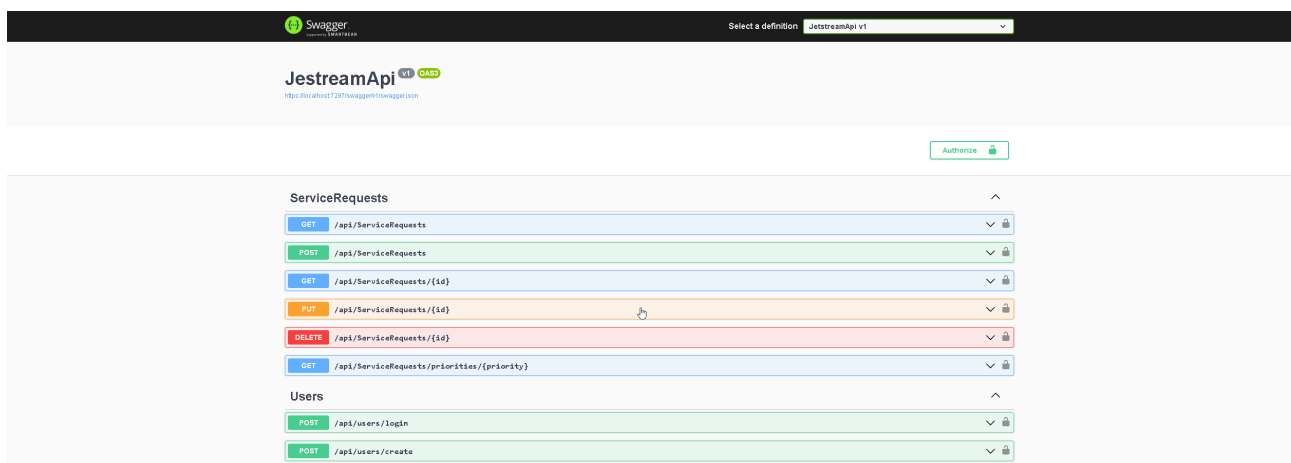


Abbildung 8: SwaggerUI - API Endpoints

21.11.2023



Abbildung 9: SwaggerUI - Schemas

13.4 Postman Collection Test Screenshot

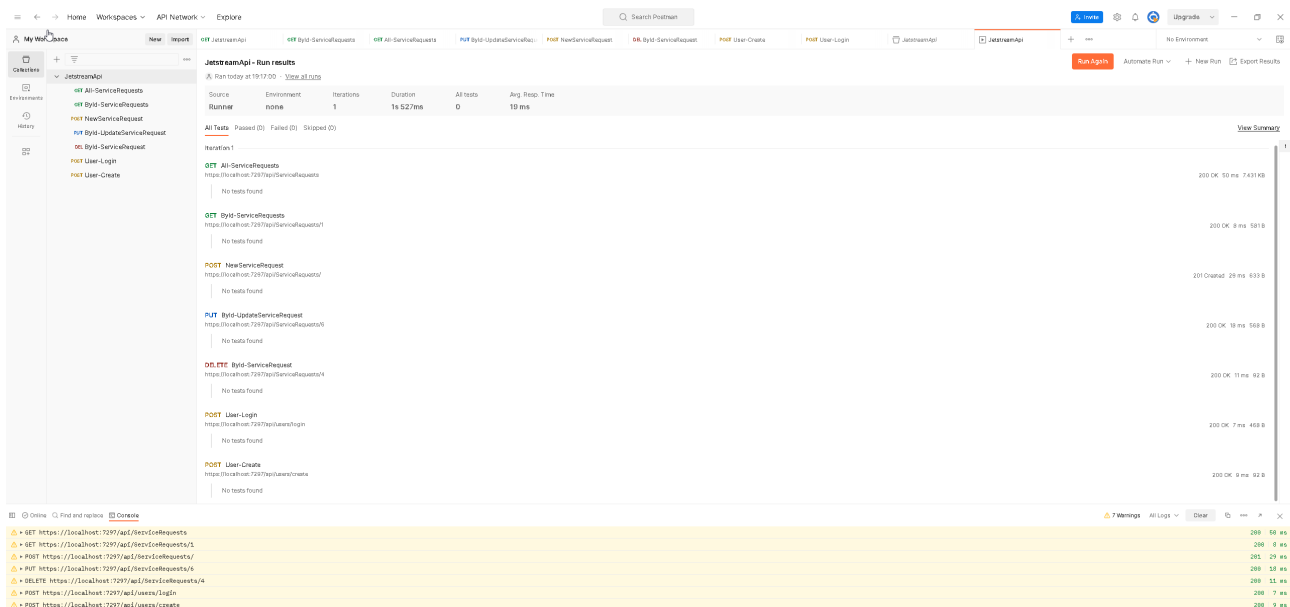


Abbildung 10: Postman Collection Test

13.5 xUnit Testergebnis Screenshot

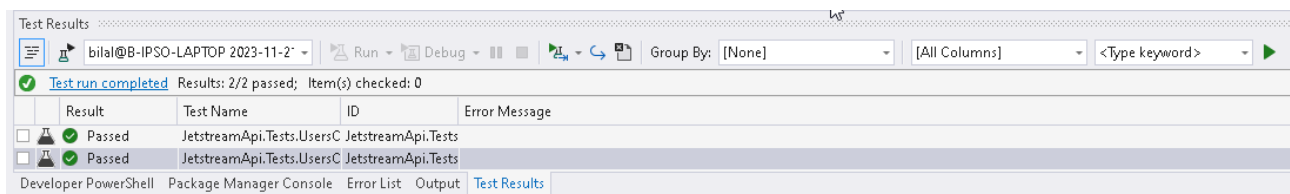


Abbildung 11: xUnit Testergebnis


13.6 Frontend Screenshots

The screenshot shows the 'Online Anfrage' form on the Jetstream Service website. The form is titled 'Online Anfrage' and is located in the center of the page. It contains several input fields and a submit button. The fields are: 'Vorname*' (First Name), 'Nachname*' (Last Name), 'E-Mail*' (Email), 'Telefon*' (Phone), 'Priorität' (Priority) with radio buttons for 'Tief' (Low), 'Standard' (Standard), and 'Express', 'Service Beginn Datum:' (Service Start Date), 'Abholdatum:' (Pickup Date), 'Ausgewählte Dienstleistung:' (Selected Service), and 'Gesamt Betrag' (Total Amount). The 'Priorität' section shows 'Tief' selected. The 'Service Beginn Datum:' and 'Abholdatum:' fields show dates '21/11/2023' and '03/12/2023' respectively. The 'Ausgewählte Dienstleistung:' field has a dropdown menu with the text 'Wählen Sie eine Option'. The 'Gesamt Betrag' field shows 'CHF 0.-'. Below the form, there is a note: 'Die mit einem Stern (*) markierten Felder sind Pflichtfelder.' (The fields marked with an asterisk (*) are mandatory.) and a 'Senden' (Send) button.

Abbildung 12: Service Request Formular

The screenshot shows the 'Mitarbeiter Login' form on the Jetstream Service website. The form is titled 'Mitarbeiter Login' and is located in the center of the page. It contains two input fields: 'Benutzername' (Username) and 'Passwort' (Password). Below the password field, there is a checkbox labeled 'Remember Me'. At the bottom of the form, there is a 'Login' button. The form is set against a light blue background with a white border.

Abbildung 13: Mitarbeiter Login




Jetstream

Service

























Service Aufträge


Mitarbeiterverwaltung

Logout



Service Aufträge

AUTRAGIDNR	NACHNAME	VORNAME	EMAIL	TELEFONNUMMER	PREIS	AUFRAGDATUM	ENDDATUM	SERVICE	PRIORITÄT	STATUS	KOMMENTAR		
2	string	string	user@example.com	1111111111	0.00	18.11.2023	18.11.2023	kleiner Service	tief	offen	string		
7	string2323	test 1	user@example.com	4915640600	0.00	19.11.2023	19.11.2023	kleiner Service	tief	offen	Postman Test ServiceRequest		
8	string2fsadfsaf323	tesdsadfsaft 1	user@evadsafsafmple.com	4915640600	0.00	19.11.2023	19.11.2023	kleiner Service	hoch	offen	Postman Test ServiceRequest		
9	string2fsadfsadfsaf323	tesdsadfsadfsadfsaft 1	usadsfaser@evadsafsafmple.com	4915640600	0.00	19.11.2023	19.11.2023	kleiner Service	hoch	offen	Postman Test ServiceRequest		
10	stradsdfafing2fsadfsadfsaf323	tesdsadfsadfsadfsadfsaft 1	usadsdfasfer@evadsafsafmple.com	4915640600	0.00	19.11.2023	19.11.2023	kleiner Service	hoch	offen	Postman Test ServiceRequest		
11	strasfing	stsadring	useradtl@esadfsample.com	86025.01729	0.00	19.11.2023	19.11.2023	RenntkIService	standard	offen	string		
12	string2fsadfsafing	stsadfsadfring	useradtl@esadfsample.com	86025.01729	0.00	19.11.2023	19.11.2023	RenntkIService	standard	offen	string		
13	Test456	Test123	user@example.com	+029.5311284291	0.00	19.11.2023	19.11.2023	RenntkIService	standard	offen	string		
6	Hampelmann2sdf	Test	user@example.com	+589146186641	0.00	19.11.2023	19.11.2023	kleiner Service	tief	offen	string		
1	Mustermann	Max	max.mustermann@example.com	1234567890	0.00	19.11.2023	20.11.2023	kleiner Service	tief	offen	Erster Kommentar		
5	Muster	Lukas	lukasmuster@example.com	3344556677	0.00	19.11.2023	24.11.2023	Fell zuschneiden	standard	offen	Fünfter Kommentar		
4	Beispiel	Anna	anna.beispiel@example.com	2233445566	0.00	19.11.2023	23.11.2023	Bindung montieren und einstellen	hoch	in Arbeit	Vierter Kommentar		



Jetstream

Über uns

Datenschutz

Kontakt

© 2023 Bobby Bial & Mohr Götzen. All Rights Reserved.

Abbildung 14: Service abrufen Frontend

13.7 Verweise

13.7.1 Quellenverzeichnis

Postman, I. (21. November 2023). *Postman API Platform*. Von Postman: <https://www.postman.com> abgerufen

Software, S. (21. November 2023). *Swagger UI*. Von Swagger.io: <https://swagger.io/tools/swagger-ui/> abgerufen

13.7.2 Tabellenverzeichnis

TABELLE 1: VERSIONSVERZEICHNIS	2
TABELLE 2: GROBE PLANUNG	5
TABELLE 3: GLOSSAR	15

13.7.3 Abbildungsverzeichnis

ABBILDUNG 1: ZUSAMMENFASSUNG ANFORDERUNGEN.....	3
ABBILDUNG 2: ZUSÄTZLICHE ANFORDERUNGEN.....	4
ABBILDUNG 3: PROJEKTSTRUKTURPLAN	5
ABBILDUNG 4: GANTT-DIAGRAMM	6
ABBILDUNG 5: SYSTEMARCHITEKTUR	7
ABBILDUNG 6: DATENBANKDIAGRAMM	7
ABBILDUNG 7: NUGET PACKAGES SCREENSHOT	16
ABBILDUNG 8: SWAGGERUI - API ENDPOINTS	16
ABBILDUNG 9: SWAGGERUI - SCHEMAS	17
ABBILDUNG 10: POSTMAN COLLECTION TEST.....	17
ABBILDUNG 11: xUNIT TESTERGEBNIS	17

21.11.2023

ABBILDUNG 12: SERVICE REQUEST FORMULAR	18
ABBILDUNG 13: MITARBEITER LOGIN.....	18
ABBILDUNG 14: SERVICE ABRUFEN FRONTEND.....	19