Azure Notizen  
  
Hinweis: Die vorliegenden Informationen gehören zu den Notizen, die ich mir beim Lesen von über 1700 Seiten gemacht habe. Abgesehen von fachbezogenen Notizen gibt es zusätzlichen auch einen kleinen Anteil an Informationen zu arbeitsbezogenen Aufgaben wie dem Ticketsystem, Outlook oder Microsoft Teams. Ich habe nach Möglichkeit versucht Themen, die zusammen gehören auch zusammen zu legen. Da es Notizen sind kann es sein das die Texte nicht vollständig alles Erklären oder weiter in der Thematik anfangen.  
  
Microsoft Azure Grundlagen

* Cloud Computing  
  Verschieben der IT-Infrastruktur in die Cloud
* Claud Namensherkunft  
  In Netzwerkdiagrammen wird für das Internet die Wolke als Symbol genutzt daher Cloud wie Wolke.

Cloud Kategorien (Angebort)

* Compute (berechnen)  
  Dienste die Rechenleistung erbringen z.B. virtuelle Maschinen, Datenträger Analyseplattformen, Webanwendungen.
* Storage  
  Alles was Daten speichert z.B. Dateien, Dateiablagen, Datenbanken
* Networking  
  Virtuelle Netzwerkkomponenten z.B. Gateways, Firewalls
* Analysis  
  Überwachung in technischer und finanzieller Hinsicht.

Cloud-Merkmale  
  
Begriffe die mit Cloud in Verbindung gebracht werden bzw. diese Auszeichnen:

* Agilität  
  Die benötigten Ressourcen können schnell nach unten oder oben skaliert werden um sich an die Erfordernisse anzupassen.
* Elastizität  
  Anpassen (skalllieren) der Ressourcen in Echtzeit (automatische Anpassung nach Erfordernissen)
* Fehlertoleranz  
  Stabilität der Umgebung durch Schaffung von Redundanzen.
* Globale Reichweite  
  Bereitstellung in einer Region nahe dem Einsatzort.
* Hochverfügbarkeit (High Availbility)  
  IT-Dienste können mit wenig Unterbrechung/ Störung angeboten werden (z.B. Unterbrechung durch Updates)
* Notfallwiederherstellung (Disaster Recovery)  
  Dienst wird nach dem Ausfall möglichst schnell und gegebenenfalls automatisch wieder bereitgestellt.
* Skalierbarkeit  
  Anpassung der Ressourcen nach dem Bedarf

Stufen der Cloud-Dienstmodelle

* Software as a Service (SaaS)  
  Bereitstellung von Soiftware ohne dafür eine Infrastruktur zu schaffen.
* Plattform as a Service (PaaS)  
  Statt Dienste auf einer eigenen virtuellen Maschine laufen zu lassen nutzt man eine Plattform auf dem der Dienst läuft. Im Hintergrund läuft immer noch eine virtuelle Maschine um dessen Wartung man sich aber nicht kümmern muss.
* Infrastructure as a Service (IaaS)  
  Das einfachste Cloud-Dienstmodell. Rechner bzw. das Netzwerk (die Infrastrucktur) wird lediglich virtuell in der Cloud abgebildet.

Pizza as a Service

* Ohne Cloud  
  Man muss alles zur Herstellung einer Pizza haben. Ofen, Tisch, Zutaten, Know How. Der Vorteil ist das man bei allen Schritten die volle Kontrolle hat. Der Nachteil ist das hohe Kosten anfallen (Pizzaofen usw.)
* IaaS  
  IaaS entspricht einer Tiefkühlpizza. Die Pizza wurde schon mit entsprechenden Produktionsverfahren vorbereitet und muss nur noch in den Ofen geschoben werden.
* PaaS  
  Man bestellt seine Pizza bei einem Lieferservice. Man muss nur noch den Tisch decken und für Getränke sorgen. Die Pizza ist fertig kann aber in ihren Eigenschaften nicht geändert werden.
* SaaS  
  Man besucht das Restaurant. Alles ist bereit. Die Pizza ist fertig. Die Getränke sind da und der Tisch ist gedeckt. Man kann sofort loslegen. Wenn einem aber die Tischdecke nicht gefällt, hat man Pech gehabt.

Cloud-Bereitstellungsmodelle

* Private Cloud  
  Betreiben einer eigenen Cloud in einem eigenen Rechenzentrum.  
  Vorteil: Man hat die volle Kontrolle  
  Nachteil: teuer
* Öffentliche Cloud  
  Man mietet sich Ressourcen über einen Anbieter. Da es viele Mieter gibt spricht man auch von einer „geteilten Umgebung“. Der Vorteil ist das man weniger Know How benötigt und man die Hardwareumgebung nicht selber aufbaut.
* Hybride Cloud  
  Eine Mischung aus privater und öffentlicher Cloud

Wirtschaftliche Cloud-Begriffe  
CapEx - Captital expenditure (Investitionskosten)  
OpEx - Operational expenditure (Operationskosten)  
  
Kurznotiz  
Kubernetes wird Kuberneties  
Bedeutung: orchestrieren (verwalten) von Prozessen oder Applikationen sowie deren Überwachung und Auswertung. Siehe Thema: K;ubernetes-Cluster

Azure-Architektur

* Azure Ressourcen  
  Alle Elemente die man in Azure erstellen kann z.B. Speicherkonten, Cluster usw. Manche Ressourcen setzen sich aus anderen Ressourcen zusammen.
* Azure Konto  
  Benutzerkonto mit dem man sich an Azure anmeldet. Dies ist ein Active Diretory oder aus Azure Active Directory kommt.
* Azure Abonnement  
  Abrechnungs- und Sicherheitskonto. Zum einen werden hier die genutzten Daten abgerechnet zum anderen müssen Benutzer über ein Abo verfügen, um Azure-Ressourcen nutzen zu können.
* Azure Ressource Gruppen  
  Alle Ressourcen müssen einer Ressourcengruppe zugeordnet sein. Ressourcengruppen sind als Container oder Ordnungseinheiten zu verstehen.
* Azure Verwaltungsgruppen  
  Man kann bis zu 6 Verwaltungsgruppen haben, in denen man Azure Abos verteilen kann.
* Azure Regionen  
  Jede Region verfügt über ein Rechenzenztrum.
* Azure Geografien  
  Mehrere Regionen werden zu einer Geografie zusammengefasst die z.B. dieselben gesetzlichen Vorschriften haben.
* Azure Verfügbarkeitszonen  
  Rechenzentren einer Region den getrennten Strom, Kühlung, Netzwerkanbindung haben. Bei Ausfall eines Rechenzentrums übernimmt ein anderes Techenzentrum aus der Verfügbarkeitszone.
* Azure Limit  
  Für bestimmte Ressourcen gibt es ein Limit pro Abo. Benötigt man mehr Ressourcen als das Limit erlaubt braucht man ein weiteres Abo.
* Azure Quota  
  Weiches Limit das kurzfristig erhöht werden kann indem man eine Quota Increase Request an den Support stellt.
* Azure Ressource Manager  
  Azure Komponenten die Anforderungen zur Erstellung einer Ressourceentgegen nimmt und umsetzt. Die Anforderung hat dabei die Form einer Json-Datei die die Einzelheiten enthält.

Notiz: CRM-System => Customer Relationship Management (z.B. Dynamics 365)  
  
Tickets erstellen  
  
In Dynamics 365 (Bin Control) -> Meine Tickets -> obere Leiste „+Ticket erstellen“

* Anfrage Titel   
  ist die Betreffzeile. Format: Firmenkürzel\_-\_Betreff  
  Beispiel: BCON\_-\_Testticket (\_ ist der Platzhalter für Leerzeichen)
* Kunde  
  Name der Firma oder für interne Verwendung „BCON“
* Ansprechpartner  
  Problemmelder/ Ansprechpartner beim Kunden  
  Intern: Leer lassen / eigener Name / oder anderer Mitarbeiter
* Projekt-Faktura (Abrechnungsprojekt)  
  Den Projektname bekommt man bei Projektzuweisung mitgeteilt. Für interne Projekte wird das Feld leer gelassen.
* Taskzuordnung/ Betreff  
  Leer lassen.
* Anfragetyp („Problem“ ist die Standarteinstellung)
* Priorität („Normal“ ist die Standarteinstellung)
* Faktura  
  Wird nicht geändert (Chefsache)

Ticket speichern  
  
Nach dem ersten speichern muss der Status von „Neu“ auf „Bearbeitung“ gestellt werden. Danach muss nochmal gespeichert werden.

Dokumentation zum Ticket erstellen  
  
In der Oberen Liste von Allgemein auf Dokumentation wechseln und den Button „+Dokumentation erstellen“

* Titel -> Hier ist kein Format vorgegeben
* Typ  
  - Leistungsbeschreibung  
  Kunde sieht Titel und Dauer  
  - Internes Memo  
  Nur Mitarbeiter sehen den gesamten Eintrag  
  - Hinweis  
  Sieht der Kunde wird aber nicht abgerechnet  
  - Beschreibung  
  Sieht nur der Autor der Dokumentation
* Verwaltung (Reiter) zeigt den Verlauf des jeweiligen Tickets an.

Dokumentation von Tätigkeiten  
  
Tätigkeiten für einen Kunden werden zum entsprechenden Kundenticket eingetragen. Interne Tätigkeiten werden in „BCon Diverser 20XX“ dokumentiert. Das Ticket findet man unter: Alle Tickets -> Suchzeile „\*divers“. Unter Divers fallen Tätigkeiten ohne Kundenbezug.

Outlook

Neben der Mail im Posteingang befindet sich ein rotes Fähnchen. Hier kann man per Klick eine Errinnerung speichern und einen Erinnerungstermin setzen.

Über das Kalendersymbol unten rechts kann man den Kalender aufrufen. Durch Rechstklick auf „Mein Kalender“ kann man über „neue Kalendergruppe“ Benutzer hinzufügen, um deren Termine einzusehen.

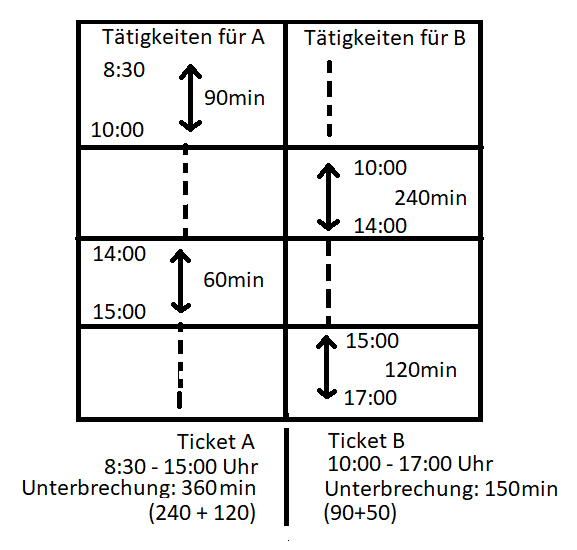
Ticket für eine Abrechnung erstellen

Zuerst wählt man „alle Tickets“. Im rechten Suchfeld filtert man nun die Auflistung mit dem Firmennamen für die man an dem aktuellen Projekt arbeitet z.B. „PAW Einführung Dynamics 365“.

Der Ticket-Titel ist in der Zeichenlänge limitiert daher muss der Titel gleichzeitig sinnvoll als auch kurz gewählt werden (Wiederholungen vermeiden). Die wichtigste Aufgabe wird zuerst genannt. Die Restlichen Aufgaben folgen danach oder werden mit „etc“ abgekürzt. Einzelne Aufgaben werden mit Kommas aufgezählt.  
Aus:  
„Anpassungen an Projektteil A, Anpassungen an Projektteil B, kleinere Arbeiten an Projektteil B.“  
wird:  
„Anpassungen an Projektteil A, Projektteil B etc.“

Zeitangaben

Bei den Zeitangaben kann nur ein Start und eine Endzeit angegeben werden. Wenn dazwischen Pausen oder andere Tätigkeiten gibt werden diese in Minuten aufsummiert und unter dem Punkt „Unterbrechung“ eingetragen. Beispiel:



Die Pausenzeit von 30min wird auf die Unterbrechung einberechnet. Das heißt:   
360 + 150 - 30 = 480

Remote Login

1. Am Laptop normal anmelden
2. Microsoft Teams lokal am Laptop starten
3. Bin-Control Webinterface starten  
   Anmelden und mit Mobiltelefon bestätigen  
   (Anmeldung wie lokal am Laptop)
4. Auf Bcon-DG-RemotePC-Access klicken  
   um sich mit dem Bürorechner zu verbinden.
5. MicrosoftTeams auf Bürorechner Bürorechner ausschalten  
   (Über den Laptop läuft Teams besser)
6. Oben am Rand kann man die Toolbar öffnen  
   Home = Laptopbildschirm  
   BCon…= remote PC
7. Bei Arbeitsende nur den Laptop herunterfahren. (Nicht den remote PC)

Notiz zum Firmen-CRM:   
Im CRM gibt es einen Bereich Kontakte. Dies kann man nutzen für den Fall das Kontakte nicht in Outlook zu finden sind.

Azure Umgebung  
  
Es gibt mehrere Möglichkeiten mit Azure zu interagieren z.B.:  
Azure-Portal – Die grafische Oberfläche  
PowerShell – Console über Eingabeaufforderung  
Azure CLI – Plattform unabhängiges Eingabetool  
  
Azure Marketplace und Dienste  
  
Der Azure Marketplace bietet verschiedene Dienste zum herunterladen von diensten an. Microsoft verteilt diese Dienste der Übersichtlichkeit wegen auf 21 Kategorien.

* AI + Machine learning  
  Dienste die künstliche Intelligenz anbieten z.B. Bots, Spracherkennung usw.
* Analytics  
  Ein Angebot aus Datenanalysetools
* Blockchain  
  Implementierung von Diensten wie Cryptowährung
* Compute  
  Angebot von virtuellen Maschinen und serverless Lösungen
* Container  
  Software die in Containern läuft und daher unabhängig vom Betriebsystem ist. Durch Container ist es möglich von einer Infrastruktur auf eine andere umzuziehen, da man nicht an das Betriebsystem der darunterliegenden virtuellen Maschine gebunden ist. Dienste die dies ermöglichen sind zB. Azure Container Instances, Azure Kubernetes Cluster oder Docker.
* Databases  
  Datenbankdienste wie Microsoft SQL und Plattformdienste wie MySQL oder PostgresSQL, sowie NoSQL Datenbanken wie Cosmos DB. Weiterhin gibt es komplexere Lösungen aus dem Big Data Bereich wie Azure Databricks oder Azure Synapse Analytics.
* Developer Tools  
  Dienste rund um die Entwicklung von Skripten, vorinstallierbare virtuelle Maschinen usw.
* Dev-Ops  
  Tools zur Softwareentwicklung und Wartungsprozessen (Unterstützung und Vereinfachung) Azure DevOps SaaS Lösungen bei der man sich nicht um die Infrastruktur kümmern muss.
* Identity  
  Lösungen zur Benutzerauthentifizierung z.B.: Azure Active Directory und Azure Active Driectory Domain Services.
* Integration  
  Lösung zum Transfer zwischen Aplikationen z.B. Azure Relay und Azure Event Hub
* Internet of Things  
  Lösungen für die Kommunikation externer Geräte
* IT und Management Tools  
  Lösungen zur Verwaltung von IT-Umgebungen z.B. Azure Log Analytics.
* Media  
  Lösungen für das Speichern, Codieren und streamen von Audio und Videoinhalten.
* Migration  
  Lösungen für die Migration von virtuellen Maschinen und Datenbanken aus der lokalen Umgebung nach Azure.
* Mixed Reality  
  Dienste für Microsoft Hololens
* Monitoring und Diagnostics  
  Dienste zur Überwachung der Azure Umgebung. Wie z.B. Container-Monitoring oder KeyVault Analytics.
* Networking  
  Virtuelle Netzwerkkomponenten für eine Azure Netzwerkinfrastruktur.
* Security  
  Sammlung von Speichermöglichkeiten. Ein wichtiges Element ist der Azure Storage Account und dessen vier Bereiche: Blob Storage, File Shares, Azure Tables und Queues.
* Web  
  Bereich für serverlose Anwendungen z.B.: Logic App, AppService Enviroment und Azure Content Delivery Service.
* Software as a Service  
  Saas Lösungen von Drittanbietern.

Sicherheitskonzepte in Azure

* Integrität  
  Vermeidung von unberechtigten Änderungen an Daten.
* Verfügbarkeit  
  Sicherstellen das Daten für berechtigte Personen verfügbar sind. Absichern von DDOS Attacken durch fluten eines Systems mit unsinnigen Anfragen)
* Vertraulichkeit  
  Zugriff auf Informationen nur für berechtigte Personen.

In Azure findet die Absicherung in mehreren Schichten statt:

1. Physische Sicherheit  
   Absicherung der Hardware an einem schwer zugänglichen Ort.
2. Identität und Zugriff  
   Wer hat zugriff auf die Infrastruktur?  
   Wie werden Änderungen an der Infrastruktur durchgeführt?
3. Perimeter  
   Filtert DDOS-Angriffe heraus bevor die Anfragen das Zielsystem erreichen.
4. Netzwerk  
   Begrenzen der Kommunikation der Systeme untereinander.
5. Compute  
   Die Compute-Schicht regelt den Zugriff auf virtuelle Maschinen-
6. Anwendungen  
   Absicherung der eigentlichen Anwendung
7. Daten  
   Das eigentliche Ziel von Angriffen befindet sich in der untersten Schicht.

Geteilte Verantwortlichkeiten für die Sicherheit

Die Verantwortlichkeiten teilen sich je nach gewählten Cloud-dienstmodell zwischen Firma und Kunden auf:

* On-Premises  
  Entspricht dem Mo0del private Cloud. Der Eigentümer ist für alles verantwortlich.
* IaaS  
  Microsoft sorgt für die Sicherheit aller physischer Hardwarekomponenten.
* PaaS  
  Zusätzlich zur physischen Sicherheit kann wenn gewünscht, Micrososft für die Sicherheit im Netzwerk, Anwendungen, Identität und Verzeichnisstruktur sorgen.
* SaaS  
  Wie bei PaaS nur das Microsoft, Netzwerk und Anwendungen sichert. Bei diesem Angebot hat man nicht die Wahl.

Azure Unterstützungsangebot  
  
Folgende Hilfsangebot gibt es in Azure

* Azure Knowledge Center  
  Das Azure Knwoledge Center ist eine Know How Datenbank zu den häufigsten Fragen rund um Azure.
* MSDN-Foren  
  Das Microsoft Developer Network Forum ist eine Anlaufstelle für Microsoftprodukte.
* Azure Feedback-Forum  
  Das Feedbackforum zu Azure über das man mit Microsoft in Kontakt treten kann. Es ist hauptsächlich dafür gedacht Bugs zu melden und Verbesserungsvorschläge einzureichen.
* Stack Overflow  
  Bekanntes Programmiererforum
* Server Fault  
  Ähnlich Stack Overflow allerdings für Systemadministratoren.
* Offizielles Azure Twitter Konto  
  <https://twitter.com/azuresupport>
* Azure Meetups  
  Azuretreffen (ähnlich einer Messe)
* Azure Q&A  
  Forum in dem Fragen eingestellt werden können.  
  <https://docs.microsoft.com/en-us/answers/products/azure?product=all>

Der Compliance Manager  
  
Tool im Service Trust Center das dabei hilft Aktivitäten die ein Unternehmen ausführen muss um Compliance-Anforderungen an die Azure, Office365 und Dynamics 365 Umgebung zu erfüllen, planen, koordinieren, zu überwachen usw.

Azure Security Center  
  
Das Azure Security Center deckt in der eigenen Infrastruktur Sicherheitsprobleme auf. Es gibt zwei Varianten:

* Die Kostenlose  
  Überprüfung beschränkt sich auf Azure Ressourcen
* Standard  
  Bietet sicherheitsbezogene Dienste für Monitoring, Erkennung von Bedrohungen, zeitgesteuerten Zugriff auf Ports usw.

Azure-Security-Komponenten für den Datenschutz  
  
Azure KeyVault

Der AzureKeyVault ist ein Dienst, der einen sicheren Tresor für geheime Daten wie Kennwörter darstellt. Ausserdem können geheime Daten bereitgestellt werden ohne das die Partei, die die Daten verwendet, Einsicht in die Daten erhält.

Azure Information Protection (AIP)  
AIP klassifiziert E-Mails durch Marker. Dies kann händisch oder automatisch von statten gehen.  
  
Azure Advanced Thread Protection (ATP)  
Sicherheitsdienst der dabei hilft Bedrohungen, Sicherheitsprobleme, kompromittierte Identitäten, schädliche Aktionen von Mitarbeitern und Risiken zu erkennen und zu untersuchen.  
  
Kosten

Die Grundlagen, um Kosten bei einer Cloud Kosten einzusparen ist es die kleinstmögliche Ressource für die eigene Umgebung zu verwenden und gegebenenfalls danach hoch zu skalieren. Außer dem sollten System die nicht benötigt werden, ausgeschaltet sein. Wartungsaufwand kann minimiert werden, indem PaaS genutzt wird. Zur Kostenersparnis ist auch ein Umdenken im vergleich zu früheren Strukturen notwendig. Anders als bei einer physischen Infrastruktur sind virtuelle Strukturen schon auf Redundanz ausgelegt. Aus diesem Grund kann zur Leistungserhöhung z.B. auch ein Raid 0 verwendet werden.

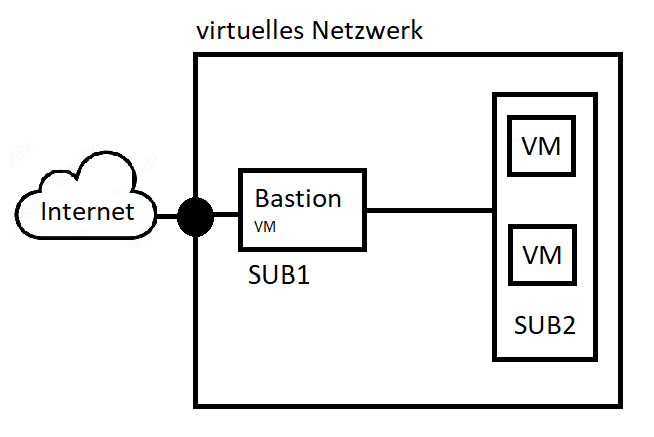
Bezugsmodelle für Azure  
  
Azure kann man sich auf folgenden Wegen beziehen:

* Über die entsprechende Internetseite von Microsoft.
* Über schon vorhandene Unternehmenslizenzen.
* Über einen Cloudserviceprovider (einem Partnerunternehmen von Microsoft) der das Thema als Dienstleistung übernimmt.
* Über ein MSDN Konto steht eine Trail-Version zur Verfügung.

Azure Preisrechner  
  
Über den Azure-Preisrechner kann man die zu erwartenden Kosten für die Abbildung der eigenen Infrastruktur in Azure vorab Informieren.  
<https://azure.microsoft.com/de-de/pricing/calculator>  
  
Im Einzelnen hängen die Kosten von folgenden Faktoren ab:

* Typ der Ressourcen
* Anzahl der VMs
* Zeit die die virtuellen Maschinen eingeschaltet sind.
* Zeit in der die IP-Adressen der virtuellen Maschinen belegt sind.
* eingehender Datenverkehr
* ausgehender Datenverkehr
* Lese/ Schreibeoperationen auf den verwalteten Festplatten
* Belegter Festplattenplatz
* Azure Region  
  Je nach Region auf der Welt können sich die Betriebskosten die Microsoft auf seine Kunden umlegt unterschiedlich sein.
* Bezugsmodel  
  Je nach Bezugsmodel (Internet, Enterprise oder Cloud Solution Provider) können die Kosten anders sein.

Azure Infrastruktur Dienste  
  
Der Bawstionhost  
  
Da es ein Sicherheitsrisiko ist die virtuellen Maschinen direkt über das Internet ansprechbar zu machen wird dazwischen ein Bastionshost verwendet. Dieser steht zwischen der virtuellen Infrastruktur und dem Internet. Die Kommunikation von und nach Draußen wird über den Bastionshost geregelt. Dabei muss der Bastionshost der auch eine virtuelle Maschine darstellt, sich in einem anderen Subnetz als der Rest des virtuellen Netzwerkes befinden:



Über den Bastionshost wird auf das Azure-Portal zugegriffen.  
  
Verfügbarkeit von virtuellen Maschinen

Die Ausfallsicherheit von vituellen Maschinen ist auch in der Cloud nicht standardmäßig gegeben. Für die virtuelle Maschinen muss dies erst eingestellt werden. Es gibt neben der Standardeinstellung (keine Infrastrukturredundanz) noch drei weitere Einstellungen.

* Verfügbarkeitszonen  
  Die VM wird in einer physisch separierten Zone innerhalb der Region ein weiteres Mal abgebildet.
* Verfügbarkeitsgruppen  
  Unter Verfügbarkeitsgruppen versteht man mindestens zwei VMs die sich in verschiedenen Fehler- und Updatedomänen befinden. Kommt es z.B. bei einem Update zu einem Fehler springt die andere VM ein.
* Multi-Region DR  
  Es wird eine weitere Verfügbarkeitszone aus einer anderen Region hinzugenommen. Fällt eine Region komplett aus übernimmt die Region/Verfügbarkeitszone die dazu genommen wurde.

Netzwerke, Firewalls und VPN

Virtuelle Netzwerke in Azure erfüllen folgende Aufgaben:

* Kommunikation virtueller Maschinen untereinander.
* Kommunikation mit dem Internet.
* Kommunikation mit lokalen Ressourcen
* Isolierung und Unterteilung von Netzwerken innerhalb eines Azure Abos. (In einem Azure Abo kann man mehrere voneinander unabhängige Netzwerke haben)  
  Ein virtuelles Netzwerk kann private Netzadressen nutzen.  
  Ein virtuelles Netzwerk kann in verschiedene Subnetze unterteilt werden und es kann bestimmt werden wie der Verkehr zwischen den Subnetzen fließen soll.

VPN-Gateways  
  
Wenn man lokale Netzwerke mit virtuellen Netzwerken in Azure verbinden möchte, benötigt man eine VPN-Gateway. Es erzeugt einen Endpunkt, der es ermöglicht über das Internet eine Verbindung aufzubauen. Die Kommunikation zwischen den beiden Endpunkten wird dabei verschlüsselt. Ein VPN-Gateway benötigt dabei dabei ein eigenes Subnetz das zwingend „Gateway Subnetz“ heißen muss. Über VPN-Gateway können drei unterschiedliche Verbindungen aufgebaut werden:

* Netzwerk zu Netzwerk  
  Kann auch genutzt werden um zwei virtuelle Netzwerke aus verschiedenen Regionen zu verbinden.
* Site-to-Site-Verbindung  
  Verbindungen zwischen externen Netzwerken und dem virtuellen Netzwerk.
* Point-to-Site  
  Benutzer meldet sich mit lokalen Rechner am virtuellen Netzwerk an.

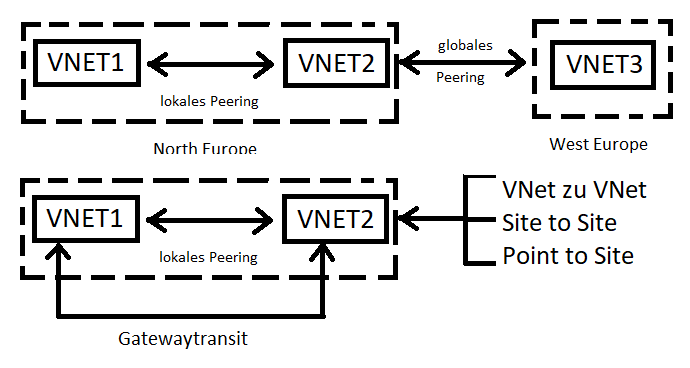
VPN-Typ  
  
Hier hat man die Wahl zwischen routenbasiert und richtlinienbasiert.

* Routenbasiert steuert den Netzwerkverkehr über Routingtabellen.
* Richtlinienbasiert steuert den Netzwerkverkehr über Richltinien. Dieser VPN-Typ kann nur für Site-to-Site Verbindungen genutzt werden.

Netzwerk-Peering

Wenn man mehrere virtuelle Netzwerke betreibt kann man diese mit Netzwerkpeering miteinander verbinden. Die Netzwerke werden nicht über das Internet sondern über das Azure-Backbone-Netzwerk verbunden. Es gibt zwei Arten von Netzwerkverbindungen über Netzwerkpeering:

* Globales Peering  
  Beim globalen Peering werden zwei Azure-Netzwerke miteinander verbunden, die sich in unterschiedlichen Regionen befinden.
* Lokales Peering  
  Es werden zwei Netzwerke in der selben Region miteinander verbunden.



Über ein Gateway lässt sich einstellen, ob eine/ die Ressource(n) eines anderen Netzwerk verbundene virtuellen Netzwerks das Gateway des anderen nutzen darf oder nicht. Im obigen Beispiel ist in VNET2 eingestellt das man mit Ressourcen aus VNet1 kommunizieren kann die an VNet2 per Gateway (Site-to-Site) hängen.

Netzwerksicherheitsgruppen (NSG)

NSG’s können einem einzelnen Netzwerkadapterwerk, einer virtuellen Maschine (NIC), oder einem ganzen Subnetz zugeordnet werden. Ist eine NSG dem Subnetz zugeordnet und gleichzeitig eine NSG einer virtuellen Maschine werden beide NSG’s getrennt ausgewertet. NSG Sicherheitsregeln bestehen aus folgenden Elementen:

* Name  
  Dient zur Identifikation der Sicherheitsregeln.
* Priorität  
  beeinflusst die Reihenfolge in der die Sicherheitsregeln bearbeitet werden (100-4096)
* Quelle/ Ziel – Ziel des Netzwerkverkehrs z.B.: Any, CIDR-Block, Dienst-Tag oder eine App-Sicherheitsgruppe.
* Protokoll  
  TCP, UDP oder Any (Netzwerkprotokoll)
* Portbereich  
  einzelner Port oder Portbereich
* Aktion

Datenverkehr erlaubt/ verboten

* AllowVnetInbound  
  Zulassen von eingehenden Verkehr im selben Subnetz.
* AllowAzureLoadBalancerInbound  
  lässt eingehenden Verkehr von Standardlastausgleich zu
* DenyAllInbound  
  Verbietet eingehenden Netzwerkverkehr. Diese Regel wird von der NSG immer zuletzt ausgewertet.
* AllowVnetOutbound  
  Diese Regel lässt ausgehenden Verkehr an alle VM’s im selben Subnetz zu, in dem sich die VM befindet, dem die NSG zugewiesen ist.
* AllowInternetOutbound  
  Die Regel lässt ausgehenden Internetverkehr zu.
* DenyAllOutbound  
  Diese Regel verbietet den ausgehenden Netzwerkverkehr komplett. (Wird als letztes Angewand.)

Dienst-Tag  
  
Bei Verwendung von NSG’s kann man Dienst-Tags angeben. Ein Dienst-Tag ist ein von Microsoft angelegtes Tag um bestimmte Netzwerkbereiche anzugeben. Wenn man mit Dienst Tags arbeitet muss man die expliziten IP-Adressen der Quelle oder des Ziels nicht kennen:

* Internet  
  Netzwerkverkehr der aus dem Internet kommt.
* Virtualnetwork  
  Netzwerkadapter aus virtuellen Netzwerken und über das Gateway eventuell verbundene Netzwerke (lokal)
* AzureLoadBalancer  
  Verkehr von Azure Load Balancer
* AzureTrafficManager  
  Verkehr von AzureTrafficManager
* Storage  
  Begrenzen des Verkehr der von Azure Storage kommt.
* SQL  
  Verkehr von SQL-Plattformen
* App Service  
  Verkehr von AppServices

Loadbalancer

Der Loadbalancer sorgt für lastenausgleich in dem dieser Daten über mehrere Systeme verteilt. Ein Loadbalancer kann dem Netzwerk vorgeschaltet sein und über eine externe IP verfügen oder sich im Netzwerk befinden mit einer internen IP.

VNet-Dienstpunkte

Anstatt aus dem virtuellen Netzwerk über das öffentliche Netz einen Plattformdienst anzusprechen kann man stattdessen Dienstpunkte einrichten. Diese Dienstpunkte verhalten sich so als wären sie teil des virtuellen Netzwerkes und sorgen dafür das die Kommunikation zum Plattformdiensten über das Azure-Backbone läuft anstatt über das öffentliche Internet. Azure-Plattform Dienste verfügen über öffentliche IP-Adressen. Diese Endpunkte werdem dem Loadbalancer bekannt gegeben. Die wichtigsten Endpunkte sind:

* Azure SQL Database
* Azure Cosmos DB
* Azure Service Bus
* Azure Key Vault
* Azure Data Lake
* Azure Storage

Azure Firewall

Die Asure Firewall ist mit zustandsbehaftete Firewall (Statefull inspection firewall) die Pakete analysiert bzw. den kompletten Kontext einer Netzwerkverbindung. In den Firewall können folgende Funktionen eingestellt werden:

* NAT-Regeln  
  Regel zur Adressübersetzung mit deren Hilfe eingehende Ziel-IP-Adressen und Zielports definiert werden können.
* Anwendungsregeln  
  Anwendungsregeln definieren vollqualifizierte Domänennamen auf die von einem Subnetz aus zugegriffen werden kann.
* Netzwerkregeln  
  Netzwerkregeln definieren Quelladresse, Protokoll, Zielport, Zieladresse

Application Gateway

Application Gateways verwalten Anforderungen die an eine Webanwendung gesendet werden und leitet diese an eine oder mehrere Webserver weiter. Das Weiterleiten bzw. Routing kann auf zwei unterschiedliche Weisen stattfinden:

* Hosten mehrerer Websites  
  Mehrere Webanwendungen über die gleiche Applikation-Gateway-Instanz. DNS-Namen werden an den IPs des Gateways registriert. So wird der Netzwerkverkehr an den richtigen Webserver geleitet. (Reverse Proxy) Für jeden DNS-Namen können Regeln festgelegt werden.
* Pfadbasiertes Routing  
  Beim pfadbasierten Routing kann man Webaufforderungen auf Grund der Pfade innerhalb der URL an unterschiedliche Backendserver gesendet werden.

Storage

Daten können in vier Kategorien eingeteilt werden:

* Strukturierte Daten  
  Strukturierte Daten oder auch relationale Daten. Diese Daten werden oft in Tabellen (Relationen) abgelegt. Die Daten stehen dabei in Beziehung (Relation) zueinander.
* Semi-strukturierte-Daten  
  Semistrukturierte Daten haben keine übergeordnete Struktur sondern bringen ein eigenes Schema mit. Die Daten haben einen internen Aufbau. Als Beispiel kann man hier Dateiformate wie XML, Json, Yaml usw. anführen Hier wird den Daten durch Tags und Klammern eine Struktur gegeben. Semistrukturierte Daten können in Dateisystemen oder NoSQL Datenbanken abgelegt werden.
* Unstrukturierte Daten  
  Unstrukturierte Daten besitzen keine klar vorgegebene Struktur. Beispiele dafür sind Bilder, Videos, Audioaufnahmen oder Office Dokumente
* Datenströme  
  Datenströme bestehen aus semistrukturierten Daten oder unstrukturierten Daten die kontinuierlich von Geräten wie Temperatursensoren gesendet werden.

Azure – Speicherkonten

Die vier Datensorten können in einem Azure-Speicherkonto abgelegt werden. Das Azure-Speicherkonto bietet dafür vier grundlegende Dienste an:

* Azure BLOB Storage  
  BLOB-Storage speichert große Binärdateien daher auch der Name „Binary Large Object“
* Azure Files  
  Hier kann man wie bei einer lokalen Netzwerkfreigabe auf einer Azure Dateifreigabe speichern.
* Azure Tabellen  
  Azure Tabellen sind ein einfacher NOSQL-Speicher. Azure-Tabellen sind dort gut einsetzbar wo man nicht gleich eine ganze Datenbank braucht und nur eine kleine Menge Daten speichern möchte.
* Azure-Warteschlangen  
  Azure-Warteschlangen (Queues) werden verwendet um Nachrichten zu speichern. Anwendungen können diese Daten wieder auslesen. Mit diesem Konstrukt kann man verhindern das empfangene Anwendung von Nachrichten überschwemmt wird.

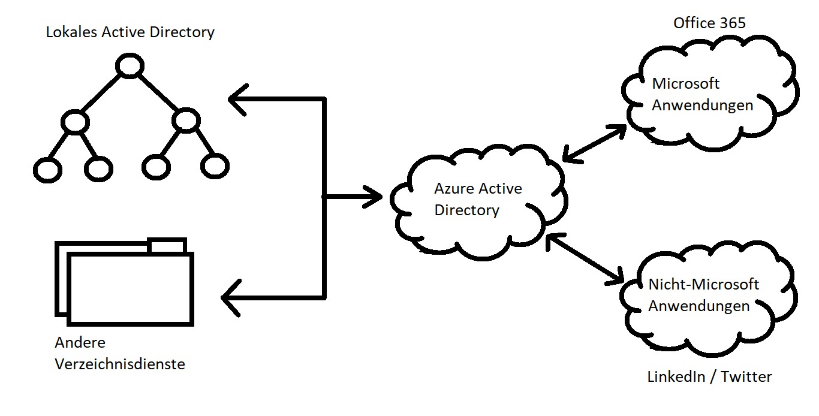
Azure Active Directory

Azure Active Directory ist ein cloudbasiertes Verzeichnis und Identitätsverwaltungsdienst. Es werden alle Zugriffe innerhalb von Azure gesteuert, um den Zugriff auf Ressourcen zu schützen. Azure Active Directory ist kein „echte“ Active Directory bzw. es verfügt nicht über alle funktionen eines herkömmlichen Active Directory (Es gibt bei AAD z.B. keine OU’s). AD und AAD können sich allerdings sich allerdings synchronisieren. Zum Authentifizieren kann man somit AAD nutzen. Azure Active Directory verfügt damit über folgende Funktionen:

* Single Sign-On  
  Unter Single Sign-On versteht man das man sich nur einmal Anmelden muss. Die Anmeldung wird an andere Systeme weitergeleitet. Meldet man sich also an seinem System an kann man die Software nutzen, ohne sich dort nochmal zu authentifizieren.
* Schutz von vertraulichen Daten und Anwendungen  
  Abgleich von Benutzerkonten mit dem Darknet und Meldung verdächtiger Vorgänge.
* Lokales Active Directory in die Cloud erweitern. Dies gibt Anwendern die Möglichkeit sich sowohl im lokalen als auch im virtuellen Netzwerk anzumelden.
* Einbindung mobiler Geräte. (Funktioniert mit Windows, macOs, IOS und Android)
* Schutz von lokalen Webanwendungen (Nutzen von AAD für die eigene Webanwendung)
* Multifaktor-Authentifizierung  
  Darunter versteht man die Anmeldung die aus verschiedenen Faktoren besteht:  
  Etwas das man weiß: Benutzername oder PIN  
  Etwas das man hat: Smartphone oder SmartCard  
  Etwas das man ist: Biometrische Erkennungsmethoden
* Self-Service Funktionen  
  Die Möglichkeit das Benutzer ihr eigenes Passwort resetten können.
* Registrierung von Geräten  
  Registrierung von mobilen Endgeräten

Es gibt Rollen, die für alle Ressourcen im AD angelegt werden:

* Besitzer  
  Dem Besitzer „gehört“ die Ressource. Er hat vollzugriff und kann anderen Zugriff gewähren.
* Mitwirker  
  Der Mitwirker kann alles was der Besitzer kann nur das er niemandem Zugriff auf die Ressource erteilen kann.
* Leser (nur Leserechte)
* Der Benutzergruppenadmin kann Zugriff auf Ressourcen verwalten.



Datenbanken

1. Ressourcengruppe erstellen
2. Server für die Datenbank anlegen  
   Da es sich um einen Plattformdienst handelt muss man sich nicht um die Wartung kümmern. Man muss ihn aber trotzdem anlegen.  
   (Hier muss eine SKU angelegt werden. Die Stock Keeping Unit gibt an wie groß der Server ist. (Kosten)
3. Auf dem Server kann danach die Datenbank angelegt werden.

Notiz: PostgreSQL ist ein Objektrelationales Datenbanksystem hier werden Daten in Objekten gehalten anstatt in Tabellen. (Wie es bei relationalen Datenbanken der Fall ist.)

Nicht relationale Datenbanken

Nicht relationale Datenbanken auch NoSQL Datenbanken genannt. Es steht für Not only SQL. Das heißt man kann Daten mit einem SQL ähnlichen Syntax abfragen. Die Daten folgen jedoch keinem festen Schema. Beispiel für NoSQL Datenbanken: Redis Cache oder Cosmos DB.

1. Ressourcengruppe anlegen
2. DB-Konto anlegen  
   Das Konto ist vergleichbar mit einer Datenbankserver. Es ist als Verzeichnis zu sehen unter den Daten/ Dateien abgelegt werden.
3. Datenbank anlegen
4. Container anlegen

Web-Apps und API’s

Anders als bei der Desktopprogrammierung in der ein großes Programm alle Funktionen enthält werden bei Cloudanwendungen, Funktionen als Dienste simultan ausgeführt. Ist die Aufteilung/ Größe dieser Funktion sehr kleinteilig spricht man dabei von Microservices.

Eine Web-App wird aus einem App-Service-Plan herraus gestratet. Der Serviceplan legt fest wann und wie viel Ressourcen für die Ausführung benötigt werden.

1. App-Service-Plan erstellen
2. Web-App erstellen
3. Berechtigungen erstellen

Web-Apps werden dabei wie Benutzer behandelt und bekommen ein Konto im AD um sich zu authentifizieren. Weiterhin bietet sich die Azure Key Vault an um eine App authentifizieren da so weder Benutzernamen noch Passwort benötigt werden. Für Web-Apps können Skallierungen (Ober- und Untergrenze) und Zeitpläne eingestellt werden.

Serverless Computing in Azure  
  
Serverless Web-Apps besitzen keinen App-Service-Plan. Der Cloudanbieter skalliert die App an die Anforderung.

1. Ressourcengruppe anlegen
2. Speicherkonto anlegen (Speicher für App)
3. Function-App anlegen

Neben App’s können auch SQL-Datenbanken „serverlos“ erstellt werden.

Azure Data Factory

Die Azure Data Factory steuert Daten von einem Service zu einem anderen und kann Daten dabei anpassen/verändern. Ein Beispiel wäre zum Beispiel CSV-Datei wird in Daten für eine SQL-Datenbank umgewandelt und übertragen. Die Azure Data Factory kann daher mehrere dieser Prozesse abwickeln. Mehrere Prozesse werden unter einer Pipeline zusammengefasst.

1. Ressourcengruppe anlegen
2. Datafactory anlegen

Pipeline einrichten

1. Speicherkonto anlegen
2. Datenbank anlegen
3. Datenbank dem Server hinzufügen
4. Firewallregeln festlegen um anderen Anwendungen die Erlaubnis zu erteilen auf die Datenbank zuzugreifen
5. Firewallregeln festlegen um von PC auf die Datenbank zugreifen zu können.
6. Schlüssel besorgen um Daten von der Datenbank in das Speicherkonto kopieren zu können.
7. Schlüssel verwenden um einen Container im Speicherkonto anzulegen.
8. Kopiervorgang starten

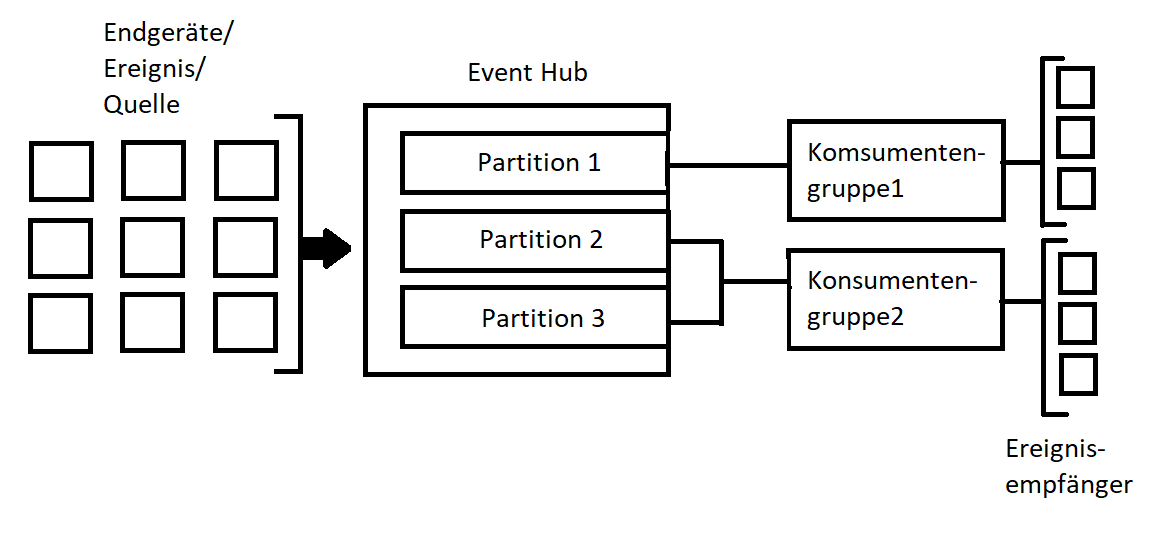
Container, Registries und Kubernetes

In Containern können Ressourcen wie Anwendungen gepackt werden. Container sind unabhängig von den darunterliegenden virtuellen Maschinen. Beim Wechsel des Cloudanbieters kann einfach der Container auf eine virtuelle Maschine des neuen Anbieters übertragen werden. Der Inhalt des Containers ist dann im Prinzip sofortwieder einsatzbereit. Container werden immer aus einem Image erzeugt. Es können beliebig viele Container aus einem Image instanziiert werden. In den Images liegen die eigentlichen Anwendungen.

Der Container-Host ist der Rechner, auf dem der Container liegt. Die Container-Engine ist die Software die Container auf dem Container-Host bereit stellt. Das Container-Netzwerk ist das Netzwerk mit dem Container miteinander und mit der Außenwelt kommunizieren. Auf einen Container-Hostkann es mehrere Containernetzwerke geben. Durch die Container-Regestry werden Images bereitgestellt und können von dort bezogen werden. Unter Azure kann man seine eigene Regestry mit der Azure Container Regestry erzeugen.  
  
Cluster und Kubernetes  
  
Funktionen können in Containern laufen. Der Vorteil ist das Container an eine virtuelle Maschine andocken und damit unabhängig von der zugrundeliegende Infrastruktur bzw. der virtuellen Maschine sind. In einem komplexen System können die Inhalte der Container miteinander kommunizieren. Es fallen bei der verwendung von Containern allerdings Aufgaben wie Skalierung, Verfügbarkeit, Fehlertoleranz und die sichere kommunikation untereinander an. Aus diesem Grund gibt es verschiedene Anbieter die das Container management ermöglichen. Einer dieser Dienste ist Kubernetes.

Azure Kubernetes Service

Kubernetes ist open Source und ordnet Container unter Knotes (Nodes) ein. Diese Nodes sind mit einem Hauptknoten, dem Control-Pane verbunden. Die Container eines Knoten laufen in sogenannten Pods. Für die Nodes muss eingestellt werden wieviele Container in einem Pod laufen (meistens einer pro Pod) und wieviele Pods einem Node zugeordnet sind. Ausserdem kann in jedem Node die Anzahl der Replikas eingestellt werden die für die Ausfallsicherheit sorgen. Bei einem Ausfall sorgt die Control-Pane das aus einem Replika-Set ein Ersatz gefunden bzw. genommen wird. Da das Einstellen eines Kubernates-Cluster viel Zeit in Anspruch nimmt gibt es Dienste wie AKS (Azure Kubernetes Service) der diese Aufgaben erleichtert.

IOT und Datenströme  
  
Bei IOT (Internet of things) werden Datenströme verarbeitet die von mobilen Endgeräten erzeugt werden. Über den Azure IOT-Hub werden Endgeräte registriert und erhalten eine Zeichenfolge über die diese Geräte eine sichere Kommunikation von und zur Cloud erhalten. Beim einrichten eines Azure IoT-Hubs muss man die SKU festlegen. Also wieviel Datenverkehr pro Tag und Einheit man erwartet. Je nach Umfang entstehen hier unterschiedliche Kosten.  
  
Azure Event HUB  
  
Der Azure Event Hub ist eine eingeschränkte Version des IoT-Hubs. Der Azure Event HUB spart allerdings den zusätzlichen Aufwand einer Geräteregistrierung.  
  


* Ereignisse werden in den Partitionen verarbeitet um zu vermeiden das Gleichzeitig eintreffende Ereignisse sich gegenseitig blockieren.
* Konsumentengruppe sind die Einheiten auf denen die Empfänger zugreifen. Sie ermöglichen Empfängern mit unterschiedlichen Übertragungsraten, Daten ohne Konflikte auszulesen.
* Die Datendurchsatzeinheit ist die zugesicherte Kapazität zur Verarbeitung von Ereignissen. Der Datendurchsatz ist damit eine Möglichkeit zur skallierung des Event Hubs.

Azure Service Bus  
  
Der Azure Service Bus bietet Lösungen die über die Fähigkeiten des Azure Event Hubs hinaus gehen. So können nicht zustellbare Nachrichten anders behandelt werden bze. Probleme gesondert behandelt werden.

Azure Storage Queues  
  
Bei Azure Storage Queues handelt es sich um eine Warteschlangentechnik mit geringeren Funktionsumfang die als einfache Methode gedacht ist Nachrichten innerhalb eines Service auszutauschen.

Cognitive Services  
  
Bei Cognitive Services handelt es sich um eine Sammlung an Diensten die Daten auswerten und Ergebnisse liefern. Beispiele sind hierfür Spracherkennung, Bilderkennung und Suchfunktionen.

Azure Synapse Analyse  
  
Azure Synapse Analytics ist eine zentrale Datenanalyse Plattform.

Azure Databricks

Webbasierte Lösung um mit Spark-Cllustern zu arbeiten und diese zu verwalten. Spark-Cluster verteilen größere Datenmengen auf einem Servercluster und analysieren diese anschließend. Dabei wird auf eine Technologie gesetzt die als „Notebooks“ bezeichnet wird. Notebooks fassen Dokumente, Dokumentationen und Programmierung sowie die Resultate der Programmierung zusammen.

Azure Bot Service  
  
Das Azure Bot-Framework hilft dabei Bots zu entwickeln. Diese Bots erlauben es ohne menschliches zutun in Chats, Informationen bereit zu stellen. Über den Bot-Framework-Composer lassen sich sogar ohne Programmieraufwand Bots zusammen stellen.

Machine learning in Azure  
  
Azure Machine learning ist ein Clouddienst zum verwalten von Machine Learning Projekten.  
  
Azure DevOps  
  
DevOps ist ein Kunstwort aus Entwicklung (Development) und Systemadministration (IT-Operations). DevOps soll für eine engere Zusammenarbeit zwischen den Abteilungen sorgen. Hier sollen Probleme vermieden werden wie zum Beispiel das Software die auf Entwicklerrechnern läuft in der Produktionsumgebung nicht funktioniert, da Hard- oder Softwarevoraussetzungen nicht erfüllt sind. Das Ziel von DevOps ist es die Auslieferzeiten der Software möglichst kurz zu halten. So werden Änderungen am Programmcode vom Versionskontrollsystem automatisch erkannt und ein neuer Buildwird automatisch erzeugt. Dies wird Continous Integration (CI) genannt. Das Ergebnis ist eine installierbare Software die verteilt werden kann. Das automatische Verteilen der Software wird Continous-Delivery (CD) genannt.  
  
Azure DevOps ist eine integrierte Webanwendung für Softwareprojekte, um diese zu planen, die Dateien in eine Versionskontrolle abzulegen und mit Werkzeugen diese kontinuierlich zu „bauen“ (CI) und zu verteilen (CD). Azure DevOps besteht dabei aus fünf wesentlichen Komponenten Bords, Repos, Pipelines, Test Plans und Artifacts.

* Bords  
  Bords dienen zu planen des Projektes und dassen einzelnen Arbeitsschritten (Work-Items)
* Repos  
  Repositories erlauben das Speichern der Projektdateien. Diese werden dann von der Versionskontrolle verwaltet.
* Pipelines  
  Pipelines dienen zum automatischen erzeugen, testen und verteilen der Projektartefakte. (Artefakte werden in DevOps die Arbeitsergebnisse genannt)
* Test Plans  
  Neben der Möglichkeit Unit-Tests und automatisierten Tests die über die Pipeline eingebaut werden, können über Testpläne weitere Testtechniken, Performancetests und Feedback eingebaut werden.
* Artifacts  
  Über Azure Artifacts können Maven-Pakete, NPM-Pakete, Nuget-Pakete in Azure DevOps abgelegt werden um diese in den Pipelines zu verwenden.

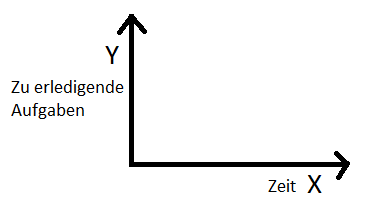
Scrum

Das Scrum-Vorgehensmodel ist ein agiles Model das transparent für den Kunden ist und abgesehen von Zeitvorgaben und Zielen dem Entwicklerteam eigenverantwortliches arbeiten ermöglicht. Ein Scrum-Team besteht aus drei Rollen. Dem Productowner, dem Entwicklerteam und dem Scrummaster. Der Kunde wird als Stakeholder bezeichnet.

* Product-Owner  
  Stellt das Product-Backlog mit der Liste der Kundenanforderungen zusammen. Die Formulierungen werden auch als User-Storys bezeichnet und beschreiben anforderungen in kurzen Sätzen (meist zwei bis drei Sätze). Das Muster dabei ist wie folgt: Als „wer“ möchte ich „was“ und „warum“
* Entwicklerteam  
  Das Entwicklerteam ist für die technische Umsetzung verantwortlich. Dafür werden aus dem Product Backlog die Ziele mit der höchsten Priorität ausgewählt. Diese Ziele werden in einem Zeitraum von zwei bis vier Wochen umgesetzt. Diesen Umsetzungszyklus nennt man in Scrum einen Sprint. In einem Sprint wird nur ein Teil des Produktes umgesetzt. Dieser Teil wird als Inkrement bezeichnet.
* Scrum-Master  
  Der Scrum-Master hilft dem Team das Vorgehen mit Scrum richtig umzusetzen.

Sprint-Planungsmeeting

Im Sprint-Planungmeeting wählen Product-Owner und Entwicklerteam, Einträge aus dem Product Backlog aus. Diese Einträge sollten während des Sprints umsetzbar sein. Die gewählten Einträge nennt man PBI’s (Product Backlog Items). Der Product-Owner legt die Reihenfolge der PBI’s fest und das Entwicklerteam die Anzahl der PBI’s die in Sprint fertiggestellt werden können. Das Entwicklerteam trägt die ausgewählten PBI’s in das Sprint Backlog ein und unterteilt die Aufgaben in detailliert Teilaufgaben. Diese Teilaufgaben werden dann Teammitglieder zugeordnet. In solchen Meetings kommt es häufig vor das das Backlog weiter wächst, da Anforderungen zu grob waren oder fehlten. So wird das Produkt Backlog während des Projektverlaufs immer detaillierter.  
  
Jeden Tag wird ein Daily Scrum durchgeführt. Hier wird besprochen wie weit ein Sprint fortgeschritten ist. Der Fortschritt wird auf dem Taskboard (Das alle PBI’s anzeigt und deren Fortschritt) festgehalten. Das Sprint-Burndown-chart zeigt die noch zu erledigenden Aufgaben in Relation zur verbleibenden Zeit an. Auf diese Weise erkennt man ob man in der Zeit liegt.



Sprint-Review  
  
Im Sprint-Review wird Feedback vom Kunden geholt. Dies kann zu weiteren Product-Backlog Einträgen führen.  
  
Sprint Retrospektive Meeting  
  
Im Sprint-Retrospektiv Meeting (Kunde nicht anwesend) wird ermittelt welche Probleme es gab, was positiv und was negativ war. Diese Erkenntnisse werden dann auf den nächsten Sprint angewendet.

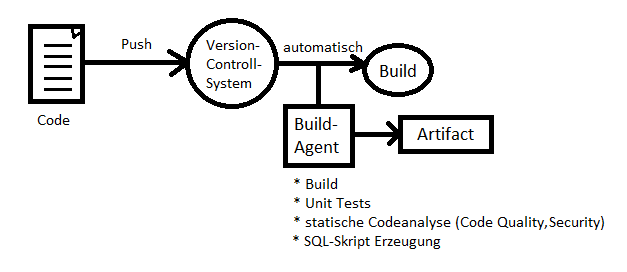
Prioritäten von Workitems

* Priorität 1  
  Zwingend erforderliche Arbeiten die sobald wie möglich umgesetzt werden müssen.
* Priorität 2  
  Wichtiges Ziel das aber nicht sofort umgesetzt werden muss.
* Priorität 3  
  Ziele die je nach Ressourcen und Zeit umgesetzt werden. Diese Ziele sind optional.
* Priorität 4  
  Arbeiten die nicht zwingend umgesetzt werden müssen.

Azure Pipelines CI (Continous Integration)  
  
Ein Problem bei der Entwicklung von Software ist das während der Entwicklung Bibliotheken und Konfigurationen erstellt werden die nur lokal auf dem Rechner des entwicklers zu finden sind. Um dies zu vermeiden ist es wichtig die Software zentral zu „bauen“ und zum testen bereit zu stellen.  
  
Wird ein neues Feature in die Software eingebaut und diese in das zentrale Repository übernommen kann automatisch ein Buildvorgang auf einem zentralen Rechner gestartet werden.  
  
Build-Agent  
  
Der Rechner der die Software „baut“ wird als Build-Agent bezeichnet. Der Build Agent ist dabei eine virtuelle Maschine die dynamisch beim starten des Build-Vorgangs aus einem Image heraus erzeugt wird. Der Buildvorgang wird dabei durch eine Folge von Jobs abgebildet die parallel laufen können. Jeder Job wiederum besteht aus Tasks die die eigentlichen Aufgaben erledigt. Beispielsweise:

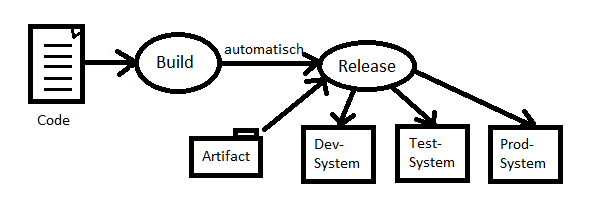
* Installation von benötigter Software
* Installation von benötigten Bibliotheken
* Builds (dot.NET, Maven usw.)
* Unit-Tests (NUnit, MSTest, XUnit usw.)
* Bereitstellen der Software

Das Ergebnis das ein Buildvorgang erzeugt wird in Azure DevOps (um Software zu testen) als Artifact bezeichnet.  
  
In einer Pipeline befinden sich die Jobs und deren Tasks. Die Pipeline wird dabei zum Konfigurieren des Buildvorgangs genutzt. (z.B. Nach wievielen Commits ein neuer Build erzeugt wird)



Hinweis: In der Build Pipeline gibt es Tasks die es erlauben PowerShell Skripte auszuführen. Weiterhin gibt es Tasks zum Erzeugen von dot.NET Core Apps.

Azure Pipelines – Release (CD-Continous-Delivery)  
  
Continous-Delivery (CD) dient dazu, die Auslieferung der Software (Deployment) zu automatisieren und diese qualitativ zu verbessern. Die Artifacts können dabei automatisiert auf Entwicklungs-, Test- und Produktivsystemen installiert werden. In Azure DevOps wird Continous-Delivery durch Release-Pipelines umgesetzt.



PowerShell  
  
Die PowerShell ist eine Automatisierungssprache deren Grundbausteine aus den folgenden Bereichen bestehen:

Cmdlets (Commandlets)

Cmdlets setzen sich aus einem Verb und einem Nomen zusammen .z.B.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Das Nomen beschreibt das Objekt das manipuliert werden soll und das voran gestellte Verb beschreibt was mit dem Objekt detan wird. Cmdlets sind oft Teil einer Familie:  
New - Service  
Remove - Service  
Set - Service  
 u.s.w.  
  
Cmdlets sind ausschließlich innerhalb der PowerShell lauffähig.

* Anwendungen (Applications)  
  Anwendungen sind auch außerhalb der PowerShell lauffähig. Anders als Cmdlets folgen sie keinen festen Namenskonventionen. Abgesehen davon sind diese Plattform abhängig. Man sollte immer ein Cmdlets einer Anwendung vorziehen. Die Ausgabe von Anwendungen ist zudem nicht standardisiert und kann alles von nichts über wenig bis unorganisiert sein.
* .Net-Befehle (Methoden)  
  Wenn es kein Cmdlet gibt das die Anforderungen erfüllt kann man zu Methoden greifen. Methoden sind kleinteilige Funktionen die aus den Bibliotheken des .Net Frameworks stammen. (auf dem die PowerShell basiert)
* Operatoren  
  Anders als in anderen Programmiersprachen basieren Operatoren hier auf Wörtern aus bis zu drei Buchstaben mit einem führenden Bindestrich.
* Literale

|  |  |
| --- | --- |
| $Variablenname | Das Dollarzeichen markiert den Anfang einer Variable |
| `r`n | Zeilenumbruch erzwingen |
| `…` | Standard Anführungszeilen wenn nur Text ausgegeben werden soll |
| “…“ | Wird verwendet wenn der Text z.B. Variablen enthält |
| [\\...\\](file:///\\...\\) oder ““…““ | Will man Anführungszeichen entwerten, nutzt man innerhalb eines Textes zwei Zeichen nebeneinander. |
| @“  Text  Text  Text “@ | Für längere Texte kann man dies nutzen anstatt der normalen Anführungszeichen |

Anmerkung: Zahlen mit denen gerechnet werden soll nutzen Punkt anstatt dem Komma. Datumsangaben sollten im ISO-Format angegeben werden:  
yyyy-MM-dd HH:mm:ss

Durch diese Konvention bleibt das Ergebnis von Datumsanfragen immer konsistent.

Kommentare  
  
# Dieser Text ist ein Kommentar  
  
<#  
 Das geht auch  
 mit mehreren  
 Zeilen.  
#>

Befehle suchen

Für die Suche nach Befehlen kann man mit Get-Command eine Suche „zusammenbauen“. Beispiele:  
  
Get - Command - Noun Archive  
Get - Command - Verb Expand  
Get - Command - Verb Expand -Noun \*Disc\*  
  
Will man das Modul finden aus dem ein Cmdlet stammt kann man folgendes machen:  
Get-Command -Module NetTCPIP  
Cmdlets haben 100 Standardverben die vor einen Namen gesetzt werden können. Diese Verben unterliegen einer Namenskonvention.  
  
Cmdlet Elemente:

* Parameter haben einen Namen und ein vorangestellten Bindestrich. Nach dem Namen folgt ein Wert:  
  
* Argumente sind Werte die den Parametern zugeordnet werden (siehe oben)
* Switch-Parameter sind Schalter die wie Parameter aussehen denen jedoch kein Argument zugewiesen ist. Sie können entweder an sein (dem Cmdlet angehangen) oder aus (dem Cmdlet nicht angehangen)  
  Beispiel: -Schalter  
  Anwendungen können auch Schalter und Parameter haben. Über Get-Command kann man nach Anwendungen suchen auch wenn es wegen der fehlenden fehlenden Namenskonventionen schwerer ist.  
    
  .Net-Methoden  
    
  Methoden verwenden:  
  Aufbau: [Klasse]::Methode  
  Beispiel: [Math]::Round(12,89)  
   [Convert]::ToString(12345)  
  PowerShell Operatoren

|  |  |
| --- | --- |
| -eq | gleich |
| -lt | Kleiner als |
| -le | Kleiner oder gleich |
| -gt | größer als |
| -ge | größer oder gleich |
| -ne | ungleich |

“PowerShell“ -like “Pow\*“  
Ist wahr (True) da Pow ein Substring von „PowerShell“ ist  
  
$w = “Dezember“,“Januar“,“Februar“  
$w -contains “Februar“  
Ergebnis ist wahr (True) da Februar im Feld (Collection) $w vorkommt.

“PowerShell“ -match “owner“  
“owner“ kommt im String “PowerShell“ nicht vor. Anders als like benötigt -match kein Wildcard(\*). Match kann auch Regexausdrücke vergleichen.  
  
Befehle verbinden  
  
Das Ergebnis eines Befehls kann mit einem Pipe Symbol zum nächsten Befehl weitergeleitet werden: Get-Date | Out-GridViews  
Hier wird das Datum geholt und das Ergebnis an Out-GridView weitergeleitet. Das Datum wird dadurch in einem grafischen Fenster angezeigt.

Das Ergebnis kann auch in einer Variable gespeichert werden: $datum=Get-Date  
  
Das Ergebnis wird direkt in der Konsole angezeigt: Get-date  
Allgemeine Notiz: Wird das Ergebnis nicht anderweitig verwertet wird das Ergebnis in der Konsole angezeigt.

$\_ steht für einen aktuellen Wert in der Pipeline. Wird z.B. ein Feld über die Pipe an den nächsten Befehl übergeben ist in $\_ das aktuelle Item des Feldes enthalten:  
1,2,3 | ForEach-Object { Write-Host $\_ }  
  
Skript-Ausführung aktivieren  
  
Mit Get-ExecutionPolicy kann man prüfen ob die Ausführung von Skripten am Computer erlaubt ist. Wird “Restricted“ ausgegeben.  
Ist dies nicht der Fall, muss man folgendes zur Aktivierung der Skriptausführung ausführen:  
Set-ExecutionPolicy -Scope CurrentUser -ExecutionPolicy RemoteSigned -Force  
  
Skript und Funktionen  
  
Um Skripte aus der PowerShell zu starten kann man folgendes eingeben:  
PS> C:\skripte\testskript.ps  
Sollte der Pfadname Sonderzeichen enthalten muss der Aufruf durch Ausrufezeichen in Text umgewandelt werden und mit Hilfe des Call-Operators (&) ausgeführt werden:  
PS> &‘c:\skripte\verschiedene skripte\test mit benutzereingabe.ps1‘  
  
**Mit:** $env:path += ‚;c:\skripte‘  
Kann man solange die Powershell Session läuft den Pfad zu den eigenen Skripten zu den Umgebungsvariablen hinzufügen. Nur durch die Angabe des Skriptnamens kann das Skript gestartet werden.

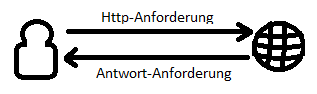
Skript können auch vom ausserhalb der Powershell gestartet werden:  
powershell.exe -File „c:\pfad\skript.ps1“  
  
weitere Parameter:

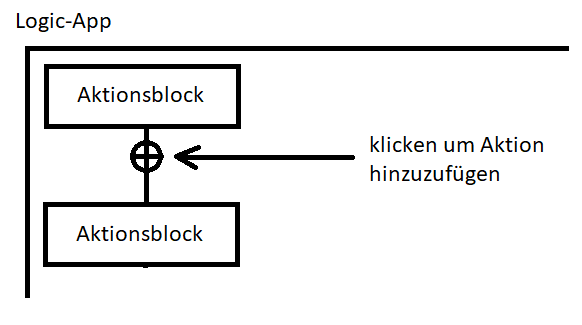
* NoProfile  
  NoProfile verhindert das Einstellungen aus den Powershell Profildateien für das Skript übernommen werden. So stellt man sicher das ein Skript nicht von Umgebungsvariablen abhängig ist und ohne Probleme auch auf anderen Rechnern verwendet werden kann.
* ExecutionPolicy  
  Hiermit kann die Skriptausführung zeitweise für das aktuelle Skript erlaubt werden.
* File  
  Hiermit wird der Pfad und die Skriptdatei angegeben. -file muss immer als letztes in der Parameterliste stehen.

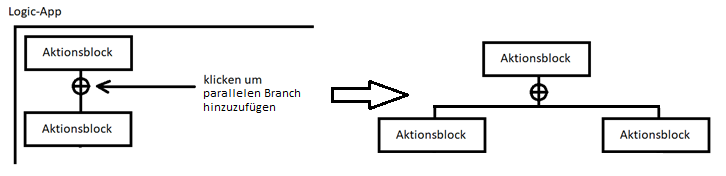
Alles zusammen kann es dann so aussehen:  
powershell -NoProfile -ExecutionPolicy Bypass -File „c:\pfad\script.ps1“

Konsolenfenster geöffnet halten  
  
Auf Enter-Taste warten: $null = Read-Host ‘Eingabe‘  
Fünf Sekunden warten lassen: Start-Sleep -Seconds 4  
  
Folgendes geht nicht im ISE-Editor  
  
Auf belibigen Tastendruck warten:  
$null = $Host.UI.RawUI.ReadKey(‘No Echo, Include Keydown‘)  
Folgendes funktioniert nur in Windows-Betriebssystemen:  
cmd.exe /c pause  
  
Encoded Commands  
Code kann in ein Base64 String umgewandelt werden den man dann ausführen kann. Encodierte Befehle sind keine Skripte und können selbst dann ausgeführt werden wenn die ExecutionPolicy das Ausführen von Skripten verbietet. Mehrere Codezeilen werden dabei mit einer Pipe verknüpft. Beispiel:  
$code = {  
 Code |  
 Code |  
 Code  
}  
  
Code wird in Bytes umgewandelt:  
$bytes = [System.Text.Encoding]::Unicode.GetBytes($code)  
  
Bytes werden in Base64 umgewandelt:  
$encoded = [convert]::ToBase64String($bytes)  
  
Kopieren des Aufrufs in die Zwischenablage:  
“powershell -noprofile – EncodedCommand $encoded“ | Set-ClipBoard  
  
Aufruf/Code auslösen:  
powershell -noprofile -encodedCommand …Inhalt aus der Zwischenablage…  
  
Parameter hinzufügen  
Parameter werden am besten nicht im Funktionskopf sondern innerhalb der Funktion definiert. Man sollte daher einen Param-Block nutzen da dieser universeller einsetzbar und besser zu kommentieren ist:  
  
function Out-TextToSpeach  
{  
 [CmdletBinding()]  
 param  
 (  
 [string]  
 $Text = “Hello User!“,  
 [Int]  
 $LanguageId=409,  
 [Int]  
 $Rate=0  
 )  
  
 $tts=New-Object -ComObject $sapi.SpVoice  
 $tts.Rate=$Rate  
 $tts.Text=“<lang langid=‘$languageId‘> $Text </lang>“  
 $null=$tts.Speak($ttsText)  
}  
Man kann Parameter zur Pflicht machen indem man folgende Annotation verwendet:  
[Parameter(Mandatory)]  
[string]  
$Text,…

Sortieren  
  
Sort-Object ist eine Piplinebefehl zum sortieren von Ergebnissen:  
 … | Sort-Object -Property Feldnamen  
oder  
… | Sort-Object -Property { Variablennamen }  
  
Bei Hashtables kann man nicht einfach den Feldnamen nutzen:  
  
#Hashtable  
$Hashtable = @{  
 Name = ‘Tester‘  
 Date = Get-Date  
 Username=$env:username  
 Role=‘Admin‘  
 }  
#Das folgende geht nicht  
$hashtable | Sort-Object -Property Name  
#Das folgende funktioniert (nach Hashtable-Schlüssel sortiert)  
$hashtable.GetEnumerator() | Sort-Object -Property Name  
#und nach Hashtable  
$hashtable.GetEnumerator() | Sort-Object -Property Value  
  
Durch das Anhängen von Unique (-Unique) kann man Dubletten entfernen.  
  
Pipeline Befehle  
  
Das Oben vorgestellte Sort-Object gehört zu den Sechs wichtigsten Pipelinebefehlen. Die anderen Befehle sind:  
  
**Select-Object**  
Legt fest welche Informationen pro Objekt angezeigt werden sollen.  
  
**Where-Object**  
Legt fest welche Objekte angezeigt werden sollen. (Der Befehl wirkt als Filter)  
  
**Measure-Object**  
Zählt die Ergebnisse und kann numerische Ergebnisse statistisch auswerten. Bei Zahlen kann die z.B. der Maximalwert oder der Durchschnitt sein. Bei Texten werden Zeichen und Worthäufungen gezählt.  
  
**For-Each-Object**  
Pipelineschleife. Der angegebene Code wird für jedes durch die Pipeline strömende Objekt ausgeführt.  
  
**Group-Object**  
Group-Object analysiert Verteilungshäufigkeit. Gruppen mit identischen Eigenschaften können gruppiert werden.  
  
**Sort-Object**  
Sortierung Ergebnisse und entfernen Duplikate auf Wunsch.  
  
**Select-Object**  
Beispiel: … | Select-Object -Property Name, Company, Info  
Nur die Spalten Name, Company und Info werden angezeigt.  
  
**-First, -Last, -Skip**  
Anzahl von Ergebnissen  
Nur die ersten X Ergebnisse  
Nur die letzten X Ergebnisse  
Überspringe X Ergebnisse  
  
-ExpandProperty  
Liefert nur den reinen Inhalt einer Eigenschaften während Property zusätzlichen Text generiert wie z.B. Linien u.s.w.  
Beispiel: … | Select-Object -ExpandProperty FullName -First 1  
  
In einer Selektion kann der aktuelle Wert ($\_) über die Mthode noch verändert werden.  
 …{[Math]::Round($\_.Length/1KB).1)),Name…  
  
Where-Objekt  
Über das Where-Object können unwichtige Objekte aus einem Abfrageergebnis entfernen: z.B. | Where Extension -eq .txt  
 | Where Extension -eq Service  
  
Group -Object  
Das Group-Object gruppiert Daten und stellt die Häufigkeit einzelner Elemente in der Gruppe fest.  
  
-noElement   
…bestimmt das die Häufigkeit bestimmter Elemente angegeben wird. Auf die Anzeige welche Elemente für die Gruppe gefunden wurden wird verzichtet.  
  
Beispielsweise zeigt nachfolgende Zeile keine Dateinamen an sondern nur wie oft welcher Dateityp wie oft vorkommt:  
| Group-Object Extensions -NoElement  
  
Measure-Object  
  
Mit: | Measure-Object  
lassen sich die Ergebnisse die ein Cmdlet liefert zählen. Zudem kann das Measure Object statische Summenfunktionen hinzufügen:  
 | Measure-Object -Sum -Average -Min -Max  
Ausserdem kann man z.B. die Ordnergröße ermitteln:  
(Get-ChildItem -Path $Home-File-Recurse -Force -ErrorAction Ignore | Measure Object -Property Length -Sum).Sum  
  
Foreach-Object  
  
Das Foreach-Object ist ein frei programmierbare Pipelineelement. Die durch die Pipeline laufenden Objekte werden durch einen Skriptblock verarbeitet. Im Folgenden werden die Zahlen von 1 bis 255 durch einen Skriptblock geschickt und generiert IP-Adressen. $\_ enthält dabei immer das aktuelle Datenelement:   
 1..255 | ForEach-Object -Process(“192.168.2.$\_“)  
  
Hashtables  
  
$hashtable=@{}  
$hashtable[‘Name‘]=‘Mustermann‘  
$hashtable[‘Datum‘]=Get-Date  
  
Abrufen kann man das Ganze mit:  
>$hashtable[‘Name‘] (Ergebnis: Mustermann)  
  
Vergleichsoperatoren  
  
Syntax: $a=10  
 -not($a -gt 10)  
$a ist nicht größer 10 die Aussage wird durch -not negiert und ist daher True. -not prüft dabei den Teil in den Klammern. Man kann -not auch ohne Klammern nutzen doch das führt zu Fehlern.  
  
Vergleiche kombinieren  
  
Zwei Vergleiche können in Klammern geschrieben werden und werden danach mit einem Operator gekoppelt:  
(($age -ge 18) -and ($gender -eq “m“))  
  
Vergleiche auf Arrays anwenden:  
1,2,3,4,3,2,1 -eq 3  
Ausgabe: 3  
 3  
Aus:  
1,2,3,4,3,2,1 -ne 3  
wird: 1  
 2  
 4  
 2  
 1  
und:  
(1,2,3,4,3,2,1 -ge 3).Count  
gibt “3“ aus  
  
If-Bedingung  
  
In einer if-Bedingung kann der Vergleich direkt genutzt werden:  
if($stunde -gt 6 -and $stunde -lt 12)  
{  
 Skriptcode  
}  
oder über boolsche Variablen:  
$vormittag = $stunde -gt 6 -and $stunde -lt 12  
if(vormittag)   
{  
 Skriptcode  
}  
  
Switch Beispiel  
$errorcode=Get-Random -Maximum 10  
switch($errorcode)  
{  
 0 { ‘OK‘ }  
 1 { ‘Error1‘ }  
 3 { ‘Error2; Break‘ }  
 default { ‘Error3‘ }  
}  
Anmerkung: Mit Break wird das Switch-Konstrukt sofort verlassen ohne weitere Einträge zu vergleichen. Wenn keiner der Werte zutreffend ist wird “default“ ausgeführt.  
  
Array-Elemente als Text zusammenführen  
  
$texte = ‘erster Text‘,‘zweiter Text‘  
$texte -join ‘,‘ (Anmerkung: Das Komma definiert das Trennzeichen)  
Ergebnis: erster Text, zweiter Text  
  
Eigene Datentypen erstellen  
  
1) Eigenen Typ definieren  
  
class MeineDreiInfos  
{  
 [string] $Name  
 [datetime] $Date  
 [Int]$Id  
}  
  
2) Neues Objekt der Klasse instanzieren:  
$object=[MeineDreiInfos]::new()  
  
3) Informationen im Objekt speichern:  
$object.Name = $env:username  
$object.Date = Get-Date  
$object.Id = 12  
  
4) Eigenes Objekt ausgeben:  
>$object  
  
Konstruktoren hinzufügen  
  
class Person  
{  
 [string]$name  
 [datetime]$Datum  
 [int]$Id  
  
(Anmerkung: Konstruktor 1)  
 Person([string]$Name,[datetime]$Datum,[Int]$id)  
 {  
 $this.Name=$Name  
 $this.Datum=$Datum  
 $this.Id=$Id  
 }  
  
(Anmerkung: Konstruktor 2)  
 Person()  
 {  
 $this.Name=‘Unbekannt‘  
 $this.Datum=Get-Date  
 $this.Id=-1  
 }  
}  
  
Anders als in Programmiersprachen können sich Konstruktoren in Powershell sich nicht gegenseitig aufrufen.  
  
Methoden zu Klassen hinzufügen  
  
Als Beispiel könnte es eine Methode geben die zurück gibt wie lange ein Mitarbeiter in einer Firma arbeitet. Dies kann wie ein Konstruktoren in die Klasse implementiert werden:  
  
[Timespan]GetEmployment()  
{  
 $heute=Get-Date  
 $zeitraum=$heute -$this.Datum  
 return $zeitraum  
}  
Anmerkung: Sobald man einen Konstruktor in eine Klasse schreibt wird der Standardkonstruktor überschrieben.  
  
Vererbung  
  
Im folgenden Beispiel erbt die Klasse Programmierer von der Klasse Person:  
class Programmierer : Person {…}  
  
Konstruktoren einer Klasse werden nicht vererbt aber es können Parameter allerdings weiterreichen: Programmierer():base()  
oder:  
 Programmierer([string]$Name[datetime]$Datum,[string]$Sprachen)  
 :base($Name, $Datum,$Id)…  
  
Simulationsmodus  
  
Der “-Whatif“ Schalter sorgt dafür das die Codeausführung nur simuliert wird. Er gilt nur für die aktuelle Funktion.  
  
Mit der Funktion “ShouldProcess“ kann man bestimmte Codeblöcke bei Verwendung des Schalters -Whatif umgehen während anderer Code tatsächlich ausgeführt wird. Der Funktion können zwei Textnachrichten übergeben werden die in der Konsole ausgeben werden.  
  
[CmdletBinding (SupportShouldProcess=“true)]  
param()  
$message1=‘‘  
$message2=‘‘  
$doit=$PSCmdlet.ShouldProcess(message1,message2)  
if(doit)  
{  
 …  
}  
  
Sicherheitsabfrage  
  
Mit dem Schalte “-Confirm“ muss jede Codezeile vor der Ausführung bestätigt werden. Mit shouldProcess können auch hier Codeblöcke gezielt ausgewählt werden.  
  
Azure Functions  
  
Azure Functions befinden sich im Wartemodus bis sie z.B. durch einen Trigger ausgelöst werden. Trigger sind unidirektional und können Daten nur empfangen während Bindings bidirektional sind und sowohl Daten empfangen als auch senden können.Bindings können daher auf in, out und inout gestellt werden (Im Azure Portal bei den Funktionseinstellungen) Über den ServiceBusQueue wird der Trigger eine Funktion immer dann angesprochen wenn er über den Bus eine Nachricht(Message) bekommt.. Anstatt des Bus kann für Bindings auch der Azure-Storage-Queue in Zusammenhang mit Datenbanken verwenden. Es gibt sowohl die Möglichkeit bei Triggern vordefinierte als auch benutzerdefinierte Trigger zu nutzen.  
  
Speicherkonten  
  
In Speicherkonten können Container für Blobs angelegt werden um z.B. Bilder auf dem Speicherkonto abzulegen. Der QueueStorage erlaubt das ablegen von Daten in eine Warteschlange.  
  
Azure Logic Apps  
  
Über das AzurePortal kann man den Bereich für LogicApps finden in dem man “Logic-Apps“ in der Suchzeile eingibt und das Ergebnis per Linksklick auswählt. Hier kann man eine neue LogicApp anlegen (+New). Es ist wichtig das man seine neue LogicApp in der Testumgebung auf Consumtion (Verbrauch) zu stellen. Die im Gegensatz zu “Standard“ fallen bei Consumtion nur Kosten an wenn die LogicApp ausgeführt wird.  
  
Hat man die LogicApp erstellt erscheint sie in der Liste mit den gegeben falls schon vorhandenen LogicApps. Durch anklicken der LogicApp bleibt links das Menu geöffnet und rechts davon erscheint ein Contentbereich. Im Menu wählt man zum bearbeiten des LogicApp-Flows den Logic-App-Designer aus. Durch die Auswahl der Verbrauchsoptionen sieht man hier die klassische Oberfläche anstatt der modernen Variante (Standardversion). In der Übersicht (Überblick) sieht man die erfolgreichen und fehlgeschlagenen Durchläufe der aktuell gewählten LogicApp. Ein weiterer klick auf einen Eintrag in dieser Historie öffnet ein Verlaufsfenster das die einzelnen Blöcke des Flows auflistet. Ein grüner Harken zeigt dabei eine erfolgreiche Ausführung an. Ein rotes X zeigt Fehler an und ein graues X weist auf Codeblöcke hin die nicht ausgeführt wurde. Jeder Codeblock kann geöffnet werden. Dies ermöglicht sich die Eingabe und die Ausgabedaten anzuschauen und Rückschlüsse auf die Vorgänge zu ziehen.



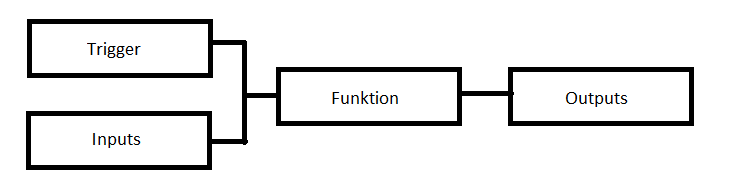
Http-Anforderung (http-Request) mit Get  
  
Bei einem http Trigger wird bei Methode „Get“ eingestellt. Bei „Add a new Parameter“ wird relativer Pfad angeharkt. Dies ermöglicht dies ermöglicht das Hinzufügen von Parametern in der URL (Get-Request). Bei relativer Pfad kann nun ein Parameter angegeben werden:   
 /{Platzhalten}  
  
Blob Storage  
  
Um einen Blob-Storage zu erstellen nutzt man wieder die Suchzeile des Azureportals. Hier sucht man nach Speicherkonten bzw. in der englischen Version nach Storage-Accounts. Ein Speicher kann dann mit „+Create“ hinzugefügt werden. Als Einstellung wählt man LRS+Standadplan. Als nächstes wählt man „Container“ aus und mit „+Container“ fügt man einen neuen Container der Auflistung hinzu. Name +Privat angeben.  
  
Containerdaten in die LogicApp einbinden  
  
  
  
Um die Verbindungsinformationen zu einem Storage-Account herauszufinden muss man unter Speicherkonten den entsprechenden Container anklicken und im Seitenmenü den Menüpunkt Zugriffsschlüssel wählen. Im oberen Bereich kann man dann „Schlüssel anzeigen“ wählen. Hier kann man den Namen des Speicherkontos und einen der beiden verfügbaren Keys kopieren. Den Namen des Containers findet man unter „Container“ in der erscheinenden Liste. Dieser gleichbedeutend mit einem Verzeichnisnamen z.B. log -> /log  
Mit den so gesammelten Informationen kann man in der Designansicht der LogicApp z.B. einen „neuen Blob“-Aktion erstellen/konfigurieren.  
  
Einen parallelen „Zweig“ (Branch) erstellen:

  
  
Begrifflichkeiten deutsch 🡨🡪 englisch  
  
Die If-condition als „Bedingung“ im deutschen Azure bezeichnet.  
Die Switch-Condition wird als „Option“ eingedeutscht.  
Until = bis, Terminate = Beenden, Scope = Bereich  
  
Api-Verbindung zu Blob-Storage ändern  
  
Will man eine Verbindung von einer LogikApp zu einem BlobStorage herstellen muss man eine API-Verbindung aufbauen. Dazu benötigt man den Storagekontonamen und den Zugriffsschlüssel. Diese Informationen kann man unter Speicherkonto 🡪 „Name des Kontos“ herausfinden. Eine gespeicherte API-Verbindung kann nicht im Logik-App-Designer gelöscht werden. Für den Fall das eine API-Verbindung überflüssig/fehlerhaft ist kann man diese unter: Ressourcengruppen 🡪 „Name der Ressourcengruppe“ 🡪 „Name der API-Verbindung“ … löschen.  
  
Tip: LogicApp/ Propertynamen herrausfinden  
Manchmal verwendet die LogicApp nicht direkt die Feldnamen aus der Datenbank sondern eine modifizierte Version die die Properties aufteilt. Aus Kunden\_Id kann dann Kunden\_Id\_Typ und Kunden\_Id-Value werden. Um dies heraus zu finden kann man in einer LogicApp einen Connector anlegen und eine Aktion wie UpdateRow folgen lassen. In der Aktion wählt man im Dropdown-Menü die Tabelle aus für die man sich interessiert und sucht sich im Vorschlagsfeld die Property herraus deren tatsächlichen Namen man erfahren möchte. Man weist der Property danach irgendeinen Wert zu. Wichtig ist hier das man die LogicApp nicht laufen lässt, sondern nur in der Aktion den Link „Peek Code“ klickt. Der Code der angezeigt wird ziegt nun die Zuweisung mit dem tatsächlichen Namen der Property an.  
  
Function App’s  
  
In Azure werden mehrere Funktionen unter einer Funktion App zusammen gefasst. Das Erstellen einer Funktion-App ist damit eher mit dem erstellen eines Ordners gleichzusetzen.

1. Funtion App erstellen  
   Compute🡪Function App🡪“+Create a Ressource“  
   - Ressoucengruppe anlegen  
   - Name angeben  
   - Runtime Stack (Das ist das was in Visual Studio eine Vorlage ist.(z.B. Dot.NET)  
   - Region angeben  
   - Tag sollte man hinzufügen um für sich selbst Informationen hinzuzufügen  
    Autor: Name-des-Autors  
    Description: Eine kurze Beschreibung   
    Enviroment: Development (oder z.B. Production)
2. Unter der FunctionApp fügt man mit „+Create“ eine neue Funktion hinzu. Hier wählt man die Art des Triggers der zum auslösen der Funktion genutzt werden soll. (z.B. Blob-Storage oder Http-Trigger)
3. Nun wird der Name der Function und weitere für die Funktion nötige Informationen angegeben. Diese können je nach Triggerart anders sein.

Funktionen im AzurePortal bearbeiten

1. FunctionApp 🡪 „Namen der FunctionApp auswählen“
2. Im nächsten Bereich „functions“ (Funktionen) auswählen und die gewünschte Function auswählen.

Integration  
  
Integration zeigt eine schematische Darstellung mit Trigger, Ein- und Ausgabe und der Funktion als zentralen Bestandteil.  
  


* In Output kann z.B. die Ausgabe des Resultats festgelegt werden. (z.B. Blob-Storage oder http)
* Unter „Code + Test“ kann man den eigentlichen Code der Funktion bearbeiten. Dieser wird in der run.csx gespeichert.
* Unter „Code + Test“ kann in Dropdownmenü die Datei geändert werden deren Code man sich ansieht.  
  Die function.json beinhaltet eine Beschreibung des Triggers im Json-Format. Das selbe bekommt man in grafischer Version wenn man unter dem Menüpunkt „Itegration“ auf dem Trigger der Funktion klickt. Hier sind alle Parameter der JSON-Datei als Texteingabefelder zu finden.
* Der Monitor dient dazu heraus zu finden ob eine FunctionApp aufrufen erfolgreich war oder nicht.  
  Code + Events + Data = Azure functions

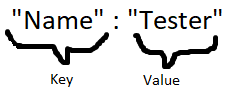
Planung  
  
Ein Problem wird in mehrere verteilte Funktionen aufgeteilt die miteinander kommunizieren. Dies wird als Event-Driven-Programming bezeichnet.  
  
Function App Phases  
  
  
  
Triggers

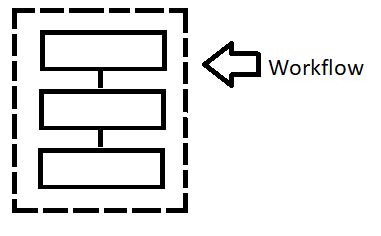
|  |  |
| --- | --- |
| Time trigger | Aufruf nach vordefinierten Zeitplan |
| GitHub trigger | Reaktion auf ein GitHub Ereignis |
| Blob trigger | Aufruf bei Blob Updates |
| Http trigger | Wird bei Http-Requests aufgerufen |
| Queue trigger | Wird durch nachrichten in der Warteschlange (Queue) aufgerufen. |
| Generic trigger | Wird durch einen http-Webhook aufgerufen |
| Service-Bus trigger | Aufruf durch einen Service-Bus Trigger |

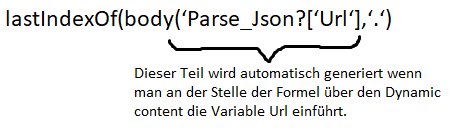
Anmerkung: Wenn man über ein externes Programm wie Visual Studio eine Funktion in eine FunctionApp auf dem Azure Portal hochladet, kann man in dieser FunctionApp keine Funktionen erstellen die im Azure Portal erzeugt wurden. Anders herum können Function Apps in denen Funktionen die innerhalb des AzurePortal erstellt wurden können keine Funktionen aus externen Programmen importiert werden. Externe funktionen können zudem nicht über das Portal editiert werden. Der Vorteil von Funktionen die im Portal erstellt wurde ist der, das diese im Portal überarbeitet werden können.

Notizen zum Azure LogicApp Workshop  
  
Tips:

* Graph Rest API zum testen der http-Trigger. Graph Rest API verfügt über Vorlagen um verschiedene Funktionen auf unterschiedlichen Datenquellen anzuwenden. Hiermit können dann Requests getestet bzw. zusammengebaut werden.
* Postman dient zum Testen bzw. als simulierte Datenquelle. Es können nicht nur Get (wie im Browser) verschickt werden, ganze Requestbodys über die Methode Post. Als Alternative zu Postman bietet sich Insomnia an.
* FunctionApps sollten mit externen Tools wie Visual Studio geschrieben werden und nicht im Portal selbst. Das hat den Vorteil das man z.B. Versionierungstools wie Git-Hub nutzen kann.

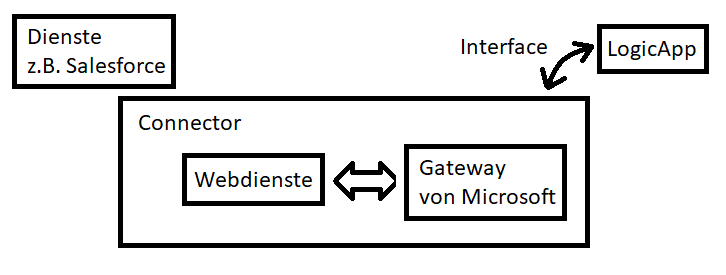
Aufbau von JSON  
  
Json-Dateien bestehen aus Key-Value-Paaren:   


Die Werztepaare werden mit Kommas separiert. Fasst mit geschweiften Klammern Objekte zusammen:  
{  
 “Key1“ : “Value1“,  
 “Key2“ : “Value2“,  
 “Key3“ : “Value3“  
}  
  
Felder wewrden in Json durch eckige Klammern erzeugt: “Zahlen“ : [1,2,3]  
  
Je nach Struktur des Json‘s können Daten innerhalb des Bodys noch einmal verschachtelt sein. Parameter können z.B. unter dem Key „value“ zusammengefasst sein. Will man auf den „Value“-Bereich zugreifen so kann das so aussehen:  
 body(‘Aktion‘)?[‘value‘]  
Das Ansprechen/auslesen von Parametern aus diesem Bereich kann dann so aussehen:  
 body(‘Aktion‘)?[‘value‘]?[0]?[‘Id‘]   
  
Tip: Filtert man innerhalb einer Aktion-Daten so kann man auf Basis des gefilterten Ergebnisses eine weitere Aktion nutzen um das Ergebnis weiter zu filtern. So kann man mehrere Filteraktionen aneinander ketten.   
  
Aufbau von LogicApps  
Als Workflow wird der gesamte Bereich aller verbundenenBausteine bezeichnet.  
  
Connectoren  
Connectoren sind eine Sammlung voon Aktionen und Triggern die thematisch eine Einheit bilden und es ermöglichen Datenarten zu bearbeiten.  
  
Trigger und Aktionen  
Trigger stoßen Vorgänge an. Trigger (Auslöser) reagieren auf verschiedene Ereignisse und starten den Workflow. Der erste Baustein in einem Workflow ist daher immer ein Trigger. Aktionen sind wiederum zur Verarbeitung der Daten gedacht und werden von triggern angestossen.  
  
Übertragung von Daten zwischen Bausteinen  
  
Zwischen Bausteinen werden Daten über den Bereich „Dynamic Content“ übertragen. Von einem Baustein abgerufene Werte landen im dynamic Content. Jeder andere Baustein kann diese Variablen aus dem dynamic Content wieder abrufen.  
  
Variablen  
  
Variablen können erst verwendet werden wenn sie initiert wurden. Erst ab dem Baustein der eine Variable initiert können alle folgenden Bausteine diese aus dem Dynamic Content nutzen.  
  
Bereiche (Scopes)  
  
Scopes teilen der Workflow in Blöcke aus mehreren Aktionen auf. Diese Blöcke sind also zu zunächst erstmal als Ordnungseinheiten zu sehen. Anders als z.B. in C# wo Variablen einen Gültigkeitsbereich haben und z.B. nur innerhalb einer Methode gültig sein können beschränken Scopes in Azure LogicApps nicht den Gültigkeitsbereich von Variablen. Jede Aktion des Workflows kann auf den Dynamic Content zugreifen.  
  
Try-Catch-Finally mit Scopes  
  
Scopes lassen sich ähnlich wie die Try-Catch-Finally Blöcke in C# verwenden. Der Try-Block ein normaler Scope-Block. Der Catch-Scope-Block schließt an den Try-Scope-Block an. Über das Dreipunktemenü (…) kann man nicht nur den Namen des Blocks ändern sondern auch Einstellungen über den Menüpunkt „Run After“ machen. Dort kann festgelegt werden das der „Catch“-Block nur startet wenn im vorhergehenden Scope die Ausführung fehlschlug. (Diese Einstellung ist das was den darauffolgenden Scope zu einem Catchblock macht.). Der Finaly-Block zeichnet sich dadurch aus das er immer ausgeführt wird. Aus diesem Grund müssen im vorangehenden „Run After“ alle Optinen angeharkt werden umden folgenden Scope-Block zu einem Finally-Block zu machen.  
  
Anmerkung: Wenn man innerhalb des Scopes verschachtelungen hat die zum Beispiel durch die Verwendung von Schleifen entstehen, werden bei Fehlern innerhalb dieser Verschachtelung die Detailinformationen zum Fehler nicht nach oben durchgereicht. Der Catch-Block erhält damit nur eine sehr allgemeine Fehlerbeschreibung. Um dies zu Umgehen muss man den Bereich in der verschachtelung ansprechen in dem man den Fehler erwartet und das von dort gelieferte Resultat auslesen. Hier ein Beispielcode für Compose-Aktionen die dies tun:  
  
Aktion\_die\_das\_Result\_holt ->   
result('Block\_in\_dem\_der\_Fehler\_erwartet\_wird')

Aktion\_die\_den\_Fehlertext\_holt -> body('Aktion\_die\_das\_Result\_holt')?[0]?['outputs']?['body']?['error']?['message']   
  
Kommunikation mit Function Apps  
  
FunctionApps haben im Kopf der Funktion die Möglichkeit, Daten entgegen zu nehmen. Dies geschiet über den Parameter HttpRequest „req“. Natürlich kann man die Funktion um weitere Parameter ergänzen aber „req“ erhält den Json Payload der zur Fuktion geschickt wurde:  
 public static Funktionsname Run(  
 [HttpTrigger (AuthorizationLevel.Anonymus,“get“,“post“)]   
 **HttpRequest req**, ILogger log) { … Code … }  
  
Eine Abfrage nach einem Key aus dem Json um einen Wert einer Property wie zum Beispiel „name“ kann so aussehen:  
 string name=req.Query[“name“];  
Der Aufruf eines Json-Body kann so aussehen:  
 string requestBody=await new StreamReader(req.Body).ReadToEndAsync();  
 dynamic data = JsonConvert.DeserializeObject(requestBody);  
 name = name ?? data ? name;  
Den Rückgabewert kann man dann so setzen:  
 return new OkObjectResult(responseMessage);  
  
Bindet man eine solche Funktion in eine LogicApp ein, so erhält die LogicApp den gesendeten Rückgabewert wieder als Json.  
  
ParseJson  
  
ParseJson ist eine Aktion die Json Daten für LogicApps benutzerfreundlich aufbereichtet. Man gibt unter Content an für welchen Teil des Json man sich interessiert z.B. „Body“ und gibt das Schema an. Das Schema beschreibt die Struktur und die darin enthaltenen Datentype. Über „Use sample payload to generate schema“ kann man dies automatisch erstellen lassen, sofern man ein Beispiel-Json hat das man in das Sample-Payload-Fenster kopieren kann. Ein erneutes verwenden des Sample-Payload-Fensters löscht jedesmal das vorhergehende Schema.  
  
Http-Block  
  
Neben Http-Trigger gibt es auch http-Aktionen mit denen man URL’s abfragen und auf diesem Wege auch Json-Daten erhalten kann.  
  
Ausgangssituation für das LogicApp Workshopbeispiel  
  
Mit einem Recurrence-Trigger (Time-Trigger) wird eine http-Aktion aufgerufen die Json-Daten von einer Picture-of-the-day Seite der Nasa abruft. Das Ganze wird mit einem ParseJson-block verarbeitet. Im darauf folgenden Schritt wird mit einer condition geprüft ob der Medientyp (media\_typ) ein „Image“ ist, da z.B. ein Video einen Fehler bei der Verarbeitung erzeugen würde. Der Parameter(media\_typ) dafür steht in dem Json das man empfangen hat.  
  
Conditions verknüpfen  
  
In Conditions können mehrere logische Bedingungen festgelegt werden die dann je nachdem, ob sie zutreffen den Workflow auf einen bestimmten Zweig umleiten. Müssen mehrere Bedingungenzusammen erfüllt sein wie zum Beispiel: Kleiner als Fünf und größer als Acht. Müssen die einzelnen Bedingungen durch ein logisches And oder Or zu einer komplexeren Bedingung verknüpft werden. Hat man mehrere komplexe Bedingungen in einem einzelnen Bedingungsblock wie Beispielsweise:  
(Wert X muss kleiner als fünf und größer als Acht sein) oder   
(Wert Y muss größer 10 und kleiner 20 sein)   
Muss man die komplexeren Bedindungen gruppieren:  
Gruppe 1 wäre in dem Fall:   
 (Wert X muss kleiner als fünf und größer als Acht sein)  
Gruppe 2 wäre in dem Fall:  
 (Wert Y muss größer 10 und kleiner 20 sein)  
Beides wird mit einem „OR“ verknüpft.  
Um eine solche Gruppierung zu erreichen harkt man die Bedingungen an die zusammen gehören. Im Fall von Gruppe 1 wäre dies (Wert X muss kleiner 5 sein) und (Wert X muss größer 8 sein). Über das Dreipunktemenü(…) wählt man „Make group“. Im Designer wird die neue Bedingungsgruppe dann sichtbar eingerückt.  
  
Compose Aktionen  
  
Eine Compose-Aktion kann als Ersatz für Variablen genutzt werden. Variablen müssen in LogicApps auf globaler Ebene initialisiert werden und können nicht in Codeblöcken wie einer For-Each-Schleife initiert werden. Die Verwendung von Variablen ist zudem in Verbindung mit paralleler Verarbeitung z.B. in For-Each-Blöcken problematisch. Ergebnisse die man aus Abrufen aus dem Datavers erhält können statt in Variablen in Compose-Aktionen geleitet werden. Ein Codebeispiel für die Inputzeile einer Compose-Aktion kann so aussehen:  
 body(‘Aktionsname‘)?[‘value‘]?[0]?[‘Eigenschaft‘]  
Über “Output“ der Compose-Aktion kann der Wert in einer anderen Aktion weiterverwendet werden. Der Vorteil einer Compose-Aktion ist das diese überall im Workflow verwendet werden kann und nicht wie Variablen im Kopf des Workflows initiert werden müssen. Der Inhalt einer Compose-Aktion bleibt selbst bei paralleler Verarbeitung gleich.  
  
Compose & Condition  
  
Ein NullCheck kann in einer Condition folgendermaßen eingestellt werden:  
 outputs(‘Compose-Aktion‘) is not equal to null  
Hinweis: Bei null handelt es sich um einen Ausdruck der nicht händisch sondern wie eine Funktion über den Expression-Beweich hinzugefügt wird.  
  
Workshopbeispiel Teil 2  
  
Sollte der Medientyp nicht „image“ sein wird die LogicApp über eine Terminate-Aktion abgebrochen. Es gibt dabei verschiedene Optionen die es erlauben Fehlercodes zu senden sodass diese im Log der App angezeigt werden.   
  
Wenn der Medientyp Image ist, wird nun ein weitere http-Aktion aufgerufen die aus dem Json die URL-Zeile zum Bild verwendet. Die http-Aktion erhält damit das Bild in Binärcode. Da Bilddateien in verschiedenen Formaten kommen können muss dafür gesorgt werden das die Bilddateien im späteren Verlauf der App mit der richtigen Dateiendung gespeichert werden. Dazu kann eine Compose-Aktion zum zusammenbauen des Dateinamens genutzen werden. Als erstes muss die Dateiendung aus dem Dateinamen als alleinstehendes String herausgeschnitten werden. Dafür muss die Position des letzten Punktes im Dateinamen ermittelt werden hinter dem die Dateiendung steht:  
  


Mit Substring wird aus dem Quellstring nur der Teil des Strings genommen ab der Position die man vorher mit LastIndexOf ermittelt hat. Auf diese Weise erhält man die Dateiendung der bilddatei. Der code dafür sieht dann so aus:  
  
substring(body(‘Parse\_Json‘)?[‘Url‘], lastIndexof(body(‘Parse\_Json‘?[‘Url‘],‘.‘))  
  
Auch für den Substringteil wurde die Variable aus dem Dynamic Content genommen und nicht händisch eingetragen. Dadurch erspart man es sich die Json-Notation selber schreiben zu müssen. Im letzten Schritt muss noch der Bildtitel vor die Datei-Endung gesetzt werden. Ein einfaches Hinzufügen von „titel“ aus dem Dynamic content vor den oben gezeigten Formelblock in der Input-Zeile der Compose-Aktion genügt dafür.  
  
Hinweis: Es kann manchmal zu dem Problem kommen das der Coursor nicht aus dem Formelfeldherauskommt. Es empfiehlt sich hier auf eine leere Stelle auf dem Bildschirm zu klicken um dies zu erreichen.  
  
CreateBlockBlob  
  
Das speichern der Bilder wird von „CreateBlockBlock“ übernommen. Als Dateiname wird der über die Composeaktion erstellte Name genutzt. Der Dateiname steht als Ausgabeergebnis in der Compose-Aktion zur Verfügung und sorgt beim speichern dafür das die Bilder in ihrem Originalformat gespeichert werden. Das Resultat einer Compose-Aktion ist in der Auflistung des Dynamic Contents unter dem Namen der Compose-Aktion als Outputs zu finden. In der CreateBlockBlob-Aktion wird der Dateiname unter „Specify name of the blob to create“ eingetragen. Unter Blob Content werden die eigentlichen Daten mit Dynamic Content „Body“ übergeben. Nach dem speichern des eigentlichen Bildes werden im Rahmen des Workshops noch zusätzliche Informationen zum Bild in einer Tabelle über die InsertEntity-Aktion abgelegt. Die Daten zu Date, Explenation und titel können einfach aus dem Dynamic Content geholt werden. RowKey dient als Primärschlüssel. Hier bietet sich die verwendung einer Guide an die eine zufällige Id erstellt. Dadurch ist es nicht norwendig händisch eine Id zu vergeben was in der vorliegenden Situation notwendig ist da das speichern automatisch abgewickelt werden soll. Im Bereich für Ausdrücke (Expression) findet man dafür die Funktion Guide().  
  
Hinweis: Für Expressions gilt das man einfache Anführungszeichen nimmt (‘). Doppelte Anführungszeichen führen zu Fehlern.  
  
Anmerkung zur Recurrence-Aktion  
  
Der Recurrence Timer löst in festgelegten Intervallen LogicApps aus. Man kann eine Zeitzone festlegen und mit Starttime ein Startdatum und eine Zeit festlegen. Die Notation ist dabei folgende: Jahr-Monat-Tag T Stunde:Minute:Sekunde Z  
 2022-06-01T12:00:00Z  
Über „at this hour“ und „at this time“ wird die regelmäßige Zeit angegeben zu der die LogicApp ausgeführt wird.  
  
Parallelität  
  
Es gibt Ablaufparallelität und Schleifenparalelität.  
  
Schleifenparalltlität  
  
Wenn man mit Schleifen z.B. Datenbanken abruft kann es sein das ja nach Verarbeitungsgeschwindigkeit Datensatzabfragen sich gegenseitig überholen. Will man die Datensätze aber in der richtigen Abfolge, kann man über der Dreipunktmenü (…) die concurrency control auf 1 oder 0 setzen.   
  
Anmerkung: Liefert man Daten an eine SQL-Aktion wird nach allen Einstellung die SQL-Aktion in eine ForEach-Schleife gepackt. Der Automatismus geht davon ausgeht das mehr als ein Datensatz angesprochen werden soll. Ist dieser Automatismus nicht gewünscht muss die SQL-Aktion aus der Schleife herrausgezogen werden. Die Schleife kann dann entfernt werden. Damit die SQL-Aktion sich auf alle Datensätze bezieht muss man anstatt des Schlüsselwortes „Item“ der Bezeichner „Body“ verwendet werden.  
  
Unterschied zwischen Recursion und SlidingWindow  
  
Bei Recursion kann immer nur eine Aufgabe gestartet werden wenn eine andere beendet wurde. Bei slidingWindow kann eine Aufgabe starten während eine andere noch läuft.  
  
Verzögerung  
  
Delay und DelayUntil verzögern Aufrufe. Beispiel: Man wartet auf eine Emailbestätigung. Wenn diese nach drei Tagen nicht eintrifft kann z.B. eine Erinnerungsaktion gestartet werden.  
  
Approval Mails  
  
Mails mit Lesebestätigung können mit „Send approval email“ aus dem Office 365 Outlook Connector gesendet werden. Unter User Options können Buttons wie Approve oder Reject festgelegt werden die daraufhin in Mail erscheinen. Über Subject kann man in der Betreffzeile JsonVariablen nutzen um den Empfänger weitere Informationen zukommen zu lassen. Wenn ein Button in der Mail gedrückt wird, wird dies an Azure über selectedOption zurückgeschickt. Das Ergebnis wird als Text (z.B. „Approved“) zurück geschickt. Der Platzhalter den man z.B. in einer condition auslesen kann heißt „Approval Email Response“. Beispiel für eine Condition: “Approval Email Response“ equals to “Approved“.

Hinweis: Approval Emails sind zwar kostenlos aber stossen an ihre Grenzen wenn es um komplexere Themen geht. Man kann zum Beispiel nicht nachvollziehen wer, welche Bestätigung geschickt hat. Dies macht Abstimmungen unmöglich. Für umfangreichere Dinge wurde mir das Microsoft-Tool Power Automate empfholen. Der Editor erinnert an den LogicApp Designer allerdings ist die Verwendung kostenpflichtig.

LogicApps (Die Stephan-Devinition)  
  
LogicApps nutzen Connectoren die eine Funktionsbiblihothek von Webdiensten darstellt. In den connectoren befinden sich auch ein Gateway das sich darum kümmert wie Daten von und zu dem Service geschickt werden. Ein Connector verbindet sich mit der anwendung und ermöglicht dort das Ausführen von Operationen.  
  


Hinweis: Connectoren gibt es in Standard und Premium, wobei Premiumdienste extra und regelmäßig bezahl werden müssen.  
  
Variablen in LogicApps ansprechen  
  
Variablen aus einem http-Response-Body auslesen:  
 body(‘Get\_records‘)?[‘value‘]  
Aktuelles Objekt aus einer Schleife ansprechen: (z.B. Id des aktuellen Arrayitems):  
 item().[‘Id‘]  
Aktuelles Objekt aus aktuellen Schleifeninhalt:  
 items(‘Feld/Aktion‘)?[ID]  
Erstes Propertie aus dem ersten Element eines Feldes lesen:  
 first(body('Passendes\_Zahlungsziel\_abrufen'))?['Property']  
Erstes Element aus einem Feld holen das sich unter values im JSON-befindet:  
 body('http-Response-Aktion')?['value']?[0]?['Property']  
Liefert eine Aktion ein Json das in einer Schleife verarbeitet werden soll muss eine Umwandlung vom Json in ein Array erfolgen:  
 json(body(‘Lade\_Json‘))  
  
Tip: In Sharepoint Unterordner abfragen  
Der Ordner Dokumente kann weitere Unterordner haben die von der normalen Benutzeroberfläche (z.B. durch Dropdownbox) nicht gefunden werden. Dies ist nur programmatisch per Filterquery möglich: FileLeafRef eq ‘Unterordner‘  
  
Tip: Der Azure Storage Explorer ist eine große standaloe Version des Azure Speicherbrowsers und bietet einen größeren Funktionsumfang.  
  
Tip: Mit Visual Studio kann man Json-Dateien öffnen und in eine lesbare Form bringen:

1. Json mit Visual Studio öffnen
2. Alles markieren (Strg + A)
3. Rechtsklick auf markierte Zeilen und Dokumente formatieren im Kontextmenü wählen.

Sollte das JSON die maximal zulässige Dateigröße überschreiten muss man anstatt Visual Studio Tools wie Notepad++ (inkl. Json Addon) nutzen.  
  
Tip: Aus einem CRM die Powerpletform öffnen

1. In Dynamics (<https://Firmenname.crm4.dynamics.com/>) rechts oben auf das Zahnradklicken und „erweiterte Einstellungen“ auswählen.
2. Oben links das Einstellungen-Manü öffnen und unter der Rubrik Anpassungen den Punkt Anpassungen auswählen.
3. Im Contentbereich der Fensters „System anpassen“ um das Fenster der Powerplattform zu öffnen, das eine Anpassung der Entitäten und Formulare des CRM’s erlaubt.

Tip: Sollte beim erstellen von Oberflächen z.B. keine comboboxen in der Parameterauswahl angeboten werden kann man diese dennoch implementieren idem man in die Codeansicht geht und die Codezeilen einer vorhandenen Kombobox kopiert.

Dataverse  
  
Dataverse ging aus dem Common Data Service von Dynamics 354 hevor und wird von Microsoft verwaltet. Dataverse wird online und nicht auf einem lokalen Server betrieben. Dataverse dient als Sammelort (Repository) für Geschäftsdaten. Eine Ganze Palette von Produkten wie z.B. PowerApps, Power Automate, Power BI oder Power Virtual Agents greift darauf zu. Dataverse ist dabei mehr als eine Datenbank und bietet eine Bandbreite an Funktionen die darüber hinaus gehen. Es bietet für viele Anwendungen die Grundlage zur Datenhaltung. Dataverse bietet Sicherheitsfunktionen die von Azure Active Directory verwaltet werden. Darunter sind Funktionen wie Zugriffssteuerung und Multifaktor-Authentifizierung bis herunter zur von Spalten und Zeilen in einer Tabelle. In der Logikschicht kann Geschäftslogik mit den Daten kombiniert werden. Die Regeln die auf diese weise angewendet werden sind dann für alle Anwendungen die diese Daten nutzen gleich (Geschäftslogik, Dublikatserkennung). In der Datenschicht werden z.B. Validierungsregeln erstellt, Daten modelliert, Kataloge oder Reports erstellt. Die Datenschicht wird in der Azurecloud abgelegt. Dort werden auch LogFiles oder Datenabfragefunktionen angelegt. Die Integrationsschicht bietet Export und Importfunktionen an.

Entwickler Terminologie in Microsoft Dataverse  
  
Je nachdem ob man in PowerApps, dot.Net SDK (Dataverse SDK) oder Web-Api arbeitet werden unterschiedliche Begriffe für die selbe Sache verwendet:

|  |  |  |
| --- | --- | --- |
| PowerApps Oberfläche | Dataverse SDK | Web-Api |
| Tabelle | Entity | EntityType |
| Column | Attribut | Eigenschaft |
| Zeile | Datensatz | Datensatz |
| Optionen | OptionSet/Picklist | OptionSet |
| Ja/Nein | Boolesch | Boolesch |

Erste Schritte im Dataverse  
  
Der Zugriff auf Geschäftsdaten kann über REST(Odata) oder SOAP erfolgen. Je nach Projektumfang bietet sich einer der beiden Möglichkeiten an.  
  
Erweiterungen die eigenen Code verwenden beziehen sich oft auf die Automatisierung von Unternehmensprozessen. Dieser Code wird üblicherweise von Ereignissen ausgelöst und vom Server ausgeführt. Die Programmierung kann mit den zur Verfügung gestellten API’s erfolgen oder durch die Nutzung von PowerPlattform Tols für VS.

Arbeiten mit Tabellendefinitionen  
  
Wichtig für die Arbeit mit Datenplatformen ist ein Verständnis von Metadaten. Metadaten sind Daten, die Daten beschreiben. Tabellen können z.B. Informationen zu den jeweiligen Spalten enthalten. So können zusätzliche Informationen beschreiben ob die Spalte benötigt wird, wann sie geändert wurde oder wer sie geändert hat.

Erstellen von Clientanwendungen und Authentifizierung  
  
Bei selbstgeschriebenen Erweiterungen müssen diese, um auf dem Server ausgeführt zu werden am Server authentifiziert werden. Der Xrm.Tooling-Connector stellt dafür Funktionen bereit um die Authentifizierung zu erleichtern.  
  
CDS (Common Data Service) bzw. Dataverse  
  
Die Entwicklungsumgebung ist über:   
 make.powerapps.com zugänglich.   
Die verwaltung von Dynamics365 erreicht man über:   
 admin.powerplatform.com  
  
Dynamics definition  
  
Dynamics ist eine Kombination aus CRM und ERP. Dynamics kann damit als Konkurrent zu S.A.P. und ähnlichen Produkten gesehen werden.  
  
In Dynamics 365 gibt es mehrere Umgebungen die man zu verschiedenen Zwecken nutzen kann z.B. Development, Test oder Produktionsumgebung. In Dynamics sind einige Tabellen (ohne Inhalt) schon vorinstalliert. Es wird empfholen diese nicht umzubenennen. Dies ist zwar möglich, allerdings gibt es an anderen Stellen im System Bezüge zu den Standardnamen. Nutzer müssten sich in dem Fall beide Bezeichnungen für ein und dieselbe Tabelle merken. Abgesehen davo hängen an diesen Tabellen Metadaten die bei doppelter Benennung für Verwirrung sorgen können. Eine weitere Empfehlung ist die Erweiterung der Standardtabellen um Beziehungen zu spezialisierten Tabellen. Statt Kunden und Mitarbeiter z.B. in einer einzelnen Tabelle zu führen. In Outlook (bzw. der Outlookanbindung) kann man z.B. dann eine einheitliche für Kontakte nutzen und auf spezialisierte Tabellen zugreifen wenn dies erforderlich ist.

Parent-Child  
  
In Dynamics gibt es die Möglichkeit, Einträge die in der gleichen Tabelle liegen zueinander in Beziehung setzen. Ein Parent-Eintrag kann mehrere Childeinträgen zugeordnet werden. Diese Technik entspricht der verwendung von Hilfstabellen. In der Classic UI kann man sich diese Beziehungen über einen Button neben dem Eintragsnamen anzeigen lassen.  
  
Auswirkung von verküpften Daten  
  
Formulare die auf einer Tabelle basieren können Informationen aus anderen Tabellen darstellen sofern deren Basistabelle mit diesen Tabellen in Beziehung stehen. Felder die sich auf diese Tebellen beziehen werden dabei genauso automatisch ausgefüllt wie Felder die mit der Basistabelle verbunden sind.  
  
Begrenzung des Emailverkehrs  
  
In Dynamics kann der zugelassene Emailverkehr zentral gesteuert werden. So können ausgehende Mails für bestimmte Kontakte deaktiviert werden z.B. BulkMails die z.B. automatisch von Aktivitäten oder Kampagnen generiert werden. Wenn Mails direkt aus dem Outlook gesendet werden greift diese Steuerung nicht.

Outlook integration in Dynamics  
  
Es gibt zwei Möglichkeiten Outlook in Dynamics zu integrieren. Dynamics 365 Full AddIn (auch als Outlook Addin Client oder Dynamics 365 für Outlook bekannt) Diese Methode gilt bei Microsoft allerdings als veraltet. Die neue Integrationsmethode nennt sich Dynamics 365 App for Outlook. Diese Variante wurde in HTML neu aufgebaut. Bevor eine Outlookintegration stattfinden kann müssen Postfächer für den Benutzeraccount erstellt werden. In Dynamics kann die Postfacherstellung automatisch ablaufen. Das ist besonders interessant für neue Benutzer die beim hinzufügen dann automatisch ein Postfach von Dynamics bekommen. Dabei werden Einstellungen zu Berechtigungen/ Rollen mit angewendet.  
  
Serverside Synchronisation  
  
Der Vorteil der Serverside synchronisation ist das Emails durch Dynamics verwaltet werden. Die Aktualisierung findet statt ohne das Outlook geöffnet sein muss. Diese Aktualisierung geschiet alle fünf Minuten bzw. je nach einstellung regelmäßig. Das Anlegen von Postfächern kann dann überprüft werden und dies Erfolgt über eine Testmail (Bereich: Test & Enable Mailbox)  
  
Integration von Outlook  
  
Die Integration erfolgt über die Microsoft Outlook App. Diese wird per Push-Methode an die Clients übertragen. Auf den Rechnern muss daher keine separate Installation vorgenommen werden. Der Server zeigt dabei an ob die Installation läuft (pending) oder ob diese fertig ist. (completed) Die Installation kann bis zu 15 Minuten dauern.  
Installation:   
Settings🡪Dynamics App for Outlook🡪 alle berechtigten Benutzer die App zuweisen.

Synchronisation von Outlook und Dynamics  
  
Die Synchronisation kann in beide Richtung erfolgen. In Dynamics werden Kopien angelegt auf die andere Benutzer zugreifen können bzw. man kann sich mit Mobilen-Endgeräten synchronisieren damit man von überall her auf sein Postfach zugreifen kann. Bei der Synchronisation geht es wie angedeutet auch darum das Daten zwischen den Benutzern synchronisiert werden und berechtigte Benutzer auf Postfachdaten zugreifen können.

Aktivitäten  
  
Aktivitäten ergänze andere Einträge. Jede Aktivität liegt eine Klasse mit dem Namen ActivityPointer zugrunde. Activity Pointer besitzen vordefinierte Felder die nicht abgeändert werden können. Die Felder enthalten Informationen wie z.B. Start-, Enddatum oder Timestamp der letzten Änderung. Aktivitäten bauen auf dieser vordefinierten Struktur auf. Es gibt verschiedene Aktivitäten für unterschiedliche Anwendungsbereiche wie z.B. Tasks, Appointments oder Emails. Zudem können Aktivitäten in Kombination mit einer Timeline genutzt werden.  
  
Timeline  
  
Aktivitäten waren früher in einer Übersicht mit verschiedenen Tabs organisiert Dies wurde durch die Timeline ersetzt in der die Aktivitäten an einem Strang der Reihe nach angezeigt werden. Per Klick kann man sich genauere Informationen zu den einzelnen Aktivitäten anzeigen lassen. Informationen werden durch die Timeline platzsparend und zentral angezeigt. Die Timeline verfügt über eine Sortierfunktion so das Aktivitäten in aufsteigender oder absteigender Reihenfolge angezeigt werden können.  
  
Customer Service  
  
Beim Customer Service werden Informationen aus Kundenanfragen, Tickets u.s.w. in eine Entity die Case genannt wird übertragen. Ein Case bildet eine zentrale Informationquelle die zu effizienten Lösen von Problemen gedacht ist. Es ermöglicht z.B. das schnelle auffinden von Kontaktinformationen. Ein Case hat einen Lebenszyklus. Angefangen damit das ein Case dabei hilft einen Kunden zu identifizieren. Auf der Suche nach einem Kunden wird zunächst versucht diesen anhand des Firmennamens zu finden und falls nötig wird danach nach dem Namen des Kunden gesucht. Eine QuickView ermöglicht dann alle relevanten Kundeninformationen einzusehen ohne die Case-Entität verlassen zu müssen. Sollten dennoch weiterführende Informationen notwendig sein führen Links in der Maske zu weiteren Details.

Beziehungen zwischen Cases  
  
Fälle (Cases) bei denen ähnliche Probleme auftraten können miteinander in Beziehung gewsetzt werden ober bei gleichen Problemen zusammengefasst werden. Beim zusammenfassen oder verbinden von Cases wird ein Case als Parent (Hauptfall dem andere Fälle untergerordnet sind) gesetzt. Beim zusammenfassen ist wichtig das „gemerchte“ Cases nur einem anderen Case untergeordnet sind aber nicht gelöscht werden. Untergeordnete Cases bekommen einen Vermerk auf den Hauptcase. Wird ein untergeordneter Case geöffnet wird man zum Hauptcase umgeleitet. Im Hauptcase lassen sich dann alle untergeordneten Cases einsehen. Es kann auch ein neuer untergeordneter Case (Fall) über den Hauptcase (Hauptfall) angelegt werden.  
  
Resolving Cases  
  
Cases müssen letztendlich nach deren Lösung geschlossen werden. Besitzt ein Case untergeordnete Cases müssen diese ebenfalls geschlossen werden. Dies kann entweder so eingerichtet werden das ein Hauptcase nicht geschlossen werden kann wenn die untergeordneten Cases noch offen sind oder man richtet ein das untergeordnete Cases automatisch geschlossen werden wenn der Hauptcase geschlossen wird. Es wird empholen das untergeordnete Cases immer in irgend einer Form geschlossen werden müssen bevor der übergeordnete Case geschlossen wird. Ist dies nicht der Fall, bleiben untergeordnete Fälle weiterhin als ungelöste verweiste Einträge in der Liste bestehen.  
  
Cases Zusatzinformationen  
  
In Cases können Informationen wie Lösungsanspruch, Service-Level-Verträge oder Agreements eingetragen werden. Zudem kann hier eingetragen werden wer zu erst auf einen Fall reagiert hat. Dies ermöglicht die Zeitspanne dazwischen zu ermitteln, wodurch auch das Setzen von Deadlines möglich wird.  
  
Telefonsupport CTI (Computer Telefonie Integration)  
  
CTI sorgt dafür das zu einem Kunden Informationen auf dem Bildschirm gezeigt werden sobald dieser Anruft oder das ohne einen Hörer abzunehmen eine Nummer über das System gewählt werden kann. Die Einträge zum Kunden können direkt mit einem Case verbunden sein um schnell Informationen zum Fall des Kunden zu haben und diesen effizient bzw. zeitsparend lösen zu können.  
  
IVR (Interaktiv-Voice Response)  
  
Durch IVR kann ein Anrufer zu einer bestimmten Abteilung durchgestellt werden. Telefonsupport kann mit verschiedenen Techniken wie Microsoft Teams oder Skype for Business eingerichtet werden.

Queue’s (Warteschlangen)

Warteschlangen zeigen ähnlich wie eine Timeline auf einer Linie an, bei welchen Arbeitsschritt sich ein Prozess befindet und wer gerade daran arbeitet. Benutzer können sich aus Warteschlangen Elemente herausholen um diese zu bearbeiten. Wird ein Element als abgeschlossen gekennzeichnet wird dies beim entsprechenden Arbeitsschritt auf der Warteschlange vermerkt. Zu jedem Arbeitschritt auf der Warteschlange sind weitere Informationen wie z.B. die Kontaktinformationen des Sachbearbeiters enthalten. Auf diese Weise können Kunden sich direkt an die Person wenden die ihren Fall bearbeitet.  
  
Routing Rules

Über Routing-Rules kann festgelegt werden ob ein Fall eskaliert wird oder normal behandelt wird. Über diese Regelungen werden also logisch Abläufe eingestellt. So können Vorgänge auf verschiedene Warteschlangen zugeordnet werden. Pro Fall kannes nur eine Routingregel geben. Routingregeln können ausserdem nur nur auf Cases angewendet werden.

Queue’s und Workloadbalancing

In Warteschlangen kann eine Logik eingebaut werden die ein Loadbalancing der Arbeitslast ermöglicht. Nach jedem Arbeitsschritt kann entschieden werden welcher Mitarbeiter den nächsten Arbeitsschritt zugewiesen bekommt. Alle notwendigen Informationen zu einem Arbeitsschritt sind in der Warteschlange enthalten sodass z.B. Emailverkehr reduziert wird.

Knowledge Base

Bei Knowledgebase handelt es sich um eine Wissenssammlung für interne und externe Zwecke. Für interne Zwecke kann z.B. Prozesswissen abgelegt werden. Nach aussen können Artikel auf ein Portal gestellt werden das dem Kunde zugänglich ist. (Self-Service) Durch diese Artikel können Kunden unter umständen ihre Probleme selbst lösen.  
  
Knowledge Artikel  
  
Es gibt zwei Arten von Knowledge Artikeln. Zum einen KB-Articles und zum anderen Knowledge Articles. Letztere können nur in der neuen Ansicht nicht aber in der alten Classic UI angesehen werden. Die neuen Artikel unterstüzen eine Reihe neuer Features wie Rich Text Editor und eine Vorschaufunktion die den Inhalt als Webportalinhalt anzeigen kann. Wichtig ist ausserdem das die Inhalte der Knowledge Articles auch die Darstellung auf mobilen Endgeräten unterstützt.  
  
Knowledge Articles Lifecycle

Der Lebenszyklus eines Knowledge-Articles wird im Business-Process-Flow abgebildet. Zunächst müssen Titel, Thema, Autor angegeben werden und danach folgt der Content. Der ausgefüllte Artikel wird mit „Mark for review“ gesetzt damit dieser nochmal inhaltlich geprüft wird. Der Reviewer gibt den Artikel dann entweder zum publishen frei oder er leht den Artikel ab (rejected). Letzteres bedeutet das der vorhergehende Schritt wiederholt werden muss. Zu den Artikeln gibt es Analyseberichte der festhält wie die Artikel bei den Lesern ankommen und es gibt einen Feedbackbereich der den Lesern ermöglicht eigene Kommentare zu hinterlassen. Ein Wissensartikel kann auch mit einem Fall (Case) verbunden werden damit zu einem Problem schneller eine Lösung gefunden werden kann. Nach erfolgreicher Prüfung geht der Artikel in die Publishphase über. In der Veröffentlichungsphase kann ein Knowledge-Article mit einem Produkt verbunden werden sodas ein Kunde später diesen Artikel findet wenn er Probleme mit disem Produkt hat das dem Artikel zugeordnet wurde. Dem Artikel kann man vor dem veröffentlichen ein Ablaufdatum geben damit dieser gelöscht oder geupdated wird. Knowledge Article können am Ende ihres Lebenszyklus archiviert oder gelöscht werden. Im letzteren Fall landet der Artikel in einem Papierkorb aus dem er gegebenenfalls wiederhergestellt werden kann.

Subject & Category  
  
Wissensartikel können einem Subject oder einer Kategorie hinzugefügt werden. Ein Wissenartikel kann nur einem Subject hinzugefügt werden aber er kann mehreren Kategorien zugewiesen werden. Sowohl Subject als auch Kategorien können in einer Hierarchie eingebunden werden. Subjects können dabei anderen Subjects untergeordnet werden. Wenn ein Artikel einer Kategorie zugeordnet wird, wird im Hintergrund eine Entität namens Knowledgecategoriesarticle angelegt der die Verbindung beschreibt und dies kann wiederum für Reports verwendet werden.

SLA’s (Service Level Agreements)

SLA sind Unternehmenssupportrichtlinien die das Vorgehen bei Servicedienstleistungen beschreiben. Wird ein SLA erstellt wird dieser mit Detailseiten ergänzt. Die Detailseiten werden SLA KPI genannt (KPI = Key Perfomance Indicator) KPI’s sind damit die „wesentlichen Leistungsindikatoren“.  
  
Leistungsindikatoren

Leistungsindikatoren enthalten Informationen wie „Wer hat auf einen Fall zuerst reagiert?“ und „Wer hat das Problem gelöst?“. Zwischen diesen beiden Punkten wird die verbrauchte Zeit festgestellt. Zudem gibt es einen Timer der Auskunft darüber gibt ob ein Problem rechtzeitig gelöst wurde (successfull) oder der Timer vorher ablief (failed). Der Timer bei den Leistungsindikatoren legt damit für die Lösung eines Falls eine Deadline fest.

SLA und Entitlement (Anspruch)

Pro Organisation darf es nur eine SLA geben. SLA’s sind mit einem Entitlement verbunden. Entitlements verwalten die vertraglich basierenden funktionen. Sie enthalten Informationen darüber wann Verträge starten und enden. Verträge können auch pausiert werden wenn bestimmte Dienstleistungen nicht in Rechnung gestellt werden sollen. Drei weitere Felder die ausgefüllt werden können sind Channel (Wodurch kam der Kontakt zustande? Webseite, Telefon…) Products(Vertragsinformationen) und Contacts(Kontaktinformationen bzw. zulässige Kontakte zu der jeweiligen Leistung). Wenn ein Entitlement mit einem SLA verbunden wird muss dieser, damit er sich aktualisiert ersteinmal deaktiviert und wieder aktiviert werden. Wird ein Entitlement mit einem Case verbunden wird automatisch der Deadlinetimer aktiviert.

Power App Portals  
  
Power App Portals dienen dazu eine externe Seite aufzubauen. Der Kunde kann sich darauf einloggen und dort eigene Tickets erstellen, Anfragen an das Unternehmen stellen oder sich in den Knowledgebase-Artikeln umsehen.  
  
Sales  
  
Im Salesbereich wird der Lebenszyklus von Produkten oder Dienstleistungen verwaltet. Um Produkte oder Dienstleistungen zu verwalten muss eine Produktfamilie erstellt werden. Innerhalb einer Produktfamilie kann eine weitere Produktfamilie erstellt werden. Unter der Familie Früchte kann eine Familie Äpfel erstellt werden. Eigenschaften(Properties) werden nicht auf Produktebene hinzugefügt sondern auf Familienebene. Die Property Farbe kann auf der Ebene Früchte implementiert werden und vererbt an alle Familien die unterhalb der Familie Früchten angelegt wurden. In diesem Fall würde „Äpfel“ die auswahl für eine Farbe vererbt bekommen. In Äpfel selber könnten eine Property hinzugefügt werden von der alle untergeordneten Familien erben. Vererbte Properties können nicht auf der Vererbungsebene geändert werden. Sie können nur versteckt oder überschrieben werden. Für die Erstellung von Produktformularen und Familienformularen wird das selbe Formular verwendet. Will man eine Produktfamilie erstellen muss das Feld für Produktinformationen leer gelassen werden.

Produkte – Sales

Produkte werden Familien zugeordnet und erben Properties ihrer Familien. Man kann auch Produkte ohne eine Familenzuordnung erstellen. In diesem Fall ist der Einsatz von Properties nicht möglich und man kann nur auf Produktebene Felder für Angaben nutzen.

Unit und Unitgroups

Ein Unit ist die kleinste Mengeneinheit in der ein Produkt gekauft werden kann (z.B. eine Banane) Eine Unitgroup basiert auf einem Unit und bildet den Verkauf mehrerer Units in festen Mengen ab. (Palette, Kiste usw.) Ein 6er Pack Bananen basiert also auf dem Unit 1xBanane)

Cross-Sale

Zwischen Produkten können Beziehungen gesetzt werden. Ein Beziehungstyp ist dabei Cross-Sale. Mit Cross-Sale sind Produkte gemeint die sich von ihrer Art her gleichen, ergänzen bzw. verwand sind. (Ein Elektronikmarkt verkauft Laptops und bietet auch Laptoptaschen, Reinigungstücker oder Mäuse an)

Preislisten und Pricelistitems

Den Produkten sind Preislisten zugeordnet. Es können mehrere Prieslisten einem Produkt zugeordnet sein. Es wird empholen mindestens eine Default-Preisliste im System zu haben. Jeder Preisliste sind Pricelistitems zugeordnet. In den Pricelistitems können dann Rabatte je nach Mengeneinheit eingestellt werden. Nachdem alle Einstellungen in den Preislisten gemacht wurden kann das ganze veröffentlicht werden (publish).

Discountlists

Discountlists werden mit Pricelistitems verbunden. In Discountlists werden eine Start und eine Endmenge festgelegt. Innerhalb dieser Spanne werden Rabatte gewährt. (Begin-/ Endquantetie) Der Mengenrabatt kann entweder aud Basis der Stückzahl oder der Ausgabenhöhe gewährt werden.  
  
Übersetzung  
  
Lead = Kontaktanbahnung

Opportunity = Angebot  
  
Quote = Angebot  
  
Invoice = Rechnung  
  
Produktlebenszyklus

1. Kontaktanbahnung (Lead)
2. Vertriebsmöglichkeiten
3. Angebot (Quote)
4. Bestellung (Order)
5. Rechnung (Invoice)

Der Lebenszyklus ergibt sich aus den oben gezeigten fünf Schritten. Die Verwaltung des Lebenszyklus bzw. deren Aufbau in der Oberfläche richten sich danach. Aus dem Lead werden Daten für Accounts und Kontakte entnommen um die Weiterverarbeitung zu gewährleisten. Leads exestieren nur für eine bestimmte Zeit. In diesem Zeitraum entscheidet sich ob ein Lead qualifiziert werden konnte und damit zu einer Opportunity wird oder abgewiesen wird weil z.B. Kontaktangaben falsch sind oder ein Kontakt nicht zustande gekommen ist.

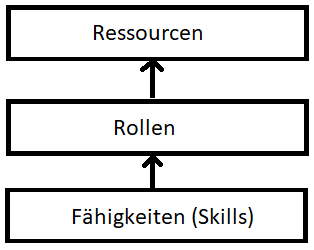
Quote

Auf Basis eines Angebotes kann ein weiteres Angebot erstellt werden indem es geklont wird. Das geklonte Angebot bekommt eine neue ID bzw. ist dann ein neues Angebot. Auf Grundlage eines Angebots kann zudem eine Bestellung/ Rechnung (Invoice) erstellt werden.

Competitors

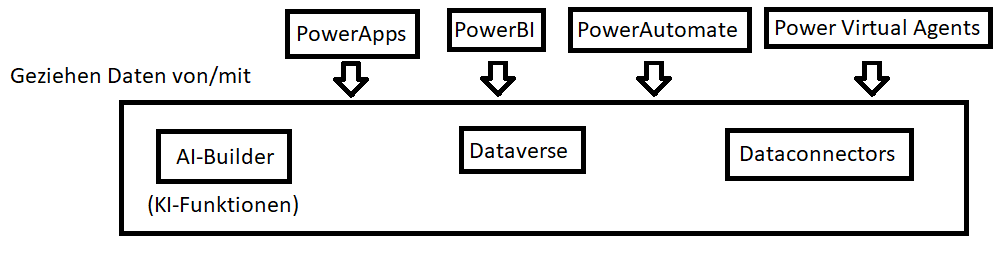
Zu einer Opportunity kann ein Competitors-Datensatz angelegt werden der Informationen wie Schwächen und Stärken der Konkurenz enthält.

Field Service und Ressource Sheduling

Beim Sheduling geht es um Zeitplanung und Arbeitskrafteinteilung. Eine Rolle kann eine Maschine (Gerät) oder ein Mensch sein der Fähigkeiten (Skills) zugeordnet sind. Rollen werden wiederum Ressourcen zugeordnet. Requirements sind Anforderungen eines Projekts denen Ressourcen zugeordnet werden die notwendig sind um diese anforderungen zu erfüllen. In Ressourcen können ausserdem Arbeitszeiten festgelegt werden in denen die Ressourcen verfügbar sind. Ein Arbeiter steht nicht 24/7 zur Verfügung sondern z.B. von 8:30 bis 17:00 Uhr. (Work Hours). Wie angesprochen spielt bei der Planung die räumliche Verfügbarkeit eine Rolle. Geräte, Maschinen, Mitarbeiter werden eher herangezogen wenn diese im Radius des Einsatzortes sind. Auf den Grundlagen von Ressourcen und Requirements können Workorders erstellt werden die in der Planungsübersicht (Schedule) angezeigt werden. Field Service ermöglicht auch die Routenplanung und Auftragsbestätigung per Unterschrift. Das Abschließen einer Lieferung führt dann zur automatischen Zuweisung eines neuen Ziels/ Auftrags. Für bestimmte Vorkommnisse gibt es die „booking Alerts“ die über Ereignisse per Nachricht informieren.

Power Platform Kurs

Die Power Platform besteht aus folgenden Bausteinen:

  
  
PowerBI – Business Intelligence

Interaktive Berichte die Daten aus verschiedenen Quellen gleichzeitig beziehen und auf einer Webobfläche visualisieren.

Power Virtual Agents

Power Virtual Agents sind virtuelle Chatbots die ca. 200 USD pro Woche kosten. Sie können einen Teil des Kundensupports übernehmen (z.B. Reklamationsverfahren), sodaß ein Anruf beim Support vermieden werden kann. Man kann dabei Chatbotvorlagen nutzen oder selber Chatbots zusammenbauen. Die Bedinoberfläche ähnelt der aus der LogicApp Designer Übersicht.

Anlegen von Umgebungen

Man kann über „Create“ mehrere Umgebungen anlegen. Beim erstellen hat man die Wahl zwischen Sandkasten, Try, Production usw. (Je nach Verwendungszweck). Nach dem Erstellen der Umgebung wird als nächster Schritt eine Datenbank angelegt in der Tabellen hinzu gefügt werden.  
  
Tip: Die Spalten können auf Timezone indipendent wählen damit keine automatische Übersetzung in das Landesformat vorgenommen wird.

Über PowerApps können zwei Arten von Oberflächen erstellt werden. (Canvas-Apps und modellbezogene Apps) Beide dienen der Darstellung der Daten auf dem Desktop, Tablet oder Mobiltelefon. Canvas-Apps erlauben es Daten, von unterschiedlichen Datenbankanbietern über Connectoren zu beziehen. Modellgesteuerte Apps beziehen ihre Daten immer aus dem Dataverse. Bei Canvas-Apps sind weitere Kosten einzuplanen wenn Connectoren, die man verwenden möchte mit einem Diamant markiert sind. Den Anbietern dieser Connectoren sind in diesem Fall Lizenzkosten zu entrichten. Canvas-Apps geben mehr Freiraum beim Erstellen der Oberfläche über den Editor. Der Editor erlaubt Komponenten per drag and drop zu platzieren. Rechts können im Editor je nach markierten Element, Abstände und andere Eigenschaften des Elements eingestellt werden. Auf der linken Seite stehen die verfügbaren Elemente die platziert werden können. Für die Oberfläche müssen Schnittstellen für die Datenquellen (zb. OneDrive) gewählt werden. Danach müssen Daten auf Elemente wie Textboxen gemappt werden. Für manche Schnittstellen ist beim einbinden noch ein Login beim jeweiligen Anbieter nötig. CanvasApps speichern Änderungen direkt in der Datenquelle. Ist dies eine Excel-Tabelle so werden diese z.B. auch dort gespeichert. Bei der Wahl der Datenquelle wird zwar von „import“ gesprochen aber die Daten werden nicht erst Anderenorts zwischengespeichert. Über Excel ähnliche Funktionen können in Canvas-Apps Suchen, Filter usw. implementiert werden:  
  
 Search(Tabellenname; Textbox1.Name;“Kategorie“;“Spaltenname“)  
  
Hinweis: In der deutschen Oberfläche werden Symikolons „;“ für Formeln als Trennzeichen genutzt während im englichen Kommas dafür verwendet werden.  
  
Wichtig ist das auch Anwender nicht nur Zugriff auf die App sondern auch auf die verwendeten Datenquellen haben müssen.

Modellgesateuerte Apps

Bei modellgesteuerten Apps muss kein mapping der Daten auf Steuerelemente vorgenommen werden, da die Datenquellen nur Dataverstabellen sein können die dem System bekannt sind. Anders als Canvas-Apps lassen sich mit modellgesteuerten Apps, Berechtigungen bis in das kleinste Detail vergeben. Es ist möglich Berechtingungen auf Tabellen, Spalten oder sogar Feldebene zu vergeben. Dies ist bei Canvas-Apps in dem Maße nicht möglich. Auch Zugangsberechtigungen können anders als bei Canvas-Apps umfangreicher eingestellt werden. So können Gruppen auf bestimmte Informationen eingeschränkt werden. (z.B. Mitarbeiterinformationen sind für Vorgesetzte nur bis zu einem bestimmten Punkt einsehbar) Bei modellgesteuerten Apps gibt es dafür weniger kreative Einstellungsmöglichkeiten. Im Gegensatz zu Canvas Apps ist hier nur ein simples Design möglich. Die Designs basieren auf Formularen und Views die wiederum auf den Tabellen aus dem Datavers basieren.

Hinweis:  
Es müssen alle Tabellen die voneinander abhängig sind implementiert werden, damit die App darauf zugreifen kann. Wenn Felder eines Formular z.B. zu den Besuchsdaten auch die Kontaktdaten darstellen sollen muss die App sowohl mit der Tabelle mit den Besuchsdaten als auch mit der Tabelle mit den Kontaktdaten verbunden werden. (Auch wenn es im Schema der Datenbank eine Beziehung zwischen den beiden Tabellen gibt)

PowerBI

Die Elemente von PowerBI bauen aufeinander auf. In PoiwerBI-Desktop werden die Daten gesammelt und in die PowerBI-Services werden diese Daten dann in einem Dashboard organisiert. Die Dashboards aus dem PowerBI-Services können dann auf PowerBI-Mobile übertragen werden.

Anmerkung: Berichte werden nicht in PowerBI-Mobil sondern in PowerBI-Desktop erstellt. Erst danach werden sie an PowerBI-Mobil übertragen.

Anmerkung: Sofern die Datenbank in der Cloud liegt kann die PowerBI Datenschnittstelle, die im Dataset angelegt sind so konfiguriert werden das diese regelmäßig aktualisiert werden.

Das Dataset ist Teil des Workspaces. Neben dem Dataset kann man von dort aus auf die Reports oder auf das Dashboard wechseln.

Tip: LogicApp/ Propertynamen herrausfinden  
  
Marketing & Leads  
  
Über die Marketingabteilung werden Leads erstellt. Wenn Leads einen bestimmten Punktewert (Score) erreichen können diese Qualifiziert werden. Aus einem Lead wird dann ein Kontakt. Die so erstellten Kontaktewerden dann zur Verkaufsabteilung (Sales) weitergeleitet. Der Vorteil Leads von den Kontakten getrennt zu halten ist das z.B. nicht alle Informationen für einen Kontakt notwenedig sind. Ausserdem wechselt der Besitzer (Owner) wenn ein Datensatz zu einem Kontakt gemacht wird. Der Datensatz wechselt vom Sales-Pool in den Kontakt-Pool. Für das Marketing ist er dann nicht mehr zugreifbar und die Verkaufsabteilung bekommt von da an Zugriff auf den Datensatz.

Marketing für Dynamics 365 ist eine zusätzliche App die nachinstalliert werden kann. Marketing unterstützt nicht das klassische Interface sondern nur das neue unified Interface. Die Marketing-App verfügt über einen eigenen Email-Server mit dem z.B. Massenmails verschickt werden können. Die Marketing-App verfügt über eine Subscription-Center. Dort sind Kunden eingetragen die Subscriber sind und Informationen per Email bekommen. Jeder Kunde kann sich selbst von dieser Liste entfernen. Die Subscriberliste kann mit Kontakten und Leads synchronisiert werden sofern dies in den Einstellungen aktiviert wurde.

Customer Journey

Customer Journeys basieren auf den Datensätzen der Kunden. Die Customer Journeys sind in einem Workflow organsisiert von dem aus Aktionen ausgelöst werden. Auf Basis des Kundenverhaltens wird z.B. eine Aktion gestartet. An einem Produktvorschlag könnte sich eine Umfrage anreihen u.s.w. die per Mail an den Kunden geschickt werden.

In diesem Workflow gibt es verschiedene Blöcke(Aktionen) die verwendet werden können:

* Marketing-Emails die an Kunden versendet werden. Diese Mails können dann andere Aktionen enthalten.
* Marketing Pages können als Block einer Mail hinzugefügt werden und sind als Link zu einer Webseite (z.B. Produktseite) zu verstehen.
* In Mail können weiterhin „Link-Blöcke“ zu Events, Surveys(Umfragen), Activities gesetzt werden.
* Es ist ausserdem möglich mit Aktionen weitere Workflows zu starten.
* Über Segment kann einer Gruppe von Kunden selektiert werden.
* Über Record-Updates kann ein neuer Kundenkontakt hinzugefügt werden.
* Der Scheduler startet nach einem Zeitplan eigestellte Aktionen.
* Trigger lösen nach angegebenen Bedingungen im Workflow Ereignisse aus.
* Splitter (und Splitter-Branches) ermöglichen es Verästelungen in den Workflow einzubinden.

Leads werden über Lead-Pages eingetragen. Wenn ein Leadscore hoch genug ist, wird der Lead als „Sales Ready“ eingestuft. Je nach vorhandenen Leaddaten kann ein Leadscore erhöht oder veringert werden. Ist ein Lead „Sales Ready“ kann dieser der Verkaufsabteilung als Kontakt übergeben werden.

Event

Events(gemeint sind hier Veranstaltungen) können in ein Eventportal eingestelt werden und ermöglichen Internet und Externe sich über Events zu informieren. Eine Einladung kann über einen zugesendeten QA-Code von statten gehen. Auf diese Weise können sich z.B. Kunden für ein Event registrieren.

Security Model

Sicherheit wird über zwei Systeme gewährleistet. Zum einen wird die Sicherheit über Azure Active Directory gewährleistet . Zum anderen verfügt Dynamics über eine Absicherung der Daten über Benutzerrollen. Die grundsätzliche Sicherheitsphilosophi bei Dynamics 365 ist das man so viele Berechtigungen bekommt um seine Rolle zu erfüllen und so wenig Berechtigungen wie nötig. Die Sicherheit von Dynamics 365 wird auf der zugrunde liegenden PowerPlatform realisiert. Unter den erweiterten Einstellungen können hier für die Benutzer diverse Rollen festgelegt werden. (Erweiterte Einstellungen 🡪 Security) Neben der Methode über die klassische Ansicht kann man auch Rollen über: Umgebung 🡪 Settings einstellen. Für diesen Weg muss man allerdings globaler Administrator sein. Über Admin.PowerPlatform.com kann man die Umgebung starten um Rollen und Sicherheitsregeln zu implementieren. (Die Arbeit mit der neuen Oberfläche wird dafür empfholen)

Struktur

Die Berechtigungsstruktur besteht aus aus einer Organisation als Wurzel von dem BusinessUnits abzweigen. Dabei können BusinessUnits weitere untergeordnete BusinessUnits haben. Untergeordnete BusinessUnits können damit ChildUnits übergeordneter Units sein. Berechtigungen werden an untergeornete Units weitervererbt. Zu beachten ist das nur in den Units in denen die Berechtigungen festgelegt wurden können diese auch geändert werden.   
  
Tip: Um Fehler bei der Berechtigungsvergabe zu vermeiden wird es empfohlen eine Basisrolle festzulegen die über minimale Berechtigungen verfügt. Die Basisrolle kann dann als Vorlage für alle weiteren Rollen genommen werden in dem man diese kopiert. Jede neue Berechtigung sollte einzeln getestet werden um ihre Auswirkung gezielt zu prüfen. Mehrere Einstellungen zusammen zu Testen führt unter umständen zu unklaren Ergebnissen. Abgesehen davo sollte man immer mit einem, für den Test vorgesehenen Konto und nicht mit dem Systemadministratorkonto testen.

Ownerschaft

Besitzer (Owner) einer Ressource kann ein einzelner Benutzer oder ein Team sein. Teams werden einem BusinessUnit zugeordnet. Auch Benutzer eines anderen BusinessUnits können in so einem Team zugeordnet werden. Einem Team können Benutzerberechtigungen zugewiesen werden die für alle Benutzer des Teams gelten.

Share

Benutzer können Shareberechtigungen vergeben die innerhalb der Organisation nach unten vererbt werden. Über die Organisationsstruktur hinweg werden Berechtigungen an Benutzer verteilt. Das Sicherheitsproblem hierbei ist das dies vom Adminitrator nicht nachvollzogen werden kann. Man weiss nicht wer , was und mit wem geshared wurde und wohin das Ganze vererbt wurde. Es werden maximal allerdings nur Rechte vergeben die man maximal selber an einer Ressource hat. Der bessere Weg ist es jedoch solche Dinge über Access-Teams zu Regeln. Access-Teams erlauben dem Administrator das Teilen von Daten nachzufolziehen.

Field Security

Für einzelne Felder können andere Berechtigungsstufen eingestellt werden. So kann verhindert werden das Benutzer die nur berechtigt sind einen Teil einer Tabelle zu ändern auch Felder ändern nicht in deren Aufgabenbereich liegen. Security Roles und Security Fields gehören zu unterschiedlichen Profilen und sind daher unabhängig voneinander zu behandeln. Hat man sich davon überzeugt das alle Berechtigungen korrekt vergeben wurden kann ein Deployment durchgeführt werden.

Teamtypen

Bei der Erstellung von Teams kann man einstellen welchen Teamtyp man wählen möchte. Bei Owner-Teams hat jedes Mitglied die selben Berechtigungen wie der eigentliche Besitzer. Accessteams sind die bessere Alternative als die Share-Berechtigungsvergabe auf Benutzerebene. Accessteams haben nicht den Besitz an einer Ressource, stattdessen kann mit Accessteams genau eingestellt werden welche Berechtigungen an einer Ressource die Teammitglieder haben. Accessteams werden keine Security-Rollen zugewiesen. Eine Zuweisung neuer Benutzer kann hier auf Benutzerebene erfolgen.

Automatische Berechtigungszuweisung

Berechtigungen können über Templates gesetzt werden, sodass jedem neuen Datensatz automatisch ein solches Template zugewiesen wird. (Datensatz bzw. auch Records genannt)

Hierarchical Security

Anstatt die Sicherheit über Rollen festzulegen kann die hierarchische Struktur einer Firma als Model für die Berechtigungen dienen. Die Position einer Person innerhalb der Firma werden Berechtigungen gewährt z.B. Geschäftsleitung, Abteilungsleiter oder Marketingmitarbeiter. Den Positionen können dann Benutzer zugewiesen werden. Berechtigungen einer Position können unten vererbt werden.

Mobil Application

Man kann zwar eine Web-Application auch auf einem mobilen Endgerät nutzen doch der Vorteil von mobilen Applicationen ist das der Bildschirm an die Bildschirmgröße des Endgerätes angepasst wird. Ausserdem sind bei mobilen Endgeräten Buttons empfehlenswerter die mit den Fingern bedienen kann, da man hier nicht auf die präzision einer Maus verzichten muss. Mobil Apps können im Gegensatz zum Webapplikationen offline laufen, da hier meist eine aktive Internetverbindung benötigt wird. Bei der Erstellung von mobilen Applikationen kann das Classic UI nicht verwendet werden. Der Mobil Client kann über [Home.Dynamics.com](http://Home.Dynamics.com) oder Portal.Office.com geöffnet werden. Ein entsprechender Login wird dabei benötigt.  
  
Mobil First Apporeach  
  
Der Mobil First Apporeach ist eine Design-Regel die besagt das eine App zuerst für den kleinsten Bildschirmbereich konzeptioniert wird und von dort aus dann für größere Bildschirme angepasst wird. Es ist wesentlich einfacher eine App die für kleine Bildschirme gedacht ist auf einen großen Bildschirm anzupassen als anders herum.

Reporting, Views, Charts und Dashboards

Im nächsten Abschnitt geht es um die Aufbereitung gesammelter Informationen/Daten in einer Übersicht. Hiermit ist es z.B. möglich sich schnell darüber zu infieren ob Geschäftsziele erreicht wurden. Für die Aufbereitung der Daten muss ein Datenmodel zugrunde liegen. Dies kann z.B. ein Model sein das mit UML erstellt wurde.

Views

Views fassen Tabellendaten in einer Übersicht zusammen in der sich die Felder, auf für den Fall aussagekräftige Informationen beschränkt wird. Die Sichten (Views) bilden die Grundlage für Dashbords. Die Kacheln des Dashbords enthalten die verbundenen Views. Bei Views wird zwischen Public/System Views und Personal Views unterschieden. Personal Views werden von einem Benutzer erstellt und können für anderen Benutzern freigegeben werden. Personal Views liegen im persöhnlichen Bereich des Erstellers. Public und System-Views sind „öffentlich“ in der Organisation sichtbar. Personal Views werden mit der Toolbar des Unified Interface erstellt, während Systemviews mit dem View-Designer erstellt werden. Weiterhin kann eine View auch über die Filteransicht erstellt werden. (persönliche View/ alternative View anstatt über die Toolbar des Unified Interface)

Charts

Charts sind Diagramdarstellungen (mit Legende u.s.w.) und können über das Datavers erstellt werden, indem man dort Daten bezieht. Die persönlichen Charts erhalten ihre Daten über das Unified Interface.

Dashbords

Dashbords vereinen Views und Charts zu einer Übersicht die es dem Benutzer erlaubt sich eine Übersicht zu verschaffen. Es wird zwischen den classic Dashboards und den neueren interactive Dashboards. Die classic Dashbords können dabei auch über das unfied Interface gelesen werden. Die interactive Dashbords verfügen über erweiterte Funktionalitäten in den Bereichen sortieren, filtern u.s.w.

Erstellen von Dashboards

1. Festlegen grundlegender Informationen  
   - Name des Dashboards  
   - Angeben der Verbindungen (Charts 🡨🡪Tabellen)  
   - Angaben der Views die Teil des Dashboards sein sollen.  
   -Einstellen des Filters und der Timeframes.  
   Hinweis zu Timeframes: Die Zeitachse kann so eingestellt sein das der Anschein bei den Nutzern erweckt werden könnte das, das Dashboard keine Daten enthält. Da Benutzer den Zeitabschnitt selbst einstellen können den diese betrachten kann es hier zu Einstellungen kommen die keine Daten anzeigen weil ein zu kleiner Ausschnitt auf der Zeitleiste gewählt wurde.
2. Hinzufügen von Streams bzw. das Hinzufügen von Views nach denen sortiert werden kann wie z.B. alle aktiven Fälle oder alle Telefonanrufe.
3. Im letzten Schritt wird das erstellte Dashboard gespeichert, validiert und am Ende geschlossen. Abgesehen vom speichern muss das Dashboard „gepublished“ werden. Man kann dabei festlegen ob das Dashboard für die ganze App oder nur für einen bestimmten Bereich veröffentlicht wird.

Forms, Views und UI Customization

Pflichtfelder werden durch einen roten Stern markiert.  
Optionale Felder werden im CRM als blauer Stern markiert und sind als Empfehlung gedacht.

CRM-Datentypen

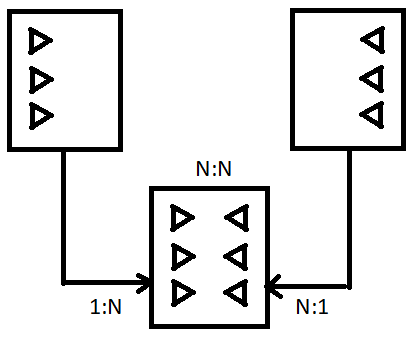
|  |  |
| --- | --- |
| Singleline of Text | String |
| Optionset (Auswahlfeld) | Einzelauswahl (intern: Zahlen-Text Zuweisung) |
| Multiselectfeld | Mehrfachauswahl |
| Two Option Field | Ja/ Nein Wert (Boolean) |
| Image Field | Base64 String für Bildspeicherung |
| Whole Number | Ganzzahl bzw. Integer |
| Floating Point Number | Gleitkommazahl mit Genauigkeitsangabe |
| Decimal Point Number | Gleitkommazahl |
| Currency | Währung + Währungseinheit |
| Multiple Lines of Text | Format für umfangreichen Text |
| Date and Time | Datums/ Uhrzeitfeld |
| Lookup Feld | Mit Daten verlinktes Auswahlfeld |
| Rollup Fields | (1:n) Felder mit einer Verbindung zu einer Tabelle |
| Calculated Fields | Berechnungsfelder die Datenver/bearbeiten  (z.B. Datum + 2 Tage) |

Beziehungen (1:N, N:1 und N:N)

Bestimmte Felder wie das Lookupfeld setzen zur Darstellung von Daten Beziehungen wie 1:N fest. Bei N:N Beziehungen nutzt man Intersect b.z.w. Hilfstabellen. Das Prinzip hinter einer N:N Beziehung lässt sich so verdeutlichen:

Mehrere Autoren schreiben an einem Buch und mehrere Bücher sind von einem Autor.  
Mehrere Patienten haben einen Arzt und mehrere Ärzte haben einen Patienten.

Um doppelte Datensätze zu vermeiden werden z.B. Ärzte und Patienten jeweils in zwei getrennten Tabellen geführt und eine dritte Tabelle weist Ärzte und Patienten einander zu. Beide Tabellen haben zur Hilfstabelle eine 1:N Beziehung mit der Hilfstabelle. Die Hilfstabelle bildet die N:N Beziehung ab:



Beziehungen zu den Feldern aus anderen Tabellen können in den Einstellungen einer Tabelle vorgenommen werden. Hier gibt es Felder und Tabellennamen an die verbunden werden sollen und die Art der Verbindung. Hilfstabellen werden im CRM nicht direkt angezeigt. Nach der Erstellung von Beziehungen werden diese generiert und versteckt.

Formulare für Tabellen werden oft in Chartform angelegt, wenn diese auf ein Dashboard angelegt werden sollen. Die Mainform wird gewählt wenn jedes Feld der Tabelle editierbar sein soll. Die Quickview zeigt Details eines Datensatzes an und die QuickCreateForm ist zum schnellen Eintragen von neuen Datensätzen da. Letztere beschränkt sich auf die wichtigsten Felder.

Processes

Der Businessflow ist im CRM im oberen Bereich als Fortschrittsbalken zu sehen.Er zeigt dem Benutzer bei welchen Schritt eines Prozesses er sich befindet. Jeder Prozessabschnitt ist mit einem Formular verbunden. Die eigentliche Erstellung eines Processflows findet in einem Editor statt. Einzelne Schritte werden in Blöcken dargestellt welche miteinander verbunden sind. Das Ganze bildet ein Ablaufdiagramm das sich von links nach rechts erstreckt und über verzweigungen verfügen kann. Die einzelnen Blöcke werden in der grafischen Ansicht des Processflows als Abschnitte dargestellt. Die Blöcke sind in einem Editor der Zugang den eigentlichen Formularen. Wie oben beschrieben gibt es im Processflow eine Businesslogic die dafür sorgt das je nach Situation, sich der Ablauf verzweigen kann. Je nach Datenquelle (Web oder Dataversedatenbank) kann z.B. der Ablauf geändert und andere Prozessschritte enthalten. Neben dem Linienverlauf erkennt man eine Verzeigung zusätzlich an dem X-Symbol. Wichtig bei der Erstellung eines Processflows ist auch die Zuweisung der Sicherheitsrollen. Unzureichende Zugriffsberechtigungen können dazu führen das man Arbeitsschritte im Businessflow nicht mehr sehen und ansteuern kann. Über „Edit Security Roles“ können die Sicherheitsrollen für die Arbeitsschritte eingestellt werden. Man kann aus einem Businessflow eine Aktion aufrufen oder einen Power Automate Flow starten. Diese Funktionalitäten können in einem Businessflow fest eingebettet sein.

Business Rules

Geschäftsregeln werden auf Formulare und dren Inhalte angewand. Geschäftsregeln lassen sich bequemer anwenden als Skripte dafür zu verwenden. Man kann z.B. mehrere Felder eines Formulares auf einmal leeren, Bedingungen an Felder koppeln sodass Felder ein/ausgeblendet oder ge/entsperrt werden. Delder können durch Geschäftsregeln validiert werden und es können Fehlermeldungen oder Empfehlungen ausgegeben werden. Businessrules können auf ein spezielles Formular, auf alle formulare oder auf eine Tabelle angewand werden. Dieses festlegen nennt man auch Bereich (Scope). Innerhalb des Formulares im Menüband können Geschäftsregeln direkt angewendet werden.