|  |
| --- |
| 0. |
| Konzeption MVP |
| THEIA für die Baustelle – App Konzept |
| Dieses Dokument beschreibt das Vorgehen und die Erkenntnisse aus der Konzeptionsphase |

Details zum Dokument

Ihre Kontaktpersonen seitens isolutions:

|  |  |
| --- | --- |
|  |  |
| Bill Staub  Executive Consultant,  Mitglied GL. isolutions | Amos Zürcher  Software Architekt  [amos.zuercher@isolutions.ch](mailto:amos.zuercher@isolutions.ch) |
| [bill.staub@isolutions.ch](mailto:bill.staub@isolutions.ch) |  |

|  |  |
| --- | --- |
| Kunde | Helion |
| Projekt | THEIA für die Baustelle |
| Version | 1.1 |
| Datum | 25.07.2024 |
| Status | Geprüft |
| Verteiler | Lukas Karrer (Leiter IT), Helion |
| Referenzierte Dokumente | [Miro Board MVP](https://miro.com/app/board/uXjVKEocZu8=/) |

Inhaltsverzeichnis

[1 Ausgangslage 4](#_Toc172013355)

[2 Anforderungen 5](#_Toc172013356)

[2.1 Ziele der Konzeption 5](#_Toc172013357)

[2.2 Vision der Lösung 5](#_Toc172013358)

[2.3 Funktionale Anforderungen 5](#_Toc172013359)

[2.4 Nichtfunktionale Anforderungen 14](#_Toc172013360)

[3 Zielbild / Konzeption 15](#_Toc172013361)

[3.1 Anbindung THEIA und Speicherung der Daten 16](#_Toc172013362)

[3.2 Anbindung Webviews (THEIA) 16](#_Toc172013363)

[3.3 Anbindung Files (SharePoint Online) 17](#_Toc172013364)

[3.4 Anbindung MS Teams 17](#_Toc172013365)

[3.5 Anbindung Formulare / Power App 18](#_Toc172013366)

[3.6 Realisierung Offlinefähigkeit 19](#_Toc172013367)

[3.7 Sicherheitsaspekte (OWASP, IAM, SSO, Transportverschlüsselung) 20](#_Toc172013368)

[3.8 Entwicklung, DevOps und Handhabung Sourcecode 21](#_Toc172013369)

[3.9 Möglichkeiten Verteilung der App 23](#_Toc172013370)

[4 Antizipierter Aufwand 31](#_Toc172013371)

[4.1 Entwicklung 31](#_Toc172013372)

[4.2 Lizenzen 33](#_Toc172013373)

[4.3 Betrieb 33](#_Toc172013374)

[5 Mögliche Roadmap 34](#_Toc172013375)

# Ausgangslage

Für die Planung und Verwaltung der Projekte auf dem Bau verwendet Helion die Software THEIA. THEIA ist ideal geeignet, um Projekte vorzubereiten und es werden alle Informationen rund um das Projekt dort ab. Zu einem Projekt gehörende Dateien und Fotos werden auf dem SharePoint von Helion abgelegt und im Projekt verlinkt.

Nun möchte Helion in einem nächsten Schritt die Möglichkeit prüfen, alle Informationen rund um ein Projekt sowie gewisse Zusatzfunktionen den Mitarbeitenden auf der Baustelle als App auf dem Smartphone zur Verfügung zu stellen, so dass sie vor Ort jederzeit gut informiert sind und wichtige Funktionen wie Nachbestellungen, Abschlüsse oder Reklamationen nutzen können. Im Rahmen dieses Konzepts sind nun wichtige Themen wie Offlinefähigkeit, Technologie-Stack und die Verteilung der App zu klären.

# Anforderungen

## Ziele der Konzeption

Das Ziel dieser Konzeption ist es hauptsächlich, die Machbarkeit einer einfachen und unterstützenden Mobilelösung zu analysieren und die wichtigsten technischen Stossrichtungen zu definieren. Dabei geht es darum, die Anforderungen möglichst so umzusetzen, dass die Handhabung einfach und qualitativ ist und der Implementierungsaufwand in Grenzen bleibt. Insbesondere geht es darum, die folgenden Fragen zu klären:

* Kann Helion seinen Mitarbeitenden ein einfaches App für iOS und Android zur Verfügung stellen, mit welchem sie einfach vor Ort auf der Baustelle auf die wichtigsten Projektdaten zugreifen können?
* Können die wichtigsten Funktionen wie beispielsweise das Abrufen und Erfassen von Fotos zu einem Projekt oder einem Abschluss auch ohne Internetempfang (z.B. im Keller) genutzt werden?
* Wie können weitere Kanäle wie z.B. Teams, Formulare oder Power Apps eingebunden respektive aufgerufen werden?
* Wie könnte die Verteilung und Aktualisierung der Lösung konkret aussehen?
* Mit welchem Aufwand müsste Helion bei der Realisierung des MVPs rechnen und wie lange würde die Umsetzung dauern (Roadmap)?

## Vision der Lösung

Die THEIA Baustellen App ermöglicht den Helion-Mitarbeitenden einfachen Zugriff auf die für sie relevanten Informationen zu ihren Projekten. Mit der App können die Mitarbeitenden alle relevanten Daten aus THEIA auf ihrem mobilen Gerät speichern und sind in der Lage, die Daten zu betrachten und zu bearbeiten, selbst wenn dem Gerät keine Internetverbindung zur Verfügung steht. Die Baustellen App läuft auf allen gängigen Smartphones und Tablets als native Applikation.

## Funktionale Anforderungen

Authentifizierung:

|  |  |  |
| --- | --- | --- |
| Nr | Name | Beschreibung |
| REQ-1 | Authentisierung | Die Helion-Mitarbeitenden müssen sich beim Einstieg in die App als erstes authentifizieren. Sie können das mit ihrem Azure AD Account tun. Erst nach Authentisierung Autorisierung kann der Benutzer auf die Inhalte zugreifen. |
| REQ-2 | Speicherung der Session | Nach dem Einloggen bleibt die Session des Benutzers im App gespeichert, damit sich der/die Benutzende nicht immer wieder neu anmelden muss. Von Zeit zu Zeit ist aber die Bestätigung erneut nötig (Azure AD bestimmt). |
| REQ-3 | Logout | Der/die Nutzende muss die Möglichkeit haben, sich wieder abzumelden und sich z.B. als einen anderen Benutzer anzumelden. |

**Projektauswahl (Startscreen)**

Der Startscreen der App gibt gleich eine Übersicht der aktuellen Projekte des eingeloggten Mitarbeitenden.

Ein Bild, das Text, Diagramm, Screenshot, Reihe enthält.

Automatisch generierte Beschreibung

|  |  |  |
| --- | --- | --- |
| Nr | Name | Beschreibung |
| REQ-4 | Projektliste | Im Startscreen unter «Meine Projekte» sieht der/die Nutzende eine Liste mit allen Projekten, welche für ihn/sie aktuell in THEIA verfügbar sind. |
| REQ-5 | Projektname | In der Liste unter «Meine Projekte» ist jedes Projekt mit dem Namen und der Ortschaft der Baustelle aufgelistet. |
| REQ-6 | Projektsuche | In einem Suchfeld kann ein Projekt nach dem Namen und der Ortschaft gesucht werden – die Liste passt sich entsprechend dem Suchbegriff laufend an. Der Suchbegriff muss wieder gelöscht werden können. |
| REQ-7 | Projektauswahl | Mittels Touch auf eines der Projekte können die Projektdetails aufgerufen werden. |
| REQ-8 | Offlinefähigkeit | Sämtliche Daten der Projektliste und Suche muss auch ohne aktive Internetverbindung auf den Smartphones möglich sein. Für den PoC: sollte eine Person mehr als 10 Projekte aktiv haben, werden einfach diejenigen 10 Projekte mit den letzten Aktivitäten synchronisiert. Alle anderen Projekte müssten manuell (über einen entsprechenden Button) heruntergeladen werden. |

Anbindung THEIA Backend: Wir gehen davon aus, dass THEIA über REST APIs wie vorbesprochen sämtliche Informationen auf den Screens in einem einfachen, user-spezifischen Service zur Verfügung stellt. Auch wird ein Search Service angeboten, der aus dem App aus konsumiert werden kann (Backend for Frontend).

**Projektdetails**

Zu jedem Projekt können die wichtigsten Details im Detailscreen angezeigt werden.

Ein Bild, das Text, Screenshot, Zahl enthält.

Automatisch generierte Beschreibung

|  |  |  |
| --- | --- | --- |
| Nr | Name | Beschreibung |
| REQ-9 | Projektdetails | Es werden die wichtigsten Infos eines Projektes gemäss obigen Mockups angezeigt. Alle diese Daten können ab THEIA API über REST Web Services abgefragt werden. Ist eine Internetverbindung vorhanden, werden Die Daten zuerst aktualisiert. |
| REQ-10 | Adressen in Maps öffnen | Kontakt- und Objektadresse können per Touch in Karten (ios) oder Maps (android) geöffnet werden. |
| REQ-11 | Projektdetails | Nach einem Touch auf das Projekt können die weiteren Projektdetails aus THEIA angezeigt werden. |
| REQ-12 | Absprung nach Teams | Wird aus THEIA ein Link zu einem Projekt Teams-Channel mitgeliefert, wird im App ein Button angezeigt, mittels welchem man nach Teams (App oder Browser) wechseln kann und dort gleich der entsprechende Chat im Channel geöffnet wird. |
| REQ-13 | Absprung nach Sharepoint | Wird aus THEIA ein Link zu einer Sharepoint-Ablage mitgeliefert, wird im App ein Button angezeigt, mittels welchem man nach Sharepoint (App oder Browser) wechseln kann und dort gleich die entsprechende Projektablage geöffnet wird. |
| REQ-14 | Logistik-Fehler melden | Durch Touch auf einen Button können Logistik-Fehler gemeldet werden. Das entsprechende Power App soll per deep Link aufgerufen werden und wenn möglich bereits mit Projektnummer ausgestattet sein. |
| REQ-15 | Notizen zum Auftrag | Über einen weiteren Button kann die Projektnotizen-Funktion aufgerufen werden. |
| REQ-16 | Inbetriebnahme | Über einen weiteren Button kann die Inbetriebnahme-Funktion aufgerufen werden. |
| REQ-17 | Zurück zur Übersicht | Mittels eines Zurück-Buttons muss wieder in die Projektübersicht gewechselt werden können. |
| REQ-18 | Offlinefähigkeit | Sämtliche Daten dieses Projekts müssen auch ohne aktive Internetverbindung auf den Smartphones abrufbar sein. Links nach Teams oder SharePoint funktionieren dann halt nicht. |

**Notizen zum Auftrag**

Zu jedem Projekt können in THEIA Notizen erfasst werden. Diese Notizfunktion soll in der App auf dem Smartphone ebenfalls nutzbar sein.

Ein Bild, das Text, Screenshot, Handy, mobiles Gerät enthält.

Automatisch generierte Beschreibung

|  |  |  |
| --- | --- | --- |
| Nr | Name | Beschreibung |
| REQ-19 | Notizen-Übersicht | In einer Liste werden sämtliche bisher erfassten Notizen zum ausgewählten Projekt angezeigt. |
| REQ-20 | Filter nach Kategorie | In einem Dropdown Feld kann die jeweilige Notizen-Kategorie gefiltert werden. Standardmässig ist kein Filter gesetzt. |
| REQ-21 | Detailansicht Notizen | In einer Detailansicht können die Notizen angezeigt werden. Externe Links (z.B. auf Word-Dokumente welche in SharePoint abgelegt sind) können angeklickt und im externen Browser (Safari oder Chrome) angezeigt werden, falls die entsprechenden Berechtigungen vorhanden sind. Eingebettete Bilder werden geladen und angezeigt. |
| REQ-22 | Notiz editieren und erstellen. | Notizen können editiert und erstellt werden, falls eine Onlineverbindung existiert. Wenn nicht, ist alles nur read-only. |
| REQ-23 | Zurück zur Übersicht | Mittels eines Zurück-Buttons muss wieder in die Notizen-Übersicht gewechselt werden können. |
| REQ-24 | Offlinefähigkeit | Sämtliche Notizen der geladenen Projekte müssen auch ohne aktive Internetverbindung auf den Smartphones abrufbar sein. Links nach Teams oder SharePoint funktionieren dann natürlich nicht. Notizen hinzufügen oder editieren muss nur bei Onlineverbindung möglich sein. |

**Anbindung THEIA Backend**: Wir gehen davon aus, dass THEIA über REST APIs wie vorbesprochen sämtliche Informationen auf den Screens in einem einfachen, user-spezifischen Service zur Verfügung stellt. Die Notizfunktionen sind gemäss Sitewerk als HTML gespeichert und die Bilder werden im Respons selbst mitgegeben.

**Inbetriebnahme**

Jedes Projekt kann mittels einer Checkliste in betrieb genommen werden.

Ein Bild, das Text, Screenshot, parallel, Design enthält.

Automatisch generierte Beschreibung

|  |  |  |
| --- | --- | --- |
| Nr | Name | Beschreibung |
| REQ-25 | Inbetriebnahme | Für jedes Projekt (Lead in THEIA) kann eine Inbetriebnahme erfasst werden. Ist bereits eine Inbetriebnahme da, kann diese editiert werden. |
| REQ-26 | Checkliste anzeigen | Es werden drei Checklisten mit erforderlichen Dokumenten, IBN SOP 2.0 und Bilder ab THEIA angezeigt. Die Checkpunkte können je Projekt verschieden sein. Jedes Checklisten-Item ist entweder Pflichtfeld oder Optional. |
| REQ-27 | Vorhandene Bilder | Sind bereits Bilder vorhanden, können diese via Ansicht-Icon angezeigt werden (aktuell nicht vorhanden im Mockup) |
| REQ-28 | Neue Bilder hinzufügen | Im App kann pro Checklisten-Item eines oder mehrere Bilder Heraufgeladen werden. Entweder werden diese gleich mit der Kamera aufgenommen, oder sie werden aus der Camera Roll hinzugefügt. Sobald die Fotos hochgeladen wurden, sind sie auch über die Ansicht verfügbar (REQ-28). |
| REQ-29 | Anmerkungen | Es können Anmerkungen in einem Textfeld erfasst oder editiert werden. Dabei ist keine Formatierung vorgesehen, also einfach ein Textfeld. |
| REQ-30 | Speichern und Inbetriebnahme | Der aktuelle Stand der Checkliste kann jederzeit gespeichert oder gespeichert und abgeschlossen werden. Ist der IBN einmal abgeschlossen, kann er im App nur noch angeschaut werden. |
| REQ-31 | Offlinefähigkeit | Sämtliche Inbetriebnahmen der geladenen Projekte (eines je Projekt) müssen auch ohne aktive Internetverbindung auf den Smartphones abrufbar sein. Auch müssen vorhandene Bilder angezeigt und neue Bilder (Multi-upload) hinzugefügt werden können. Sobald eine Onlineverbindung wieder verfügbar ist, werden die Daten über die APIs ins THEIA geladen. |

**Anbindung THEIA Backend**: Wir gehen davon aus, dass THEIA über REST APIs wie vorbesprochen sämtliche Informationen auf den Screens in einem einfachen, user-spezifischen Service zur Verfügung stellt. Die Items der Checklisten sowie die dazugehörigen Bilder werden über die APIs mitgeliefert oder referenziert. Allfällige Race conditions (wenn z.B. mehrere User die gleiche Inbetriebnahme bearbeiten oder gleichzeitig jemand die Abnahme abschliesst und jemand anders editiert) werden über das API Backend von THEIA gelöst.

**Projektdaten**

Bei Klick auf das Projekt im Detailscreen sollen weitere Daten zum Projekt angezeigt werden.

Ein Bild, das Text, Screenshot, Zahl, Smartphone enthält.

Automatisch generierte Beschreibung

|  |  |  |
| --- | --- | --- |
| Nr | Name | Beschreibung |
| REQ-32 | Projektdaten | Für jedes Projekt können die Detaildaten angezeigt werden. Dabei werden die Felder aus dem THEIA backend angezeigt. |
| REQ-33 | Editierbarkeit | Die Projektdetails sind über das App nicht editierbar sondern werden einfach angezeigt. Wo ein Link hinterlegt ist, kann dieser in einem externen Browserfenster geöffnet werden. |
| REQ-34 | Offlinefähigkeit | Sämtliche Daten der geladenen Projekte müssen auch ohne aktive Internetverbindung auf den Smartphones abrufbar sein. |

**Anbindung THEIA Backend**: Wir gehen davon aus, dass THEIA über REST APIs wie vorbesprochen sämtliche Informationen auf den Screens in einem einfachen, user-spezifischen Service zur Verfügung stellt.

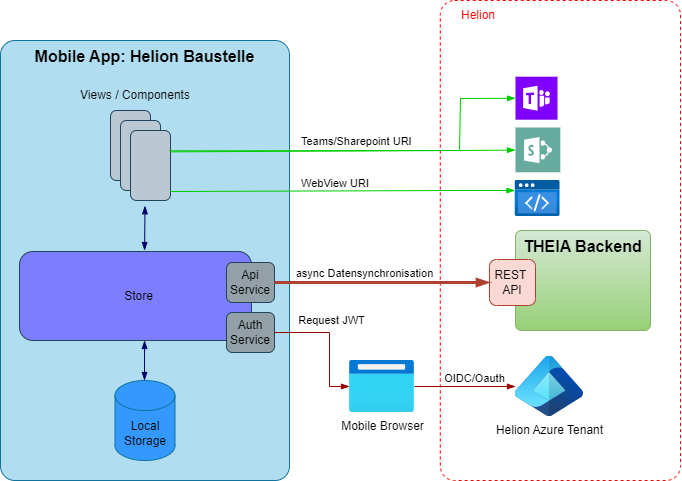
## Nichtfunktionale Anforderungen

Nichtfunktionale Anforderungen (NFRs) sind die Einschränkungen oder Anforderungen, die an eine App gestellt werden, um ihre Qualitätsmerkmale zu definieren. Im Gegensatz zu funktionalen Anforderungen, die beschreiben, **was** das System tun soll, beschreiben nichtfunktionale Anforderungen **wie** das System es tun soll. Hier sind einige Beispiele für nichtfunktionale Anforderungen:

|  |  |  |
| --- | --- | --- |
| Nr | Name | Beschreibung |
| REQ-NF-1 | Leistung | Die App soll schnell und reaktionsschnell sein, insbesondere beim Laden von Inhalten und beim Wechseln zwischen den Ansichten. |
| REQ-NF-2 | Benutzerfreundlichkeit | Die Benutzeroberfläche sollte intuitiv gestaltet sein, um eine nahtlose Experience zu gewährleisten. |
| REQ-NF-3 | Sicherheit | Die App sollte sicher sein, sowohl in Bezug auf Datenübertragung wie auch lokale Datenspeicherung. Authentifizierung über Azure AD muss möglich sein. |
| REQ-NF-4 | Verteilbarkeit | Die App sollte möglichst einfach auf den Geräten der Mitarbeitenden installiert werden können. |
| REQ-NF-5 | Wartbarkeit | Die App sollte wartbar sein, neue Versionen sollen möglichst einfach auf die Geräte der Mitarbeitenden verteilt werden können. |
| REQ-NF-6 | Offlinefähigkeit | Damit die App auch auf der Baustelle und im Keller verwendet werden kann, sollen die Funktionen auch offline verfügbar sein. Nicht geladene Inhalte sollen bei Onlineverbindung heruntergeladen werden können. Für Online-only Inhalte sollte eine passende Fehlermeldung angezeigt werden. |
| REQ-NF-7 | Gerätekompatibilität | Die App soll auf Android Version 12 und neuer sowie iOS 16 und neuer lauffähig sein. |

# Zielbild / Konzeption

Die Lösung für besteht aus einer nativen Applikation für mobile Endgeräte mit Android- oder iOS-Betriebssystem. Die App muss native umgesetzt werden, da Webtechnologien und PWA bezüglich Offlineabgleich und in der lokalen Speicherung von Daten sehr beschränkt sind. Die besondere Herausforderung ist die asynchrone Kommunikation zwischen THEIA Backend und der Mobile App. Die Daten aus dem THEIA Backend müssen offline zur Verfügung stehen und ergänzt werden können. Das stellt besondere Anforderungen an die Synchronisation der Daten hinsichtlich Konsistenz und zusammenführen von Daten ohne Konflikte. Die entsprechende Abgleichlogik muss im App umgesetzt werden.



*Abbildung 1: Zielarchitektur Abgleich Daten und Anbindung Mobile App*

Das System besteht aus einer Mobilen App, welche alle Anforderungen abdecken soll. Algorithmen für die Datensynchronisation und Konsistenzprüfung werden in der mobilen Applikation realisiert.

## Tenants und Authentifizierung

Sowohl Helion wie auch Theia betreiben einen eigenen Azure Tenant, die miteinander verbunden sind. Theia greift gemäss unserer Abklärung mittels Service Principals auf Ressourcen (z.B. SharePoint Inhalten) aus dem Helion Tenant zu.

Die SharePoint Dateien, welche von Theia aus referenziert werden, liegen im Helion Tenant. Auf diese Ressourcen können die Helion Benutzer über die App problemlos mit ihrer Entra ID Identity zugreifen. Eine Federation der Tenants würde die Sache nur verkomplizieren und ist in dem Fall nicht notwendig. Es ist einzig darauf zu achten, dass die App ebenfalls Entra ID integriert ist und der User Token verwendet wird, um auf SharePoint zuzugreifen.

So wird der Zugang zu den Helion Ressourcen über die neue App mittels Single Sign-on für bestehende AD-Benutzer von Helion nutzbar sein.

## Anbindung THEIA und Speicherung der Daten

Das THEIA Backend stellt eine **API** basierend auf dem OpenAPI Standard zur Verfügung. Die API wird anhand der Anforderungen an die Baustellen App erweitert respektive erstellt. In der App wird die Anbindung über einen Store für das State Management realisiert. Dieser Kommuniziert mit dem Backend, wenn eine Verbindung besteht und setzt die Algorithmen für die Datenkonsistenz um. Als App State wird hier der Zustand der lokalen Daten bezeichnet. Dieser State speichert der Store im lokalen Storage in einer **IndexedDB** als Key Value Paare oder als Datenobjekte in einer **SQLite Datenbank**. Das UI wird als Views oder Komponenten umgesetzt. Diese beziehen die Daten oder den für die jeweilige Komponente relevanten State aus dem Store. Änderungen am State werden den Komponenten proaktiv mitgeteilt und dem Benutzer angezeigt.

## Anbindung Webviews (THEIA)

Die Einbindung von bestehenden (mobilefähigen) Webinhalte in der App als so genannte Webviews ist durchaus möglich. Beispielsweise könnten so künftig Inhalte von THEIA oder auch anderen Webapplikationen oder Webseiten im app dargestellt werden.

A white rectangular object with lines

Description automatically generated

Der THEIA Webview Bereich lädt den Inhalt von THEIA. Damit das möglich ist, muss THEIA den Inhalt als HTML-Dokument ausliefern. Der Bereich mit der Webview in der Baustellen App funktioniert dabei ähnlich wie ein Browser. Für die Authentifizierung wird dem Aufruf an THEIA für die Webview das JWT-Token als Cookie mitgeliefert. Damit kann THEIA den Benutzer authentifizieren.

**Webviews nur bei onlineverbindung**

Die nichtfunktionale Anforderung REQ-NF-6 an die offline Fähigkeit gilt insbesondere für die THEIA Daten. Die Einbindung von Webviews würde die Erfüllung dieser Anforderung verhindern. Wo gewünscht können THEIA Webviews in der Applikation angezeigt werden, mit der vorher beschriebenen Einschränkung.

## Anbindung Files (SharePoint Online)

Projekt relevante Dokumente werden auf SharePoint abgelegt und in den Projekten verlinkt. In der Projekt Detailansicht in der mobilen App werden Links zu der Dokumentablage auf SharePoint als Button dargestellt. Ein Klick/Touch auf den Button leitet den/die Nutzende auf die Webansicht oder die SharePoint App auf dem Gerät weiter und öffnet die verlinkte Dokumentsammlung. Dies setzt eine Verbindung mit dem Internet voraus und funktioniert nicht, wenn das Gerät im offline Modus ist. Kann keine Verbindung mit dem Internet hergestellt werden, wird der Button als deaktiviert dargestellt und kann nicht angeklickt werden. Direkte Links zu Dokumenten auf SharePoint (in den Auftragsdetails) sind möglich, können jedoch nur mit aktiver Internetverbindung aufgerufen werden. SharePoint unterstützt die Authentifizierung mit OAuth. Beim Aufruf des Links zu SharePoint, kann das Identity Token in der Baustellen App mitgegeben werden, welches ebenfalls via OAuth vom Helion EntraID gelöst wurde. Damit kann der Benutzer in SharePoint authentifiziert werden.

## Anbindung MS Teams

Links zu Microsoft Teams Chats oder Channels können in einem THEIA Projekt über die Desktop-Anwendung gespeichert werden. In der Baustellen App wird dieser Link als Button dargestellt. Links zu Chats und Teams Channels können in der Teams Applikation erstellt und kopiert werden. Es gibt zwei Varianten, wie diese Links aufgebaut werden müssen.

Direkter Aufruf von MS Teams:

Damit ein Channel oder Chat direkt in Teams geöffnet wird, muss der Link mit „msteams://teams.microsoft.com/…“ beginnen. Diese Variante setzt allerdings voraus, dass MS Teams auf dem Gerät installiert ist. Ansonsten funktioniert der Link nicht.

**Aufruf von MS Teams via Browser:**

Mittels normaler http Links können ebenfalls Chats und Channels geöffnet werden. MS Teams selbst erstellt http Links, welche auf die entsprechenden Chats und Channels zeigen. Erkennbar ist das daran, dass diese Links mit „https://teams.microsoft.com/...“ beginnen. Der Link öffnet einen Browser und von dort kann der Benutzer entweder die Webversion von Teams öffnen oder wenn Teams lokal installiert ist, kann verlinkte Channel oder Chat in der lokalen MS Teams App geöffnet werden. Der https Link funktioniert allerdings nur, wenn das Gerät online ist. Deshalb ist der Button für diese Art Links im offline Modus deaktiviert und kann nicht angeklickt werden.

## Anbindung Formulare / Power App

Formulare und Ansichten aus PowerApps (beispielsweise das Reklamationsformular) können als Webview in die Baustellen App integriert werden. Das würde analog zu der Integration von THEIA Webviews funktionieren. Oder die Power App und Formulare werden in der Baustellen App verlinkt und der Benutzer wird auf die Power App oder das Formular weitergeleitet. In jedem Fall wird ein Deeplink auf das Formular oder die View der Power App wird als Quelle benötigt und kann je nach Anforderung in der Baustellen App integriert werden.

Ein Bild, das Text, Screenshot enthält.

Automatisch generierte Beschreibung

Wie auch bei den Webviews wird bei der Integration oder Verlinkung von Power Apps auch die Onlineverbindung vorausgesetzt. Ist die App offline, wird ein entsprechender Hinweis angezeigt. Es ist sowohl bei den Webviews wie auch beim Öffnen im Browser möglich, das Power App im Mobile Power Apps App zu öffnen. Ob bei diesem Vorgang der Deep Link zum jeweiligen Formular mitgereicht wird, müssen wir noch eruieren.

## Realisierung Offlinefähigkeit

Die Fähigkeit, offline Projektdaten zu betrachten und zu bearbeiten, ist eine Herausforderung für die Mobile App, insbesondere da jederzeit die Datenkonsistenz zu gewährleisten ist.

Es gibt viele Algorithmen und Entwurfsmuster, um das Problem der Datenkonsistenz zu lösen. Im MVP wird das Entwurfsmuster «**Last Write Wins**» angewendet. Das ist ein einfacher Ansatz für die Synchronisation von offline Applikationen. Im Gegensatz zu anderen komplexeren Entwurfsmustern stellt dieser Ansatz weniger hohe Anforderungen an Speicherplatzbedarf und Bandbreite. Ausserdem ist das THEIA Backend so aufgebaut, dass für den gewählten Ansatz keine Anpassungen am Backend diesbezüglich nötig wären. Dadurch wird er Aufwand für die Implementierung der Datensynchronisation wesentlich vereinfacht.

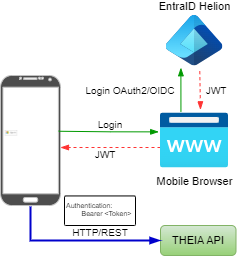
Konkret: Jeder Datensatz pflegt ein Datumsfeld, welches jedes Mal aktualisiert wird, wenn der Datensatz bearbeitet wird. Dieses Datum kann die Baustellen Applikation abfragen und mit dem lokalen Datum des Datensatzes abgleichen. Dieser Abgleich findet statt, wenn die Mobile App Konnektivität zum THEIA Backend erhält. Mit anderen Worten, wenn sich das Gerät mit dem Internet verbindet. Basierend auf diesem Abgleich entscheidet die Baustellen-App, ob die lokalen Datensätze an das Backend gesendet werden oder ob Daten vom Backend abgefragt werden. Dieser Sync kann im Hintergrund ausgeführt werden, ohne dass die App sichtbar ist. Voraussetzung ist, dass die Baustellen Applikation mindestens einmal geöffnet und somit gestartet wurde und Hintergrund-Jobs auf dem jeweiligen Gerät aktiviert sind. Wird die App durch das Betriebssystem oder den Benutzenden gestoppt, funktioniert der Sync nicht mehr.

Sämtliche Projektdaten von den **zehn** zuletzt geöffneten Projekten werden lokal gespeichert. Zusätzlich steht dem Benutzer eine Funktion zur Verfügung, damit Projekte **gezielt offline gespeichert werden können**. Es wird der lokale Gerätespeicher verwendet. Verfügt das Gerät nicht über genügend Speicherplatz, wird dem Benutzer ein Hinweis angezeigt, dass nicht genügend Speicher vorhanden ist und Helion Projekte nur bedingt heruntergeladen und offline verwendet werden können. Lokal gespeichert werden die Projektdaten, Metadaten welche nötig sind, um Formulare und Checklisten generisch darzustellen sowie Bilder. Diese Daten werden initial vom Backend heruntergeladen, sofern dort verfügbar. Änderungen sind beispielsweise das Ausfüllen einer Checkliste oder hinzufügen von Bildern. Das geschieht lokal und wird direkt mit dem Backend synchronisiert, wenn das Gerät online ist. Ist das Gerät offline wird die Änderung nur lokal durchgeführt und gegebenenfalls synchronisiert.

**Bilder** werden auf dem Filesystem des Gerätes gespeichert. Sowohl in Android wie auch in iOS hat eine App einen Filesystemkontext. Bilder werden in diesem Kontext gespeichert. Auf diesen Kontext kann nur die App selbst zugreifen. Externer Zugriff auf die lokalen Bilder ist von daher nicht möglich. Ausserdem wird der Kontext und damit die Bilder entfernt, wenn die App gelöscht wird.

## Sicherheitsaspekte (OWASP, IAM, SSO, Transportverschlüsselung)

THEIA ist personalisiert und verlangt einen Userkontext, damit THEIA eine Autorisierung des Benutzers durchführen kann. THEIA hat einen eigenen Authentifizierungsprozess implementiert kann jedoch mit JSON Web Token (JWT) umgehen, um Single-Sign-On (SSO) zu ermöglichen.



Für die Authentifizierung in der Baustellen App wird Microsoft als Identity Provider verwendet. Der Login Prozess verwendet das Microsoft Webinterface für die Authentifizierung. Klickt der Benutzer in der Baustellen App auf Login, wird der Standard-Browser geöffnet und das Microsoft Login weitergeleitet. Der Browser verwendet für die Authentifizierung OAuth2 mit OpenID-Connect. Ist der Benutzer authentifiziert kommt er zurück in die App und bringt ein JWT-Token von Microsoft mit. Dieses Token wird nun verwendet für die Kommunikation mit dem THEIA Backend und wird als Bearer Token den API-Anfragen angehängt. Die Abfragen werden mittels Transport Layer Security (TLS) gegen das Abfangen des Tokens und mitlesen der Daten gesichert. Auf dem Gerät wird das Token verschlüsselt in einem sicheren Speicher abgelegt. Damit wird unbefugter Zugriff auf das Token von anderen Apps oder Code verhindert. Die Authentifizierung via Browser, Ablage des Tokens im Secure Store des Gerätes und mit TLS verschlüsseltes Senden via HTTPS gelten heute als Best-Practice für maximale Sicherheit und entsprechen den Empfehlungen von OWASP.

## Entwicklung, DevOps und Handhabung Sourcecode

Der Sourcecode für die Baustellen App soll in einem Repository in Gitlab von Helion aufbewahrt werden. Gitlab bietet alles Nötige, um ein Build und Deployment Pipeline für die Baustellen App einzurichten.

**Das richtige Cross Platform Framework**

Um die umfangreiche Offlinefähigkeit zu gewährleisten, soll eine native App zum Einsatz kommen. Damit nicht für jedes Betriebssystem eine eigene App entwickelt werden muss, gibt es Cross-Plattform Frameworks. Mit einem Framework wird die Applikation auf einer Codebasis entwickelt und in native Apps übersetzt. Als geeignete Frameworks kommen Metas React Nativ, Googles Flutter, Microsoft .Net MAUI oder Ionic in Frage.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Kriterium | React Native | Flutter | Ionic | .net MAUI |
| Hintergrund Tasks | Headless JS | Isolates | Background Tasks / Cap | Custom |
| Speicherkapazität | Verfügbarer Gerätespeicher | Verfügbarer Gerätespeicher | Verfügbarer Gerätespeicher | Verfügbarer Gerätespeicher |
| Native Controls | Kern Komponenten sind auf native Komponenten gemappt. Weitere Native Controls können mit «Native Components» gebaut werden. | Klein direktes Mapping  -  Native Controls via Android oder iOS Views | Kein direktes Mapping  -  Native Controls müssen als Capacitor Plugin verfügbar gemacht werden | MAUI Controls mappen direkt auf native Controls |
| Technologie | Javascript, React Syntax | Dart | Angular, React oder VueJS | XAML, C#, Blazor |
| Build und Deployment | Azure DevOps, GitHub Actions | Azure DevOps, GitHub Actions | Azure DevOps, GitHub Actions | Azure DevOps, GitHub Actions |

**React Native** ist die Meta eigene Entwicklung eines Cross Plattform Frameworks. Es wird durch Meta und die Community weiterentwickelt und gewartet.

**Ionic** basiert auf der Abstraktionsschicht Cordova, welcher die native API der jeweiligen Geräte abstrahiert. Es unterstützt die Entwicklung mit gängigen UI-Frameworks wie Angular, React oder VueJS. Ionic ist sehr flexibel, da es verschiedene UI-Frameworks unterstützt. Entwickler mit Erfahrung mit Frontend Frameworks erzielen schnell Ergebnisse mit Ionic. Die essenziellen Features von Ionic sind kostenlos. Ein paar zusätzliche Funktionen sind jedoch Bezahlfeatures.

**Flutter** wurde von Google entwickelt und verwendet eine eigene Sprache namens Dart. Gemäss Umfragen ist Flutter das beliebteste Cross-Plattform Framework. Wie auf einer Zeichenfläche zeichnet Flutter das Benutzerinterface Pixel für Pixel auf den Gerätebildschirm. Laut Flutter reagiert das UI besonders flüssig und performant.

**MAUI** ist die Cross Plattform Technologie von Microsoft und verwendet XAML und C# als Codebasis. MAUI ist noch jung, verwendet jedoch robuste und gängige und bekannte Entwicklungssprachen. Ein Vorteil von MAUI ist, dass die UI Controls auf echten nativen Controls gemappt werden. Das Benutzererlebnis entspricht einer echten nativen Applikation.

Sowohl .Net MAUI wie auch React Native decken die Anforderungen ans GUI und die Abdeckung der Anforderungen inkl. Controls, Kamerazugriff und Hintergrundsynchronisation gleichermassen ab.

**Unser Ansatz: .Net MAUI**

Für den MVP der THEIA Baustellen Applikation wird empfohlen auf .**Net MAUI** zu setzen. Das Framework erfüllt alle Voraussetzungen, um die Anforderungen umzusetzen, bietet native Controls und die Programmiersprache C# ist weit verbreitet. Des Weiteren haben wir bereits gute Erfahrungen mit .Net MAUI gemacht und es gibt eine grosse Community.

## Möglichkeiten Verteilung der App

Eine weitere Entscheidung muss im Bereich der möglichen Deployment Strategie gefällt werden. Dabei geht es darum zu definieren, wie die App initial auf die Geräte kommt und wie später Updates verteilt werden.

Grundsätzlich gibt es die Möglichkeit, die App in den beiden öffentlichen App Stores von Google und Apple bereitzustellen. Für interne Line-of-Business Apps empfiehlt sich grundsätzlich eher das Deployment über eine interne Website (pragmatisch) oder mittels eines Mobile Device Managements (MDM). Gerne vergleichen wir die Varianten im Folgenden.

### Deployment public App Store

Das Bereitstellen der Apps im öffentlichen App Store und der selbständige Download der Apps über den nativen App Store (Google oder Apple) wäre eine Möglichkeit, die App ohne eigene Tools und Management bereitzustellen.

**A diagram of a computer

Description automatically generated**

*Figure 1: Verteilung via Public App Stores*

Klarer Nachteil ist aber, dass die App nur auf Geräte, Versionen und Länder eingeschränkt werden kann. Das heisst, nebst den Helion Mitarbeitenden in der Schweiz könnten es auch beliebige andere Menschen herunterladen, auch wenn das App für diese Personen dann nicht brauchbar ist. Das wiederum führt zu unzufriedenen Kunden und schlechten Bewertungen in den App Stores. Zudem ist der Aufwand des Publizierens im jeweiligen App Store nicht zu verachten, und man schafft sich eine unnötige Abhängigkeit. Gerade bei Apple kann nicht garantiert werden, dass die App initial oder auch ein Update sicher in den Store aufgenommen wird. Es bleibt das Risiko einer Zurückweisung.

**Unlisted und Private Apps für Public Store**

Es ist bei beiden App Stores auch möglich, Apps für eine gewisse Gruppe von Benutzern einzuschränken. Mittels nicht gelisteten (**Unlisted Apps**) Apps können Apps im Apple AppStore publiziert werden, stehen dann aber nicht der ganzen Welt zur Verfügung wie sonst üblich, sondern nur für Personen, welche den direkten Link kennen. Es ist also nicht kontrollierbar, ob nur interne Mitarbeitende das App laden – wer den Link kennt, kann es herunterladen. Bei Google existiert keine solche Funktion für nicht gelistete Apps im Public App Store. Dafür lassen sich im Google Store zu testzwecken Apps bereitstellen, genannt **Private Apps**. Google Private Apps müssen aber auf User oder Organisationen zugewiesen sein. Die User müssen also entweder alle einzeln erfasst oder der selben Google Workspace Organisation angehören. Bei Apple sowie Google unterlaufen die Apps nach wie vor der generellen Reveiw, was bei inhouse-Apps mit direktem Login manchmal schwierig ist.

### **Deployment über interne Webseite oder Inhouse Store**

Die einfachste und pragmatischste Variante wäre, die jeweiligen Apps einfach zu einem installierbaren Package zu schnüren und dann auf einer internen Webseite (z.B. Intranet) den Mitarbeitenden zur Verfügung zu stellen.

A diagram of a software application

Description automatically generated with medium confidence

*Figure 2: Verteilung über interne Website*

Die ausführbare App wird jeweils für Android und iOS auf einem Blob Storage abgelegt und auf einer statischen Webseite verlinkt. Die Mitarbeiter können auf die Webseite und dort die App herunterladen und installieren. Neue App Versionen werden auf den Blob Storage deployed.

Bitte beachten: Die folgenden Voraussetzungen müssen gegeben sein, damit eine Distribution von Apple-Apps über eine interne Website funktionieren wird.

* Apps müssen im .ipa-Format vorliegen und mit einem internen Bereitstellungsprofil erstellt worden sein.
* Sie müssen über eine XML-Manifest-Datei verfügen.
* Sie müssen von einer Website geladen werden, deren Adresse mit HTTPS beginnt.
* Sie müssen mit einem Zertifikat signiert sein, dem auf dem Gerät vertraut wird.
* Ihre Netzwerkkonfiguration muss Geräten den Zugriff auf einen Server bei Apple ermöglichen.

Da diese Art von Deployment keine Update-Mechanismen beherrscht, müsste man in der App einen speziellen Prüfalgorithmus integrieren, der regelmässig überprüft, ob die aktuelle Version installiert ist oder ob es eine neue Version gibt. Wenn ja, wird dem Benutzer ein Hinweis angezeigt, dass er eine neue Version herunterladen kann oder soll. Die Installation bleibt dann aber dem Benutzer überlassen. Nachteil dieser Lösung ist ganz klar, dass wir zu keiner Zeit den Überblick haben, wer welche App Version installiert hat und man kann auch nicht sicherstellen, dass die App deinstalliert oder eine neue Version installiert wird.

**Inhouse Store**

Abhilfe schaffen so genannte Inhouse Stores, welche genau tracken, wer wann welche Datei heruntergeladen hat.

**Apple Business Manager**

Der Apple Business Manager (ABM) ermöglicht es Unternehmen, alle ihre Apple-Geräte (MDM) sowie Custom- und Standard-Apps von einer zentralen Plattform aus zu verwalten. Mit der **automatisierten Geräteanmeldung** (ADE) können Unternehmen neue, von ihnen bereitgestellte Geräte automatisch in ihre Verwaltung einbinden und konfigurieren. Privat gekaufte Geräte (BYOD) können leider nicht automatisch angemeldet werden. Der MDM-Teil von ABM kann mit Entra ID und Intune verbunden werden, um das manuelle Erfassen zu vereinfachen. Unternehmen können so benutzerdefinierte Apps sicher an ihre Mitarbeiter verteilen, ohne sie im öffentlichen App Store zu veröffentlichen. **Die Verteilung mit ABM ohne MDM** wird weiter unten ausgeführt.

Die App wird nach der Entwicklung im Apple Business Manager hochgeladen und für die Verteilung vorbereitet. Dies kann auch über diverse Apple-Schnittstellen bis zur Prüfung von Apple automatisiert geschehen. Der Entwickler konfiguriert die Verteilungseinstellungen beim Aufsetzen von ABM, einschliesslich der Auswahl der Zielgeräte oder Benutzer. Wird die App freigegeben, greifen anschliessend die Voreinstellungen und die App kann je nach Einstellung **direkt gepusht oder zum Download zur Verfügung** gestellt werden. Updates werden auf demselben Weg an Benutzende verteilt. Der Apple Business Manager selbst ist kostenlos, es können jedoch Kosten für das Apple Developer Program anfallen, das erforderlich ist, um benutzerdefinierte Apps zu erstellen und zu verteilen (Siehe Lizenzen).

Der Onboarding-Prozess für ein persönliches Gerät mit ABM und Entra ID beginnt damit, dass der Benutzer eine E-Mail mit einem Registrierungslink erhält. Der Benutzer authentifiziert sich dann mit seiner Entra ID. Nach der Authentifizierung installiert der Benutzer das MDM-Profil auf seinem Gerät. Sobald das Profil installiert ist, wird das Gerät gemäss den Richtlinien der Organisation konfiguriert und die notwendigen Apps werden bereitgestellt oder direkt gepusht. Der Benutzer kann dann auf seinem Gerät getrennt auf Arbeits- und persönliche Daten zugreifen. Das Gerät erhält weiterhin Updates und bleibt konform mit den Richtlinien der Organisation. Für Updates erhalten Benutzer gemäss Voreinstellung in ABM eine Benachrichtigung durch Tags oder Hinweise im App Store und können die neueste Version der App herunterladen und installieren, sofern sie nicht direkt gepusht wird. Wenn automatische Updates aktiviert sind, wird das Update direkt installiert. Eine manuelle Deinstallation der alten Version ist nicht erforderlich.

**Ohne MDM in ABM** werden eigene Apps im Business Manager wie oben beschrieben bis zur Prüfung eingereicht und durchlaufen den Apple App-Store Review-Prozess. Nach der Freigabe können Entwickler einzigartige Einlösungscodes für die App ausstellen und diese Codes per E-Mail oder über eine dedizierte Website verteilen. Benutzer folgen dem Link(Code), um die App aus einer versteckten Version des Apple App Stores herunterzuladen. Updates verhalten sich identisch wie mit MDM, können jedoch nicht direkt auf das Gerät gepusht werden. Die Anzahl der eingelösten Codes kann nachverfolgt werden, jedoch sind Informationen zum Betriebssystem, Gerät, Version und zur Nutzung der App sowie die Möglichkeit zur Entfernung der App nicht verfügbar.

**Managed Google Play Store**

Der Managed Google Play Store (MGPS) bietet sich für Android Geräte als Pendant zum Apple Business Manager (ABM) an. Sie sind in vielen Aspekten ähnlich aufgestellt. Der MGPS und ABM unterscheiden sich hauptsächlich in der Methode der App-Verteilung, der Integration und Verwaltung sowie den Benachrichtigungsmethoden.

Hinsichtlich der Nutzung ohne MDM erhalten Benutzende beim MGPS Einladungslinks, um auf alle genehmigte Apps im MGPS zuzugreifen. Im Gegensatz dazu verwendet ABM Einmalkauf-Codes (Redemption Codes) pro App für deren Verteilung. Genauer bedeutet dies, dass im Gegensatz zu ABM, die Google Accounts im Google EMM erfasst werden und der Link den Benutzenden so zugespielt wird. Bei ABM hingegen ist eine Erfassung der Benutzer zur Generierung des Redemption Codes nicht notwendig.

Bei den Benachrichtigungsmethoden informiert der Google Managed Play Store Benutzer über Updates und Benachrichtigungen im Google Play Store. ABM hingegen zeigt Benachrichtigungen für Updates als Tags oder Hinweise im App Store an.

### Deployment mit MDM (Microsoft Intune)

Möchte man mehr Kontrolle darüber haben, wer welche Version installiert hat oder wer welche Version installieren oder updaten muss, führt kein Weg an einer Mobile Device Management Lösung vorbei. Dies ist auch die empfohlene Variante von Apple und Google, wenn es um interne Apps geht.

A diagram of a devops application

Description automatically generated

*Figure 3: Verteilung über Intune Management MDM*

Sobald ein Gerät in der Organisation gemanagd wird, kann die Verteilung kontrolliert werden. Es kann also genau eingestellt werden, welche Geräte welche Version der App erhalten und kann die Updates automatisch vornehmen, wenn das gewünscht ist. Auch kann sichergestellt werden, dass die App deinstalliert wird, wenn beispielsweise das Gerät abhandenkommt. Und man kann dafür sorgen, dass alle Mitarbeitenden zeitnah die neueste Version installiert haben.

### Vergleich der Möglichkeiten

**Generelle Ansätze für Deployment**

Die drei verschiedenen Möglichkeiten zum Verteilen der Apps im direkten Vergleich:

|  |  |  |  |
| --- | --- | --- | --- |
| Kriterium | Public Store | Interne Website | MDM |
| Einfachheit Administration | Kompliziert | Mittel | Mittel |
| Deployment Aufwand | Gross | Klein | Klein |
| Automatisierung | Manuelles Deployment, Update durch Benutzer nötig | Manuelles Deployment, Update und Einwilligung durch Benutzer nötig | Automatisiertes Deployment |
| Kosten | Apple Developer License nötig | Keine | 5-50 Franken pro Device (Intunes ist schon lizenziert) |
| Abhängigkeiten | Starke Abhängigkeit von Apple | Keine (bei ABM Abhängigkeit von Apple Review) | keine |
| Speed für neue Versionen | Update kann bis zu 2 Wochen dauern | Sofort (bei ABM 1-2 Tage) | sofort |
| User Adoption | Braucht App Store Login | Braucht Zugriff auf Location oder über Direktlink. | Muss das Company Portal installieren und dort sein Gerät für die Firma freigeben. |
| Kontrolle über Version auf dem Gerät | Begrenzt | Keine | Volle Kontrolle |

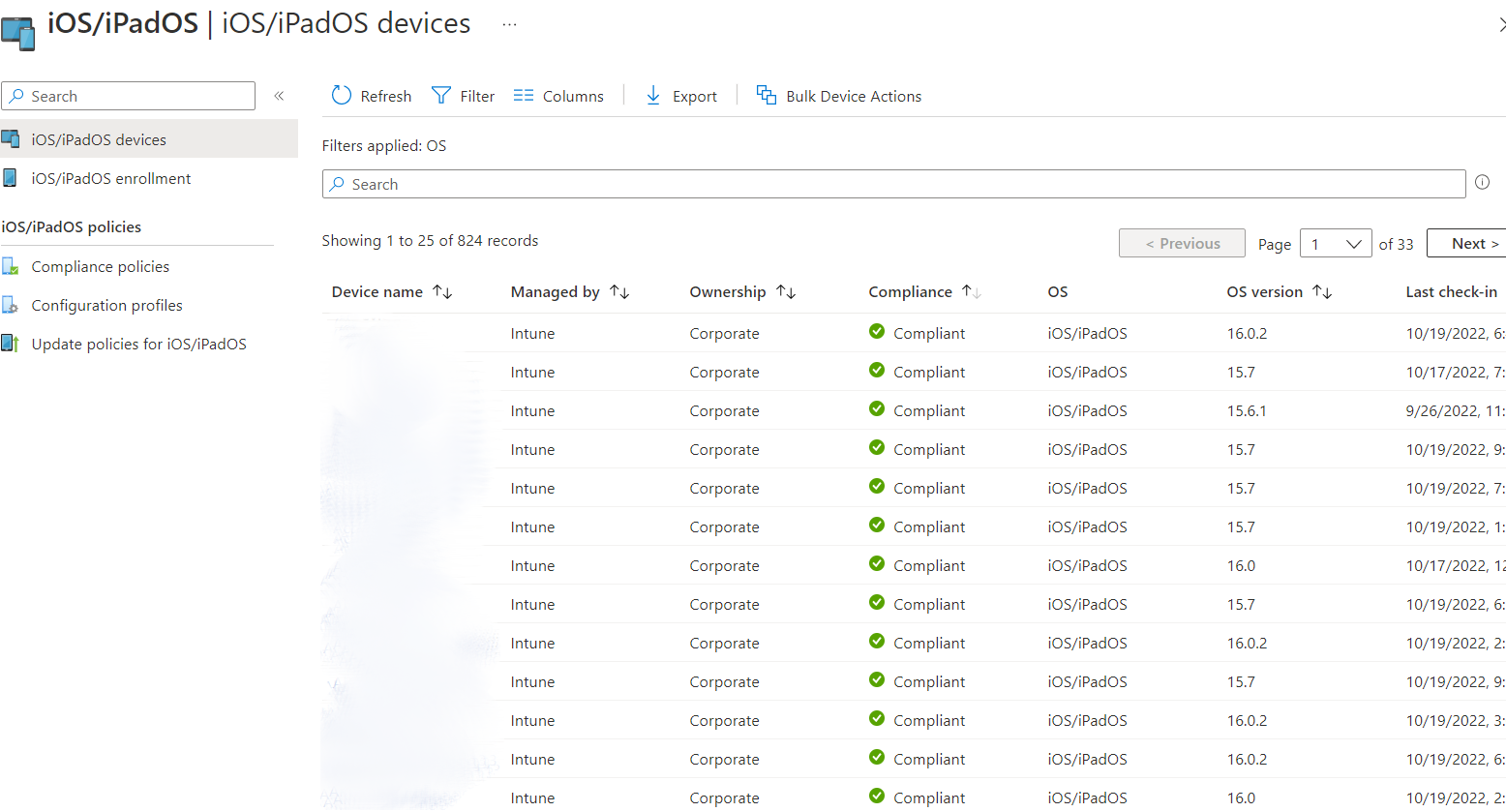
**Inhouse Distribution Systeme (ohne MDM)**

|  |  |  |  |
| --- | --- | --- | --- |
| Kriterium | Apple Business Manager | Managed Google Play Store | Appaloosa (und Weitere kommerzielle Tools) |
| Einfachheit Administration | Mittel | Mittel | Mittel |
| Deployment Aufwand | Komplex (inkl. App Review) | Komplex (inkl. Google Review) | Mittel |
| Automatisierung | Kein Self-Onboarding möglich, | Self-Onboarding möglich ohne Admin-Aufwand? | Self-Onboarding möglich mit minimalem Admin-Aufwand via API |
| Kosten | Gratis | Gratis | Je nach Funktionsumfang, ca. $4.99 pro Device |
| Abhängigkeiten | Apple Review, Verfügbarkeit Apple Dienste | Google Review, Verfügbarkeit Google Dienste | Verfügbarkeit Appaloosa Dienste |
| Speed für neue Versionen | Apple-Review dauert 1-2 Tage | Sofort | Sofort |
| User Adoption | Braucht Zugriff auf Location oder über Direktlink. | Braucht Zugriff auf Location oder über Direktlink. | Braucht Zugriff auf Location oder über Direktlink. |
| Kontrolle über Version auf dem Gerät | Keine | Keine | Information, wer was geladen hat. |

**Unsere Empfehlung: Deployment mit MDM – Verwendung Intune**

In den meisten Fällen empfehlen wir für die internen App Verteilung ein MDM System. Da es in der Regel aber aufwändig ist, ein neues System einzuführen, muss das Verhältnis von Aufwand und Nutzen gut überprüft werden, gerade bezüglich Kosten.

Da Helion aber mit Intune bereits ein System mit integrierter MDM-Funktionalität im Einsatz hat, gestaltet sich die Nutzung und Einführung des Systems als äusserst einfach. Zusätzlich zu den verwalteten Laptops können einfach zusätzlich auch die Smartphones verwaltet werden. Bei Geschäfts- Smartphones wird das Company Portal automatisch in vorinstalliert und bei privaten Smartphones muss dies durch den Mitarbeitenden geschehen. Erst einmal aufgenommen, kann nun die App im Intune registriert und auf den gewünschten Geräten automatisch installiert werden.



*Figure 4: Hinzufügen und Managen von Smartphones per Intune auf einfachste Art und Weise*

Ein Bild, das Text, Software, Computersymbol, Multimedia-Software enthält.

Automatisch generierte Beschreibung

*Figure 5: App registrieren und den Geräten zuweisen in Intune.*

Da wir bereits für helion die Clients (Laptops) über intunes verwalten, wäre es ein Leichtes, dort auch die Smartphones zu verwalten. Der Aufwand wäre minimal.

Da sich mit Intune die Synergien der bestehenden Systeme am besten und preiswertesten nutzen lassen, empfehlen wir an dieser Stelle den Einsatz von Intune. Wir nutzen übrigens Intune auch bei anderen Kunden bereits für Smartphones, was sehr gut funktioniert.