**Begrifflichkeiten**

**Backend/Frontend**

* Schichteinteilung der Funktionalität einer Web-App
* Frontend näher am Nutzer
* Backend näher am System
* <https://de.wikipedia.org/wiki/Front-End_und_Back-End>

**Benefit**

* Synonym für Vorteil
* <https://de.wikipedia.org/wiki/Benefit>

**Business Case**

* Prüfverfahren ob ein Geschäftsszenario rentabel ist
* <https://de.wikipedia.org/wiki/Business-Case>

**Cascading Style Sheets (CSS)**

* Style sheet scripting language für festlegung des Erscheinungsbildes der Webseite
* Migration in HTML sowie JavaScript
* <https://de.wikipedia.org/wiki/Cascading_Style_Sheets>

**Continuous Delivery Pipeline**

* Sammlung von
  + Techniken
  + Prozessen
  + Tools
* Für die Verbesserung des Softwareentwicklungsprozesses
* <https://de.wikipedia.org/wiki/Continuous_Delivery>

**Deployment**

* Prozesse zur Installation von Software auf Rechnern
  + Automatisierte Installation durch Anweisung von der Ferne möglich
* Findet vor allem in der Software-Entwicklung statt
  + Temporäres Erstellen von Testumgebungen
* <https://de.wikipedia.org/wiki/Softwareverteilung>

**DotNet (.Net)**

* Beinhaltet Net.Core 5.0
  + Entwicklung und Ausführung von Anwendungsprogrammen
* Grundlage für Web API „ASP.NET“
* <https://de.wikipedia.org/wiki/.NET_5>

**Entity-Relationship-Modell (ER-Modell)**

* Verfahren für eine Datenmodellierung
* Beinhaltet Akteure (Entity) sowie Beziehungen (Relations) zwischen den Akteuren
* Entities beinhalten Merkmale (Attribute)
* <https://de.wikipedia.org/wiki/Entity-Relationship-Modell>

**Framework**

* Allgemein
  + Dient als Programmiervorlage
  + Findet insbesondere Anwendung in
    - Objektorientierte Softwareentwicklung
    - Komponentenbasierte Entwicklungsansätze
  + <https://de.wikipedia.org/wiki/Framework>
* Testframework
  + Bestandteil zur Testautomatisierung
  + Erstellen einer temporären Testumgebung
    - Liefert Testergebnisse
    - Löscht danach Testumgebung
  + Continuous Delivery Pipeline für Testautomatisierung
  + <https://de.wikipedia.org/wiki/Testautomatisierung>
* Webframework
  + Software für die Entwicklung von
    - Dynamischen Websites
    - Web Apps
    - Webservices
  + <https://de.wikipedia.org/wiki/Webframework>

**NAS Server**

* Speicher der für ein Netzwerk auf einen Server zur Verfügung gestellt wird
* <https://de.wikipedia.org/wiki/Network_Attached_Storage>

**Objektorientiertes Programmieren (OOP)**

* Programmieren von Klassen als Vorlage von Objekten
  + Generierung von mehreren Objekten, welche unterschiedlich agieren
* Interfaces
  + Klassen mit leeren Methoden und Attributen als Schablone
* Templates
  + Universelles Attribut für den Einsatz beliebiger Datentypen
* Aufbau der Klassen in
  + Methoden
  + Attributen
* Vererbung von Klassen
  + Vererbung der Methoden
  + Vererbung der Attribute
* Sicherheitsabstufungen
  + Public
    - Zugriff von überall
  + Protected
    - Zugriff nur über eigene Klasse sowie vererbte Klassen
  + Private
    - Zugriff nur über eigene Klasse
* <https://de.wikipedia.org/wiki/Objektorientierte_Programmierung>

**Programmablaufplan (PAP)**

* Ablaufdiagramm für eine Anwendung
* <https://de.wikipedia.org/wiki/Programmablaufplan>

**Progressive Web Application (PWA) vs. Native Application (NA)**

* <https://topflightapps.com/ideas/native-vs-progressive-web-app/>
* PWA
  + Hybrid aus Website mit Responsive Design undApp
  + Besitzt Elemente einer Native Application
  + <https://de.wikipedia.org/wiki/Progressive_Web_App>
* NA
  + Mobile Application
  + An einer bestimmten Plattform angepasst
  + <https://de.wikipedia.org/wiki/Mobile_App>

**Proof of Concept**

* Als Meilenstein zu betrachten
* Beleg für die Durchführbarkeit eines Vorhabens
* <https://de.wikipedia.org/wiki/Proof_of_Concept>

**Remote Desktop Protocol (RDP)**

* Ermöglicht Remote Control
* <https://de.wikipedia.org/wiki/Remote_Desktop_Protocol>

**Repository**

* Speicher für Softwarepakete
* Tabelle mit Inhalten in Repository mit Metadaten gespeichert
* <https://en.wikipedia.org/wiki/Software_repository>

**Scaffolding**

* Gerüst für eine vollständige Orientierungsgrundlage in Form von
  + Anleitungen
  + Denkanstößen
  + und weitere Hilfestellungen
* <https://de.wikipedia.org/wiki/Scaffolding>

**SQL vs. NOSQL**

* SQL (Structured Query Language)
  + Scriptsprache für Anfragen in relationalen Datenbanken
  + <https://de.wikipedia.org/wiki/SQL>
* NOSQL (Not only SQL)
  + Aufbrechen der relationen Datenbankstruktur
  + Keine festgelegten DB-Schemata mehr notwendig
  + Kann Joins vermeiden
    - Performancesteigerung
  + <https://de.wikipedia.org/wiki/NoSQL>

**Technische Analyse**

* Form einer Finanzanalyse
* Untersucht Verhalten des Finanzmarktes
* <https://de.wikipedia.org/wiki/Technische_Analyse>

**Toolchain**

* Betrachtet den Funktionszusammenhang von Tools/Frameworks + Hardware
  + Für die Erfüllung einer Aufgabe
* <https://de.wikipedia.org/wiki/Toolchain>

**Web API**

* Konzept für die Entwicklung von Web-Apps
* Web Application Programming Interface (Web API)
  + Web server
  + Web browser
* <https://en.wikipedia.org/wiki/Web_API>

**Workflow**

* Beschreibt einen Arbeitsablauf
* Räumliche und zeitliche Reihenfolge von zusammengehörenden Arbeitsvorgängen
  + funktional
  + physikalisch
  + technisch
* <https://de.wikipedia.org/wiki/Arbeitsablauf>

**Virtuelle Umgebung**

* Simulation eines Rechnersystems
* Bildet Hardware eines hypothetischen Rechners nach

<https://de.wikipedia.org/wiki/Virtuelle_Maschine>