TAU INF303 Software Engineering Projekt

Projektdokumentation

2021

Melde- und Nachverfolgungssystem für Studentenclubs

Product Owner: Önder Tombuş

Autoren:

Name	Email
Arif Erdem Avcı	<e180503043@stud.tau.edu.tr>
Helin Öykü Demircioğlu	<e160503114@stud.tau.edu.tr>
İhsan Kürt	<e180503042@stud.tau.edu.tr>
Pelin İşeri	<e180503039@stud.tau.edu.tr>

Rümeysa Durak <e180503034@stud.tau.edu.tr>

Dokumentenverwaltung

Dokument-Historie

Version	Status *)	Datum	Verantwortlicher	Änderungsgrund

*) Sofern im Projekt nicht anders vereinbart sind folgende Statusbezeichnungen zu verwenden

(in obiger Tabelle und am Deckblatt):

Entwurf / in Review / gültig / ungültig

Dokument wurde mit folgenden Tools erstellt:

Microsoft Office Word

Inhaltsverzeichnis

1 Einleitung

1.1 Motivation

Als Mitglied von InfX, dem Informatik Club unserer Universität, haben wir vor einigen Jahren mit dem Verkauf von Stickern begonnen, um Clubveranstaltungen zu finanzieren. Jedes Mitglied, welches Verkäufe tätigte, musste die Einnahmen und die Anzahl der verkauften Sticker melden, um die Inkonsistenz zwischen der Anzahl der verkauften Sticker und den erziehlten Einnahmen zu schützen. Darüber hinaus war es notwendig zu informieren, an wen und wie viel verkauft wurde. Mit steigendem Umsatz wurde dieser Prozess komplexer und anspruchsvoller. Da wir Verkaufs- und Produktinformationen auf komplexe Weise in Excel speichern, hatten wir einige Probleme wie Leistungsbilanzdefizite, Produktverluste usw. Außerdem denken wir, dass ein System, in dem wir Verkaufsanalysen durchführen können, von Vorteil wäre. Zum Beispiel können Mitglieder, die mehr verkaufen, innerhalb des Clubs mehr Berechtigungen erhalten, weil sie aktiver sind als andere. Beispielsweise kann diese Person in ihrer eigenen Abteilung innerhalb des Clubs mehr Mitspracherecht haben. Oder wir können die Bestellung entsprechend der Nachfrage nach den Produkten erhöhen oder verringern. D.h., mehr verkaufte Produkte werden mehr vom Verkäufer bestellt, wodurch der Gewinn erhöht wird. Wir stellten fest, dass für den Verein andere Produkte als Sticker verkauft werden konnten und dass andere Vereine an unserer Universität in einer ähnlichen Situation waren. Ähnliche Schwierigkeiten gibt es nicht nur in Clubs, sondern auch bei der Verfolgung der Verkäufe von lizenzierten Universitäts Produkten wie Sweatshirts, Tassen usw.

Darauf aufbauend haben wir uns entschieden, eine Website zu entwickeln, die als Sales-Tracking-System (oder Verkaufsverfolgungssystem) für die Clubs unserer Universität genutzt werden kann. Auf diese Weise können alle Clubs auf einer einzigen Website leicht Verkäufe verfolgen und eine Verkaufsstrategie usw. erstellen, indem sie Statistiken verwenden.

1.2 Zweck des Dokuments

Der Zweck dieses Dokuments besteht darin, eine Beschreibung des Entwurfs eines Systems bereitzustellen, die vollständig genug ist, um der Softwareentwicklung zu ermöglichen, mit einem Verständnis dessen fortzufahren, was gebaut werden soll und wie es gebaut werden soll.

1.3 Begriffsbestimmungen und Abkürzungen

Auflistung von Begriffsdefinitionen und Abkürzungen, die für das Verständnis der Grobspezifikation wichtig sind (EDV-Fachausdrücke und Begriffe aus der Anwendungsdomäne) - unabhängig davon, ob diese in einem anderen Dokument (z.B. Pflichtenheft) bereits erklärt wurden.

MVC Steht für "Model-View-Controller". MVC ist ein Anwendungsdesignmodell, das aus drei miteinander verbundenen Teilen besteht. Dazu gehören das Modell (Daten), die Ansicht (Benutzeroberfläche) und der Controller (Prozesse, die Eingaben verarbeiten).

UML Die Unified Modeling Language (UML) ist eine universelle Entwicklungs- und Modellierungssprache im Bereich der Softwareentwicklung, die eine Standardmethode zur Visualisierung des Entwurfs eines Systems bieten soll

SOLID SOLID ist ein beliebter Satz von Designprinzipien, die in der objektorientierten Softwareentwicklung verwendet werden. SOLID ist ein Akronym, das für fünf zentrale Designprinzipien steht: Single-Responsibility-Prinzip, Open-Closed-Prinzip, Liskov-Substitutionsprinzip, Interface-Segregation-Prinzip und Dependency-Inversion-Prinzip

SQL Structured Query Language

ASP.NET *ASP.NET* ist ein serverseitiges Open-Source-Webanwendungs-Framework, das für die Webentwicklung entwickelt wurde, um dynamische Webseiten zu erstellen. Es wurde von Microsoft entwickelt, um Programmierern die Erstellung dynamischer Websites, Anwendungen und Dienste zu ermöglichen.

HTML5 HTML5 ist eine Auszeichnungssprache, die zum Strukturieren und Präsentieren von Inhalten im World Wide Web verwendet wird. Es ist die fünfte und letzte große HTML-Version, die eine Empfehlung des World Wide Web Consortium (W3C) ist. Die aktuelle Spezifikation ist als HTML Living Standard bekannt.

CSS Cascading Style Sheets (CSS) ist eine Stylesheet-Sprache, die zum Beschreiben der Präsentation eines Dokuments verwendet wird, das in einer Auszeichnungssprache wie HTML geschrieben wurde

Bootstrap Bootstrap ist ein kostenloses Open-Source-CSS-Framework, das auf die responsive Front-End-Webentwicklung ausgerichtet ist. Es enthält CSS- und (optional) JavaScript-basierte Designvorlagen für Typografie, Formulare, Schaltflächen, Navigation und andere Schnittstellenkomponenten.

ORM Objektrelationales Mapping in der Informatik ist eine Programmiertechnik zum Konvertieren von Daten zwischen inkompatiblen Typsystemen unter Verwendung objektorientierter Programmiersprachen. Dies erzeugt in der Tat eine "virtuelle Objektdatenbank", die innerhalb der Programmiersprache verwendet werden kann.

EF **E**ntity Framework (EF) ist ein Open-Source-Framework zur relationalen Zuordnung (ORM) für ASP.NET.

LINQ Language Integrated Query (LINQ, ausgesprochen "Link") ist eine Microsoft .NET Framework-Komponente, die .NET-Sprachen native Datenabfragefunktionen hinzufügt

DD *Deployment Diagram*

AD *Activity Diagram*

CD *Class Diagram*

SD *Sequence Diagram*

UD *Use Case Diagram*

2 Projekt Management

PRODUCT SPRINT PLAN

PROJECT NAME	START DATE	END DATE
de- und Nachverfolgungssystem für Studentenclubs		

AT RISK	TASK NAME	FEATURE TYPE	RESPONSIBLE	STORY POINTS	START	FINISH	DURATION (DAYS)	STATUS
	Sprint 1		Pelin İşeri		15.11.21	30.11.21	15	In Progress
	Klassen, Datenbank, Migrationen		Rümeysa Durak				15	In Progress
	Produkt und Kategorie(Controller,View)		Pelin İşeri				15	In Progress
	Abteilungen und Kunden(Controller,View)		Helin Öykü Demircioğlu				15	In Progress
	Sprint 2		Arif Erdem Avcı		5.1.21	15.1.21	7	Not Started
	Angestellte und Verkaufen(Controller,View)		İhsan Kürt				7	Not Started
	Rechnung?? Ladung??		?				?	Not Started
	Statistiken,Todo Seiten		Arif Erdem Avcı				7	Not Started
	Sprint 3		Helin Öykü Demircioğlu		15.1.21	27.1.21	12	Not Started
	Login/Register		Rümeysa Durak/İhsan Kürt				10	Not Started
	Mitteilungen		Arif Erdem Avcı				10	Not Started
	Berechtigen		Pelin İşeri/ Helin Öykü Demircioğlu				10	Not Started
	andere kleine Details und Designverbesserung		Alle Mitglieder				7	Not Started

Wir beginnen unser Projekt, indem wir zuerst die Datenbank verbinden. Wir werden jedoch die Klassen und Attribute bestimmen, die wir benötigen. Diese Aufgabe stellt hauptsächlich Rümeysa Durak dar. Später werden wir die ermittelten Klassen gleichmäßig verteilen und mit der Datenbank verbinden.

Wie in der Tabelle zu sehen ist, wird Pelin İşeri für die Erstellung der Produkt- und Kategorietabelle verantwortlich sein, die notwendigen Funktionen im Controller und die kleinen Details, die von diesen Klassen benötigt werden, schreiben, während Helin Öykü Demircioğlu für die Klassen abteilungen und kunden und İhsan Kürt für die Klassen angestellte und kaufen. Dabei helfen wir uns bei Problemen gegenseitig.

Dann dachten wir, es sollte eine Statistik und eine Todo-Liste sein. Grund dafür ist die Effizienz im Vertrieb. Wie viele von welchem Produkt wurden an wen verkauft oder das meistverkaufte/teuerste usw. Wir halten es für wichtig, den Benutzern Komfort zu bieten, indem wichtige Funktionen identifiziert werden. Arif Erdemir Avcı wird sich auf dieses Thema konzentrieren.

Wenn wir am Ende dieser Teile angelangt sind, gehen wir auf die Details des Jobs ein und kümmern uns um den Teil Login/Registrierung, Autorisierung und Nachrichten. Zum Schluss gehen wir auf den Weg, das von uns festgelegte Design zu verschönern / kleine Details hinzuzufügen.

3 Produktumfang

Unser Verfolgungssystem dient dazu, die Produkte unserer Universität in systematischer Weise verkaufen zu können.
Man wird in der Lage sein, die Produkte der Clubs oder Vereine, oder auch der Universität selbst zu kaufen.

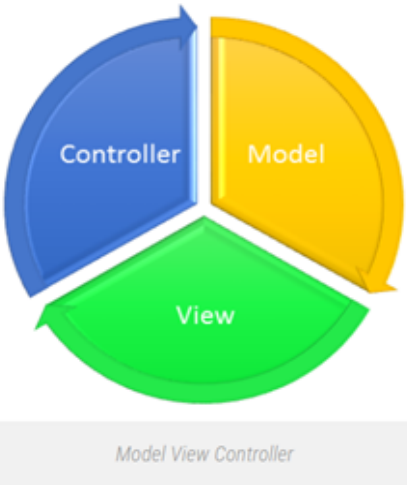
Der Verkäufer kann sich dort anmelden und seine Produkte hochladen.
Produkte können jeder Zeit gelöscht werden.
Der Verkäufer kann ein Club in der Universität sein wie zum Beispiel der Club infX.
Außerdem kann auch die Üniversität selbst, eigene Produkte verkaufen, bei möglichen Produkten der Universität speziell.

Der Benutzer, genauer gesagt der Käufer kann sich ein Account erstellen und sich die Produkte anschauen, die im System hochgeladen sind.
Diese Produkte kann er jederzeit kaufen.

3.1 Technische Infrastruktur

Als Entwicklungsumgebung haben wir uns in diesem Projekt für ASP.Net MVC5 entschieden. Bevor wir ASP.Net MVC erklären, lassen Sie uns darüber sprechen, was MVC bedeutet.

MVC ist die Abkürzung für Model View Controller und ist eines der Architekturmuster, das einen sehr wichtigen Platz in der Anwendungsentwicklung einnimmt. Wir zerlegen Code in Strukturen, die verschiedenen Zwecken dienen. So können wir unseren Code einfacher entwickeln, testen und weniger fehleranfällig machen.



Modell: Stellt die in der Anwendung verwendeten Daten dar und ist der Abschnitt, in dem die Datenverarbeitungslogik gespeichert ist.

View: Dies ist die Ebene, die die Benutzeroberfläche enthält, die Benutzer sehen. Im View-Layer können neueste Technologien wie HTML5 und CSS3 verwendet werden.

Controller: Verarbeitet die Anfrage des Clients, stellt die Kommunikation zwischen dem Modell und der Ansicht her. Es überträgt die von den Benutzern aus der Ansicht entnommenen Daten in das Modell und zeigt die aus dem Modell entnommenen Daten dem Benutzer über der Ansicht.

MVC5: Neueste Version von MVC

ASP.NET MVC ist das von Microsoft entwickelte Framework, um das MVC-Pattern zu ASP.NET hinzuzufügen.

Warum sollten wir ASP.NET MVC wählen?

- Mit MVC besteht eine große Möglichkeit der Kontrolle über die Ausgabe, die als Reaktion auf die Anfrage des Clients erzeugt wird, und die Adressen, an die die Anfrage gesendet wird. Auf diese Weise wird sichergestellt, dass die am besten geeignete Ausgabe produziert werden kann und die Adresse einen vollständigen Bezug zum Inhalt hat.

- Es macht es uns sehr einfach, testbare Anwendungen mit MVC zu entwickeln. Weil die Schichten voneinander getrennt sind. Es verfügt auch über eine Architektur, die die testgetriebene Entwicklung erleichtert.

- Ein weiterer Vorteil der Trennung von MVC-Schichten besteht darin, dass jede Schicht in anderen Projekten verwendet werden kann. Somit ist es möglich, mit MVC wiederverwendbaren Code zu generieren.

Welche Grundsätze verfolgt die Architektur?

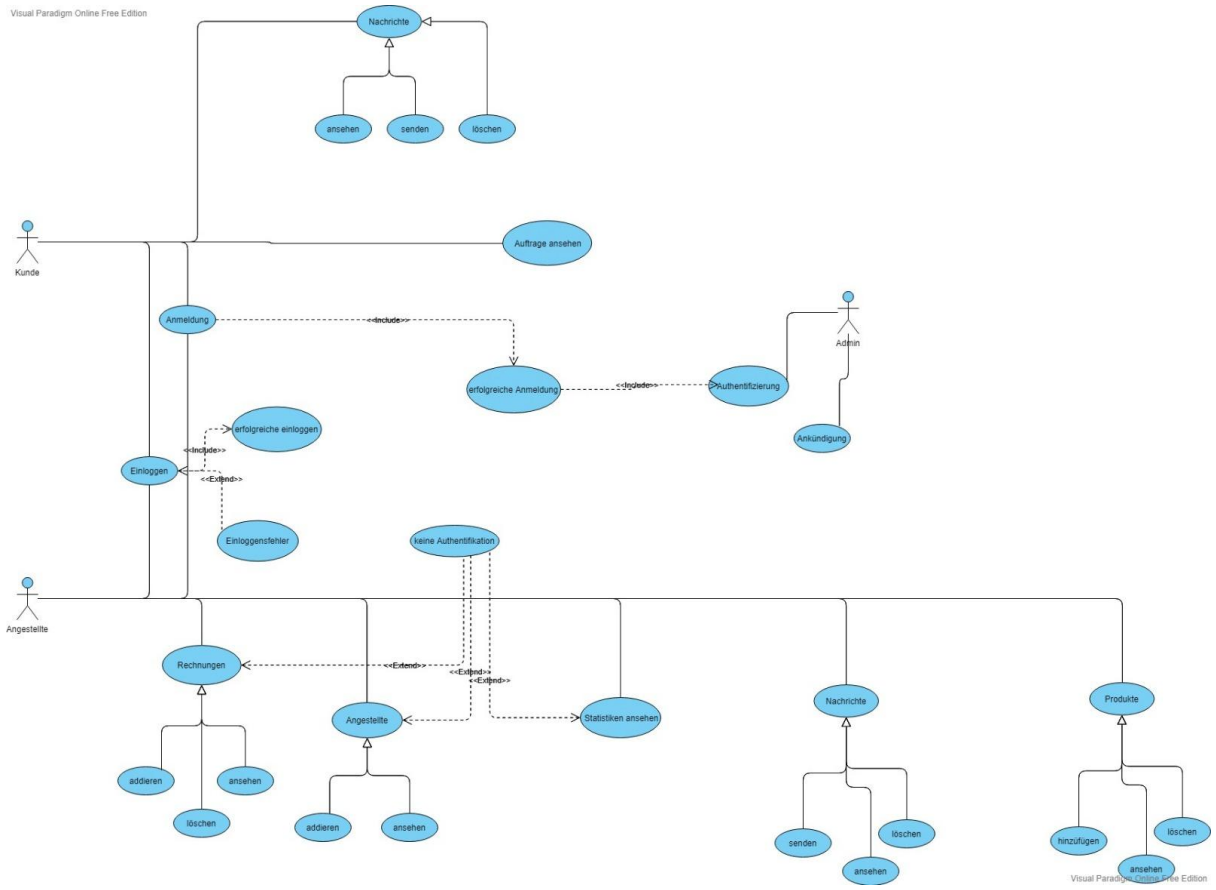
Einer der Vorteile der Verwendung von MVC besteht darin, dass es dem S-Prinzip in SOLID folgt.

SOLID ist eine mnemonische Abkürzung für fünf Designprinzipien, die darauf abzielen, Softwaredesigns verständlicher, flexibler und nachhaltiger zu machen.

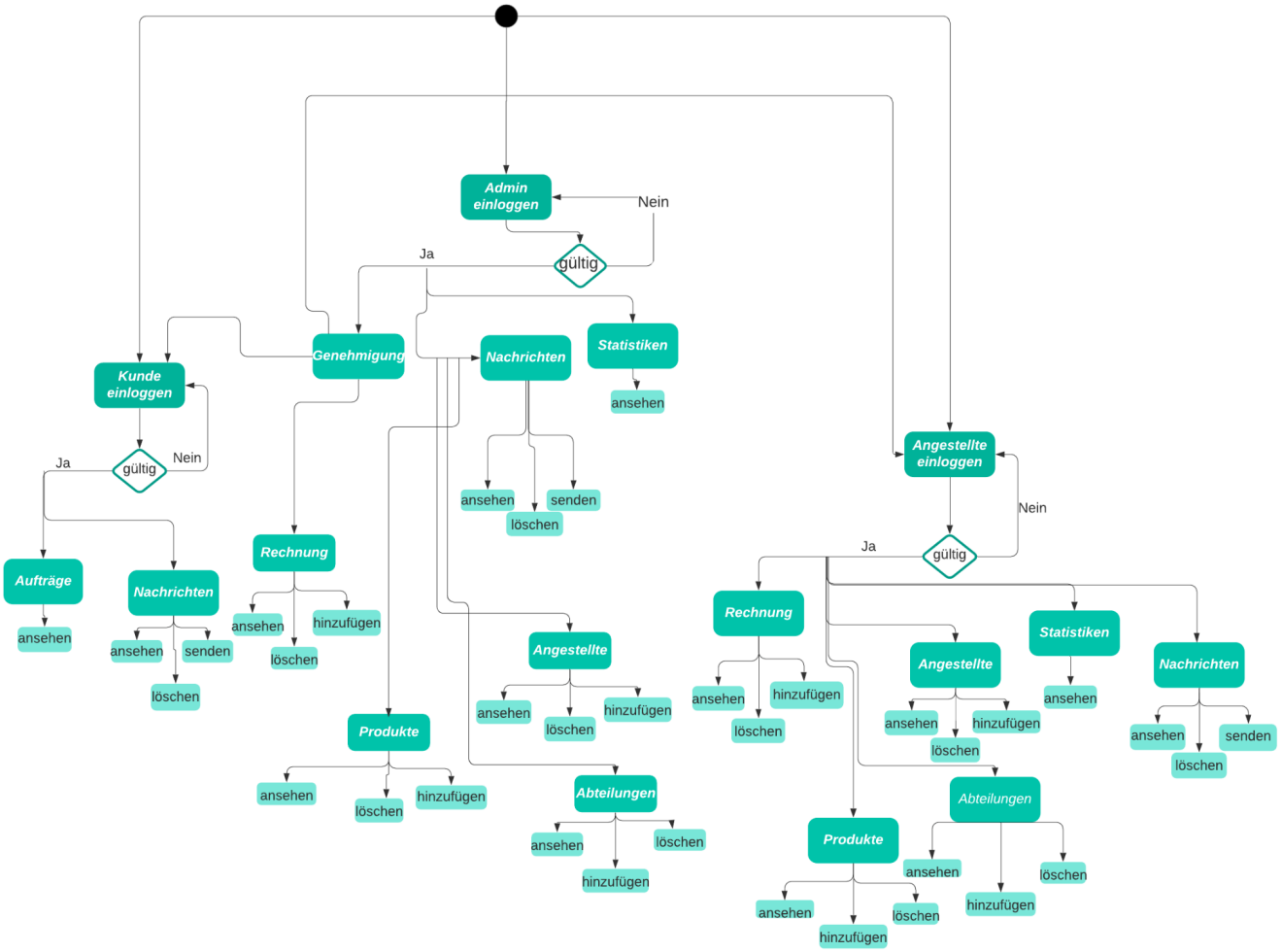
- **S — *Single-responsibility principle*:** Unsere Klassen sollten eine einzige, genau definierte Verantwortung haben.
- **O — *Open-closed principle:*** Unsere Klassen sollten für Änderungen geschlossen sein, aber offen für das Hinzufügen neuer Verhaltensweisen.
- **L — *Liskov substitution principle:*** Wir sollten in der Lage sein, die abgeleiteten Klassen (Unterklasse) anstelle der übergeordneten Klasse (Basisklasse) zu verwenden, von der sie abgeleitet sind, ohne Änderungen an unserem Code vorzunehmen.
- **I — *Interface segregation principle:*** Wir sollten lieber spezialisiertere Verträge als einen einzigen Vertrag für den allgemeinen Gebrauch erstellen.
- **D — *Dependency Inversion Principle:*** In mehrschichtigen Architekturen sollten Module der obersten Ebene nicht direkt von Modulen der unteren Ebene abhängig sein.

MVC folgt aber auch anderen Prinzipien. Das Hinzufügen neuer UI-Elemente ist beispielsweise einfach. Dazu müssen wir weder das Modell noch den Controller ändern, da die View getrennt ist. Damit folgt auch sie dem Open/Closed-Prinzip. Und andere Prinzipien werden auch verwendet.

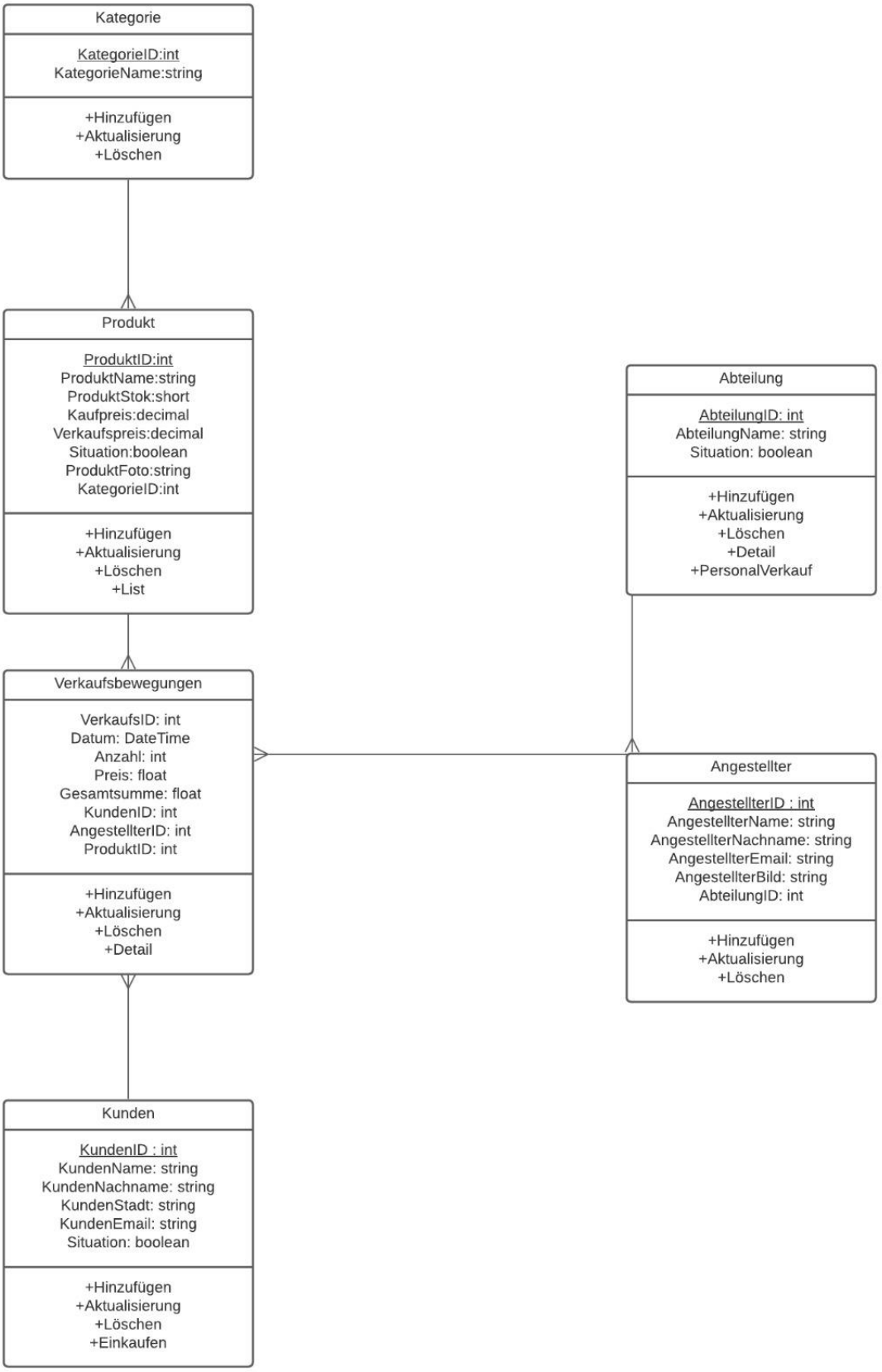
3.2 Anforderungsanalyse / Use Cases



UML Use Case Diagram



UML Activity Diagramm



UML Class Diagramm

3.3 Architektur

Eine der wichtigsten Komponenten in unserem Projekt sind Kategorien. Hier behalten wir die Kategorien der Produkte, die wir verkaufen werden. Zum Beispiel Sticker, Schlüsselanhänger, Kleidung, Kurs usw . und zusätzlich zu den Funktionen zum Hinzufügen, Löschen und Aktualisieren der Kategorien können wir auf der Detailseite sehen, welche Produkte in einer bestimmten Kategorie enthalten sind. Hier wird unser Schlüssel CategoryeID sein. CategoryName ist eines der Attribute, die wir im UML-Diagramm erwähnt haben.

Es besteht eine 1-N-Beziehung zwischen der Kategorietabelle und der Produkttabelle. Während die Hauptfunktionen der Produkttabelle das Hinzufügen und Löschen von Updates sind, sind unsere Attribute ID, Name, Lager, Kauf- und Verkaufspreis, Statusfoto und Kategorie-ID für die Kategoriebeziehung. Die Produkttabelle hat jedoch auch eine Beziehung zur Verkaufstransaktionstabelle. Denn bei jedem Verkauf müssen wir wissen, welches Produkt wie viel und zu welchem Preis verkauft wird. Dementsprechend sollte der Lagerstatus automatisch aus der Produkttabelle abgezogen werden.

In der Verkaufstransaktionstabelle werden Funktionen zum Hinzufügen von Details zur Aktualisierung hinzugefügt. In dieser Tabelle können wir sehen, wer wie viel und von welchem Produkt verkauft hat, während wir andere Verkaufsmerkmale im Detail sehen können. Als Attribut gibt es ID, Datum, Nummer, Preis, Gesamtbetrag, Kunden-ID, Umsatz-ID und Angestellt-ID.

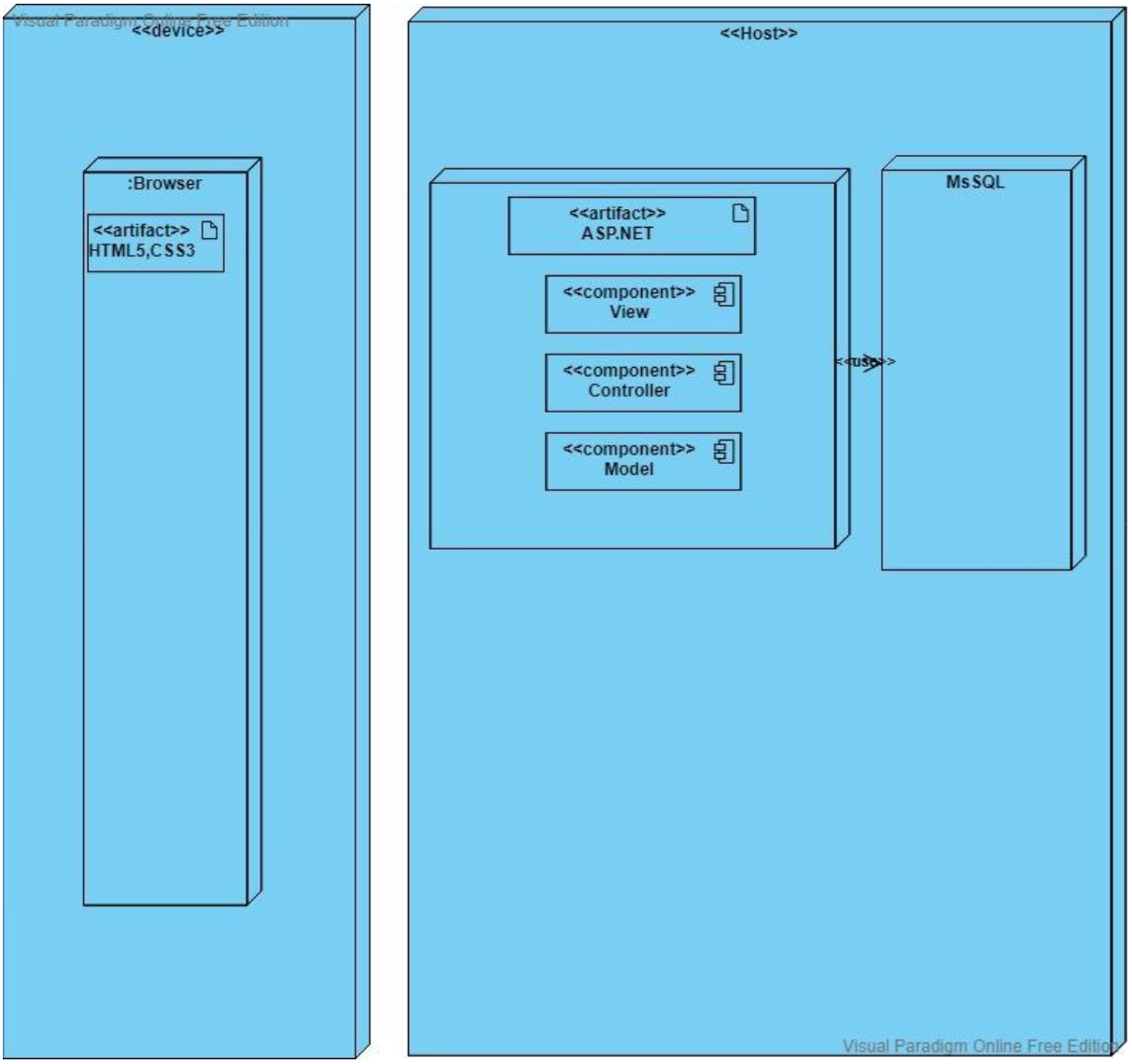
Wie wir den ID-Beziehungen entnehmen können, ist die Verkaufsvorgangstabelle mit insgesamt drei Tabellen verknüpft: Kunde, Angestellter und Produkt.

Um eine gesündere Kommunikation innerhalb der Univesitaet herzustellen, sollte auch eine Tabelle mit den Informationen der Studenten (eigentlich der Studenten der Uni) vorhanden sein. Während der Kunde über Funktionen zum Hinzufügen, Löschen, Aktualisieren und Verkaufsverlauf verfügt, werden ID-Name / Nachname, Stadt, E-Mail und Status als Zuweisung gespeichert. Im Falle einer Ankündigung wird die Öffentlichkeitsarbeit der Vereins die Kunden unter Verwendung dieser Informationen kontaktieren. Der Grund, warum wir die Stadt gekauft haben, ist, dass wir während der Quarantänezeit einige Frachtverkäufe getätigt haben. Wir haben ein solches Attribut als Vorsichtsmaßnahme gegen solche Situationen erstellt.

Studentenvereins mit einem System haben in der Regel viele Abteilungen. Obwohl dieser Club je nach Bereich variiert, in dem er tätig ist, sind im Allgemeinen Abteilungen für Öffentlichkeitsarbeit, Veranstaltungen und soziale Medien in allen verfügbar. Für die Clubleitung ist es wichtig zu sehen, welche Abteilungen und welche Mitglieder mehr verkaufen und aktiver sind. Denn unter Berücksichtigung dieser Bedingungen werden je nach Tätigkeit einige Rechte gewährt. Aus diesem Grund haben wir uns entschieden, die Abteilungen der Mitglieder, die den Verkauf tätigen, in unserem System zu behalten. Neben den Funktionen zum Hinzufügen/Löschen von Aktualisierungen/Detail/Verkauf behalten wir auch die ID/Abteilungsbezeichnung und den Status der Abteilung (aktiv/inaktiv) als Attribut bei. Es besteht eine 1-N-Beziehung zwischen der Abteilung und dem angestelter.

Schließlich bezieht sich unsere Tabelle Angestellter sowohl auf die Abteilungs- als auch auf die Verkaufstransaktionstabelle. Zusätzlich zu unseren Funktionen zum Hinzufügen/Löschen/Aktualisieren gibt es als Attribute ID, Vorname, Nachname, Foto, Abteilung, E-Mail und Abteilungs-ID.

3.4 Deployment View



4 Teststrategie und Testplanung

Die oberste Priorität ist es, die definierten Use Cases des Produkts zu validieren. Hier soll es beschrieben werden wie das Produkt getestet wird.

[REQ4] Als Projektanforderung soll es für jeden Use Case mindestens einen Test Case definiert, implementiert und durchgeführt werden.

[ÖT1]

QUELLE:

https://en.wikipedia.org/wiki/SOLID
http://aliozgur.net/2018/03/02/solid/