*Architektur-dokument*

**Bestell-Applikation “Zum Gelben Bach”**

*Hans Bloching (3009455)*

*Dennis Haaf (3009578)*

*Vincent Knapp (3009328)*

*Rene Weber (3009036)*

Table of Contents

[Änderungshistorie 2](#_Toc184932977)

[Einführung und Ziele 2](#_Toc184932978)

[Qualitätsziele 3](#_Toc184932979)

[Stakeholder 3](#_Toc184932980)

[Randbedingungen 3](#_Toc184932981)

[Kontextabgrenzung 4](#_Toc184932982)

[Business Context 4](#_Toc184932983)

[Technical Context 4](#_Toc184932984)

[Lösungsstrategien 5](#_Toc184932985)

[Bausteinsicht 6](#_Toc184932986)

[Architekturentscheidungen 7](#_Toc184932987)

[Technologie 7](#_Toc184932988)

[Datenbanksystem 7](#_Toc184932989)

[Nutritionix API 7](#_Toc184932990)

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Erstellt** | **Geprüft** | **Freigegeben** |
| **Name, Vorname** | Bloching, Hans  Haaf, Dennis  Knapp, Vincent  Weber, Rene | Vincent Knapp | Vincent Knapp |
| **Position / Rolle** | Autoren | Produkt owner | Produkt owner |
| **Datum** | 22.11.2024 | 13.12.2024 | 13.12.2024 |

## Änderungshistorie

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Datum** | **Geändert durch** | **Änderungsgrund** |
| 0.1 | 09.12.2024 | Vincent Knapp | Aufbau |
| 0.5 | 12.12.2024 | Vincent Knapp | Erweiterung |
| 1 | 13.12.2024 | Hans Bloching | Erweiterung |

## Einführung und Ziele

Das Projekt „Zum gelben Bach“ umfasst die Abwicklung von Bestellung innerhalb eines Restaurants, sowie die Darstellung der bestellten Speisen für das Küchenpersonal. Dabei sollen die Kunden aus der Liste aller möglichen Speisen/Getränke (S/G) wählen können. Nach der Auswahl der S/G sollen die Kunden diese in einem Warenkorb nochmal bestätigen/bearbeiten. Nach absenden einer Bestellung soll diese im System erfasst und an das Küchenpersonal weitergeleitet werden. Dieser Prozess erleichtert die Arbeit des Servicepersonals und kann ebenfalls für das Küchenpersonal eine effizientere Koordination der Speisezubereitung ermöglichen.

## Qualitätsziele

|  |  |
| --- | --- |
| **Ziel** | **Beschreibung** |
| Prozessintegration & Automatisierung | Die Software soll den Bestellprozess vollständig unterstützen, einschließlich der automatisierten Weiterleitung von Bestellungen an die Küche und der Rechnungserstellung, um den Workflow der Mitarbeiter effizient zu gestalten. |
| Datenintegrität & Sicherheit | Die Informationen der Kunden und deren Bestellungen sollen sicher verarbeitet werden. Der Datenaustausch zwischen Datenbank und Website soll minimalisiert werden. |
| Übersichtlichkeit | Der Kunde soll eine gute Übersicht über alle S/G haben. Auch die Küche soll schnell und übersichtlich alle bestellten S/G erfassen können. |

## Stakeholder

|  |  |
| --- | --- |
| **Wer?** | **Erwartungshaltung?** |
| Kunden | Schneller, einfacher Bestellvorgang. |
| Küchenpersonal | Übersichtliche Darstellung der bestellten S/G. |
| Servicepersonal | Weniger Arbeitsaufwand durch autom. Bestellprozess. |
| IT-Bereich | Simple Integration der Website auf vorhandene Hardware. |

## Randbedingungen

|  |  |
| --- | --- |
| **Randbedingung** | **Erklärung** |
| Hardware | Die Website & die Datenbank sollen auf einem System lauffähig sein. |
| Monetär | Die Entwicklung des Projekts darf keine finanziellen Aufwände verursachen. Auch soll die Software keine laufenden Kosten (abgesehen von Strom und Hardware (RAM/Verschleiß)) verursachen. |
| Zeit | Das Projekt soll bis zum festgelegten Termin (spätestens jedoch am 17.01.2025) abgeschlossen sein. |
| Methodisch | Das Projekt ist in einer Scrum-Methodik zu erstellen. |
| Dokumentation | Für den Zeitraum des Projekts sollen mehrere Sprint-Backlogs, sowie ein Product-Backlog erstellt und gepflegt werden. |

## Kontextabgrenzung

Das System „Zum Gelben Bach“ dient als Bestellapplikation für ein Restaurant und interagiert mit verschiedenen Kommunikationspartnern: Benutzern (Kunden und Mitarbeitern), einer zentralen Datenbank und einer externen API. Die Abgrenzung des Systems erfolgt durch die Definition der Schnittstellen zu diesen Kommunikationspartnern.

### Business Context

|  |  |  |
| --- | --- | --- |
| **Kommunikationspartner** | **Eingaben** | **Ausgaben** |
| Kunden (Benutzer der Website) | Produktauswahl, Anpassung im Warenkorb, Bestellabsendung | Bestätigung der Bestellung, Kostenübersicht |
| Küche | Eingehende Bestellungen | Übersicht kumulierter Bestellungen |
| nutrionx (Externe API) | Produktkatalog | Kalorien der Speisen |
| Datenbank | Anfrage nach gespeicherten Daten | Speichern von neuen Bestellungen, Kundeninformationen |

#### Business Context – Erklärung

* Der Kunde interagiert mit dem System über das Frontend (der Website), indem er S/G angezeigt bekommt, auswählt und bestellt.
* Die Küchenmitarbeiter nutzen das System als Informationsgrundlage zur Zubereitung der Speisen.
* Nutrionx wird zum Anreichern der Datengrundlage genutzt. Dabei werden die Speisen um die Anzahl der Kalorien erweitert.
* Die Datenbank stellt das zentrale Speichermedium aller Informationen dar.

### Technical Context

|  |  |  |
| --- | --- | --- |
| **Kanal** | **Eingabe** | **Ausgabe** |
| http/https | Anfragen der Nutzer (GET/POST) | HTML/CSS/JavaScript-Content |
| SqLite | Anfragen ausgelöst durch Nutzeranfragen auf Website | Tabellen |
| Nutrionx | Produkt (nach Neuanlage auf Datenbank) | JSON |

# Lösungsstrategien

|  |  |  |
| --- | --- | --- |
| **Qualitätsziel** | **Szenario** | **Lösungsansatz** |
| Benutzerfreundlichkeit | Die Nutzer sollen das Menü einfach bedienen können. | Eine tabellarische Darstellung der Gerichte, ein klar strukturierter Aufbau von Speisekarte und Warenkorb sowie minimalistische Gestaltung zur Vermeidung von Ablenkungen. |
| Skalierbarkeit | Möglichkeit die Anzahl der Kunden Problemlos zu erhöhen. | Ermöglichung des Austauschs der Datenbank durch eine leistungsstärkere Alternative. |
| Erweiterbarkeit | Möglichkeit zur Integration zukünftiger Funktionen wie Bewertungen, Analytics. | Ein modularer, API-basierter Ansatz für Auswertungen und Analysen. |

## Ein Bild, das Diagramm, Plan, technische Zeichnung, Reihe enthält.Bausteinsicht

## Architekturentscheidungen

### Technologie

**Entscheidung:** C#, Blazor (beides Frontend u. Backend)

**Begründung:** Geringere Komplexität durch Frontend + Backend Entwicklung in C#. Durch die modernen Features von C# und Blazor können Tasks simultan ausgeführt werden, was Ladezeiten verringern kann. Blazor selbst ermöglich die Entwicklung von Frontends ohne auf JavaScripte angewiesen zu sein.

**Alternative:** JavaScript, HTML, CSS. Wir haben uns gegen diese Alternative entschieden, da uns der Einarbeitungsaufwand höher erschien (gerade im Bezug auf JavaScript).

**Konsequenz:** schnelle Entwicklung durch Blazor mit einfacher Wartbarkeit. Lernphase des Teams kann daher kompensiert werden.

### Datenbanksystem

**Entscheidung:** SqLite als Datenbank

**Begründung:** Die SqLite Datenbank biete eine hohe Performance mit simplerer Einrichtung. Auszugleichende Schwächen können durch C# und Blazor kompensiert werden. Es entstehen durch die Nutzung der SqLite keine Kosten.

**Alternative:** … wäre eine MySql/MariaDB oder ein Microsoft Sql Server, die eine höhere Skalierbarkeit bieten, jedoch für dieses Projekt wahrscheinlich zu leistungsstark sind.

**Konsequenz:** Die Datenbank bietet alle benötigten Aspekte für dieses Projekt. Sollte das Projekt jedoch in größerem Maße ausgerollt werden, sollte man eine Leistungsfähigere Datenbank anbinden.

### Nutritionix API

**Entscheidung:** Nutzung der Nutritionix API zum erfassen von Kalorien der Speisen.

**Begründung:** Vorgabe vom Kunden.

**Alternative:** DieNutritics API wäre ein valider Ersatz für die Nutritionix API. Da der Kunde jedoch auf die Anbindung der Nutritionix API bestand wurde davon abgesehen.

**Konsequenz:** Die API erfordert während der Datenabfrage eine stabile Internetverbindung. Dies erweitert unser Projekt um ein weiteres Risiko das es zu beachten gilt. Durch die API Anbindung binden wir uns auch an die NutritionixAPI was in Zukunft dazu führen kann, dass wir uns an die Vorgaben & Änderungen der API anpassen müssen.