POC 1:

Implementation eines Publish-Subscribe -Nachrichtenservice mittels “RabbitMQ “

Exit:

Androidclient

1. kann Topics abonnieren
2. erhält asynchron Nachrichten über abonnierte Topics
3. kann auf Topics publishen

Node-JS-Client

1. kann auf Topics publishen

Fail:

1. Nachrichten zu abonnierten Topics werden nicht an Androidclient übermittelt.
2. Im RabbitMQ-Manager sind die Consumer und die erstellten Queues nicht aufgeführt.

Fallback:

Implementation eines Publish-Subscribe Nachrichtenservice mittels “Faye”

POC 2:

Prüfung von Arzneimittelinteraktionen, Kontraindikation mittels des “ifapWebservice”

Exit:

1. kostenfreier Zugriff auf Webservice vom Betreiber ifap gestattet
2. Zurückgelieferte Daten des Webservice können verwendet werden, um Kontroindikationen und Nebenwirkungen zu prüfen mittels den Daten der Patientenakte
3. Anbindung in den Node Server gelingt

Fail:

1. Zugriff auf Webservice nicht kostenfrei gestattet
2. Anbindung des Webservices an das MDKS (Medikations-System) misslingt

Fallback:

prototypische Simulation der Arzneimittelinteraktions- und Kontraindikationsprüfung auf Grundlage selbstverfasster Datensätze

---

POC 3:

Sichere Datenübermittlung vom Android-Client zum Node-Server durch SSL oder TLS

Exit:

1. Eine https Verbindung vom mobilen Client zum NodeJS-Server ist möglich.
2. Eine Abfrage vom Client mit dem https-Protokoll ist erfolgt.

Fail:

1. Eine https Verbindung vom mobilen Client zum NodeJS-Server ist nicht möglich.
2. Eine Abfrage vom Client mit dem https-Protokoll ist nicht erfolgt.

Fallback:

unverschlüsselte Datenübertragung innerhalb eines VPNs

POC 4:

Implementation eines Ausweichservers

Exit:

Bei einen Ausfall des Hauptserversausfall werden Anfragen der Clients automatisch auf einen Ausweichserver umgeleitet und der Abruf ist trotzdem möglich.

Fail:

Der Abruf vom Client an den Server ist nicht mehr möglich, nachdem ein Server ausfällt. Die Abfrage des Clients wird nicht auf einen Ausweichserver umgeleitet.

Fallback:

"Failback" (Zurücksetzen des Systems auf einen vorherigen Zustand)

POC 5:

Implementation eines Notification-Service auf einem Android-Client für Benachrichtigungen, die vom Node-Server angestoßen werden.

Exit:

1. Ein Service kann im Android-Client implementiert werden
2. Eine Notification kann erstellt werden.
3. Benachrichtigungen werden auch außerhalb der Applikation innerhalb der Androideigenen Notification Area angezeigt, auch wenn die Applikation geschlossen ist.
4. Benachrichtigungen werden durch Nachrichten auf ein Topic des RabbitMQ-Servers angestoßen.

Fail:

1. Ein Service kann nicht im Android-Client implementiert werden
2. Eine Notification kann nicht erstellt werden.
3. Benachrichtigungen können nicht außerhalb der Applikation innerhalb der Androideigenen Notification Area angezeigt werden auch wenn die Applikation geschlossen ist.
4. Benachrichtigungen können nicht durch Nachrichten auf ein Topic des RabbitMQ-Servers angestoßen werden.

Fallback:

Sollte die Implementation scheitern, so muss sichergestellt werden, das in der Applikation intern ohne den Androidsystemservice zu nutzen, ein Benachrichtigungssystem implementiert wird, wodurch der Nutzer über Neuigkeiten informiert wird.
Eine Benachrichtigung per E-Mail ist eine hinreichende Alternative.

POC 6:

Implementierung einer MySQL-Datenbank in einem NodeJS-Server und synchroner Datenaustausch zwischen Android-Anwendung und NodeJS-Server und anschließende Speicherung der Daten in einer MySQL-Datenbank

Exit:

1. Implementierung der MySQL-Datenbank in NodeJS ist erfolgt
2. Daten werden vom Android-Client fehlerfrei in der Datenbank abgelegt
3. Daten sind über die Android-Anwendung fehlerfrei abrufbar

Fail:

1. Implementation der MySQL-Datenbank in NodeJS nicht möglich
2. Daten werden nicht oder fehlerhaft gespeichert
3. Abruf der gespeicherten Daten schlägt fehl

Fallback:

Implementierung einer MongoDB-Datenbank im NodeJS-Server