Projekt „Terminverwaltung“

**Inhaltsverzeichnis**

1. Beschreibung des Ist- und Sollzustandes (?) ......................................................... 4

1.1. Istzustand.............................................................................................................................. 4

1.2. Veranlassung für das Projekt ............................................................................................. 4

1.3. Sollzustand (Produktbeschreibung) .................................................................................. 4

1.4. Zielbestimmung .................................................................................................................... 4

1.5. Produkteinsatz...................................................................................................................... 4

2. Funktionale Anforderungen (?) ............................................................................... 5

2.1. Featureliste mit User Stories .................................................................................................... 5

2.2. Use-Case Beschreibung zum Gesamtsystem ......................................................................... 6

2.3. Produktfunktionen ..................................................................................................................... 7

3. Nebenbedingungen (?) ............................................................................................19

4. Produktdaten (?) ......................................................................................................19

5. Skizze der Systemarchitektur (?) ...........................................................................20

6. Schnittstellen (?) ......................................................................................................20

7. Lieferumfang (?) .......................................................................................................20

8. Abnahmekriterien (?) ...............................................................................................21

9. Glossar (?)..................................................................................................................22

Inhalt

[2 Einführung/Kernfunktion 1](#_Toc98848667)

[3 Funktionen 1](#_Toc73233546)

[3.1 Übersicht über die Produktfeatures 1](#_Toc2065302221)

[3.2 Use-Case Diagramm 1](#_Toc733269293)

[3.3 Beschreibung der Use Cases 1](#_Toc109793212)

[4 Technische Details 1](#_Toc1882451039)

[4.1 Technologiestack: 2](#_Toc869727549)

[4.2 Architektur: 2](#_Toc2003152763)

[4.3 Datenschutz und Sicherheit: 2](#_Toc223599108)

[4.4 Entwicklungsprozess 2](#_Toc1545930671)

[4.5 Fazit 2](#_Toc457917555)

[5 User Stories für Administratoren, Kunden und Mitarbeiter für die Terminverwaltungswebanwendung: 3](#_Toc296911219)

[6 Use Cases: 4](#_Toc1596793002)

[6.1 Use Case 1.1: 4](#_Toc2140774118)

[6.2 Use Case 1.5 5](#_Toc1586714761)

[6.3 Use Case 1.6 5](#_Toc183429130)

[6.4 Use Case 1.7 6](#_Toc1931961044)

[6.5 Use Case 1.8 7](#_Toc63437226)

[6.6 Use Case 1.9 8](#_Toc1168520995)

[-Der Salon-Manager meldet sich in der Beauty-Salon-Webapplikation an. 11](#_Toc413749373)

[7 Masken und Controls 16](#_Toc929022155)

[7.1 Anmelde-Maske 16](#_Toc2144040223)

[7.2 Administrations-Dashboard 16](#_Toc720848511)

[7.3 Kunden-Dashboard 16](#_Toc835500135)

[7.4 Mitarbeiter-Dashboard 17](#_Toc23194315)

[8 Detailbeschreibungen für die Entwicklung: 17](#_Toc1717563195)

[8.1 Anmelde-Maske: 17](#_Toc1756445118)

[8.2 Administrations-Dashboard: 17](#_Toc2135013254)

[8.3 Kunden-Dashboard: 17](#_Toc2145652926)

[8.4 Mitarbeiter-Dashboard: 18](#_Toc610435447)

[8.5 2. Repository-Klassen: 18](#_Toc708126156)

[8.6 3. Controller-Klassen: 18](#_Toc696148077)

[8.7 4. Service-Klassen: 18](#_Toc367134575)

[8.8 5. Sicherheits-Klassen: 19](#_Toc1124251637)

[9 Hier sind Epics (große, übergreifende Aufgaben) für die zuvor erstellten User Stories für die Terminverwaltungswebanwendung: 19](#_Toc85328518)

[9.1 Administratoren: 19](#_Toc2101297686)

[9.2 Kunden: 19](#_Toc905401685)

[9.3 Mitarbeiter: 20](#_Toc1688007367)

[10 Entwicklung der Terminverwaltungswebanwendung 20](#_Toc1872844568)

[11 Terminplan 22](#_Toc542144627)

# Einführung/Kernfunktion

Die Webanwendung wird für das Kleinunternehmen „Schwester Liebe“ entwickelt. Die Webapplikation ermöglicht es den Kunden online einen Termin zu buchen. Des Weiteren kann das Unternehmen Termine für Mitarbeiter bzw. Kunden verwalten. Es gibt die Möglichkeit Rabatte auf Termine in gewissen Zeiträumen (zB: Weihnachten oder in Ferien) zu gewähren. Die Mitarbeiter und die Leiterin können sich außerdem einloggen, um einige Änderungen vorzunehmen. Die Anwendung konzentriert sich hauptsächlich auf Terminverwaltung.

# Funktionen

## Übersicht über die Produktfeatures

Die Webapplikation wird die folgenden Hauptfunktionen bieten:

* Ein Loginsystem:
  + Erlaubt den Mitarbeitern bzw. der Leiterin sich in der Anwendung anzumelden, um Änderungen zu machen.
* Terminverwaltung:
  + Ermöglicht den ~~Mitarbeitern bzw.~~ der Leiterin Termine zu vergeben bzw. zu bearbeiten
  + Kunden können online Termine vereinbaren und eine Mitarbeiterin für die Dienstleistung wählen.
  + Mitarbeitern werden Termine zugeteilt.
  + Kunden können Buchungen stornieren
  + Admin bzw. Leiterin kann Termine bei Terminabsage löschen.
* Rabattverwaltung:
  + Admin kann saisonale Angebote wie Weihnachtsangebote, Herbstangebote, Black Friday-Specials, Silvesterangebote und Sommerangebote machen.
* Benachrichtigungen:
  + Bei Terminabsage von Terminen erhält der Admin eine Benachrichtigung.
  + Beim Löschen von Terminen aufgrund von Stornierung bekommen Kunden per E-Mail eine Benachrichtigung

## Use-Case Diagramm

## Beschreibung der Use Cases

# Technische Details

## Technologiestack:

Die Webapplikation wird unter Verwendung von folgenden Technologien entwickelt:

- Programmiersprachen: HTML, CSS, C#,

- Front-End-Framework: ASP.Net

- Back-End-Technologie: C# (Visualstudio) ADO.Net

- Datenbank: MongoDB oder MySQL

- Server: AWS, Azure oder Google Cloud

- Sicherheitsprotokolle: HTTPS für Datenverschlüsselung

- Versionierung: Git

## Architektur:

Die Anwendung wird in einer clientseitigen (Single-Page-Application) oder serverseitigen (traditionellen MVC) Architektur entwickelt, abhängig von den Anforderungen und Präferenzen des Unternehmens. Die Kommunikation zwischen Front-End und Back-End erfolgt über APIs.

## Datenschutz und Sicherheit:

Die Anwendung wird strenge Datenschutzrichtlinien und Sicherheitsmaßnahmen implementieren, um die persönlichen Daten der Kunden und Mitarbeiter zu schützen. Dies umfasst die Verwendung von sicheren Authentifizierungsverfahren und Verschlüsselung der Datenübertrag.

## Entwicklungsprozess

Die Entwicklung erfolgt in mehreren Phasen:

- Anforderungsanalyse: Festlegung der genauen Anforderungen in enger Zusammenarbeit mit dem Kunden.

- Front-End- und Back-End-Entwicklung: Implementierung der Funktionalitäten und Integration von Datenbanken.

- Testing und Qualitätssicherung: Umfangreiche Tests, um Fehler zu identifizieren und zu beheben.

- Bereitstellung und Wartung: Hochladen der Anwendung auf den Hosting-Server und laufende Wartung und Aktualisierung.

## Fazit

Die vorgeschlagene Webanwendung zur Terminverwaltung ermöglicht es dem Dienstleistungs-KMU, saisonale Angebote wie Weihnachtsangebote, Herbstangebote, Black Friday-Specials, Silvesterangebote und Sommerangebote effizient zu organisieren und anzubieten. Durch die Nutzung moderner Technologien und bewährter Sicherheitsmaßnahmen wird die Anwendung den Anforderungen des Unternehmens gerecht und ermöglicht eine reibungslose Verwaltung von Terminen und Kundenbeziehungen.

**---bis hier Tugba hat gemacht!!!**

# User Stories für Administratoren, Kunden und Mitarbeiter für die Terminverwaltungswebanwendung:

**Administratoren**:

1. Als Administrator möchte ich mich in das System einloggen können, um Zugriff auf alle administrativen Funktionen zu haben.

2. Als Administrator möchte ich neue Mitarbeiter hinzufügen können, indem ich ihre persönlichen Informationen und Arbeitszeiten festlege.

3. Als Administrator möchte ich Mitarbeiter bearbeiten können, um ihre Daten oder Arbeitszeiten zu aktualisieren.

4. Als Administrator möchte ich Mitarbeiter deaktivieren oder löschen können, wenn sie das Unternehmen verlassen.

5. Als Administrator möchte ich Rabattzeiten definieren können, indem ich festlege, welche Zeiten als Stoßzeiten gelten und Rabatte gewährt werden.

6. Als Administrator möchte ich die Übersicht über alle Kunden und ihre Buchungen haben.

7. Als Administrator möchte ich Berichte über die Auslastung der Mitarbeiter und den Umsatz generieren können.

8. Als Administrator möchte ich E-Mail-Benachrichtigungen für wichtige Aktivitäten im System erhalten, wie z.B. neue Buchungen oder Terminänderungen.

**Kunden**:

1. Als Kunde möchte ich mich in das System einloggen können, um meine Buchungen anzuzeigen und neue Termine zu buchen.

2. Als Kunde möchte ich die Verfügbarkeit der Mitarbeiter sehen und Termine in freien Zeiten auswählen können.

3. Als Kunde möchte ich eine Bestätigungs-E-Mail erhalten, sobald ich einen Termin gebucht habe.

4. Als Kunde möchte ich Erinnerungsbenachrichtigungen per E-Mail oder SMS für meine bevorstehenden Termine erhalten.

5. Als Kunde möchte ich Termine bearbeiten oder stornieren können, wenn sich meine Pläne ändern.

6. Als Kunde möchte ich Informationen über Rabatte für Termine außerhalb der Stoßzeiten sehen und sie bei der Buchung nutzen können.

**Mitarbeiter**:

1. Als Mitarbeiter möchte ich mich in das System einloggen können, um meine Verfügbarkeit und Buchungen zu verwalten.

2. Als Mitarbeiter möchte ich meinen Arbeitszeitplan einsehen und aktualisieren können.

3. Als Mitarbeiter möchte ich neue Buchungen akzeptieren oder ablehnen können, um meine Verfügbarkeit zu verwalten.

4. Als Mitarbeiter möchte ich eine Liste meiner bevorstehenden Termine sehen und Benachrichtigungen für neue Buchungen erhalten.

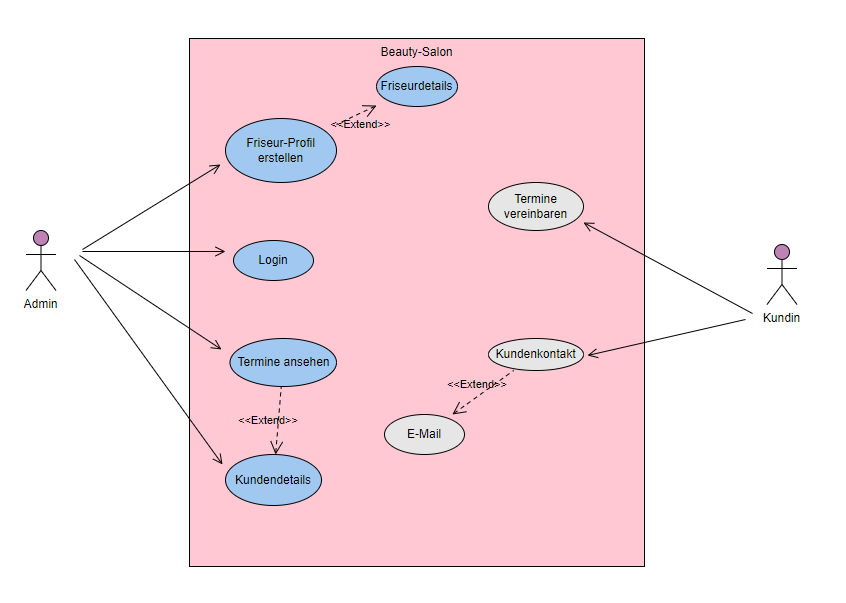
5. Als Mitarbeiter möchte ich die Möglichkeit haben, Notizen zu meinen Terminen hinzuzufügen, um wichtige Informationen zu speichern.

6. Als Mitarbeiter möchte ich eine Übersicht über die von mir erzielten Einnahmen und Rabatte haben.

Diese User Stories dienen als Ausgangspunkt für die Entwicklung und werden im Laufe des Projekts möglicherweise weiter verfeinert und ergänzt. Sie helfen dabei, die Anforderungen und Erwartungen der verschiedenen Benutzergruppen klar zu definieren.

**---bis hier ist fertig!**

**Use-Case Beschreibung zum Gesamtsystem**



**Produktfunktionen**

Use Case 1 **Login System**

|  |  |
| --- | --- |
| **Use Case 1 Einloggen in das System** | |
| Name | Einloggen in das System |
| Kurzbeschreibung | Mitarbeiter und Leiterin melden sich an, um Änderungen vorzunehmen. |
| Akteure | Administrator, Mitarbeiter |
| Auslöser | Benutzer möchte auf das System zugreifen. |
| Ergebnis(se) | Erfolgreicher Login; Zugriff auf entsprechende Funktionen. |
| Vorbedingungen | Benutzerkonto vorhanden; System ist erreichbar. |
| Eingehende Daten | Benutzerkonto vorhanden; System ist erreichbar. |
| Ablaufbeschreibung | 1. Benutzer öffnet die Login-Seite.  2. Benutzer gibt Benutzername und Passwort ein.  3. System überprüft die Anmeldeinformationen.  4. Bei erfolgreicher Überprüfung wird der Benutzer zum entsprechenden Dashboard weitergeleitet. |
| Fehlerverhalten | - Ungültiges Datum oder Uhrzeit: Fehlermeldung und Möglichkeit zur erneuten Eingabe.  - Überschneidende Termine: Warnung und Möglichkeit zur Änderung. |
| Variationen | - Zwei-Faktor-Authentifizierung: Zusätzliche Sicherheitsschicht.  - Passwort zurücksetzen: Möglichkeit, das Passwort zurückzusetzen. |

### Use Case 2.**Terminverwaltung:**

|  |  |
| --- | --- |
| **Use Case 2: Buchung und Verwaltung von Terminen** | |
| Name | Buchung und Verwaltung von Terminen |
| Kurzbeschreibung | Mitarbeiter und Leiterin vergeben und bearbeiten Termine; Kunden buchen und verwalten Termine online. |
| Akteure | Administrator, Mitarbeiter, Kunden |
| Auslöser | Benutzer möchte einen Termin vergeben, bearbeiten oder buchen. |
| Ergebnis(se) | Erfolgreiche Terminvergabe, -bearbeitung oder -buchung. |
| Vorbedingungen | Benutzer ist eingeloggt  Verfügbarkeit ist festgelegt. |
| Eingehende Daten | -Datum, Uhrzeit, Kundeninformationen (für Mitarbeiter und Leiterin)  -Kundeninformationen, gewünschter Termin (für Kunden) |
| Ablaufbeschreibung | 1. Mitarbeiter/Leiterin öffnen das Terminverwaltungsmodul. 2. Mitarbeiter/Leiterin vergibt oder bearbeitet Termine.  3. Kunden öffnen das Online-Buchungsmodul.  4. Kunden wählen einen verfügbaren Termin und buchen ihn. |
| Fehlerverhalten | - Ungültiges Datum oder Uhrzeit: Fehlermeldung und Möglichkeit zur erneuten Eingabe.  - Überschneidende Termine: Warnung und Möglichkeit zur Änderung. |
| Variationen | - Neue Buchungen erstellen: Standardverfahren oder Schnellbuchungsoption.  - Terminbearbeitung: Reguläre Bearbeitung oder Massenbearbeitung.  - Terminbestätigung: Automatische oder manuelle Bestätigung.  - Stornierungsoptionen: Kostenlose Stornierung oder Stornierungsgebühr.  - Integrationen: Kalenderintegration oder Zahlungsintegration. |

### Use Case 3 **Rabattverwaltung:**

|  |  |
| --- | --- |
| **Use Case 3: Rabattverwaltung durch Administrator** | |
| Name | Rabattverwaltung durch Administrator |
| Kurzbeschreibung | Administrator erstellt saisonale Angebote und Rabatte. |
| Akteure | Administrator |
| Auslöser | Administrator möchte Rabatte definieren. |
| Ergebnis(se) | Erfolgreiche Erstellung von Rabatten |
| Vorbedingungen | Administrator ist eingeloggt. |
| Eingehende Daten | Rabatttyp, Zeiträume, Details des Rabattangebots. |
| Ablaufbeschreibung | 1. Administrator öffnet das Rabattverwaltungsmodul.  2. Administrator erstellt ein neues Rabattangebot und legt Zeiträume fest. |
| Fehlerverhalten | - Ungültige Zeiträume oder Details: Fehlermeldung und Möglichkeit zur Überarbeitung. |
| Variationen | Rabattarten: Prozentsatz, Festbetrag.  - Zeitraum: Weihnachten, Herbst, Black Friday, Silvester, Sommer. |

### Use Case 4. Benachrichtigungen:

|  |  |
| --- | --- |
| **Use Case 4** Benachrichtigungen bei Terminänderungen | |
| Name | Benachrichtigungen bei Terminänderungen |
| Kurzbeschreibung | Administrator und Kunden erhalten Benachrichtigungen über Terminänderungen. |
| Akteure | Administrator, Kunden |
| Auslöser | Terminänderung oder Stornierung durch Administrator oder Kunde. |
| Ergebnis(se) | Erfolgreicher Versand von Benachrichtigungen. |
| Vorbedingungen | Benutzer ist eingeloggt, System ist erreichbar. |
| Eingehende Daten | Terminänderung oder Stornierung |
| Ablaufbeschreibung | 1. Administrator oder Kunde ändert oder storniert einen Termin.  2. System sendet automatisch Benachrichtigungen an betroffene Parteien. |
| Fehlerverhalten | - Fehler beim Versand der Benachrichtigung: Fehlermeldung und erneuter Versuch. |
| Variationen | Keine vorhanden |

### Use 6. Einloggen als Mitarbeiter:

|  |  |
| --- | --- |
| **Use Case 6.** Einloggen als Mitarbeiter | |
| Name | Einloggen als Mitarbeiter |
| Kurzbeschreibung | Mitarbeiter loggt sich in das System ein, um auf seine Verfügbarkeit und Buchungen zuzugreifen. |
| Akteure | Mitarbeiter |
| Auslöser | Mitarbeiter möchte auf das System zugreifen. |
| Ergebnis(se) | -Erfolgreicher Login  -Zugriff auf Verfügbarkeit und Buchungen. |
| Vorbedingungen | Mitarbeiterkonto vorhanden; System ist erreichbar. |
| Eingehende Daten | Benutzername, Passwort. |
| Ablaufbeschreibung | 1. Mitarbeiter öffnet die Login-Seite.  2. Mitarbeiter gibt Benutzername und Passwort ein.  3. System überprüft die Anmeldeinformationen.  4. Bei erfolgreicher Überprüfung wird der Mitarbeiter zum Dashboard weitergeleitet. |
| Fehlerverhalten | - Falsche Anmeldeinformationen: Fehlermeldung und Möglichkeit zur erneuten Eingabe.  - Konto deaktiviert: Benachrichtigung über deaktiviertes Konto.  - Systemfehler: Generelle Fehlermeldung mit Kontakthinweis. |
| Variationen |  |

**Arbeitszeitplan einsehen/aktualisieren:**

|  |  |
| --- | --- |
| **Use Case 1.10** Arbeitszeitplan einsehen/aktualisieren | |
| Name | Arbeitszeitplan einsehen/aktualisieren |
| Kurzbeschreibung | Mitarbeiter sieht seinen Arbeitszeitplan ein und aktualisiert ihn bei Bedarf. |
| Akteure | Mitarbeiter |
| Auslöser | Mitarbeiter möchte seinen Arbeitszeitplan einsehen oder aktualisieren. |
| Ergebnis(se) | Aktualisierter Arbeitszeitplan. |
| Vorbedingungen | Mitarbeiter ist eingeloggt; Neue Buchungsanfrage vorhanden. |
| Eingehende Daten | Neue Arbeitszeitinformationen (falls erforderlich). |
| Ablaufbeschreibung | 1. Mitarbeiter öffnet das Modul für Arbeitszeitverwaltung.  2. Mitarbeiter sieht seinen aktuellen Arbeitszeitplan ein.  3. Bei Bedarf aktualisiert der Mitarbeiter den Arbeitszeitplan. |
| Fehlerverhalten |  |
| Variationen |  |

### Neue Buchungen akzeptieren/ablehnen:

|  |  |
| --- | --- |
| **Use Case 1.11** Neue Buchungen akzeptieren/ablehnen | |
| Name | Neue Buchungen akzeptieren/ablehnen |
| Kurzbeschreibung | Mitarbeiter akzeptiert oder lehnt neue Buchungen ab, um die Verfügbarkeit zu verwalten. |
| Akteure | Mitarbeiter |
| Auslöser | Neue Buchungsanfrage eingegangen |
| Ergebnis(se) | Akzeptierte oder abgelehnte Buchung. |
| Vorbedingungen | Mitarbeiter ist eingeloggt; Neue Buchungsanfrage vorhanden. |
| Eingehende Daten | Details der Buchungsanfrage. |
| Ablaufbeschreibung | 1. Mitarbeiter erhält eine Benachrichtigung über eine neue Buchungsanfrage.  2. Mitarbeiter öffnet das Buchungsverwaltungsmodul.  3. Mitarbeiter akzeptiert oder lehnt die Buchung ab. |
| **Abgrenzungen:** |  |
| Variationen: |  |

Bevorstehende Termine einsehen:

|  |  |
| --- | --- |
| **Use Case 1.12** Bevorstehende Termine einsehen | |
| Name | Bevorstehende Termine einsehen |
| Kurzbeschreibung | Mitarbeiter sieht eine Liste seiner bevorstehenden Termine ein. |
| Akteure | Mitarbeiter |
| Auslöser | Mitarbeiter möchte bevorstehende Termine überprüfen. |
| Ergebnis(se) | Liste der bevorstehenden Termine. |
| Vorbedingungen | Mitarbeiter ist eingeloggt. |
| Eingehende Daten | - |
| Ablaufbeschreibung | 1. Mitarbeiter öffnet das Modul für bevorstehende Termine. 2. Mitarbeiter sieht eine Liste seiner bevorstehenden Termine. |
| **Abgrenzungen:** |  |
| Variationen: |  |

### Notizen zu Terminen hinzufügen:

|  |  |
| --- | --- |
| **Use Case 1.13** Notizen zu Terminen hinzufügen | |
| Name | Notizen zu Terminen hinzufügen |
| Kurzbeschreibung | Mitarbeiter fügt Notizen zu seinen Terminen hinzu. |
| Akteure | Mitarbeiter |
| Auslöser | Mitarbeiter möchte wichtige Informationen zu einem Termin speichern. |
| Ergebnis(se) | Notizen zu einem Termin hinzugefügt. |
| Vorbedingungen | Mitarbeiter ist eingeloggt; Termin ist vorhanden. |
| Eingehende Daten | Textuelle Notizen. |
| Ablaufbeschreibung | 1. Mitarbeiter öffnet das Modul für Termindetails.  2. Mitarbeiter fügt Notizen zu dem gewünschten Termin hinzu. |
| **Abgrenzungen:** |  |
| Variationen |  |

# Masken und Controls

## Anmelde-Maske

* Benutzername-Feld
* Passwort-Feld
* Anmelde-Button
* Passwort vergessen-Link

## Administrations-Dashboard

* Übersicht über Mitarbeiter
* Verwaltung von Mitarbeitern (Hinzufügen, Bearbeiten, Löschen)
* Übersicht über Kunden und Buchungen
* Rabattverwaltung

## Kunden-Dashboard

* Terminbuchungskalender
* Terminauswahl
* Buchungsformular
* Terminverwaltung (Bearbeiten, Stornieren)
* Rabattinformationen

## Mitarbeiter-Dashboard

* Verfügbarkeitskalender
* Buchungsanfragen
* Annahme/Ablehnung von Buchungen
* Terminverwaltung
* Einnahmeübersicht

# Detailbeschreibungen für die Entwicklung:

## Anmelde-Maske:

- Benutzername-Feld: Ein einfaches Textfeld, in das der Benutzer seinen Benutzernamen eingibt.

- Passwort-Feld: Ein Passwortfeld, in das der Benutzer sein Passwort eingibt (Text wird maskiert).

- Anmelde-Button: Ein Button, um den Anmeldevorgang auszulösen.

- Passwort vergessen-Link: Ein Link, über den Benutzer ein Passwortwiederherstellungsverfahren starten können.

## Administrations-Dashboard:

- Übersicht über Mitarbeiter: Eine Tabelle oder Liste, die die Mitarbeiter und ihre grundlegenden Informationen anzeigt.

- Verwaltung von Mitarbeitern: Eine Funktion zur Hinzufügung, Bearbeitung und Löschung von Mitarbeitern, die Formulare für Mitarbeiterdaten verwendet.

- Übersicht über Kunden und Buchungen: Eine Übersicht über Kunden und deren gebuchte Termine.

- Rabattverwaltung: Eine Funktion zur Festlegung von Rabattzeiten und -beträgen.

## Kunden-Dashboard:

- Terminbuchungskalender: Ein Kalender, der die Verfügbarkeit der Mitarbeiter anzeigt.

- Terminauswahl: Ein Steuerelement, mit dem Kunden verfügbare Termine auswählen können.

- Buchungsformular: Ein Formular, in dem Kunden ihre Buchungsdetails eingeben können.

- Terminverwaltung: Eine Übersicht über gebuchte Termine, mit der Möglichkeit, Termine zu bearbeiten oder zu stornieren.

- Rabattinformationen: Die Anzeige von Rabatten für Termine in Stoßzeiten.

## Mitarbeiter-Dashboard:

- Verfügbarkeitskalender: Ein Kalender, der die Verfügbarkeit des Mitarbeiters anzeigt.

- Buchungsanfragen: Eine Liste der Buchungsanfragen, die der Mitarbeiter erhalten hat.

- Annahme/Ablehnung von Buchungen: Optionen, um Buchungsanfragen anzunehmen oder abzulehnen.

- Terminverwaltung: Eine Übersicht über gebuchte Termine und die Möglichkeit zur Bearbeitung.

- Einnahmeübersicht: Informationen über die Einnahmen des Mitarbeiters.

Die Detailbeschreibungen dienen als Grundlage für die Entwicklung der einzelnen Masken und Steuerelemente. Jedes Element muss entsprechend den Anforderungen und Benutzererwartungen entwickelt werden, um eine reibungslose Funktionsweise der Terminverwaltungswebanwendung sicherzustellen.

## 2. Repository-Klassen:

- `EmployeeRepository.java`: Repository-Klasse für die Datenbankzugriffe auf Mitarbeiterdaten.

- `CustomerRepository.java`: Repository-Klasse für die Datenbankzugriffe auf Kundendaten.

- `AppointmentRepository.java`: Repository-Klasse für die Datenbankzugriffe auf Termindaten.

- `DiscountRepository.java`: Repository-Klasse für die Datenbankzugriffe auf Rabattdaten.

## 3. Controller-Klassen:

- `AdminController.java`: Controller-Klasse für Administrator-Funktionen.

- `CustomerController.java`: Controller-Klasse für Kunden-Funktionen.

- `EmployeeController.java`: Controller-Klasse für Mitarbeiter-Funktionen.

## 4. Service-Klassen:

- `EmployeeService.java`: Service-Klasse für die Geschäftslogik im Zusammenhang mit Mitarbeitern.

- `CustomerService.java`: Service-Klasse für die Geschäftslogik im Zusammenhang mit Kunden.

- `AppointmentService.java`: Service-Klasse für die Geschäftslogik im Zusammenhang mit Terminen.

- `DiscountService.java`: Service-Klasse für die Geschäftslogik im Zusammenhang mit Rabatten.

## 5. Sicherheits-Klassen:

- `SecurityConfig.java`: Konfigurationsklasse für die Sicherheit der Anwendung.

- `CustomUserDetailsService.java`: Implementiert den benutzerdefinierten Benutzerdetaildienst für die Authentifizierung.

Diese Struktur dient als Ausgangspunkt und kann je nach den spezifischen Anforderungen der Anwendung erweitert und angepasst werden. Bitte beachten Sie, dass dies nur eine Beispielstruktur ist und die tatsächliche Implementierung von den Anforderungen und Präferenzen Ihres Projekts abhängt.

# Hier sind Epics (große, übergreifende Aufgaben) für die zuvor erstellten User Stories für die Terminverwaltungswebanwendung:

## Administratoren:

**1. Benutzer- und Mitarbeiterverwaltung:**

- Epic: Verwalten von Benutzern und Mitarbeitern

- Beschreibung: Dieses Epic konzentriert sich auf die Erstellung, Bearbeitung und Verwaltung von Benutzerkonten (Administratoren, Kunden und Mitarbeiter) sowie auf die Verwaltung der Mitarbeiterdaten.

**2. Rabattverwaltung:**

- Epic: Rabattverwaltung für Stoßzeiten

- Beschreibung: Dieses Epic konzentriert sich auf die Einrichtung und Verwaltung von Rabatten für Stoßzeiten sowie auf die Anzeige von Rabatten für Kunden während der Buchung.

## Kunden:

1**. Buchung und Verwaltung von Terminen:**

- Epic: Buchung und Verwaltung von Terminen

- Beschreibung: Dieses Epic ermöglicht Kunden die Buchung von Terminen, die Anzeige von Buchungen und die Verwaltung von Terminen, einschließlich Bearbeitung und Stornierung.

## Mitarbeiter:

1. Verfügbarkeitsverwaltung und Buchungsannahme:

- Epic: Verwaltung der Verfügbarkeit und Annahme von Buchungen

- Beschreibung: Dieses Epic konzentriert sich auf die Verwaltung der Verfügbarkeit des Mitarbeiters, die Annahme oder Ablehnung von Buchungen und die Verwaltung der eigenen Termine.

Diese Epics helfen dabei, die verschiedenen Aufgaben und Funktionen der Anwendung zu organisieren und ermöglichen eine bessere Planung und Priorisierung der Entwicklung. Sie können als übergreifende Aufgaben oder Projektabschnitte betrachtet werden, die in kleinere User Stories und Aufgaben aufgeteilt werden können, um die Umsetzung zu erleichtern.

# Entwicklung der Terminverwaltungswebanwendung

**1. Projektsetup und Planung:**

- Dies umfasst das Einrichten des Entwicklungsumfelds, die Festlegung von Projektzielen und -anforderungen sowie die Erstellung eines Entwicklungs- und Testplans.

**2. Datenbankdesign und -entwicklung:**

- Beginnen Sie mit der Erstellung der Datenbankstruktur, einschließlich der Definition von Schemas und Beziehungen zwischen den Datenmodellen. Implementieren Sie auch die Datenbankanbindung.

**3. Benutzer-Authentifizierung und Sicherheit:**

- Implementieren Sie die Benutzer-Authentifizierung und die Sicherheitsmaßnahmen für die Anwendung, um unberechtigten Zugriff zu verhindern.

**4. Datenmodelle und Repository-Klassen:**

- Erstellen Sie die Java-Klassen für Datenmodelle und Repository-Klassen für den Zugriff auf die MongoDB-Datenbank.

**5. Controller-Klassen:**

- Entwickeln Sie die Controller-Klassen für die verschiedenen Benutzerrollen (Administrator, Kunde, Mitarbeiter), um die Anfragen zu verarbeiten.

**6. Service-Klassen und Geschäftslogik:**

- Implementieren Sie die Service-Klassen, um die Geschäftslogik zu entwickeln, wie z.B. Buchungen, Verfügbarkeit und Rabatte.

7. **Benutzeroberfläche (Front-End):**

- Entwickeln Sie die Benutzeroberfläche (Front-End) für die Anwendung, einschließlich der Benutzeroberflächenkomponenten und -seiten.

8. **REST-APIs (falls erforderlich):**

- Implementieren Sie RESTful APIs, falls die Anwendung eine API für die Kommunikation zwischen Front-End und Back-End benötigt.

**9. Validierungen und Tests:**

- Fügen Sie Validierungen für Formulare und Eingaben hinzu und schreiben Sie Unit-Tests und Integrationstests, um die Funktionalität sicherzustellen.

**10. Sicherheitstests und Penetrationstests:**

- Überprüfen Sie die Anwendung auf Sicherheitslücken und führen Sie Penetrationstests durch, um Schwachstellen zu identifizieren und zu beheben.

**11. Dokumentation und Benutzerhandbuch:**

- Erstellen Sie Entwicklerdokumentation und Benutzerhandbuch, um die Verwendung der Anwendung zu erleichtern.

12. **Deployment und Produktionsvorbereitung:**

- Bereiten Sie die Anwendung für das Deployment in einer Produktionsumgebung vor, einschließlich der Konfiguration von Datenbankverbindungen und Sicherheitsmaßnahmen.

13. **Schulung und Einführung:**

- Führen Sie Schulungen für die Benutzer und Administratoren durch und leiten Sie sie in der Anwendungsnutzung ein.

14. **Überprüfung und Qualitätssicherung:**

- Führen Sie abschließende Überprüfungen und Qualitätssicherungsprozesse durch, um sicherzustellen, dass die Anwendung ordnungsgemäß funktioniert.

15. **Inbetriebnahme und Start:**

- Nehmen Sie die Anwendung in Betrieb und starten Sie den produktiven Einsatz.

Diese Reihenfolge dient als grobe Richtlinie, und es kann notwendig sein, Aufgaben zu parallelisieren oder in Abhängigkeit von den spezifischen Anforderungen Ihres Projekts anzupassen. Es ist wichtig, während des gesamten Projekts auf die Prioritäten zu achten und regelmäßig Updates und Überprüfungen durchzuführen, um sicherzustellen, dass das Projekt im Zeitplan bleibt und die Anforderungen erfüllt werden.

Ein Meilensteinplan für 32 Wochen kann helfen, die wichtigsten Etappen und Ziele für Ihr Projekt im Laufe der Zeit zu organisieren. Hier ist ein Beispiel für einen Meilensteinplan:

# Terminplan

**Woche 1-2: Projektstart und Planung**

- Projektinitialisierung

- Anforderungsanalyse und Spezifikation

- Teamzusammenstellung und Ressourcenplanung

- Erstellung eines Projektplans und Zeitplans

- Beginn der Datenbankentwicklung

**Woche 3-4: Benutzer-Authentifizierung und Sicherheit**

- Implementierung der Benutzer-Authentifizierung

- Einrichtung von Sicherheitsmaßnahmen

- Entwicklung von Validierungen und Authentifizierungstests

**Woche 5-6: Datenbankentwicklung**

- Fortsetzung der Datenbankentwicklung

- Erstellung der MongoDB-Schemas

- Implementierung der Repository-Klassen

- Integrationstests für die Datenbank

**Woche 7-8: Controller und Service-Klassen**

- Entwicklung der Controller-Klassen

- Implementierung der Service-Klassen und Geschäftslogik

- Beginn der Front-End-Entwicklung

**Woche 9-10: Benutzeroberfläche (Front-End)**

- Fortsetzung der Front-End-Entwicklung

- Gestaltung von Benutzeroberflächenkomponenten und -seiten

- Integration mit Back-End-Services

**Woche 11-12: REST-APIs (falls erforderlich)**

- Implementierung von RESTful APIs

- Schnittstellenentwicklung und Dokumentation

- Validierung und Tests der APIs

**Woche 13-16: Buchungsfunktionen und Kundenverwaltung**

- Implementierung der Buchungsfunktionen für Kunden

- Verwaltung von Kundendaten

- Buchungstests und Validierung

**Woche 17-20: Mitarbeiterverwaltung und Verfügbarkeit**

- Entwicklung von Mitarbeiterverwaltungsfunktionen

- Verfügbarkeitsverwaltung für Mitarbeiter

- Test und Validierung der Mitarbeiterfunktionen

**Woche 21-24: Rabattverwaltung und Sicherheitstests**

- Implementierung der Rabattverwaltung für Stoßzeiten

- Sicherheitstests und Penetrationstests durchführen

- Behebung von Sicherheitslücken und Schwachstellen

**Woche 25-28: Dokumentation und Benutzerhandbuch**

- Erstellung von Entwicklerdokumentation

- Erstellung eines Benutzerhandbuchs

- Schulungsvorbereitung für Benutzer und Administratoren

**Woche 29-30: Produktionsvorbereitung und Deployment**

- Vorbereitung der Anwendung für das Produktionsdeployment

- Konfiguration von Datenbankverbindungen und Sicherheitsmaßnahmen

**Woche 31: Inbetriebnahme und Schulung**

- Inbetriebnahme der Anwendung

- Schulungen für Benutzer und Administratoren

Woche 32: Projektabschluss und Qualitätssicherung

- Letzte Überprüfung und Qualitätssicherung

- Abschlussbericht und Projektdokumentation

- Projektabschluss und Übergabe an die Stakeholder

Bitte beachten Sie, dass dies nur ein Beispiel für einen Meilensteinplan ist, und Sie sollten die Zeiten und Aufgaben entsprechend den spezifischen Anforderungen und Ressourcen Ihres Projekts anpassen. Es ist auch wichtig, regelmäßig den Fortschritt zu überprüfen und den Zeitplan bei Bedarf anzupassen.