Teams App Development

Learnings, Architektur und Deployment einer Custom App

Tobias Kritten

Cloud Security ist meine Leidenschaft

Seit über 15 Jahren helfe ich Unternehmen beim sicheren Einsatz von Microsoft-Technologien: zu Beginn mit meinem eigenen Systemhaus, später durch das Hosting von Exchange, SharePoint- und Microsoft-basierten Applikationen und heute schwerpunktmäßig in der Cloud durch den Einsatz von Microsoft Security-Lösungen und Konzepte wie Azure AD, der Defender-Produktreihe oder Zero Trust.

Cloud Security Portfolio (Auszug)

- Einführung von SOC-Prozessen und Tools (u.a. Defender, Azure AD, Sentinel, Intune, …)
- Durchführung von Security-related Best Practice Workshops und Audits
- Einführung von und Beratung rund um Zero Trust-Konzepte

Cloud Automation Portfolio (Auszug)

- Automatisierung von Inter-Service-/Applikations-Prozessen
- Vernetzung von Microsoft 365 mit Gebäude-Automatisierung, Sicherheitssystemen und weiteren 3rd Party-Systemen
- Umfangreiches Tool-Set (Azure Functions, Logic Apps, Power Apps, C#/.NET, etc.)

azure geek.io

AGENDA

Einführung

Teams Custom App – wieso ich mich dafür entschieden habe und welche Use Cases umgesetzt werden. Außerdem: wie ist eine Teams App aufgebaut?

Architektur

Aus welchen Komponenten besteht eine Teams App, was ist ein Frontend und Backend und wie wird eine Teams App eigentlich betrieben?

Security & Authentifizierung

Welche Rolle spielen Security, Authentifizierung und Zero Trust in einer Teams Custom App?

Deployment

Wie gelangt der Source Code der App in die Infrastruktur und wie sorgen wir dafür, dass die App ohne nennenswerte Wartungsfenster aktualisiert wird?

Einführung

Teams Custom App – wieso ich mich dafür entschieden habe und welche Use Cases umgesetzt werden. Außerdem: wie ist eine Teams App aufgebaut?

Use Cases: was wollen wir eigentlich erreichen?

Wifi Voucher-Management für Angestellte und Gäste

Der Zugriff auf das Gäste-WiFi wird per Voucher-Code geregelt. Ein Voucher-Code kann bisher nur manuell durch das IT-Team vergeben warden. Die App löst folgende Use Cases:

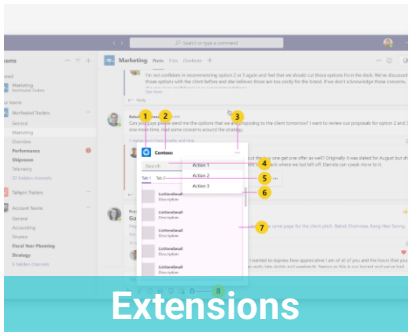
- Verwaltung von Mitarbeiter-Voucher Codes
- Sicherstellen von maximal 1 Voucher je Mitarbeiter
- Verwaltung von Gäste Voucher-Codes
- Statusanzeigen und Voucher-Austausch

Zugangskontrollsystem-Management

Das zentrale Zugangssystem regelt den Zutritt zu Gebäuden. Zur Verbesserung der Abläufe bietet die App folgende Funktionen:

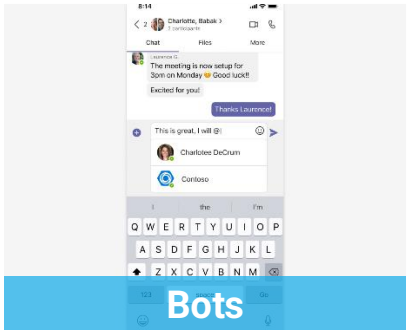
- PIN-Verwaltung durch den Nutzer selbst
- Anzeige von erlaubten Zutritts-Profilen
- Anzeige von Zutritts-Versuchen und ggf. Verständlichen Fehlermeldungen
- Zukünftig: Besucher-Verwaltung

Teams App: welche Typen existieren und welche nutzen wir?



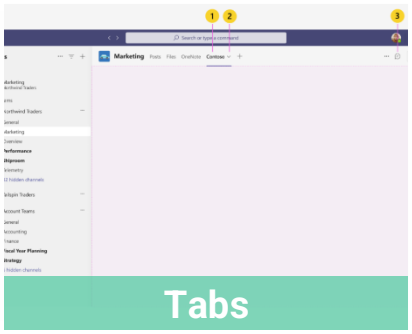
Extensions

Messaging-Extensions erweitern das Nachrichten-Eingabefenster und Chats durch zusätzliche Menü-Punkte, die das Einblenden von individuellem Content und das Auslösen von Aktionen – z.B. der Nutzung eines Web Service – erlauben.



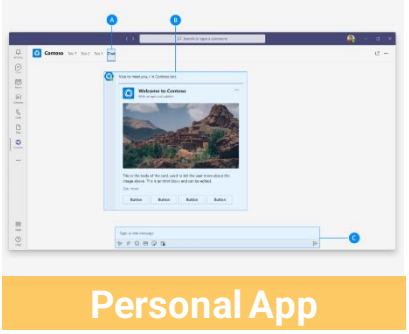
Bots

Ein Bot interagiert durch Chat-Nachrichten mit Benutzern. Die Interaktion basiert in der Regel auf mittels Machine Learning antrainierten Antworten und findet häufig im Customer Service Anwendung.



Tabs

Teams kann jede beliebige Website direkt in einem Kanal, Gruppen-Chat oder einer Personal App darstellen. Unterstützt die Website zusätzlich die Teams-Funktionen, ist ein Austausch von Daten (z.B. Authentifizierung) mit der Tabs-App möglich.



Personal App

Eine Personal App kann aus einem Bot, einem Personal Workspace mit z.B. mehreren Tabs oder beidem bestehen. Im Gegensatz zu anderen App-Typen ist die Personal App über ein Icon im Teams-Menü erreichbar und der Scope der App ist immer an den jeweiligen User gebunden.

azure geek.io

Sonderfall: PowerApps – Klicken statt Coden

01

Klicken statt Coden

Microsoft PowerApps ist ein zusätzlicher Service, mit dem Power User Apps auf einem “Canvas” grafisch zusammenstellen können

02

Datenanbindung im Excel-Stil

Die Anbindung von Daten erfolgt über vordefinierte oder individuelle Connectoren, die über Formeln angesprochen werden

03

Schnell und einfach, aber kostenpflichtig

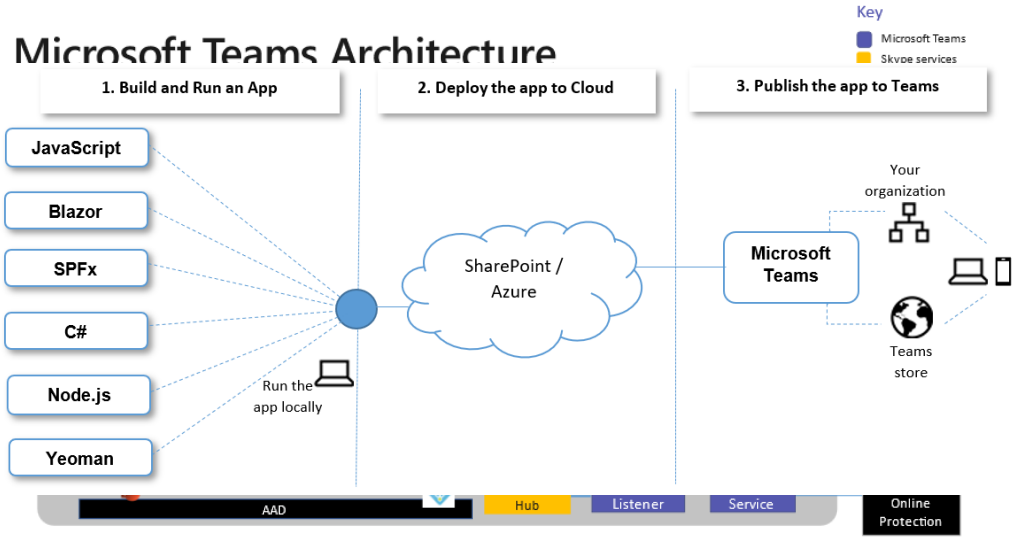
Mit PowerApps lassen sich auch durch wenig erfahrene Entwickler Apps im Handumdrehen bauen – Zugriff haben allerdings nur zahlende “Konsumenten”

Architektur

Aus welchen Komponenten besteht eine Teams App, was ist ein Frontend und Backend und wie wird eine Teams App eigentlich betrieben?

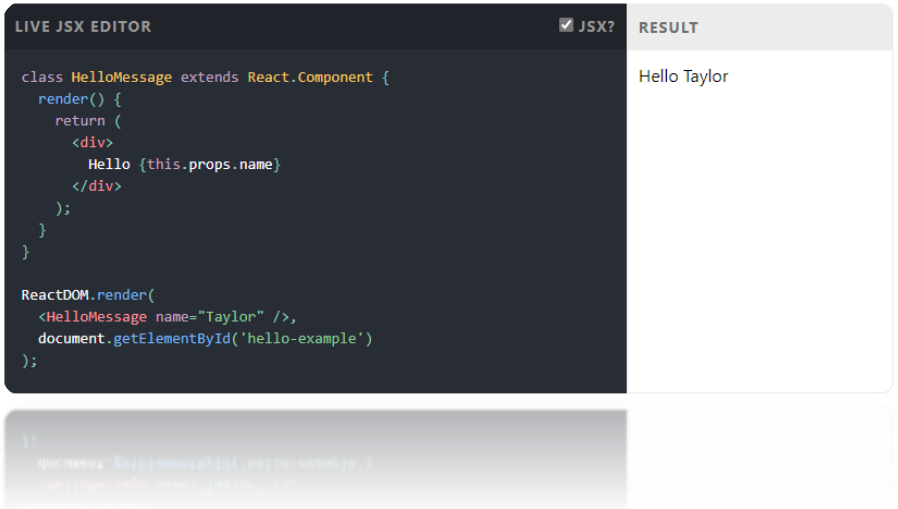
Frontend: was ist eigentlich eine Teams Personal Tab App?

- Teams basiert auf **Electron**, einem Open Source-Projekt, das plattformunabhängige Apps mit HTML, CSS und Javascript ermöglicht
- Die Teams-App kommuniziert zur Bereitstellung der Funktionen mit unterschiedlichen **Backend-Services**
- Tab Apps sind **statische Web-Seiten**, die im Teams-Client angezeigt werden
- Ein Tab entspricht dabei üblicherweise einer Web-Seite (**Single Page Application**)
- Unterschiedliche Frameworks können für die Erstellung der Web-Seite(n) verwendet werden, u.a. Plain HTML, JavaScript/React, Blazor/C#, Yeoman, SPFx
- Austausch von Daten zwischen Teams und der Tab App mittels **Teams Javascript Library** oder Teams FX
- Hosting, also **Speicherort** der Web-Seite, ist **außerhalb** von Teams und kann ein z.B. eigener Webserver oder ein Azure Storage Account sein



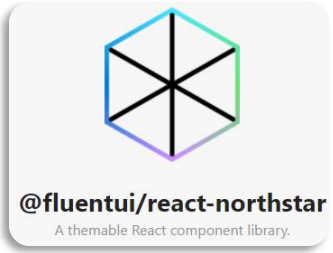
Frontend: React – ein kleiner Exkurs

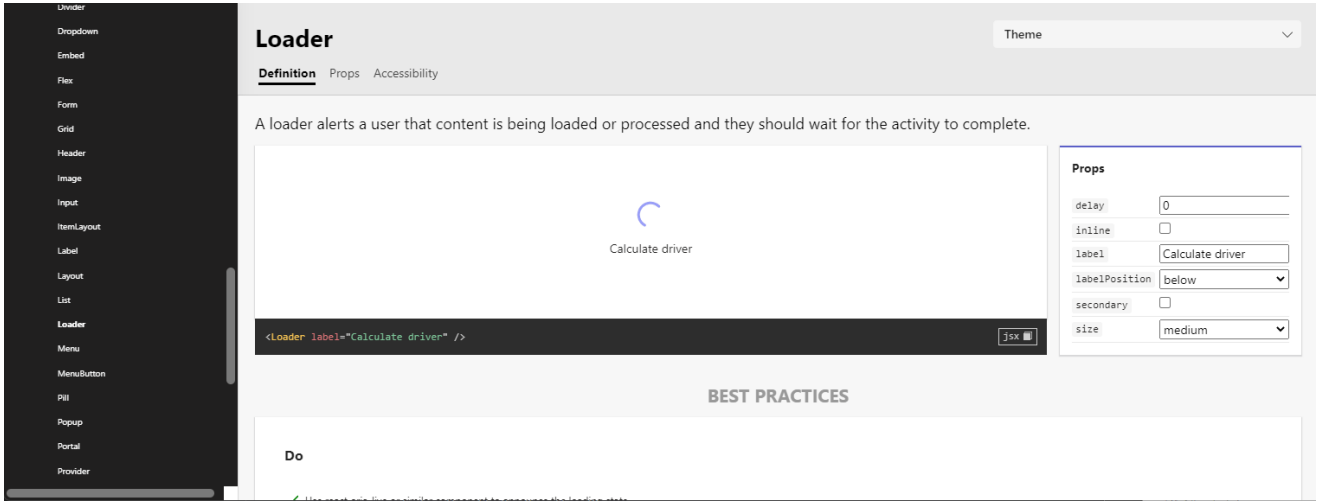
- JavaScript-Bibliothek zur Erstellung von User Interfaces
- Deklarativ:
  Code definiert das User Interface und React aktualisiert Inhalte automatisch – ohne, dass codeseitige Mechanismen dafür implementiert werden müssen
- Komponenten-Basiert:
  Gekapselte Komponenten erleichtern die Entwicklung umfangreicherer Applikationen
- Einfach zu erlernen, weit verbreitet, gut dokumentiert
- Bekannte Konzepte aus klassischer Entwicklungs-Welt möglich (TypeScript erlaubt typensicheres JavaScript, muss aber kompiliert werden)



Frontend: FluentUI – ein kleiner Exkurs

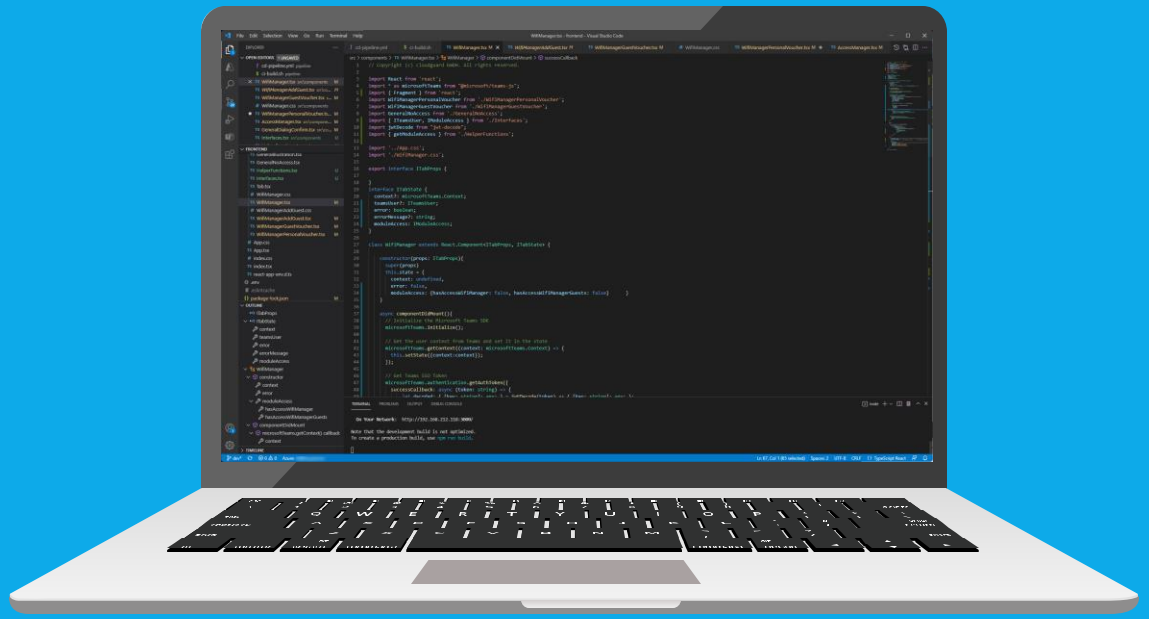
- UX-Library zur Erstellung von cross-plattform Apps
- FluentUI – React-Northstar: Library basierend auf React / JavaScript, die Teams-spezifische Design-Komponenten bereitstellt
- Sorgt für den typischen „Teams“ Look & Feel in Web-Anwendungen und unterstützt Teams Theaming
- Bietet eine interaktive Demo-Site mit allen Komponenten und der Möglichkeit, die Komponenten in einer Sandbox zu testen



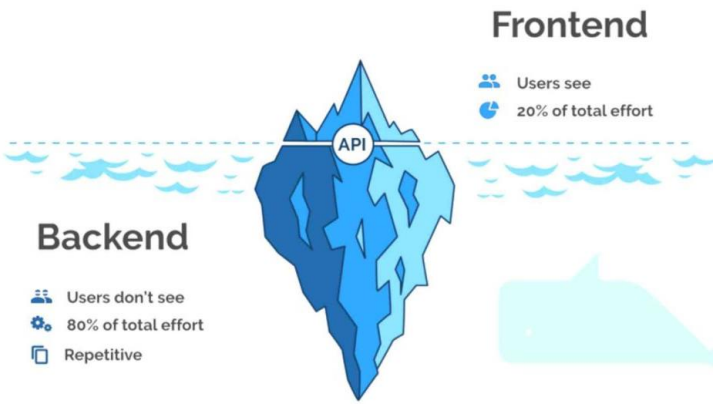


DEMO

- Kurze Einführung in den Frontend-Code
- Frontend im Browser
- React-Components



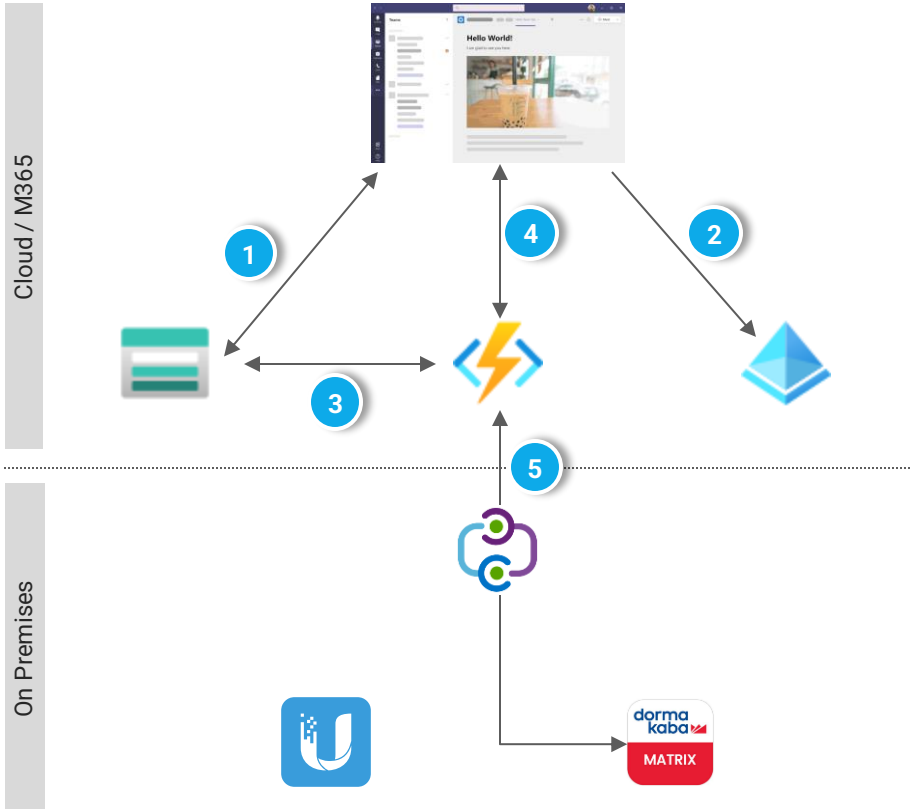
Backend: Wofür benötigt man ein Backend?



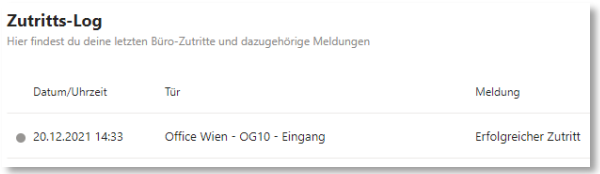
Quelle: https://sidtechtalks.in/designer-vs-frontend-vs-backend-in-web-development/

- Frontend dient zur Visualisierung der vom Backend abgerufenen Daten, Umsetzung der Bedien-Logik, Verifikation von Eingaben und Weitergabe der Eingaben an das Backend
- Das Backend verarbeitet eingehende Daten des Frontend und gibt Daten an das Frontend weiter, meist in einem standardisierten Format wie JSON
- Abhängig vom Zweck des Backend werden Daten an weitere Systeme über deren „Backends“ (API) weitergegeben
- Unterscheidung findet häufig wegen der unterschiedlichen Technologien / Programmiersprachen statt: Frontend ist meist Script-lastig (HTML, CSS, JavaScript, etc.), während Backends eher in „echten“ Programmiersprachen wie Java, .NET, etc. entwickelt werden

OfficeManager App: beispielhafter Flow „Zutritte anzeigen“



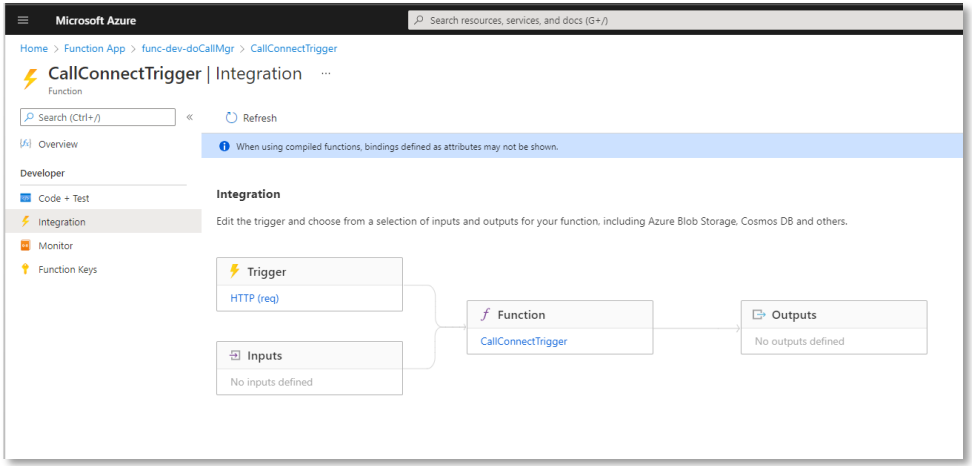
1. Bei Öffnen der Teams Tab-App: Abruf der statischen Tab App Web-Site von einem Storage Account mit statischem Webhosting
2. Authentifizierung an Azure AD
3. Abruf von Informationen im Table Storage des Storage Account
4. Aufruf der Function App-Methode „GetAccessLog“
5. Aufruf der MATRIX Pro-API über eine Hybrid Connection (On Premises)



azure geek.io

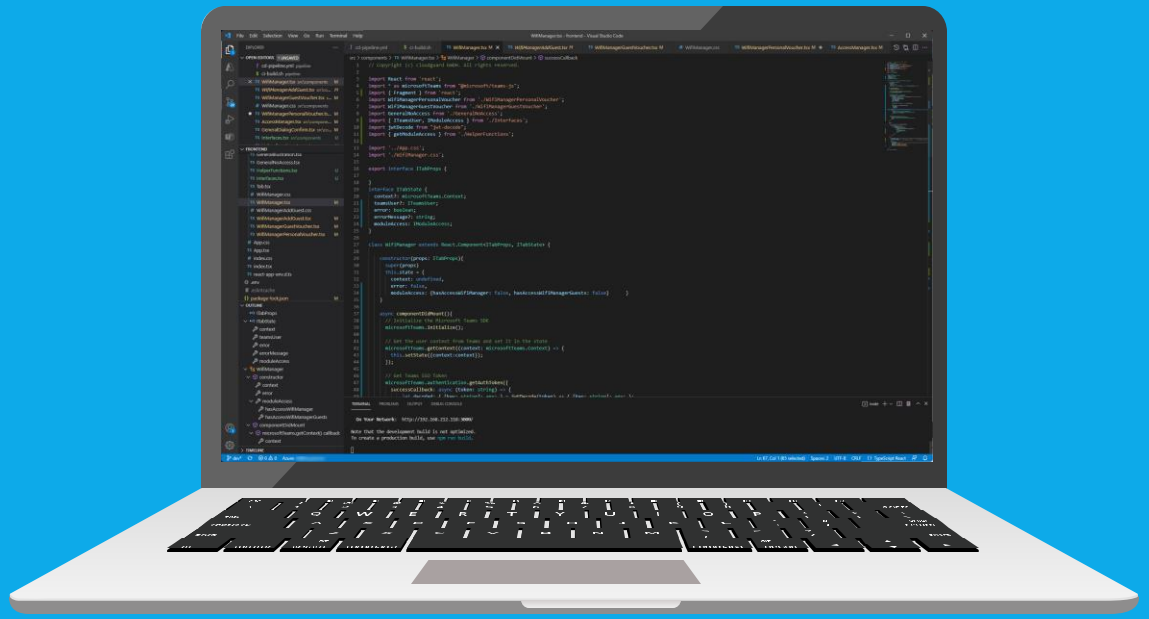
Exkurs: Azure Functions – Serverless Backend

- Azure Functions bieten eine Ausführungsumgebung für Programmcode ohne die Notwendigkeit, eigene Infrastruktur betreiben zu müssen
- Umfangreiche Ausführungsumgebungen werden unterstützt, z.B. C#, JavaScript, F#, PowerShell, Java
- Code muss immer durch einen Trigger gestartet werden, z.B. per HTTP Call oder über eine Message Queue
- Integration in On Prem-Netzwerke ist z.B. über VPN oder Hybrid Connections möglich
- Integration in Azure Active Directory nimmt dem Entwickler Authentifizierung und Authorisierung ab
- Deployment mit einfachen Tools aus Azure DevOps oder Github möglich



DEMO

Our mission is to make you stand apart in each of your presentation whether it is related to your project management, planning, sales, branding, or anything in this regard.

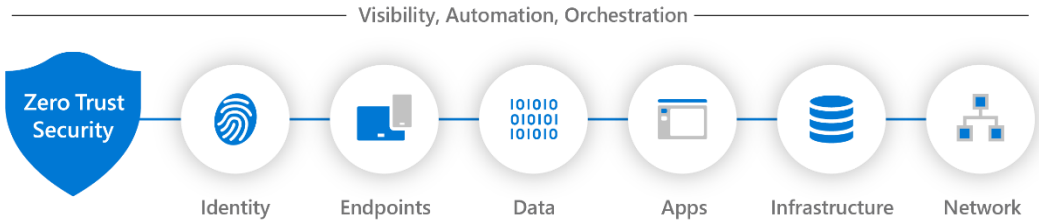


Security & Authentifizierung

Welche Rolle spielen Security, Authentifizierung und Zero Trust in einer Teams Custom App?

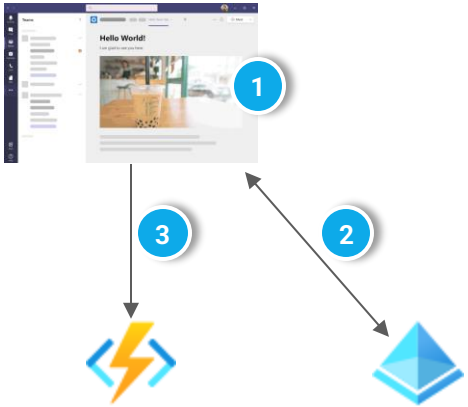
Security, Authentifizierung, Zero Trust, … wieso das?

- Teams Apps sind direkt aus dem Internet erreichbar – es muss sichergestellt sein, dass nur autorisierter Zugriff erfolgt
- Anstelle der Entwicklung eigener Identity und Access-Provider zu entwickeln, wird der Einsatz bewährter Technologien wie Azure Active Directory empfohlen
- „Zero Trust“ ist ein modernes Sicherheitsmodell, das perfekt zu Teams Apps (und weiteren Cloud Services) passt: im Gegensatz zu konventionellen Modellen wird jede Anforderung neu überprüft und aus folgenden Bausteinen besteht:



- Als einer der wichtigsten Bausteine wird “Identity” betrachtet
- Um Zugriffe auf Teams Apps abzusichern, kommt u.a. Single Sign On mittels oAuth sowie Azure Active Directory zum Einsatz

So ist Single Sign On umgesetzt



1. Abruf eines SSO-Token über Teams Javascript Library

```
// Get Teams SSO Token
microsoftTeams.authentication.getAuthToken({
  successCallback: async (token: string) => {
    let decoded: { [key: string]: any; } = jwtDecode(token) as { [key: string]: any; };

    let moduleAccess: IModuleAccess = this.state.moduleAccess;

    try {
      moduleAccess = await getModuleAccess(token);
      this.setState({
        moduleAccess
      });
      microsoftTeams.appInitialization.notifySuccess();
    }
    catch(e) {
      microsoftTeams.appInitialization.notifyFailure({
        reason: microsoftTeams.appInitialization.FailedReason.Other,
        message: (e as Error).message
      });
    }
  },
```

Decoded Token | Claims


```
{
  "typ": "JWT",
  "alg": "RS256",
  "x5t": "Mr5-AUibfBii7Nd1jBebaxboXW0",
  "kid": "Mr5-AUibfBii7Nd1jBebaxboXW0"
}.{
  "aud": "api://teamsapp[redacted]net/a880e3b4-2716-4792-a9cb-089d0b331d1a",
  "iss": "https://sts.windows.net/b3201e87-[redacted]70d4a/",
  "iat": 1643310217,
  "nbf": 1643310217,
  "exp": 1643315471,
  "acr": "1",
  "aio": "AVQAq/8TAAAA+5nQBlCUcr6DL8ujJItoW36KbajbM71MFFPLvFVAPC0dxoLjcVU1
  "amr": [
    "rsa",
    "mfa"
  ],
  "appid": "5e3ce6c0-2b1f-4285-8d4b-75ee78787346",
  "appidacr": "0",
  "deviceid": "[redacted]-5c4f-4ccc-9812-50e798ec1f32",
  "family_name": "Kritten",
  "given_name": "Tobias",
```


2. Authentifizierung an Azure AD mit Teams SSO-Token, um erweiterten SSO-Token zu erhalten

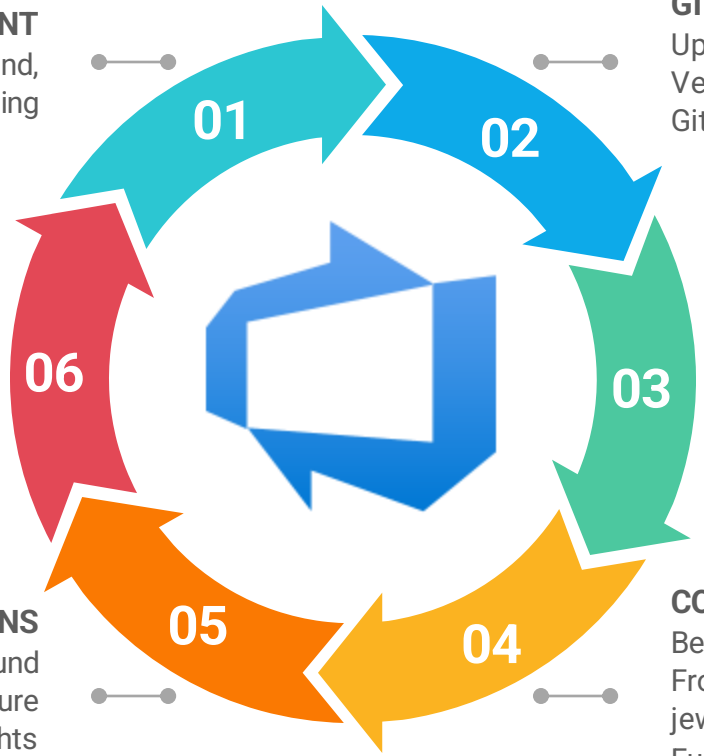
3. Authentifizierung an Function App mit integrierter Azure AD-Authentifizierung und Validierung des Token

azure geek.io

Deployment

Wie gelangt der Source Code der App in die Infrastruktur und wie sorgen wir dafür, dass die App ohne nennenswerte Wartungsfenster aktualisiert wird?

Der Lebenszyklus einer Teams App



DEVELOPMENT
Entwicklung von Front- und Backend, lokales Testing

GIT COMMIT/PUSH
Upload des Source Code in ein Versions-Verwaltungstool (z.B. Github oder Azure Devops)

CONTINOUS INTEGRATION
Ausgelöst durch den Source Code-Upload werden Tests gestartet und bei Erfolg wird der Source Code kompiliert

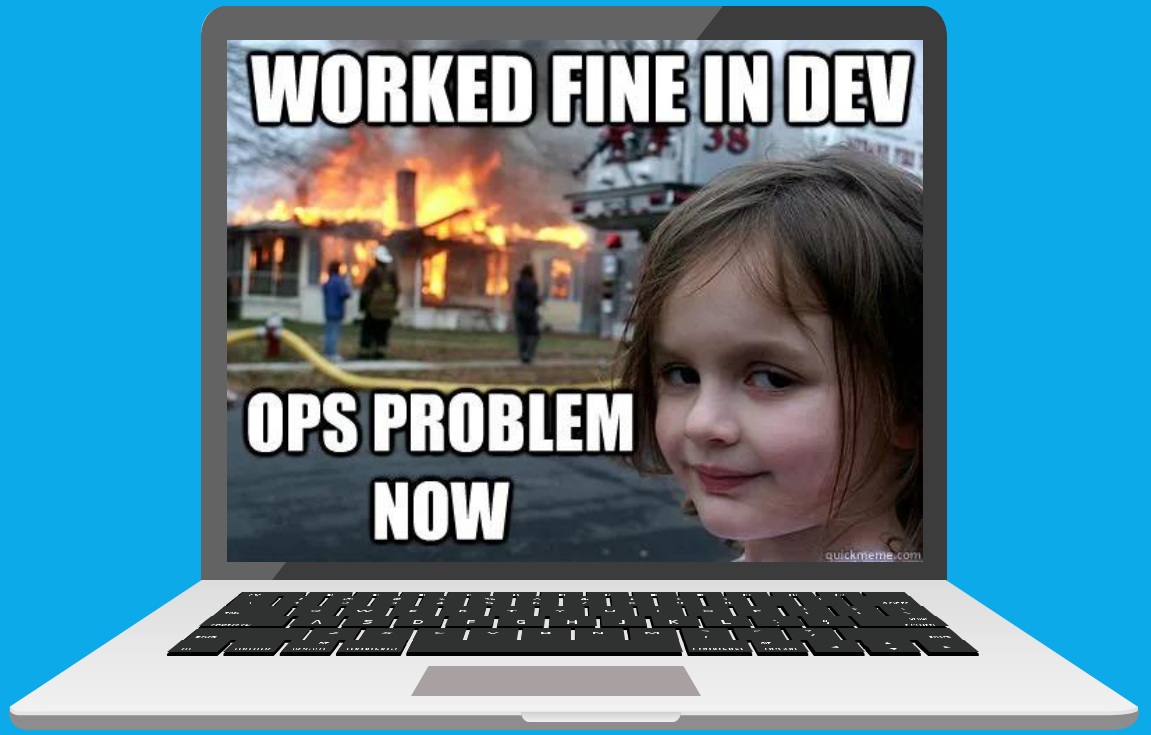
CONTINOUS DEPLOYMENT
Bereitstellung des kompilierten Front- und Backend in den jeweiligen Plattformen, z.B. Azure Functions und Storage Accounts. Nutzung von TeamsFX.

OPERATIONS
Die App ist nun im Regelbetrieb und wird kontinuierlich mit Azure Monitor und Application Insights überwacht.

USER FEEDBACK
Basierend auf dem Feedback der Nutzer warden “User Stories” erstellt und in das Development gegeben.

DEMO

Our mission is to make you stand apart in each of your presentation whether it is related to your project management, planning, sales, branding, or anything in this regard.

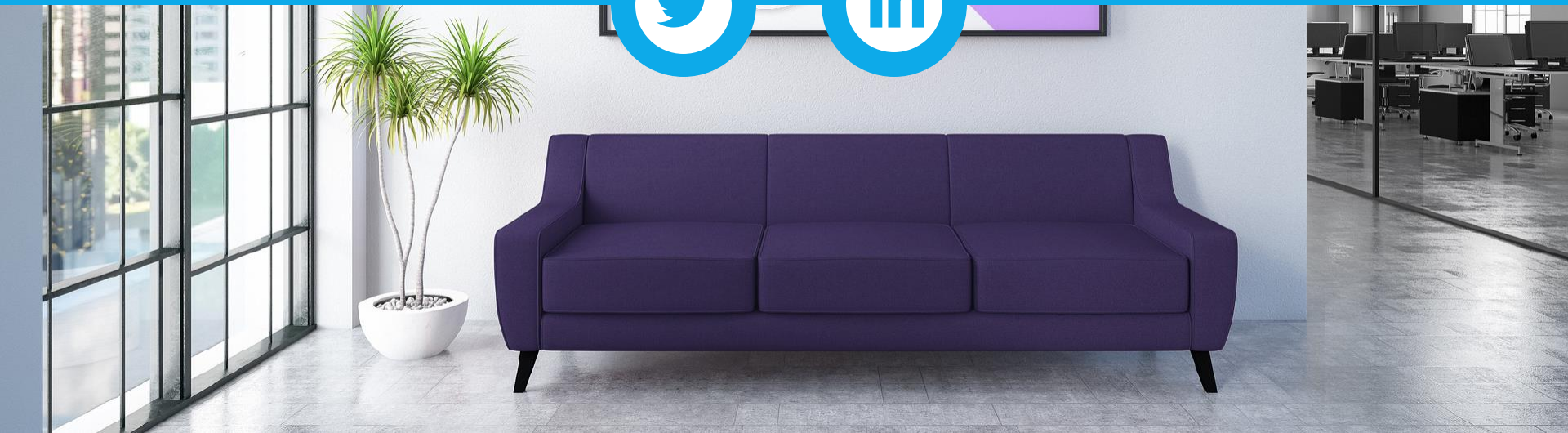


Vielen Dank für eurer Interesse!

Gerne stehe ich bei Fragen, Ideen und zur Diskussion zur Verfügung

Austria – Wien
Germany – NRW

+49 151 116 18 108

tobi@azuregeek.io

azuregeek.io