Hinweis zur Dokumentation:
Diese Version wurde für mein persönliches Portfolio angepasst. Sämtliche Originalbilder des Kunden wurden durch neutrale Platzhalterbilder ersetzt.

Aus rechtlichen Gründen kann der Quellcode leider nicht veröffentlicht werden, da das Projekt im Rahmen eines Auftrags für die Firma *Exxcellent solutions* entstand und sämtliche Rechte am Code nun beim Unternehmen liegen.

Dennoch war das Projekt aus technischer und organisatorischer Sicht ein voller Erfolg. Ich bin sehr stolz auf das Ergebnis und auf unsere Teamleistung.

Kundenfeedback: *Sehr gut*
Bewertung: *Note 1,3*

FIGURE 1

Teamorientiertes Projekt Automatische Arbeitsplatz-ubersicht

Team B
Sommer Semester 2025

Technische Hochschule Ulm
Teamorientiertes Projekt
Prof. Dr. Alfred Franz

Inhaltsverzeichnis

1. Einleitung

Im nachfolgenden Abschnitt werden die Motivation des Projekts im Sinne einer Projektvision, sowie die nach dem SMART Verfahren definierten Projektziele aufgezeigt.

1.1 Projektvision

Im Rahmen des Projekts „Automatisierte Arbeitsplatzübersicht“ soll die in der Firma eXXcellent solutions bestehende Raumbuchungslosung erweitert und verbessert werden. HierfOr ist die Erstellung einer Smartphone Anwendung, mit deren Hilfe die Raumbuchung automatisch und ohne Interaktion der Mitarbeitenden erfolgt, gewOnscht. Des Weiteren soll im Eingangsbereich von eXXcellent solutions ein gror!er Bildschirm installiert werden, auf dem eine GesamtObersicht der anwesenden Mitarbeiter zu sehen ist. Vor den einzelnen BOros sollen kleinere Monitore installiert werden, welche eine detailliertere Obersicht der aktuell an//*wesenden Mitarbeitenden anzeigen. Sowohl die Gesamtansichten als auch die BOroansichten sollten Ober einen Web-Service im Browser abrufbar sein und sich dynamisch der Gror!e des anzeigenden Monitors anpassen.

1.2 Projektziele

Die Projektziele des Projekts wurden nach SMART definiert, hierbei ist jedes Ziel:

- Spezifisch
- Messbar
- Erreichbar
- Relevant
- Zeitgebunden

1	Die Projektplanung des Projekts “Automatische Arbeitsplatzübersicht” wird bis zum 18.03.2025 im Rahmen des 1. Projekt-Meilensteins vollstandig abgeschlossen und dokumentiert sein.
2	Bis zum 01.04.2025 soll eine erste Demoversion des Projekts “Automatische Arbeitsplatzübersicht” entwickelt werden. Diese umfasst einen Mobile-App-Client und eine Web-Anwendung mit eingeschrankten Funktionalitaten. Wichtige Funktionalitaten sind das Buchen von Arbeitsplatzen per Mobile-App-Client Ober die desk.ly-API, sowie das Anzeigen einer GesamtObersicht der gebuchten Arbeitsplatze Ober die Web-Anwendung, sowie alle dafOr benotigten Unter-Features. Die Demoversion gilt als Erfolg, wenn diese Funktionalitaten implementiert, getestet und funktionsfahig sind.
3	Der Mobile-App-Client und die Web-Anwendung des Projekts “Automatisierte Arbeitsplatzübersicht” werden bis zum 22.04.2025 vollständig entwickelt und bereitgestellt. Ab diesem Zeitpunkt sollen nur noch Robustheitstests und Bugfixes erforderlich sein, ohne dass gror!ere Entwicklungsarbeiten notwendig sind. Die Fertigstellung ist erreicht, wenn alle geplanten Funktionen implementiert wurden.

4	Das Projekt “Automatische Arbeitsplatzübersicht”, einschließlich der vollständigen Dokumentation, soll bis zum 29.04.2025 abgeschlossen sein. Ab diesem Zeitpunkt sollen keine weiteren inhaltlichen Arbeiten erforderlich sein, sodass bis zum 02.05.2025 nur noch ausschlier!lich die Projekt-Abschlusspra-sentation erstellt und eingeObt werden muss. Der Abschluss des Projekts ist erreicht, wenn alle definierten Features implementiert und die Dokumentation finalisiert wurde.
5	Die Abschlusspräsentation des Projekts “Automatische Arbeitsplatzübersicht” wird als zweiter Meilenstein am 02.05.2025 gehalten. DafOr wird die Prasentation bis zu diesem Termin vollstandig erstellt und eingeObt, sodass sie Oberzeugend prasentiert werden kann. Der Meilenstein gilt als erreicht, wenn die Prasentation vorbereitet, intern geObt und termingerecht durchgefOhrt wurde.
6	Die Abgabe der Projektdokumentation und der Quelltexte stellt den dritten Meilenstein des Projekts “Automatische Arbeitsplatzübersicht” dar und erfolgt bis 04.05.2025. Dieses Ziel gilt als erfolgreich abgeschlossen, wenn die vollstandige Projektdokumentation und alle finalen Quelltexte fristgerecht eingereicht wurden.

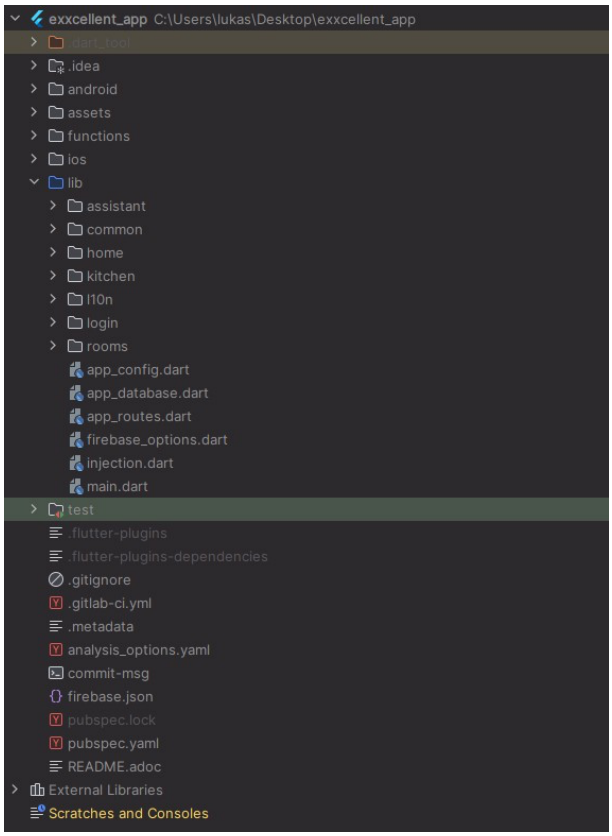
2. Technischer Hintergrund und Vorarbeiten

Im Rahmen dieses Projekts wurden eine Web-Anwendung und ein Mobile-App Client entwickelt. Dabei baut der von uns im Rahmen dieses Projekts entwickelte Mobile-App Client auf der Vorarbeit von eXXcellent solutions hauseigener Mobilanwendung namens „eXXcellent App“ auf. Im nachfolgenden Abschnitt soll genauer auf diese Vorarbeit eingegangen werden.

Vorarbeiten von eXXcellent solutions

Zu Beginn des Projekts „Automatische ArbeitsplatzObersicht“ Obernahm das Mobile-App Team unseres Projekt Teams ein geforktes Repository der von eXXcellent solutions hauseigenen Mobilanwendung eXXcellent App. Nachfolgend wird veranschaulicht, welche Inhalte bereits implementiert und im Laufe des Projekts relevant waren.

Die Ordnerstruktur der 0berordner sah bei der Repository 0bernahme so aus:



Wahrend des Projekts waren folgende 0berordner relevant:

- android (Android Build Dateien/Konfigurationen, sowie native Implementierungen)
- assets (Dart-Define Konfigurationen, sowie Logos von eXXcellent solutions)
- ios (XCode Projekt Dateien, sowie native Implementierungen)
- lib (Dart Quellcode der Anwendung)
- test (Verschiedenste Anwendungstests)

Stand des Android Build-Workflows bei der Projektubernahme:

Der Android Build Workflow der eXXcellent App warf zu Beginn eine Fehlermeldung aus, da die von Firebase benotigte Konfigurationsdatei namens g*oogle-services.json* nicht gefunden werden konnte. Diese Datei befand sich lediglich an der falschen Stelle innerhalb des *android*-Verzeichnisses. Durch das Verschieben an den korrekten Speicherort konnte der Fehler schnell behoben werden, wodurch der Android-Build-Workflow lauffahig war.

Stand des iOS Build Workflows bei der Projektubernahme:

Der iOS Build Workflow war bei der ProjektObernahme nicht lauffahig. Dieser wurde im Laufe des Projekts repariert und erweitert.

Auflistung relevanter bereits Implementierte Features:

- lib/common Fur Android aktivierte Lokale Benachrichtigungen und Logger.
- lib/I10n Vorhandene App Lokalisierungen.
- lib/login Funktionaler Microsoft Single Sign-On Login via Firebase Authentication.
- lib/newProject Startpunkt der fur unser Projekt in der eXXcellent App eingerichtet wurde.

Commit-Hash des 0bernahmestands

Wir haben das Projekt ursprOnglich vom Branch „students/main“ übernommen. Hierbei war der Commit-Hash des letzten Commits vor der Repository Obernahme:

5a27bd28808c6522ffbf97ab4aa19cdef6d1718b

3. Anforderungen

Dieser Abschnitt der Projektdokumentation geht genauer auf die Anforderungen des Projekts ein. Hierbei wird zwischen funktionalen und nicht funktionalen Anforderungen unterschieden.

3.1 Funktionale Anforderungen

Funktionale Anforderungen beschreiben Funktionen, welche eine Software erfOllen muss, um die BedOrfnisse der Nutzer zu erfOllen. Sie sind direkt mit der Nutzung und dem gewOnschten Verhalten des Systems verbunden und bilden die Grundlage fOr die Entwicklung der Software.

	Anforderung	**Prioritat**
1	Die Arbeitsplatzbuchung soll Ober einen Mobile-App-Client erfolgen. Dazu soll ein fOr jeden Arbeitsplatz einzigartiger NFC-Tag ausgelesen werden, anhand dessen Informationen die Arbeitsplatzbuchung entsprechend abgewickelt wird. Von den Mitarbeitenden soll hierfOr keine weitere Interaktion, als das Smartphone Ober den NFC-Tag zu halten, erforderlich sein.	HOCH
2	Ober eine Web-Anwendung soll eine GesamtObersicht aller aktuell im Unternehmen anwesenden Mitarbeitenden realisiert werden. Diese GesamtObersicht soll auf einem gror!en Monitor im Eingangsbereich von eXXcellent solutions Ober den Browser dargestellt werden.	HOCH
3	Ober eine Web-Anwendung soll fOr jedes BOro eine detailliertere Ansicht der aktuell im BOro anwesenden Mitarbeitenden ermoglicht werden. Diese BOroObersicht soll jeweils auf kleineren Monitoren vor jedem BOro von eXXcellent solutions Ober den Browser dargestellt werden.	HOCH
4	Um einen Arbeitsplatz zu buchen, sollen sich die Mitarbeitenden von eXXcellent solutions im Mobile-App-Client mittels Microsoft Single Sign-On einloggen konnen.	MITTEL
5	Die Darstellung der Gesamtansicht und der einzelnen BOroansichten Ober den Browser soll keine Runtime Umgebung benotigen, damit sichergestellt ist, dass die Darstellung auf allen Monitoren ohne Probleme erfolgen kann.	HOCH
6	Im Eingangsbereich von eXXcellent solutions soll direkt neben der EingangstOr ein iBeacon installiert werden, welcher dauerhaft ein Signal zum Ausloggen einer aktuellen Arbeitsplatzbuchung vorbeilaufender Mitarbeiter sendet. Damit sollen Arbeitsplatzressourcen effizient und automatisiert wieder freigegeben werden.	NIEDRIG

3.2 Nicht Funktionale Anforderungen

Nicht funktionale Anforderungen legen die Qualitatsmerkmale und Leistungsstandards fest, die eine Software erfOllen muss, ohne sich auf spezifische Funktionen zu beziehen. Sie betreffen Aspekte wie die Performance, Benutzerfreundlichkeit, Sicherheit, VerfOgbarkeit und Skalierbarkeit des Systems. Diese Anforderungen definieren, wie gut das System seine Aufgaben erfOllen soll, z. B.: durch schnelle Reaktionszeiten oder hohe Ausfallsicherheit. Sie sind entscheidend, um sicherzustellen, dass das System unter realen Bedingungen effizient und zuverlassig funktioniert.

	Anforderung	**Prioritat**
1	Die Arbeitsplatzbuchung muss fOr die Mitarbeitenden moglichst einfach sein, damit diese akzeptiert und angewendet wird.	HOCH
2	Der Quellcode unserer Implementierungen soll hohe Qualitat aufweisen. Hierzu sollen die Architekturen, sowie die Wart- und Erweiterbarkeit der Implementierungen bereits in der Planungsphase berOcksichtig und in der Entwicklung entsprechend umgesetzt werden.	HOCH
3	Der Mobile-App-Client soll eine intuitive Benutzeroberflache bieten.	MITTEL
4	Der Mobile-App-Client soll so optimiert sein, dass er auf den ausfOhrenden Mobilgeraten moglichst wenig Akku verbraucht.	NIEDRIG
5	Die Mobile-App soll auf den Plattformen Android und iOS umgesetzt werden.	HOCH

4. Methodik und Vorgehen

lm nachfolgenden Abschnitt wird auf die Methodik und das Vorgehen bei der Projektumsetzung eingegangen. Hierzu wird zuerst auf das Technische Konzept dieses Projekts eingegangen. Danach wird die Software-Architektur des Projekts genauer beleuchtet. lm Anschluss daran wird die Projektorganisation betrachtet. Abgerundet wird das Kapitel Methodik mit den Beitragen der einzelnen Teammitglieder am Projekt.

4.1 Technisches Konzept

Die Wahl der richtigen Technologien ist ein kritischer Bestandteil fOr die erfolgreiche Umsetzung eines Projektvorhabens. Die ausgewahlten Technologien sollten ausgereifte, skalier- und wartbare Losungen sein, mit welchen sowohl die funktionalen als auch die nicht funktionalen Anforderungen des Projekts umgesetzt werden konnen.

4.1.1 Technologieauswahl des Mobile-App Clients:

Vorgegeben durch die in Abschnitt 2 erwahnte Vorarbeit von eXXcellent solutions.

1. Programmiersprache:
    - Dart
    - Kotlin
2. Framework:
    - Flutter
3. Tools:
    - Android Studio (Entwicklungsumgebung)
    - Git (Versionskontrolle)
    - GitLabs Cl (Automatisiertes Cl/CD)

4.1.2 Technologieauswahl der Web-Anwendung:

1. Programmiersprachen:
    - JavaScript
2. Designsprachen:
    - HTML
    - CSS
3. Frameworks:
    - NodeJS

- Express
- Jest
- Supertest

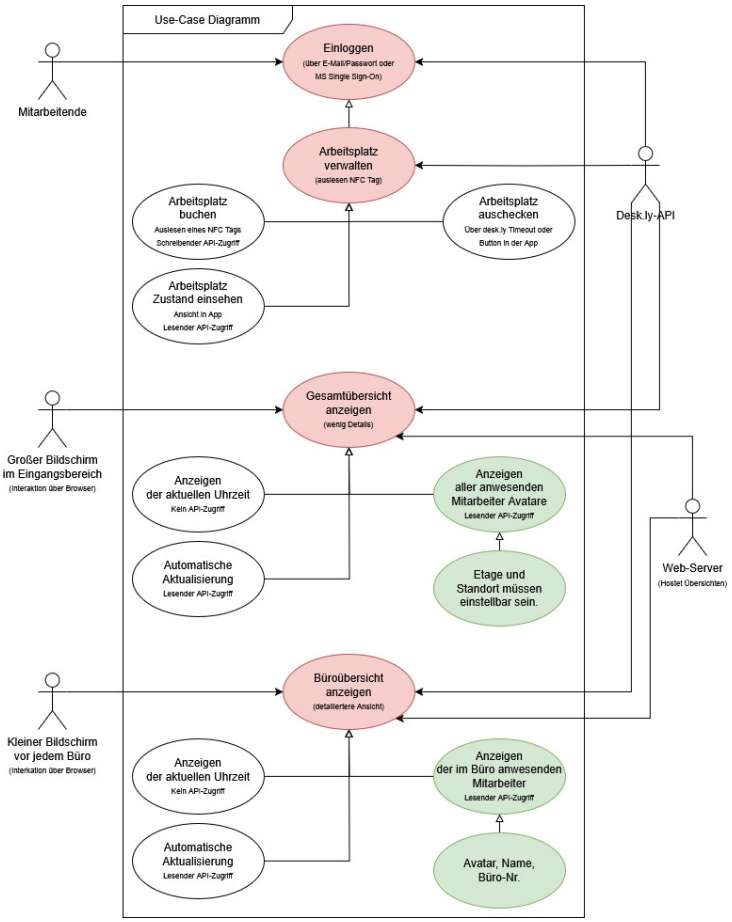
4. Tools:

• Android Studio	(Entwicklungsumgebung)
• Git	(Versionskontrolle)
• GitLabs CI	(Automatisiertes CI/CD)
• Docker	(Einfaches Deployment)

4.2 Softwarearchitektur

Nachfolgend werden alle relevanten UML-Diagramme des Mobile-App Clients und der Web-Anwendung unseres Projekts aufgezeigt und erklart, um dem Leser einen umfangreichen Einblick in die Softwarearchitektur dieses Projekts zu geben. Es gilt zu beachten, dass nur die im Rahmen dieses Projekts erstellten Teile der Mobile-App beleuchtet werden.

4.2.1 UML-Use-Case Diagramm

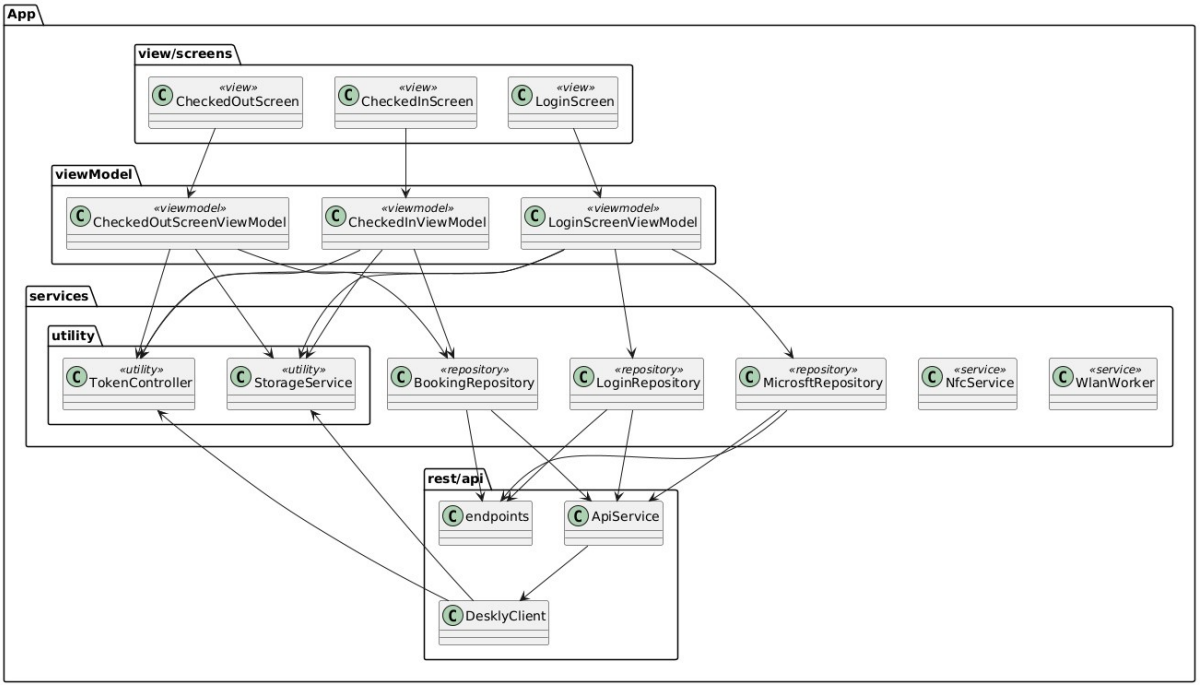


Das Use-Case Diagramm des Projekts zeigt, dass die Mitarbeitenden Ober den Mobile-App Client nach einer Authentifizierung in der Lage sein sollen ihren Arbeitsplatz zu verwalten. Hierzu zahlen Funktionen wie das Ein- und Ausbuchen an einem Arbeitsplatz, sowie das Einsehen des aktuellen Buchungszustands.

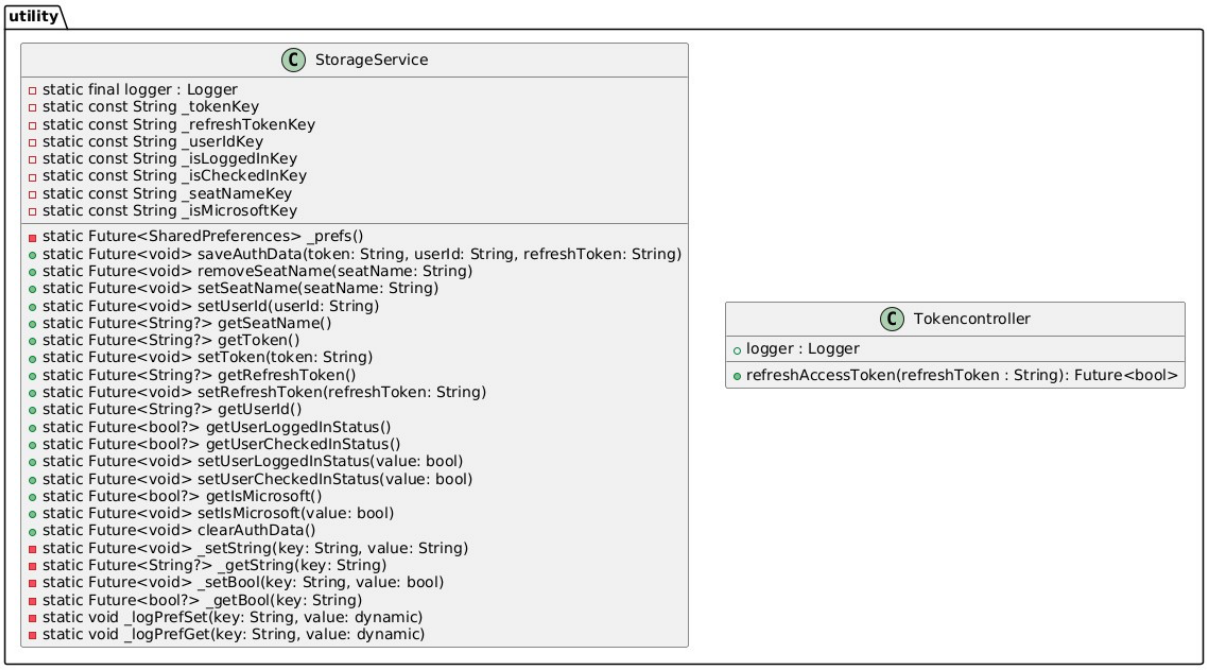
Des Weiteren wird gezeigt, dass die GesamtObersicht dazu in der Lage sein muss, die aktuelle Uhrzeit, sowie alle anwesenden Mitarbeitenden anzuzeigen. Hierbei soll sich die Ansicht automatisch aktualisieren und entsprechend dem Standort und der Etage konfigurierbar sein.

Auch die BOroansicht muss in der Lage sein die aktuelle Uhrzeit, sowie die aktuell im BOro anwesenden Mitarbeitenden anzuzeigen. Auch hierbei soll sich die Ansicht wieder automatisch aktualisieren.

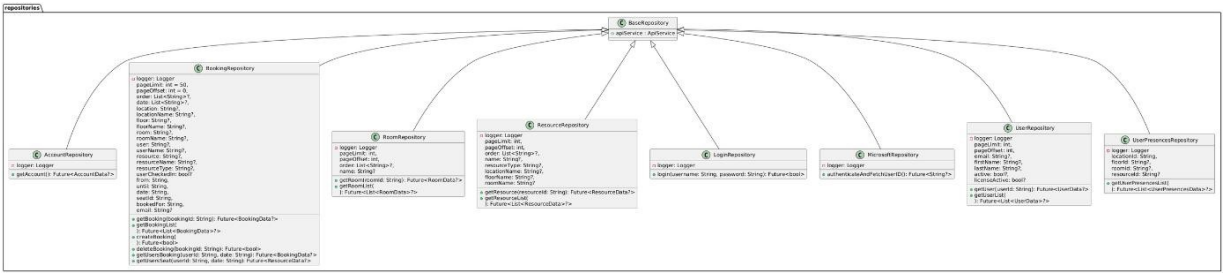
4.2.2 UML Klassen Diagramm



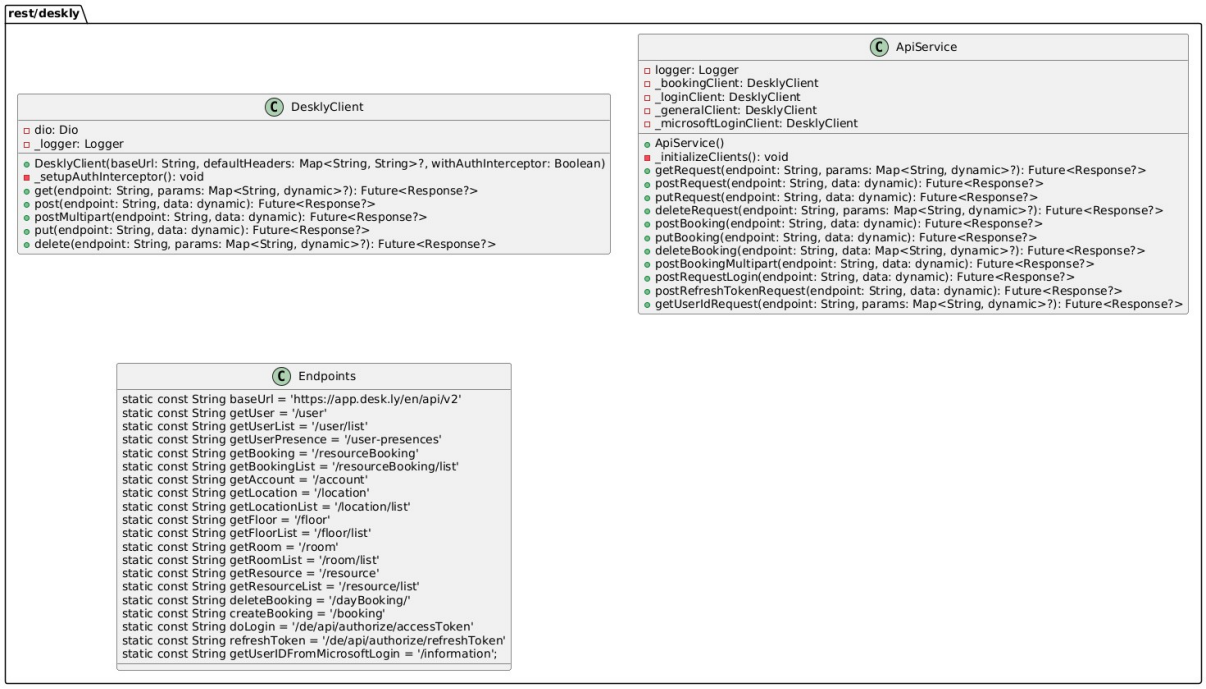
Hier ist eine grobe Obersicht, um die Zusammenhange der folgenden Diagramme besser darstellen zu konnen, die Klassen wurden bewusst leer gelassen, um nur auf die Zusammenhange einzugehen. Die NFC-Implementierungen und ihre Beziehungen werden weiter unten genauer erklart.



Nun kommen wir zu unserem “Utility”-Paket, das Klassen enthalt, die Oberall verwendet werden konnen. Sie helfen der Anwendung, ihren aktuellen Zustand zu halten und diesen nach dem Beenden der Anwendung wiederherzustellen. Der Token Controller hat nur eine Aufgabe und das ist die Aktualisierung des Bearer Tokens, nachdem der alte Token abgelaufen ist.



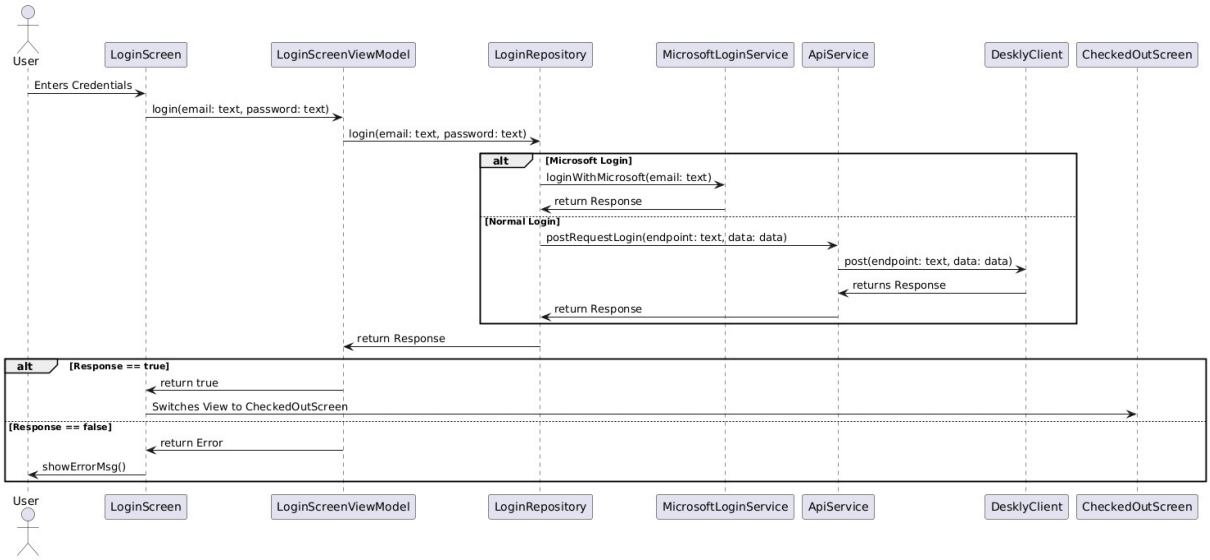
Hier werden nun alle Services aufgelistet, die unsere Anwendung verwendet, diese werden als Schnittstelle zwischen den View Models und den API-Anfragen betrachtet.



Alle Funktionen der App werden zentral in diesem Modul verarbeitet. Dabei unterscheiden wir zwischen vier verschiedenen Client-Typen:

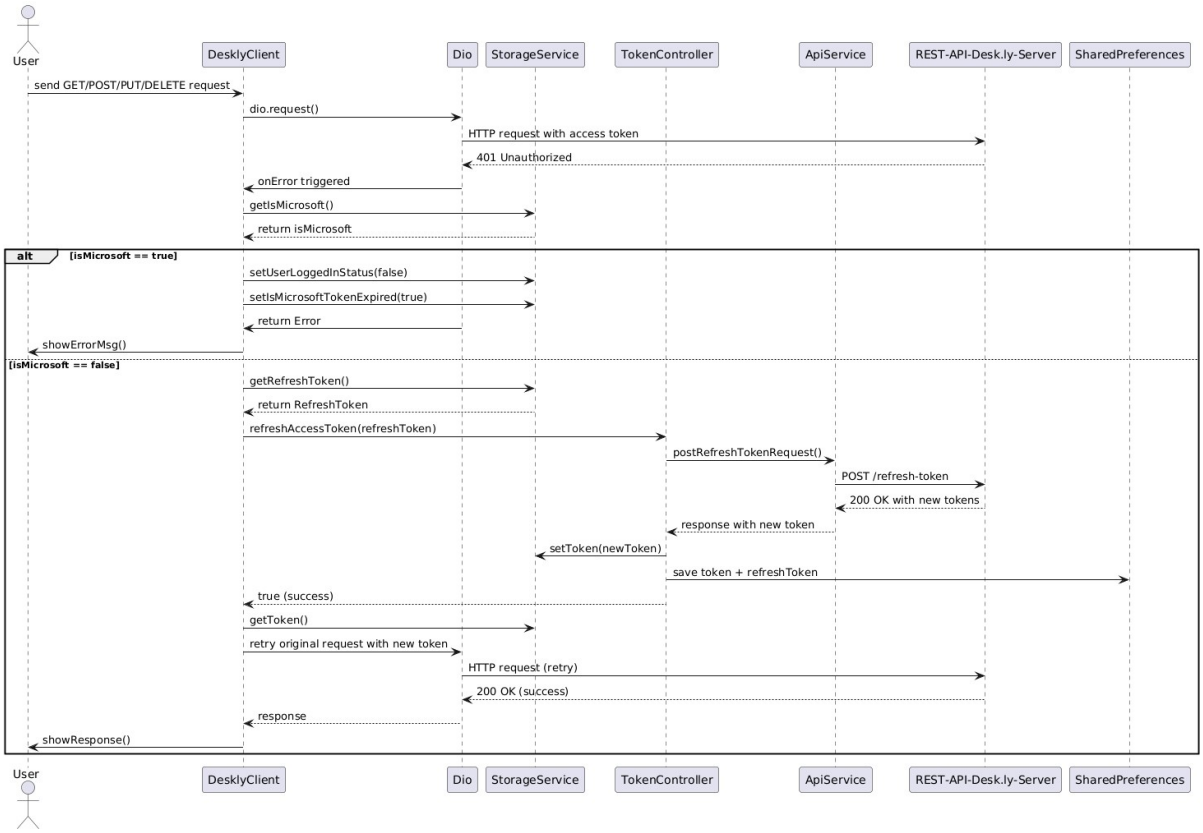
- API-Client: Verwendet die offizielle REST-API.
- Nicht-Authentifizierter Client: Ermoglicht den Login ohne vorherige Authentifizierung.
- Microsoft-SSO-Client: Ermoglicht die Anmeldung Ober ein Microsoft-Konto.
- Token-basierter Client: Nutzt einen Bearer Token zur Authentifizierung, um Funktionen zu ermoglichen, die von der offiziellen API noch nicht unterstOtzt werden.

4.2.3 Login App-Client UML Sequenz Diagramm



- Benutzereingabe:
  - Der User gibt seine Anmeldedaten (E-Mail und Passwort) im LoginScreen ein.
- Verarbeitung:
  - Der LoginScreen ruft die Methode login(email, password) im LoginScreen-ViewModel auf.
  - Das ViewModel leitet die Login-Daten an das LoginRepository weiter.
- Login-Methodenwahl:
  - Microsoft Login:
    a) Das LoginRepository ruft den MicrosoftLoginService auf, um sich Ober Microsoft zu authentifizieren.
    b) Der MicrosoftLoginService gibt eine Antwort zurOck.
  - Normaler Login (mit Benutzername/Passwort):
    a) Das LoginRepository ruft den ApiService auf, um die Login-Daten an den Server zu senden.
    b) Der ApiService sendet eine POST-Anfrage an den DesklyClient, der die Anfrage verarbeitet und eine Antwort zurOckgibt.
    c) Der ApiService gibt die Antwort an das LoginRepository zurOck.
- Antwort zurOck an das ViewModel:
  - Das LoginRepository gibt die erhaltene Antwort an das LoginScreenViewModel zurOck.
- Entscheidung – Erfolgreicher Login?
  - Ja (Response == true):
    a) Das LoginScreenViewModel gibt true zurOck.
    b) Der LoginScreen zeigt den Erfolg an und wechselt zur Checked-OutScreen.
- Nein (Response == false):
  - Das LoginScreenViewModel gibt einen Fehler zurOck.
  - Der LoginScreen zeigt dem User eine Fehlermeldung an.

4.2.4 Refresh Expired Token App-Client Sequenz Diagramm



Ein Benutzer startet eine HTTP-Anfrage, z. B. ein GET oder POST.
Diese Anfrage wird vom DesklyClient an die HTTP-Bibliothek Dio weitergeleitet. Dio hangt das aktuelle Access Token an und sendet die Anfrage an den Server.

Antwortet der Server mit einem 401-Fehler, bedeutet dies, dass das Token abgelaufen oder ungOltig ist. In diesem Fall greift die integrierte Fehlerbehandlung des Clients.

Der DesklyClient prOft zunachst, ob der Benutzer mit einem Microsoft-Login angemeldet ist. Ist dies der Fall, wird wie folgt verfahren:

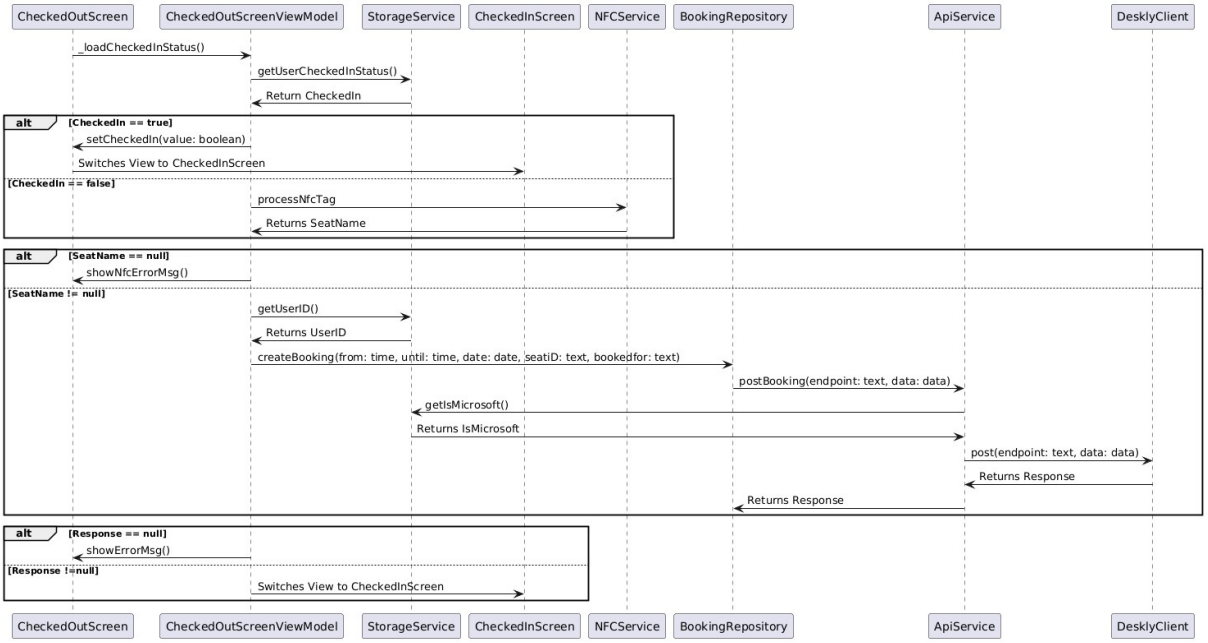
- Das Token wird nicht automatisch erneuert.
- Stattdessen wird der Benutzer intern als ausgeloggt markiert.
- Zusatzlich wird gespeichert, dass der Microsoft-Token abgelaufen ist.
- Die ursprOngliche Anfrage wird nicht wiederholt.
- Der Benutzer erhalt eine entsprechende Fehlermeldung.

Handelt es sich nicht um eine Microsoft-Anmeldung, wird anders verfahren:

- Der Client holt sich den gespeicherten Refresh-Token.
- Mit diesem versucht er, beim Server ein neues Access Token anzufordern.
- Gelingt dies, wird der neue Token gespeichert.

- Danach wird die ursprOngliche Anfrage automatisch wiederholt, diesmal mit einem gOltigen Token.
- Wenn alles funktioniert, erhalt der Benutzer wie gewohnt eine erfolgreiche Antwort vom Server.
- Schlagt die Token-Erneuerung fehl, wird der Benutzer ebenfalls ausgeloggt und Ober den Fehler informiert.

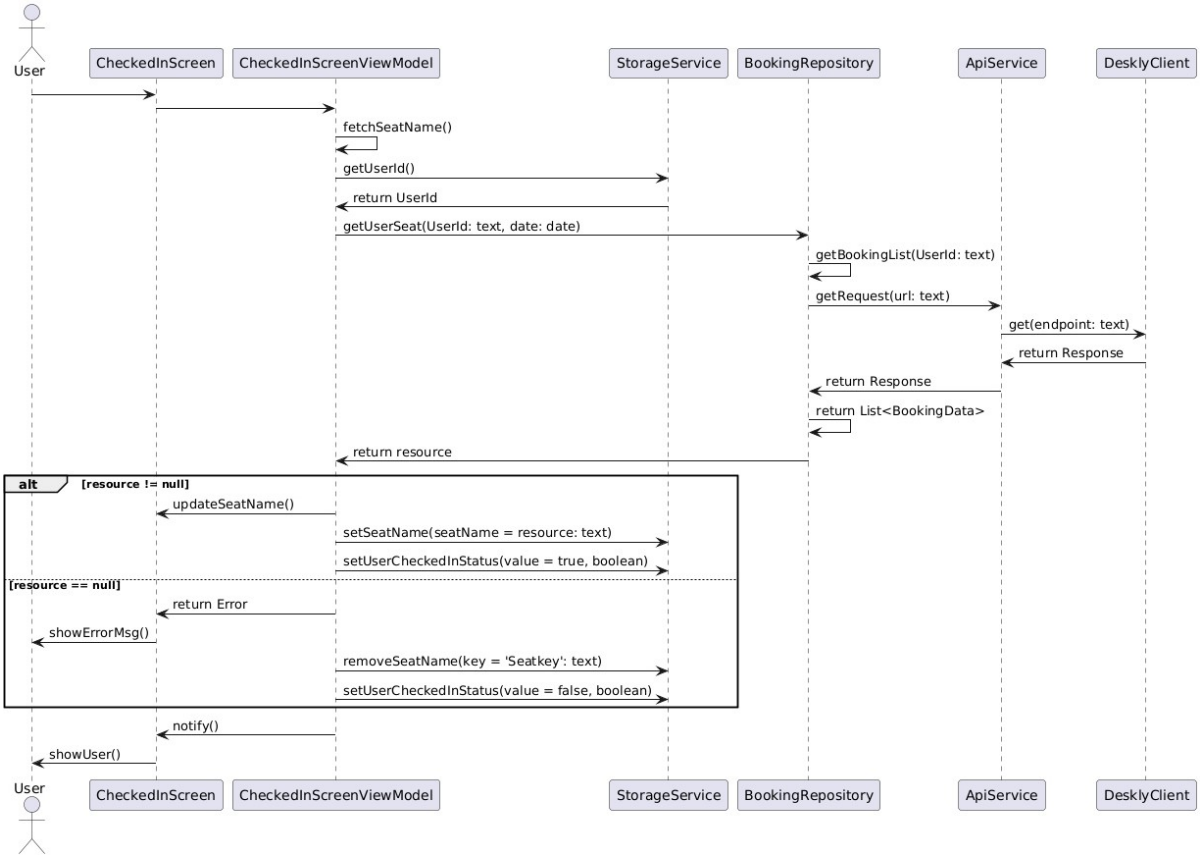
4.2.5 CheckedOutScreen App-Client UML Sequenz Diagramm



Dieses UML-Sequenzdiagramm beschreibt den Ablauf beim Offnen der Anwendung durch den Benutzer und den automatischen Check-in-Vorgang:

- Start durch den Benutzer: Der Benutzer offnet die Anwendung bzw. den CheckedOutScreen.
- Statusabfrage: Die View fragt das CheckedOutScreenViewModel nach dem aktuellen Check-in-Status, der aus dem StorageService geladen wird.
- Abhangig vom Status:
  - Wenn der Benutzer bereits eingecheckt ist: Wechselt direkt zum CheckedInScreen.
  - Wenn der Benutzer noch nicht eingecheckt ist: Ein NFC-Tag wird eingelesen.
    a) Kein Sitzplatz gefunden: Es wird eine Fehlermeldung angezeigt.
    b) Sitzplatz gefunden: Die App ruft die User-ID ab, erstellt eine Buchung Ober das BookingRepository, welches Ober den ApiService mit dem Server (DesklyClient) kommuniziert.
- Nach dem Buchungsversuch:
  - Fehlgeschlagen: Der Benutzer erhalt eine Buchungsfehlermeldung.
  - Erfolgreich: Die Ansicht wechselt zum CheckedInScreen, der den Benutzer benachrichtigt.

4.2.6 CheckedInScreen App-Client UML Sequenz Diagramm



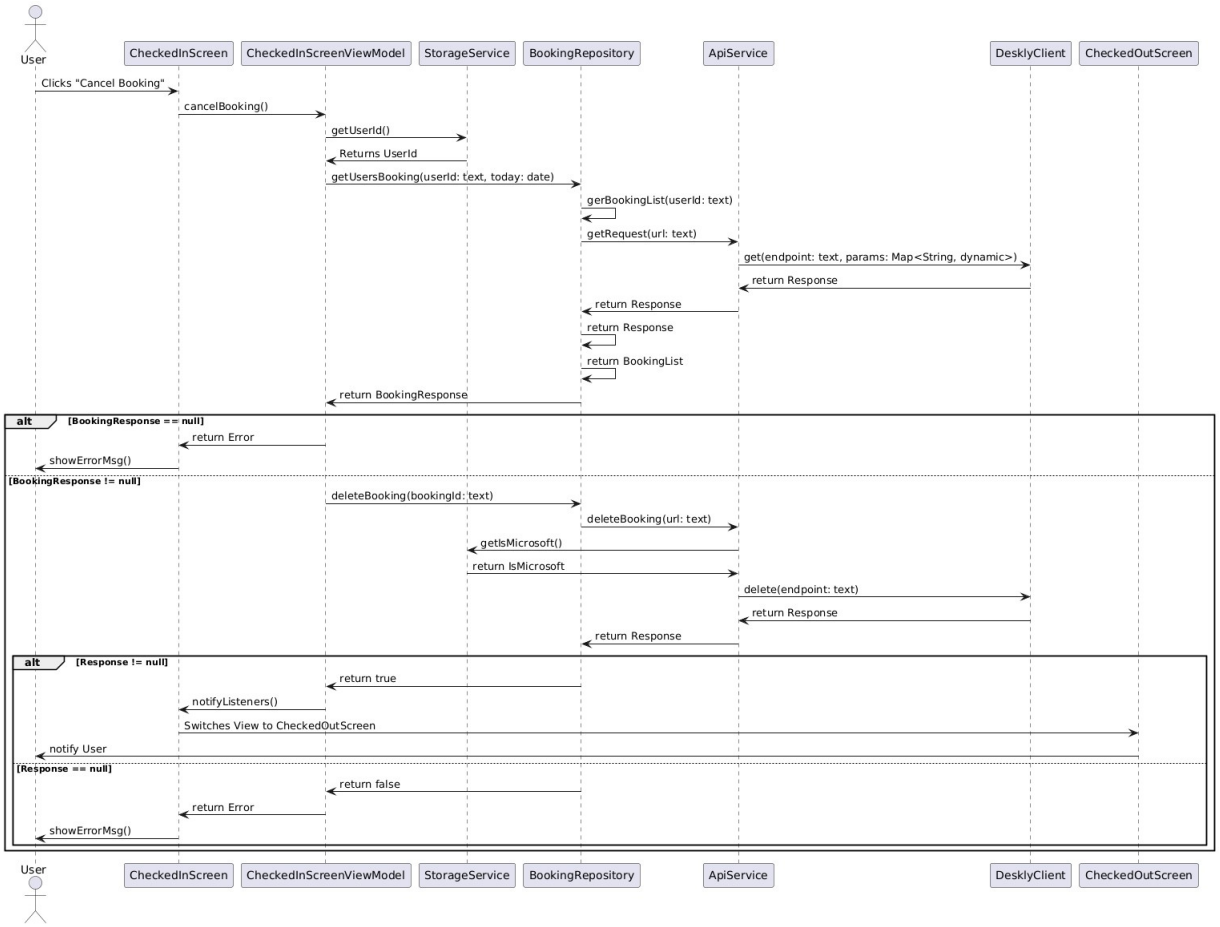
Dieses UML-Sequenzdiagramm zeigt den Ablauf des Abrufs des Sitzplatznamens (SeatName) fOr einen eingecheckten Benutzer in der Anwendung:

- Benutzereingabe: Der Benutzer offnet den CheckedlnScreen.
- Verarbeitung:
  - Der CheckedlnScreen ruft das CheckedlnScreenViewModel auf.
  - Das ViewModel ruft die Methode fetchSeatName() auf.
  - Es fragt den StorageService nach der Userld des aktuellen Benutzers.
- Buchung holen:
  - Mit der erhaltenen Userld fragt das ViewModel beim BookingRepository die Sitzplatzreservierung fOr das aktuelle Datum ab.
  - Das BookingRepository sendet Ober den ApiService einen GET-Request an den DesklyClient.
  - Die Antwort des Servers wird von ApiService und BookingRepository verarbeitet und als Liste von Buchungen (List<BookingData>) zurOckgegeben.
- Entscheidung - Buchung vorhanden?
  - Ja (resource != null):
    a) Das ViewModel ruft updateSeatName() auf dem CheckedlnScreen auf.
    b) Es speichert den Namen des Sitzplatzes im StorageService.
    c) Aur!erdem setzt es den CheckedlnStatus auf true.
  - Nein (resource == null):
    a) Das ViewModel meldet der CheckedlnScreen einen Fehler.
    b) Die CheckedlnScreen zeigt dem Benutzer eine Fehlermeldung an.
    c) Im StorageService wird der gespeicherte Ortsname entfernt.

d) Der CheckedInStatus wird auf false gesetzt.

- Notification & Completion:
  - Das ViewModel benachrichtigt den CheckedInScreen, dass der Vorgang abgeschlossen ist.
  - Die CheckedInScreen zeigt dem Benutzer die aktualisierte Schnittstelle an.

4.2.7 Delete Booking Function App-Client UML Sequenz Diagramm

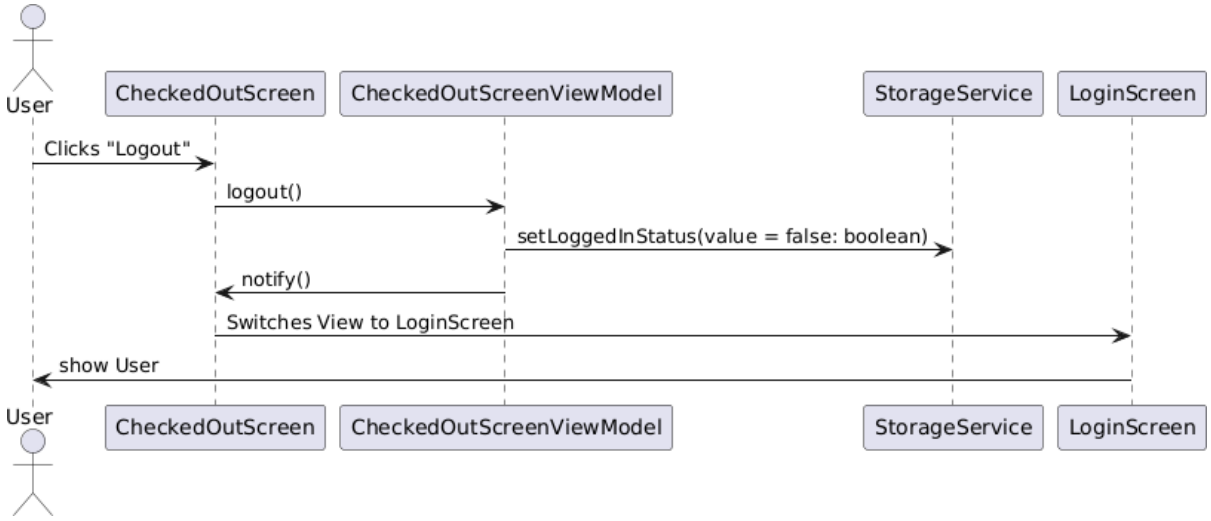


Dieses UML-Sequenzdiagramm zeigt den Ablauf eines Buchungsstornierungsprozesses in der Anwendung:

- Benutzereingabe:
  - Der Benutzer klickt auf „Cancel Booking“ im CheckedInScreen.
- Verarbeitung:
  - Der CheckedInScreen ruft die Methode cancelBooking() im CheckedInScreenViewModel auf.
  - Das ViewModel fragt den StorageService nach der User-ID.
  - Das ViewModel fragt das BookingRepository nach der aktuellen Buchung des Benutzers.
- Buchung ermitteln:
  - Das BookingRepository sendet einen GET-Request an den API-Service, um die Buchung des Benutzers zu ermitteln.
  - Die Antwort des Servers (die Buchung) wird zurOckgegeben.
- Entscheidung - Existiert eine Buchung?
  - Wenn keine Buchung gefunden wurde (BookingResponse == null):

    a) Das ViewModel zeigt eine Fehlermeldung im CheckedInScreen an.
    b) Der Benutzer sieht eine Fehlermeldung.
  - Wenn eine Buchung vorhanden ist (BookingResponse != null):
    a) Das ViewModel ruft deleteBooking() im BookingRepository auf.
    b) Eine DELETE-Anforderung wird an den API-Dienst gesendet, um die Buchung zu loschen.
    c) Der API-Service fragt den StorageService, ob der Benutzer eine Microsoft-Anmeldung verwendet.
    d) Die Antwort vom Server wird verarbeitet.
- Entscheidung - War das Loschen erfolgreich?
  - Wenn das Loschen erfolgreich war (Antwort != null):
    a) Das ViewModel wird mit true zurOckgegeben.
    b) Der CheckedInScreen wird aktualisiert und zeigt eine erfolgreiche Benachrichtigung an.
    c) Die Anwendung wechselt zum CheckedOutScreen.
    d) Der Benutzer wird Ober das erfolgreiche Auschecken informiert.
  - Wenn das Auschecken fehlgeschlagen ist (Response == null):
    a) Das ViewModel gibt false zurOck.
    b) Eine Fehlermeldung wird ausgegeben.
    c) Der Benutzer erhalt eine Fehlermeldung.

4.2.8 Logout App-Client UML Sequenz Diagramm



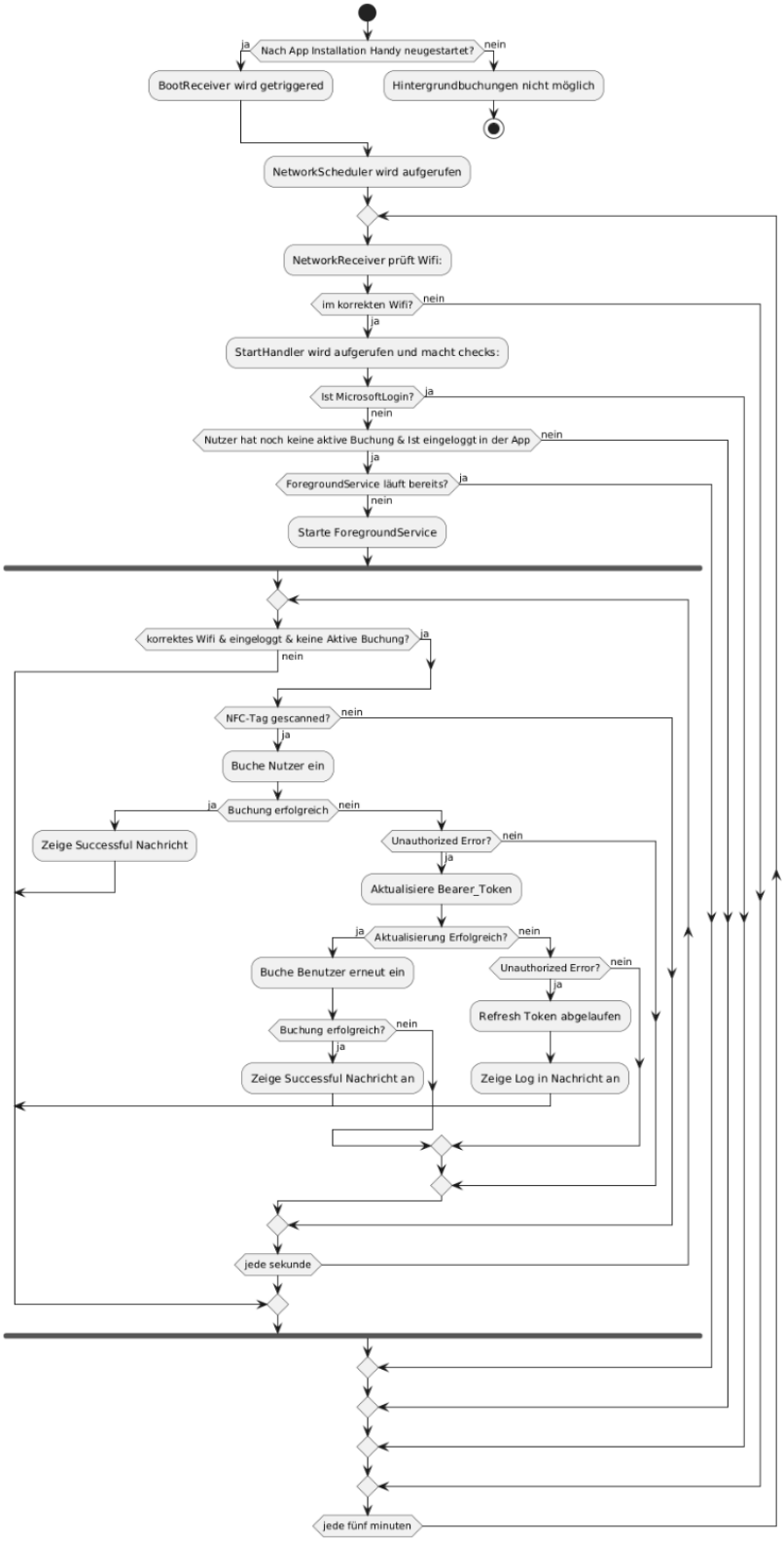
Dieses UML-Sequenzdiagramm zeigt den Ablauf eines Logout-Vorgangs in der Anwendung:

- Benutzereingabe: Der Benutzer klickt auf "Logout" im CheckedOutScreen.
- Verarbeitung:
  - Der CheckedOutScreen gibt den Befehl an das CheckedOutScreenViewModel weiter.
  - Das ViewModel setzt den Login-Status im StorageService auf false, was bedeutet, dass der Benutzer nun abgemeldet ist.
- Benachrichtigung & Ansicht andern:
  - Das ViewModel benachrichtigt das CheckedOutScreen, dass die Abmeldung abgeschlossen ist.

- Die Anwendung wechselt dann zum LoginScreen.

- Feedback an Benutzer: Der LoginScreen wird angezeigt und der Benutzer sieht wieder die Login-Oberflache.

4.2.9 Ablaufdiagramm Foreground-Service



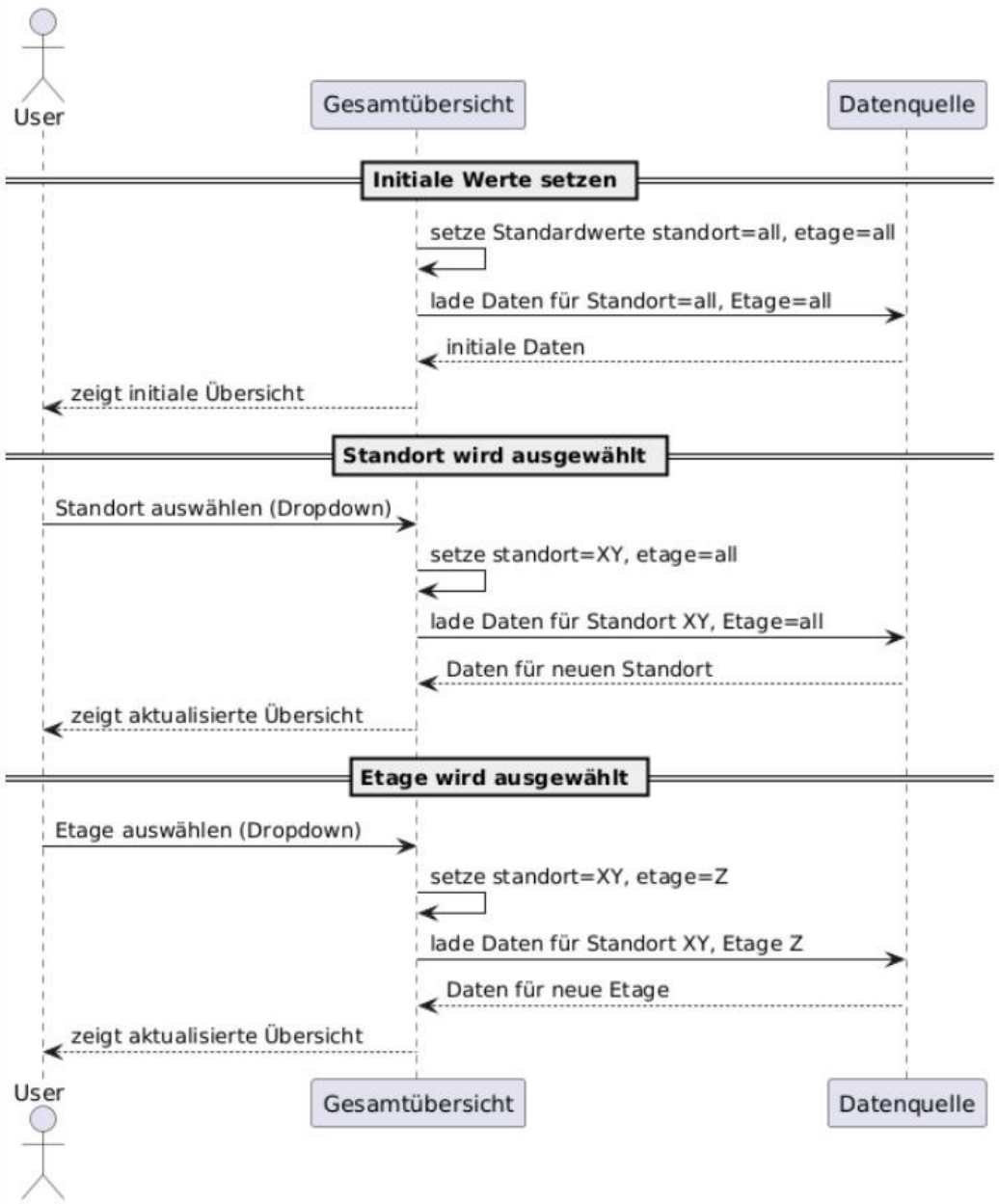
Dieser gesamte Prozess ermoglicht es einem Nutzer auf Android, sich mit sehr wenig Interaktion einbuchen zu lassen. Es startet einen Foreground-Service (Ein Hintergrunddienst, der eine Nachricht anzeigen muss, damit er von Android nicht beendet werden kann) falls notig und lasst den Nutzer mit scannen eines gOltigen NFC-Tags einbuchen. Der Foreground-Service wird gestartet falls bestimmte Anforderungen gOltig sind, wie im Diagramm zu sehen. Der Nutzer muss nach App-Installation alle Berechtigungen (Abschnitt 5.2.3) akzeptieren und lediglich einmal das Handy neu starten, damit dieser gesamte Prozess initialisiert wird.
Der Foreground-Service startet, sobald ein Nutzer sich mit einen der exxcellent Routern verbindet und bestimmte Anforderungen passen.
Dazu muss er nur sein Handy entsperren und das gOltige NFC-Tag scannen und wird erfolgreich eingebucht, ohne jedes Mal die App offnen zu mOssen.

Wichtig: Der Foreground-Service unterstOtzt kein Microsoft-Login.

Der Ablauf, lasst sich im Diagramm oben nachvollziehen.

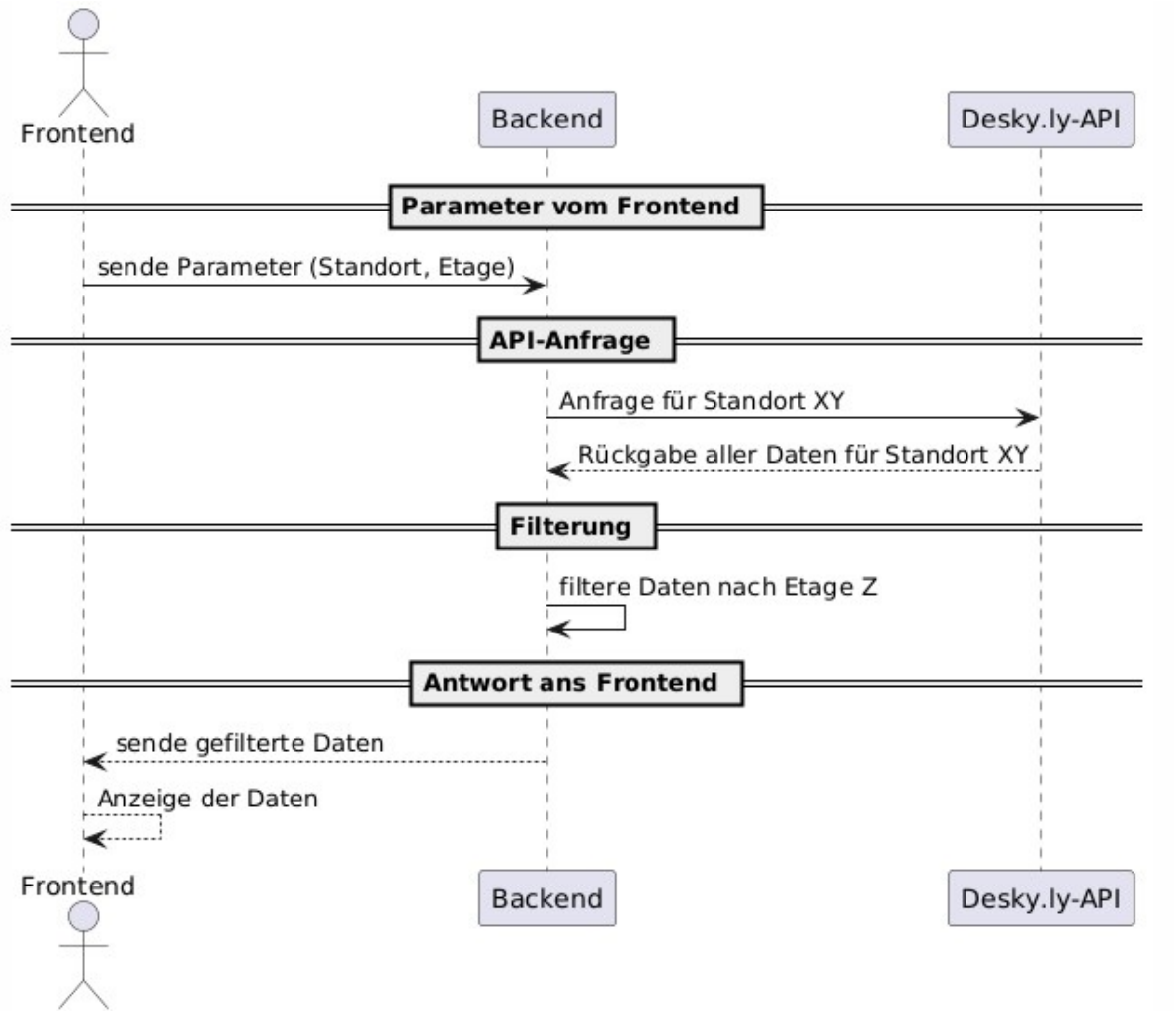
Der Code lasst sich unter:
android/app/src/main/kotlin/de/exxcellent/app/workspace
einsehen.

4.2.10 Gesamtubersicht UML Sequenz Diagramm



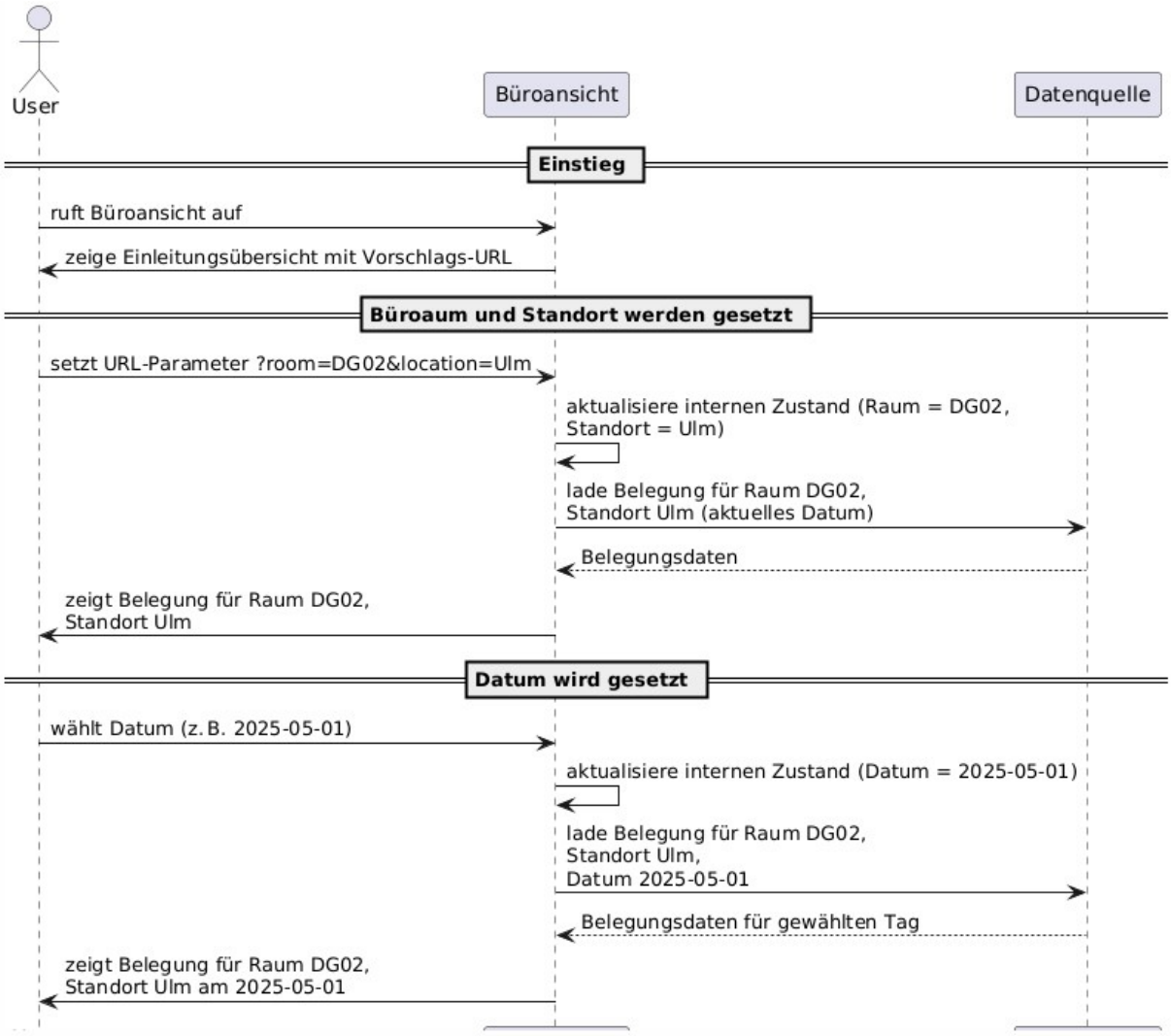
Das Sequenzdiagramm beschreibt den internen Ablauf, der stattfindet, sobald ein Nutzer die Seite ladt und im Dropdown-Menu einen Standort und eine Etage auswahlt. “GesamtObersicht” steht fOr den browserseitigen JavaScript-Code und “Datenquelle” ist stellvertretend für das Backend. Zu Beginn werden die Werte der Dropdown-Menus auf den Standardwert “all” gesetzt und es wird eine Get-Request an das Backend gestellt. Jede Anfrage enthalt die aktuellen Werte der beiden Menus als Parameter. Bei jeder Veranderung der Dropdown-Menus wird eine weitere Anfrage ausgelost. Die Daten werden anschlier!end in der GesamtObersicht dargestellt. Die Werte fOr die Etage werden dynamisch fOr den jeweiligen Standort generiert. Somit kann für die Etage nur ein Wert außer “all” vorliegen, wenn zuerst ein Standort festgelegt wurde.

4.2.11 UML Sequenz Diagramm Gesamtubersicht-Backend-Interaktion



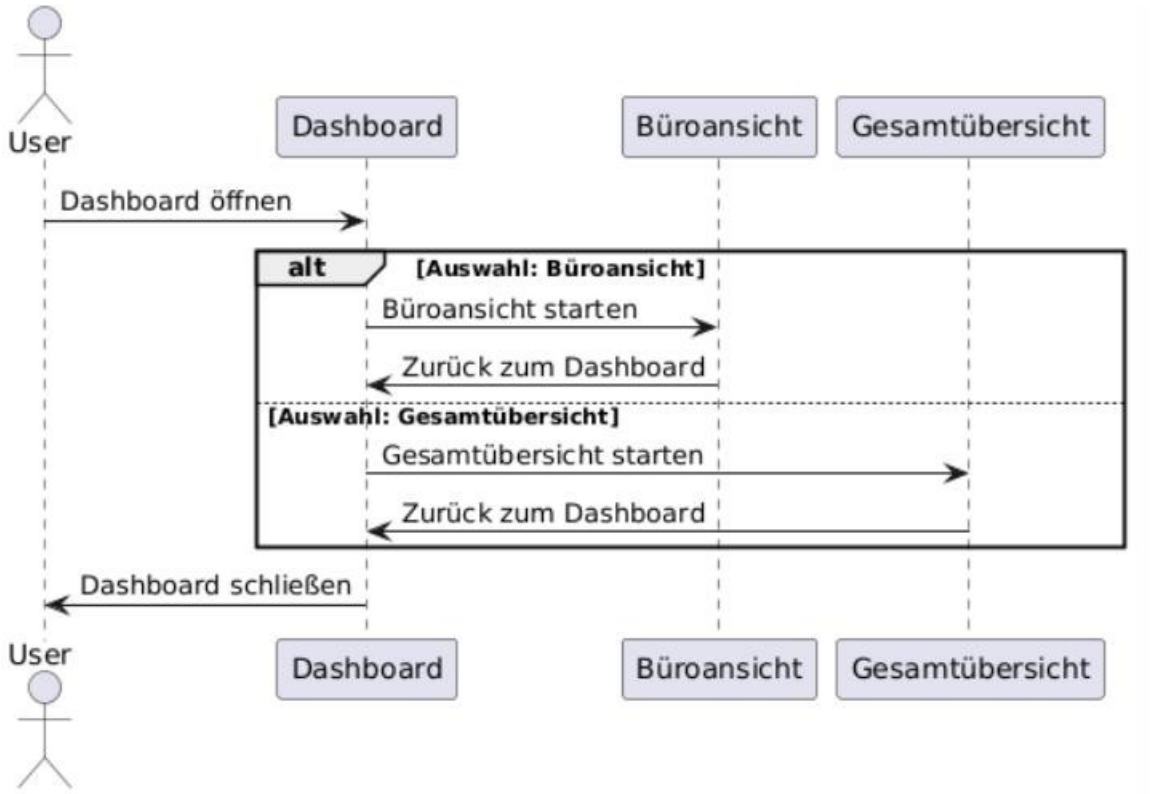
Hier wird die Interaktion mit dem Backend genauer beschrieben. Das Diagramm zeigt eine Anfrage mit zwei gesetzten Werten fOr Standort und Etage. Zuerst wird der Parameter fOr den Standort dazu verwendet, um eine Anfrage an die Desk.ly-API zu stellen. Die Antwort der API wird daraufhin so gefiltert, dass nur noch die Daten fOr die richtige Etage Obrigbleiben. Zuletzt werden die finalen Daten zurOck an den Browser gesendet, wo sie angezeigt werden. Sollte fOr einen der beiden Parameter oder beide der Standardwert “all” gesetzt sein, werden je nachdem automatisch Anfragen fOr jeden Standort gesendet, die Antwort nicht gefiltert oder beides.

4.2.12 Buroansicht UML Sequenz Diagramm



Nach Laden der BOroObersicht, wird dem Nutzer zuerst eine Obersicht angezeigt, welche beschreibt wie die Query-Parameter in der URL aussehen mOssen. Das Sequenzdiagramm zeigt was passiert, wenn die vorgeschlagenen Parameter gesetzt werden. Durch Anderung der URL wird eine Anfrage ans Backende ausgelost. Diese liefert die Daten fOr die genannten Parameter und das aktuelle Datum. Zusatzlich besteht die Moglichkeit ein anderes Datum zu wahlen. In dem Fall, dass ein anderes Datum gewahlt wird, wird bei der darauffolgenden Anfrage ein 3. Parameter mitgesendet, der als Wert das ausgewahlte Datum enthalt.

4.2.13 Dashboard UML Sequenz Diagramm



Bei Aufruf der Base-URL wird dem Nutzer das Dashboard angezeigt. Das Dashboard stellt zwei Auswahlmoglichkeiten bereit: Die GesamtObersicht und die BOroansicht. Je nachdem fOr welche Option sich der Nutzer entscheidet, wird er zur jeweiligen Ansicht weitergeleitet.

4.3 Projektorganisation

Im Abschnitt Projektorganisation wird genauer auf die Teamorganisation, den Projektablauf und das Projektmanagement eingegangen.

4.3.1 Teamorganisation

Vorgehensmodell: SCRUM

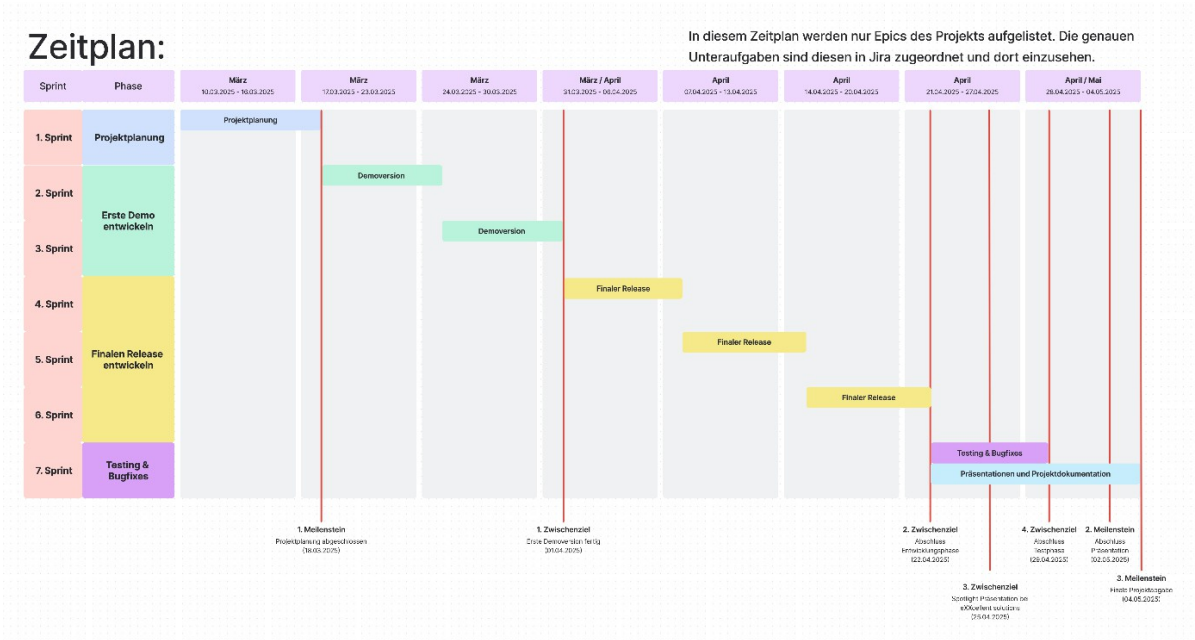
Unser Team setzte sich wie folgt zusammen:

	Name	Rolle
1	Alton Bekolli	Development-Team
2	Christian Kasper	Scrum-Master
3	Niklas KOmmel	Development-Team
4	Felix Kussmann	Development-Team
5	Lukas Jeckle	Proxy Product-Owner
6	Lovro Lupis	Development-Team

Die tatsachlichen Product-Owner des Projekts sind die Betreuer von eXXcellent solutions.

	Name	Rolle
1	Wolfram Gulde	Project-Owner
2	Wolfgang Janz	Project-Owner

4.3.2 Projektablauf



Der ursprOngliche Projekt-Zeitplan des Projekts „Automatische Arbeitsplatzübersicht“ wurde bereits vor Beginn des Projekts erstellt, und wahrend des Projekts iterationsweise OberprOft und bei Bedarf aktualisiert.

Aufgrund von Platz- und ObersichtsgrOnden wurden im Zeitplan nur die Jira Epics des Projekts aufgelistet, wobei die genauen Unteraufgaben der gelisteten Epics diesen jeweils in Jira zugeordnet und dort einzusehen waren.

Der 1. Sprint war rein der Projektplanung gewidmet. In diesem Teil des Projekts wurden wichtige Infrastrukturen und Dokumente eingerichtet und aufgebaut.

Die Sprints 2 und 3 dienten der Erstellung einer ersten Demo-Version aller fundamental wichtigen Anforderungen des Projekts, sozusagen als „Proof of Concept“ fOr das weitere Vorgehen des Projekts.

In den Sprints 4 bis 6 wurden die bis dahin erstellten Demo-Versionen unseres Mobile-App Clients und unserer Web-Anwendung zu einem vollstandigen finalen Release weiterentwickelt.

Sprint 7 und der daran angrenzende Projektabschluss-Zeitraum wurden für „Testing & Bugfixes“, sowie für das Erstellen der Prasentationen und der Projektdokumentation verwendet.

4.3.3 Projektmanagement

FOr das Aufgabenmanagement des Projekts verwendeten wir Jira. Um Dateien bzw. Unterlagen zentralisiert zuganglich zu machen kamen Microsoft SharePoint und Confluence zum Einsatz. FOr die Kommunikation und den Austausch im Team verwendeten wir Discord und fOr Sprint Reviews mit eXXcellent solutions, sowie die wochentlichen Statusbesprechungen mit Hr. Franz kam Microsoft Teams zum Einsatz.

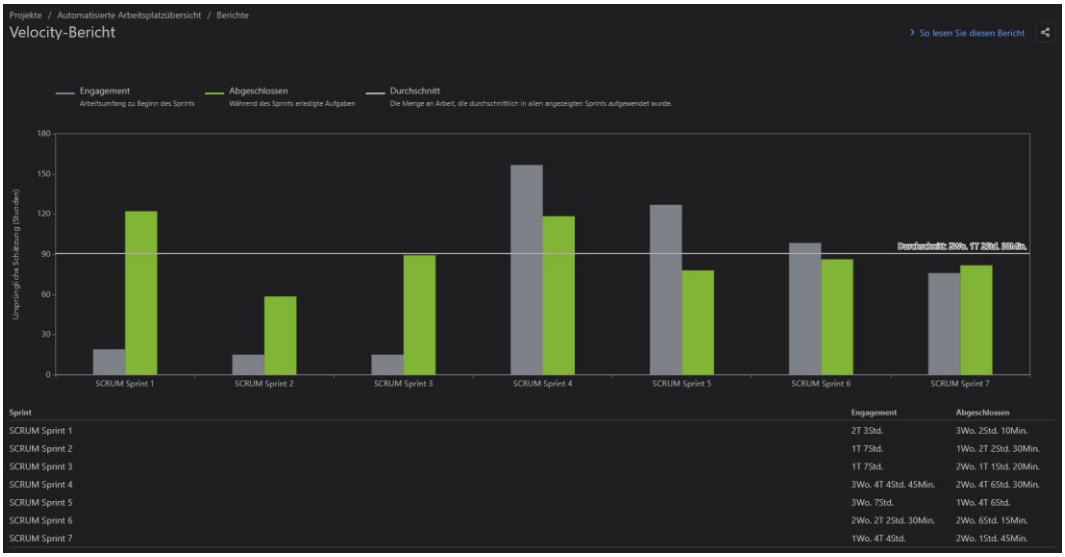
Unser Projekt war in sieben wochentliche Sprints aufgeteilt, sowie den Projektabschlusszeitraum vom 29.04.2025 bis 04.05.2025.

Aus dem Velocity Report geht hervor, dass wir durchschnittlich pro Sprint Aufgaben fOr 2 Wochen, 1 Tag und 38 Minuten abschlier!en konnten, wobei dieser Wert wahrend des ganzen Projekts, mit Ausnahme des zweiten Sprints, gut gehalten werden konnte.

Bei den Burndown Charts der Sprints gibt es immer wieder kleine Ausreir!er nach oben, welche den zeitlichen Aufwand wahrend des Sprints in die Hohe getrieben haben. Diese zusatzlichen Zeitaufwande sind auf unvorhergesehene Hindernisse bzw. Blockaden zurOckzufOhren gewesen.

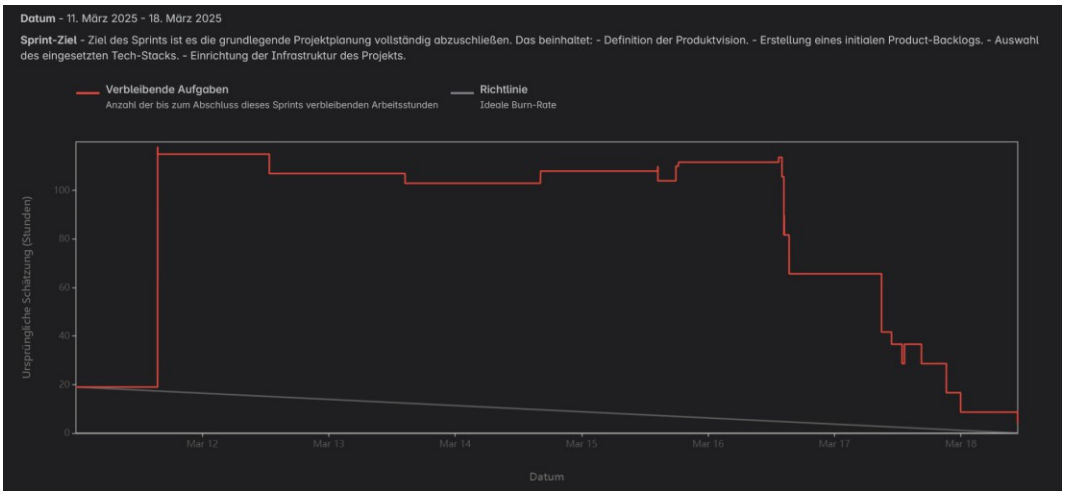
Wahrend der Ersten drei Sprints wurden die Aufgaben erst nach Beginn des Sprints in den Sprint-Backlog von Jira gezogen, aus diesem Grund sind in den Burndown Charts der Ersten drei Sprints am Anfang so gror!e Anstiege zu verzeichnen. Nach Hinweis von Hr. Balser bei der Planung im dritten Sprint Planungsmeeting wurde dies ab dem vierten Sprint berichtigt, sodass die initialen Zeitaufwande ab dem vierten Sprint richtig in den Burndown Charts angezeigt werden konnten.

Velocity Report:

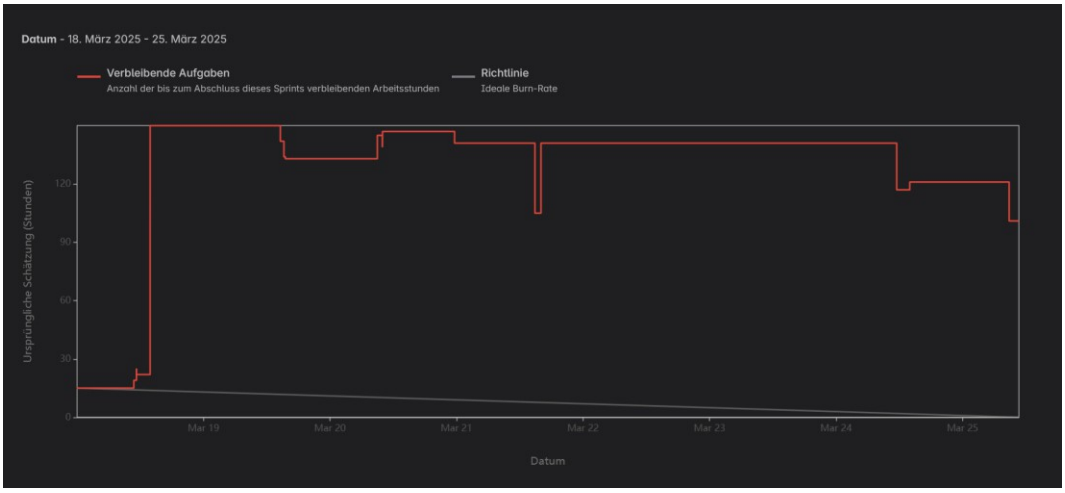


Sprint	Engagement	Abgeschlossen
SCRUM Sprint 1	2T 3Std.	3Wo. 2Std. 10Min.
SCRUM Sprint 2	1T 7Std.	1Wo. 2T 2Std. 30Min.
SCRUM Sprint 3	1T 7Std.	2Wo. 1T 1Std. 20Min.
SCRUM Sprint 4	3Wo. 4T 4Std. 45Min.	2Wo. 4T 6Std. 30Min.
SCRUM Sprint 5	3Wo. 7Std.	1Wo. 4T 6Std.
SCRUM Sprint 6	2Wo. 2T 2Std. 30Min.	2Wo. 6Std. 15Min.
SCRUM Sprint 7	1Wo. 4T 4Std.	2Wo. 1Std. 45Min.

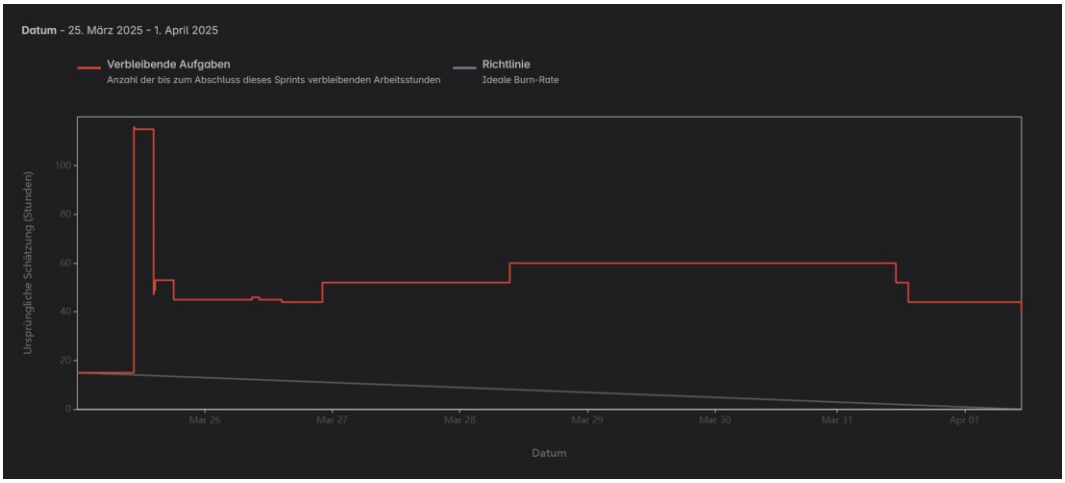
Sprint 1 Burndown-Chart:



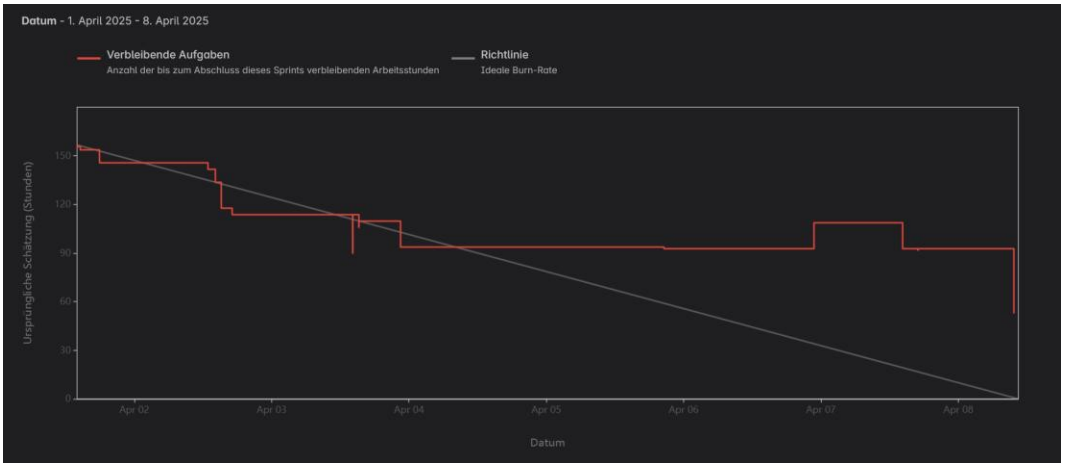
Sprint 2 Burndown-Chart:



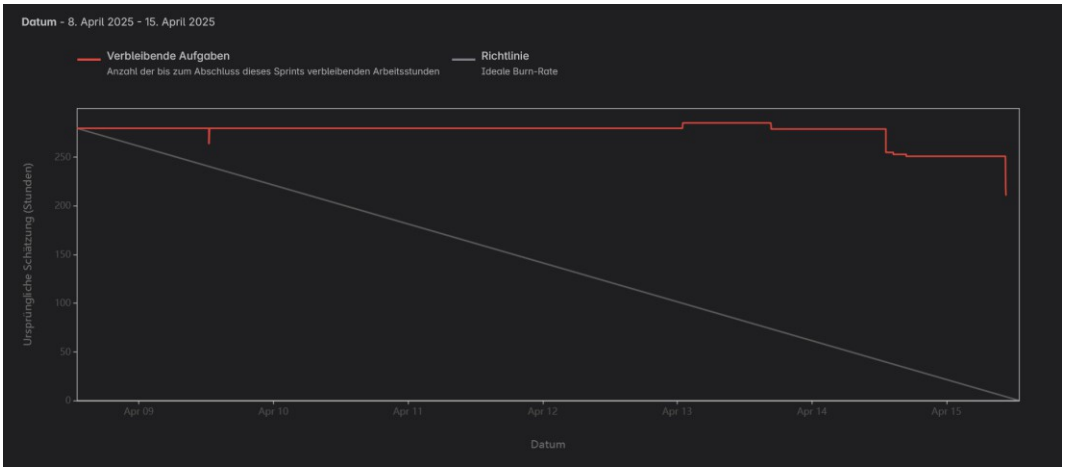
Sprint 3 Burndown-Chart:



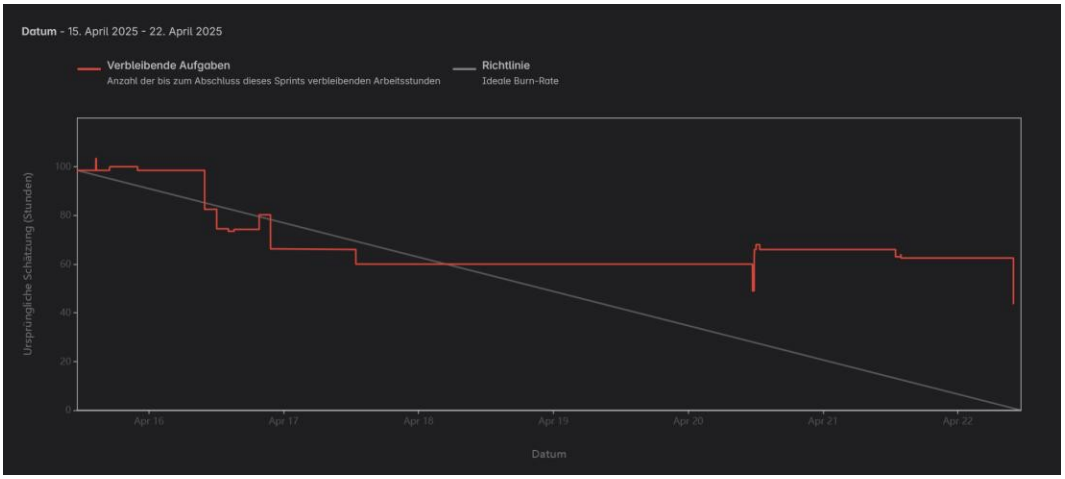
Sprint 4 Burndown-Chart:



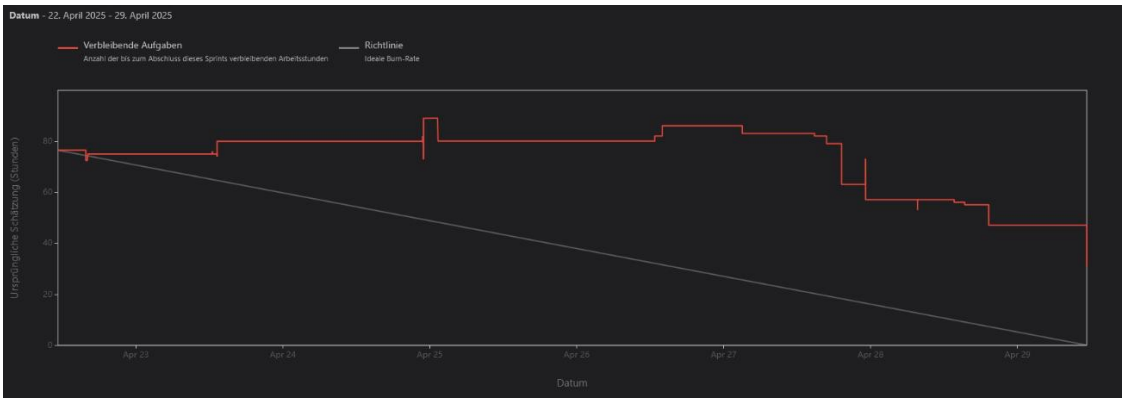
Sprint 5 Burndown-Chart:



Sprint 6 Burndown-Chart:



Sprint 7 Burndown-Chart:



4.4 Beitrage der Mitglieder am Projekt

lm nachfolgenden Abschnitt werden die einzelnen Beitrage der Teammitglieder an diesem Projekt aufgezeigt.

4.4.1 Beitrag Alton Bekolli

Im Projekt „Automatisierte Arbeitsplatzübersicht“ lag mein Hauptverantwortungsbereich im Bereich Frontend-Entwicklung. Dabei habe ich mar!geblich an der Gesamt-Obersicht, der BOroansicht sowie am Dashboard gearbeitet und viele zentrale Funktionen eigenstandig oder im Team umgesetzt. Zusatzlich habe ich bei der Backend-Anbindung fOr zukunftsgerichtete Buchungen mitgewirkt, um die Anzeige von Belegungen auch fOr zukOnftige Tage zu ermoglichen.

Gesamtubersicht und Buroansicht (Web-App)

Ein besonderer Fokus lag auf der Umsetzung der GesamtObersicht, bei der ich unter anderem folgende Aufgaben realisiert habe:

- Umsetzung des Frontends mit HTML, CSS und JavaScript fOr die GesamtObersicht / BOroansicht.
- Entwicklung der Web-App: GesamtObersicht mit visuell ansprechender Darstellung, dynamischen Filtern und einer intuitiven NutzerfOhrung
- Umsetzung der Etagen- und Standortfilter sowie deren VerknOpfung mit den entsprechenden API-Daten.
- Implementierung der Funktion zum Dynamischen Abholen von URL-Parametern fOr die BOroansicht
- Entwicklung der BOroansicht mit Echtzeitdaten, bei der die Slots automatisch befOllt werden, je nach Anwesenheit bzw. Buchung.
- UnterstOtzung bei der Fehlerbehebung und Testing dieser Komponenten.
- Verbesserung der Responsivitat, Schriftgror!en und visuellen Anpassungen.

Dashboard

Aur!erdem war ich verantwortlich fOr die Gestaltung und Umsetzung des Dashboards, das als zentrale Einstiegsseite dient. Dieses Dashboard verknOpft die Gesamt- und BOroansicht miteinander. Ich habe dabei sowohl das Layout entwickelt als auch die dynamische Verlinkung zu den jeweiligen App-Bereichen eingerichtet.

Automatisierung & zukunftige Buchungen

In Zusammenarbeit mit Niklas habe ich bei der Erweiterung der Buchungsfunktionalitat fOr zukOnftige Zeitraume mitgewirkt. Diese Funktion ermoglichte es, nicht nur die aktuelle Anwesenheit, sondern auch zukOnftige Buchungen zu visualisieren – inklusive einer Datumsauswahl (Date-Picker) in der BOroansicht.

Weitere Beitrage

Zusatzlich habe ich an folgenden Aufgaben mitgewirkt:

- Branching-Strategie definiert und dokumentiert, um eine klare Struktur und Zusammenarbeit im Git-Repository zu gewahrleisten.
- Merge- und Pull-Request-Handling festgelegt und dokumentiert, um saubere und nachvollziehbare Entwicklungsprozesse sicherzustellen.
- Projektstruktur erstellt, um die technische Basis und einheitliche Verzeichnisse fOr Frontend und backend zu schaffen.
- Dokumentationsstrategie entwickelt, um eine transparente und vollstandige Nachvollziehbarkeit der technischen Umsetzung und des Projektfortschritts zu ermoglichen.
- lmplementierung der automatischen Raum-Anzeige auf Basis der Ressourcenstruktur.
- URL-Parsing zur Identifikation von Etage und Platz fOr die korrekte Slot-Zuordnung.
- Sanfte Seitenrotation mit Fade-Effekt zur visuellen Verbesserung bei Ansicht mit mehreren Seiten.
- UnterstOtzung bei der vollstandigen Obersetzung der Kommentare ins Englische fOr internationale Nachvollziehbarkeit.

Mein Beitrag umfasste sowohl die eigenstandige Umsetzung zahlreicher Frontend-Funktionen als auch die enge Abstimmung mit dem Backend-Team (Niklas), um eine konsistente Datenanbindung und Nutzererfahrung sicherzustellen.

4.4.2 Beitrag Christian Kasper

lm Laufe des Projekts habe ich verschiedene Aufgaben Obernommen, angefangen mit der Integration der Desk.ly API in Dart. Meine ersten Schritte waren die lmplementierung der API-Funktionen und das Parsen der API-Antworten in Dart-Objekte.

Spater stellten wir fest, dass einige der benotigten Funktionen nicht von der offiziellen Desk.ly-API unterstOtzt wurden. Daher war es meine Aufgabe, diese Funktionen mit Hilfe von BurpSuite nachzubilden. Dies beinhaltete insbesondere die Abbildung von HTTP-Requests sowie die Verwaltung von Autorisierungstokens. Dazu gehorte auch die lmplementierung der korrekten Einbindung und die regelmar!ige Aktualisierung der Token, da diese nach 24 Stunden ablaufen. Durch die Nachbildung des Refresh-Prozesses mit einem „Refresh-Token“ konnte die Gültigkeit einer Session auf bis zu einen Monat verlangert werden.

Da der Zugriff auf ein gOltiges Token nur Ober ein Login moglich war, habe ich zusatzlich einen Login-Screen entwickelt. Dort wurde die nachgebildete Login-Funktion eingebaut, um die Tokens korrekt zu extrahieren und weiter zu verwenden. Spater kam noch die Anforderung hinzu, Microsoft Single Sign-On zu unterstOtzen. Auch hier konnte ich mit Hilfe von BurpSuite den gesamten Authentifizierungsprozess nachbilden und die benotigten Tokens extrahieren. Da die Microsoft Token jedoch keine direkte UUID des Benutzers enthielten, mussten weitere Workarounds entwickelt werden, damit die Anwendung den Benutzer eindeutig identifizieren konnte. Auch diesen Mechanismus habe ich erfolgreich nachgebaut und implementiert.

Neben der Backend-Integration war ich auch fOr die lmplementierung und Erweiterung der App selbst verantwortlich. Unsere App wurde dabei in die bestehende App-Sammlung der Firma exxcellent solutions integriert. Die ursprOnglichen UI-Designs

sowie wertvolle UnterstOtzung im Bereich Frontend und Microsoft-Login erhielt ich von meinem Teamkollegen Lukas Jeckle.

Zusammengefasst bestanden meine Aufgaben:

- Integration der Desk.ly-APl in das App-Backend inklusive Erstellung von Model-Klassen (z.B. Buchung, Benutzer, Ressourcen) und lmplementierung der APl-Requests gemar! der offiziellen Dokumentation.
- Abstraktion der APl-Clients in drei spezialisierte Klassen: ein Client ohne Authentifizierung, ein Client mit Bearer- oder Microsoft-Token fOr Buchungen und ein allgemeiner Client fOr regulare APl-Anfragen.
- Nachbildung undokumentierter Funktionen mit Hilfe von Burp-Suite, z.B. Login mit E-Mail/Passwort, Token-Update, Buchungsprozesse mit unterschiedlichen Authentifizierungen.
- Erstellung von Klassendiagrammen und Sequenzdiagrammen, die die geplanten Ablaufe in der Anwendung darstellen, mit anschlier!ender Anpassung an die tatsachliche lmplementierung.
- Oberarbeitung der Views von Lukas Jeckle und lmplementierung von ViewModels zur sauberen Trennung von Logik und Ul.
- Lokalisierung aller relevanten User Messages in Deutsch und Englisch.
- Integration von Firebase Push Notifications und deren Anbindung an unsere App-Views.
- Optimierung des App-Zyklus, insbesondere der Obergange zwischen den drei Haupt-Screens.
- Integration aller wichtigen APl-Funktionen wie Login, Logout, Buchen und Stornieren von Buchungen.
- Integration und Anpassung des Microsoft SSO-Logins, vorbereitet von Lukas Jeckle.
- Fehleranalyse und -behebung, z.B. ein Bug mit sich andernden Seat UUlDs, den ich durch intensives Debugging erfolgreich losen konnte.

4.4.3 Beitrag Niklas Kummel

lch habe bei der Erstellung des Webservers mitgearbeitet. Mein Schwerpunkt lag dabei auf der Backend-Entwicklung sowie auf der Sicherstellung eines funktionierenden Entwicklungs- und Testprozesses. DarOber hinaus war ich fOr die Kommunikation zwischen Frontend und Backend zustandig und habe an der Dokumentation und lnfrastruktur des Projekts mitgearbeitet.

Teststrategie & Testumgebung:

- Entwicklung der Teststrategie fOr das Backend
- Aufsetzen und Konfiguration der Testumgebung
- Erstellung und Testen einer Anleitung fOr die Installationen aller benotigten Tools

Backend-Entwicklung:
Ein zentraler Teil meiner Arbeit war die Strukturierung und lmplementierung des Backends. Dabei habe ich sowohl grundlegende Strukturen aufgebaut als auch spezifische Endpunkte zur Kommunikation mit dem Frontend entwickelt.

- Strukturierung des Backends (Services, Controller, Routes, Hilfsfunktionen)

- Erstellung mehrerer APl-Endpunkte im Backend (bspw. zur Abfrage der Daten fOr die GesamtObersicht und BOroansicht
- Kommunikation zwischen Frontend und Backend Ober Get-Requests und Generierung passender Responses
- Schnittstellen zur Desk.ly-API u.a.
    - Abfrage: Welche Mitarbeiter sind vor Ort (GesamtObersicht)
    - Abfrage: Welche Standorte gibt es

Dokumentation und Infrastruktur:
Neben der Entwicklung habe ich zur Dokumentation und Containerisierung des Projekts beigetragen.

- Erstellung eines Docker-Files fOr den Webserver
- Einbindung und Nutzung der .env-Datei mit docker-compose
- Mitarbeit an Sequenzdiagrammen zur Visualisierung der Webserver-Prozesse

4.4.4 Beitrag Felix Kussmann

In unserem Projekt “Automatische Arbeitsplatzübersicht” war ich Teil des WebApp-Teams. Meine Aufgabe bestand hauptsachlich darin, an der Entwicklung der Webanwendungen “Büroansicht” und “Gesamtübersicht” mitzuwirken. Zu Beginn habe ich Mockups fOr beide Ansichten erstellt. Dabei war es besonders wichtig, die relevanten Informationen Obersichtlich und benutzerfreundlich darzustellen. Bei der Gesamt-Obersicht lag der Fokus darauf, moglichst viele Mitarbeiter auf einer Seite abzubilden, wahrend es bei der BOroansicht darum ging, auf einen Blick zu erkennen, wie viele Arbeitsplatze ein BOro hat, ob sie belegt sind und wenn ja, von wem.

Nach der Konzeptphase ging es an die Umsetzung. Hier habe ich mit HTML, CSS und JavaScript gearbeitet. Besonders bei der Gestaltung der Gesamtansicht habe ich eng mit meinem Teamkollegen Alton Bekolli zusammengearbeitet. Gemeinsam standen wir vor einigen Herausforderungen:

- Dynamische Anpassung der Ansichten an unterschiedliche Bildschirmauflosungen
- Farbgestaltung in Anlehnung an das Design der Firma eXXcellent solutions
- Einbindung von Daten, wie Datum, Uhrzeit, Mitarbeiteranzahl usw.
- Einbindung von Funktionen, wie Dropdown-MenOs und Date-Picker
- Erhaltung der Obersichtlichkeit und Schaffung eines stimmigen Gesamtbilds

lm Anschluss an die Umsetzung der Web-App war ich aur!erdem an der Erstellung der Web-App Wiki mar!geblich beteiligt. Ziel war es, die Inhalte moglichst einfach und verstandlich darzustellen, damit sich alle Mitarbeitenden schnell und problemlos in der Anwendung zurechtfinden konnen.

Zum Abschluss habe ich noch an unserem Projektplakat mitgewirkt. Hier ging es darum, unser gesamtes Projekt Obersichtlich und ansprechend auf einer DIN A1-Seite zu prasentieren, ebenfalls mit dem Anspruch, die wichtigsten Punkte klar und einfach verstandlich zusammenzufassen.

4.4.5 Beitrag Lukas Jeckle

Vorarbeit vor Projektbeginn

Die Arbeit am Projekt begann fOr mich bereits ungefahr zwei Wochen vor Projektbeginn. Zu diesem Zeitpunkt hatte ich mir als Proxy Product Owner bzw. Projekt Manager von Team B bereits Gedanken Ober den Ablauf und die Umsetzung des Projekts gemacht. Hierbei habe ich zunachst begonnen die Infrastruktur des Teams aufzubauen, darunter fiel das Einrichten von Jira, Confluence und unserem Team Discord. Des Weiteren habe ich mich bereits vor Beginn des Projekts mit dem Thema Projektmanagement befasst, und erste fOr den Meilenstein „Projektplanung“ wichtige Tasks in Jira im Backlog hinterlegt. Ferner habe ich vorab bereits eine Erste Version unseres Projekt Zeitplans erstellt. Zu guter Letzt habe ich bereits vor dem Projekt ein Treffen von Team B arrangiert. Zweck dieses Treffens war es direkt vorab konkrete Umsetzungsideen und Umsetzungsmoglichkeiten fOr unser Projektvorhaben festzuhalten und diese gesammelt mit unseren noch offenen Fragen als PDF bei der Ersten Kontaktaufnahme an die Betreuer von eXXcellent solutions weiterzugeben. Durch diese Vorarbeit war uns ein deutlicher „Head Start“ ins Projekt möglich.

Wahrend des Projekts

Während des Projekts übernahm ich größtenteils „Overhead“ Aufgaben.
Darunter fielen Aufgaben wie, das Abstimmen von Terminen, die Projekt- und Aufgabenplanung, die Team-Koordinierung, die Pflege der Produkt- und Sprint-Backlogs, das Abklaren von Zugriffen auf Inhalte, das Anfragen von Benutzer Accounts, das Erstellen der wochentlichen Statusberichte fOr Herrn Franz, sowie viele weitere unscheinbare Aufgaben, um den Teamkollegen bestmoglich den ROcken fOr eine effiziente Entwicklung freizuhalten und einen effizienten Projektablauf zu gewahrleisten.

Neben all diesen „Overhead“ Aufgaben habe ich mich auch aktiv in die Entwicklung des Mobile-App Clients eingebracht. Hierbei Obernahm ich die Implementierung der ersten Versionen unserer UI-Screens, die Vervollstandigung und Erweiterung des bereits vorhandenen iOS Build-Prozesses, sowie die Anpassung des Firebase Authentifizierungsprozesses, um dort den Microsoft Access-Token bei einem Microsoft Login abzuspeichern und diesen in unseren Authentifizierungs-Workaround abholen zu konnen.

4.4.6 Beitrag Lovro Lupis

In unserem Projekt "Automatisierte ArbeitsplatzObersicht" lagen meine Aufgaben in der Mobile-App. Ich habe mich am Anfang um die NFC-Tags gekOmmert, wie die Definition der Daten auf den NFC-Tags, sowie einen NFC_Flutter_Service implementiert, der zur Auslesung des Textes auf den NFC-Tags dient und diesen dann in das Buchungssystem integriert. Damit konnte man dann mit geoffneter App einen NFC-Tag auslesen und sich einbuchen lassen.

Meine Hauptaufgabe lag allerdings dabei, die Nutzung der App maximal interaktionslos zu ermoglichen. Durch lange Recherche habe ich festgestellt, dass dieser Prozess nur auf Android moglich sein wird, da iOS laufende Hintergrunddienste stark einschrankt. Ich habe dies sogenannt als Bonus fOr Android Nutzer gesehen, indem diese nicht einmal die App offnen mOssen, um sich einbuchen zu konnen. Schritt fOr

Schritt und mit vielen Problemen, habe ich dann eine vollstandig funktionsfahige Software auf Kotlin implementiert, in dem der Nutzer sich ganz einfach und interaktionslos einbuchen kann. Zu meinen Aufgaben gehorten:

- Einen NFC_Flutter_Service implementiert, der sich um das Auslesen der Daten auf einem NFC-Tag kOmmert, solange die App offen ist, dies ist auf Android und iOS unterstOtzt.
- Einen Kotlin_Boot_Receiver implementiert, der bei Handy Reboot einen Broadcast von Android empfangt und dadurch einen Kotlin_Alarm_Manager initialisiert.
- Einen Kotlin_Network_Receiver implementiert, der jede 5 Minuten durch einen Kotlin_Alarm_Manager aufgerufen wird, um die derzeitige Wifi-SSID auslesen zu konnen.
- Einen Kotlin_Booking_Handler implementiert, der sich um Netzwerkoperationen kOmmert, wie das Einbuchen eines Nutzers und die Abfrage, ob ein Nutzer bereits eingebucht ist. Aur!erdem wird in dieser Klasse auch ein Bearer_Token automatisch mithilfe eines Refresh_Tokens aktualisiert.
- Einen Kotlin_Start_Handler implementiert, der sich um die Abfrage kOmmert, ob ein Kotlin_Foreground_Service gestartet werden soll.
- Einen Kotlin_Nfc_Helper implementiert, der automatisch aufgerufen wird, wenn ein NFC-NDEF-Tag gelesen wird. Diese Klasse wird selbst aufgerufen, wenn die App nicht geoffnet ist.
- Einen Kotlin_Foreground_Service implementiert, der gestartet wird, wenn die Wifi-SSID "exxcellent" im Namen hat und Kotlin_Start_Handler eine erfolgreiche OberprOfung liefert. Dieser Kotlin_Foreground_Service, wartet bis ein NFC-Tag gelesen wird, und dadurch dann mit bestehenden Daten, wie den Bearer_Token etc. den Nutzer versucht einzubuchen.
- Es werden auch im Hintergrund Nachrichten angezeigt, wie zum Beispiel: Buchung ist erfolgreich, Sitzung abgelaufen, bitte App offnen und einloggen.

Code:

android/app/src/main/kotlin/de/exxcellent/app/workspace

lib/workspace/services/nfc_service.dart

Herausforderungen:

- NFC-Tag lesen und bearbeiten, ohne dass die App geoffnet ist.
- Integration von Kotlin-Code in ein bestehendes Flutter-Projekt.
- Da dieser gesamte Hintergrund-Buchungsprozess ohne geoffnete App laufen soll, ist es nicht moglich unseren Flutter-Buchungs-Code aufzurufen, dadurch musste ich die Buchung auch noch separat auf Kotlin schreiben.
- Im Hintergrund arbeiten und gegen die gesamten Hintergrundrestriktionen (zum Beispiel Zeitlimits, Berechtigungen, Energieverwaltung) der Betriebssysteme ankampfen.

Erkenntnisse:

- Technischer Aufbau und Einsatz von NFC-Tags (NDEF-Format)

- Kotlin-Entwicklung und Integration nativer Android-Komponenten in Flutter.
- Einschrankungen und Besonderheiten bei Hintergrundprozessen unter Android und iOS.

Zusatzlich war ich im Verlauf des Projekts jederzeit ansprechbar und habe UnterstOtzung angeboten, sowohl bei der Organisation als auch bei technischen Aspekten. Ich habe stetig beim Erstellen der Dokumentation mitgearbeitet und bin gezielt auf die Anforderungen sowie das Feedback der Betreuer eingegangen, um ein benutzerfreundliches Ergebnis zu schaffen.

5. Ergebnis

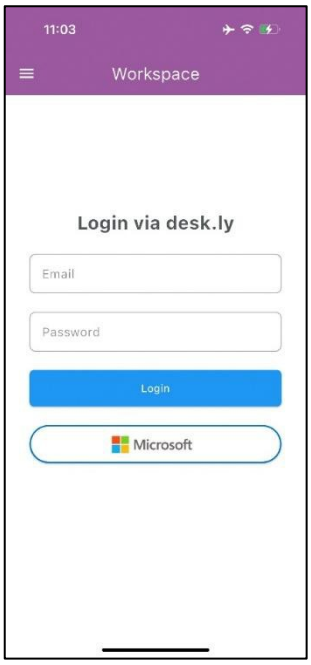
In diesem Abschnitt soll dem Leser ein Einblick in das Ergebnis unserer Projektarbeit gegeben werden. Um dies zu erreichen, wird zunachst eine Tour aus Nutzersicht gegeben, in der auf die verschiedenen UI-Komponenten der im Projekt erstellten Implementationen eingegangen wird. Im Anschluss daran folgt im Abschnitt Benutzerdokumentation eine Sammlung der Benutzeranleitungen, die vom Projekt-Team wahrend des Projekts fOr den Kunden eXXcellent solutions erstellt wurden. Da manche unserer Funktionen nicht direkt in der Desk.ly API vorhanden waren, mussten diese per HTTP-Requests nachgebaut werden, dabei wurden fOr bestimmte Funktionen andere Base-URLs verwendet:

Name	Inhalt	Beschreibung
_BASE_URL	https://app.desk.ly/en/api/v2	Offizielle Desk.ly REST-API Schnittstelle
_BOOKING_URL	https://exxcellent.desk.ly/en/api	URL mit Microsoft oder normale Login Authentifizierung
_LOGIN_URL	https://app.desk.ly	URL ohne Authentifizierung

5.1 Tour aus Nutzersicht

Im nachfolgenden Abschnitt wird auf die UI-Komponenten des App-Clients und der Web-Anwendung eingegangen, sowie deren Use-Case Szenarien erlautert, um dem Leser einen Einblick in die wahrend des Projekts erstellten Software-Implementationen zu bieten.

5.1.1 App-Client - Login Screen



Im Login Bereich des App Clients konnen Nutzer zwischen zwei Authentifizierungsmethoden wahlen. Hierbei ist wichtig anzumerken, dass es sich bei beiden Login

Varianten um „Workarounds“ handelt, welche nicht mit der offiziellen API von desk.ly interagieren. Das hangt damit zusammen, dass die offizielle API-Schnittstelle von desk.ly keine Methoden zur Nutzerauthentifizierung bietet. FOr die Umsetzung der Funktionalitat der Authentifizierungen wurden die HTTP-Requests zunachst mittels der Pentesting Software Burp-Suite ausgelesen und anschlier!end im Code rekonstruiert.

E-Mail/Passwort Authentifizierung:

Dabei werden die Anmeldedaten an einen bestimmten Endpunkt des Dienstes gesendet. In diesem Fall war es der Endpunkt “/de/api/authorize/accessToken” von Desk.ly.

Nach erfolgreichem Login erhalten wir ein Access-Token und ein Refresh-Token. FOr einen genaueren Ablauf kann man dies im Kapitel 4.2.3 nachvollziehen.

Microsoft Single Sign-On Authentifizierung:

Um den Microsoft Login des App Clients zu ermoglichen, wurde der bereits bestehende Firebase Authentifizierungsprozess, welcher ursprOnglich von eXXcellent solutions implementiert wurde erweitert.

HierfOr wird wahrend des Firebase Authentifizierungsprozesses der Microsoft Access-Token abgefangen und im Hintergrund gespeichert. Dieser Prozess ermoglicht es, einen fOr eine Stunde gOltigen Access-Token zu erhalten.

Dieses Microsoft-Access-Token wird dann an Desk.ly an den Endpunkt “/information” gesendet, um die lnformationen Ober den Desk.ly-Benutzer abzurufen. Darunter zahlt die UUlD des Nutzers die innerhalb von Desk.ly einzigartig ist. Diese lnformationen werden spater zur ldentifizierung innerhalb der Anwendung benotigt, um Buchungen zu erstellen oder zu loschen.

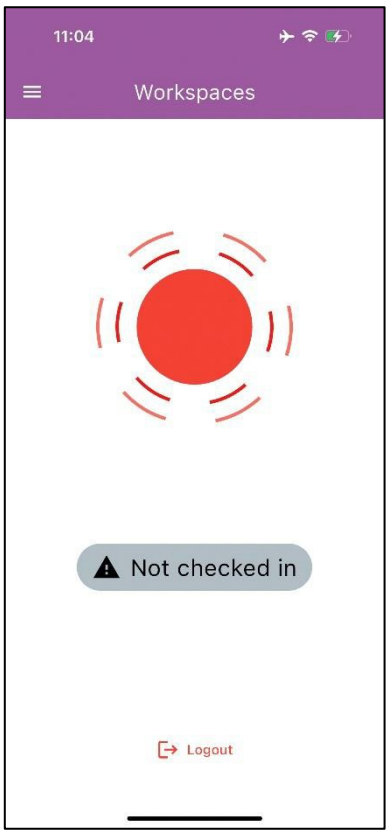
Aufgrund dessen, dass bei der Firebase Authentifizierung kein Refresh-Token mitgeliefert wird, ist es nicht moglich diesen wie bei der E-Mail/Passwort Authentifizierung nachdem ablaufen automatisch zu erneuern.

Das resultiert darin, dass sich Nutzer nach einer Stunde erneut in der Workspace Sektion der eXXcellent App authentifizieren mOssten.

Um diesen Prozess so angenehm wie moglich zu gestalten, ist es empfehlenswert, die Microsoft Zugangsdaten im App-Client zu speichern, damit eine „One-Click“ Authentifizierung durchgefOhrt werden kann.

Auf weiteres Verbesserungspotenzial dieses Ansatzes wird in Kapitel 7 eingegangen.

5.1.2 App-Client - Checked-Out Screen



Im Checked-Out Screen hat der Nutzer die Moglichkeit einen validen NFC-Tag zu scannen, um an einem Arbeitsplatz eingebucht zu werden, oder aber auch sich auszuloggen. Um dem Nutzer deutlich zu signalisieren, dass er aktuell an keinem Arbeitsplatz eingebucht ist, wird in diesem Bildschirm ein rotes Hauptelement verwendet, sowie klar der Text „Not checked in“ angezeigt, wobei dieser Text Ober App-Locale gesteuert wird und damit auch eine deutsche Obersetzung angezeigt werden kann.

Validen NFC-Tag scannen:

Wie die NFC-Tags konfiguriert werden, wird in Abschnitt 5.2.2 anschaulich erklart.

Um eine Buchung durchzufOhren, muss am Gerat die NFC-Funktion aktiviert sein. Danach das Smartphone mit der App im Vordergrund an den NFC-Tag halten und die Buchung wird automatisch durchgefOhrt.

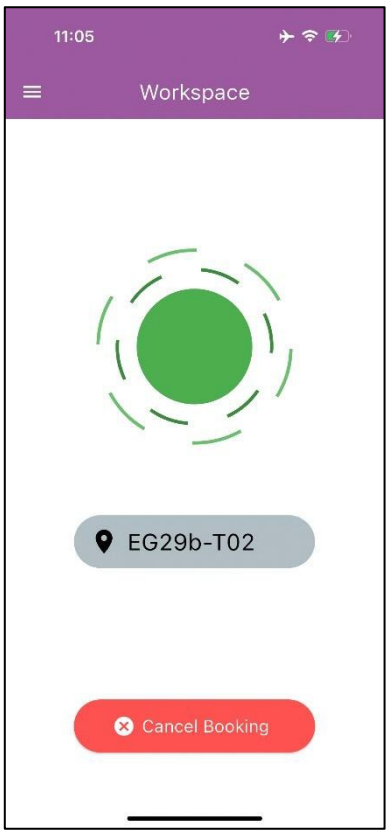
FOr Android Benutzer gibt es zusatzlich noch eine Hintergrundbuchung. Auf deren Umsetzung wurde bereits in Abschnitt 4.2.9 genauer eingegangen.

Nach der erfolgreichen Buchung wird der Nutzer in den Checked-In Screen weitergeleitet.

Ausloggen:

Dieser Ablauf der technischen Implementation wurde bereits technisch genauer in Abschnitt 4.2.8 erlautert.

5.1.3 App-Client - Checked-In Screen



Im Checked-In Screen wird dem Nutzer seine aktuelle Buchung visuell klar und Übersichtlich angezeigt. Hierzu wird ein grOnes Hauptelement verwendet, sowie dem Nutzer deutlich angezeigt, an welchem Arbeitsplatz er sich eingebucht hat.

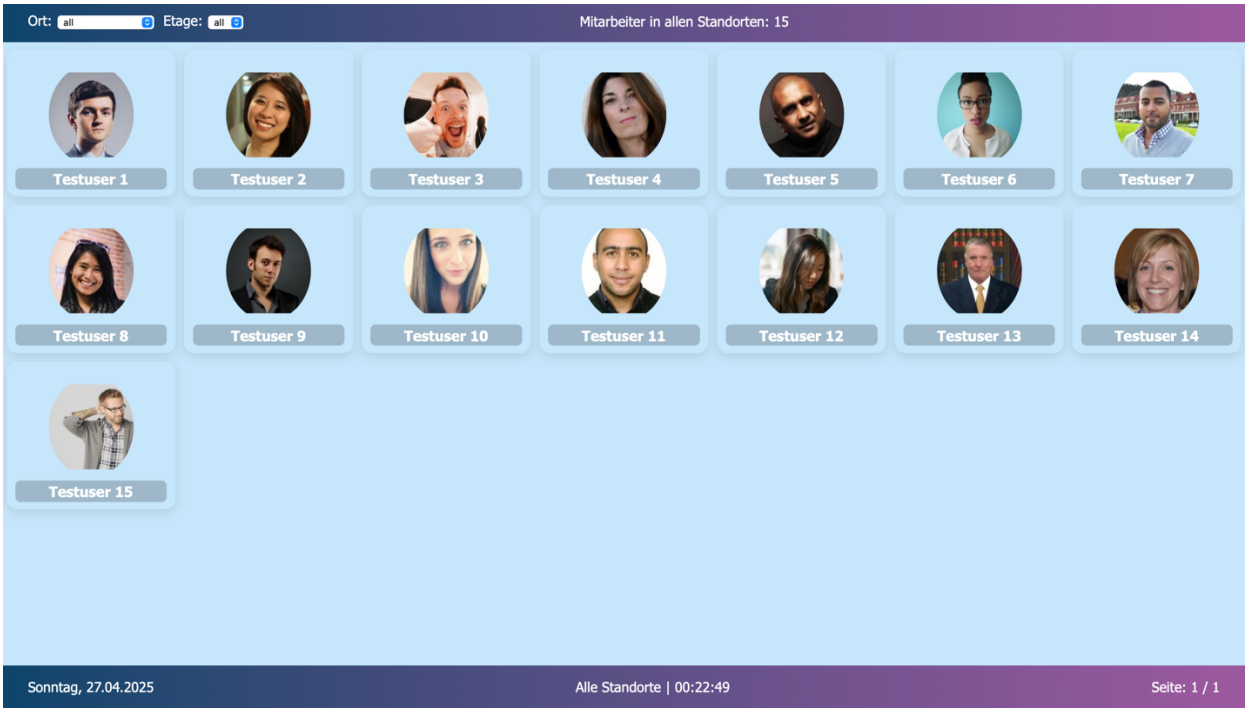
Buchung stornieren:

Sollte dem Nutzer auffallen, dass er sich versehentlich am falschen Arbeitsplatz eingebucht hat, kann er die aktuelle Buchung über den „Cancel Booking“ ganz einfach und unkompliziert stornieren.

Nach der Stornierung wird der Nutzer wieder in den Checked-Out Screen zurOckgeleitet.

Der Ablauf der technischen Implementation des Stornierungsprozesses wurde bereits genauer in Abschnitt 4.2.7 beleuchtet.

5.1.4 Web-App - Gesamtubersicht

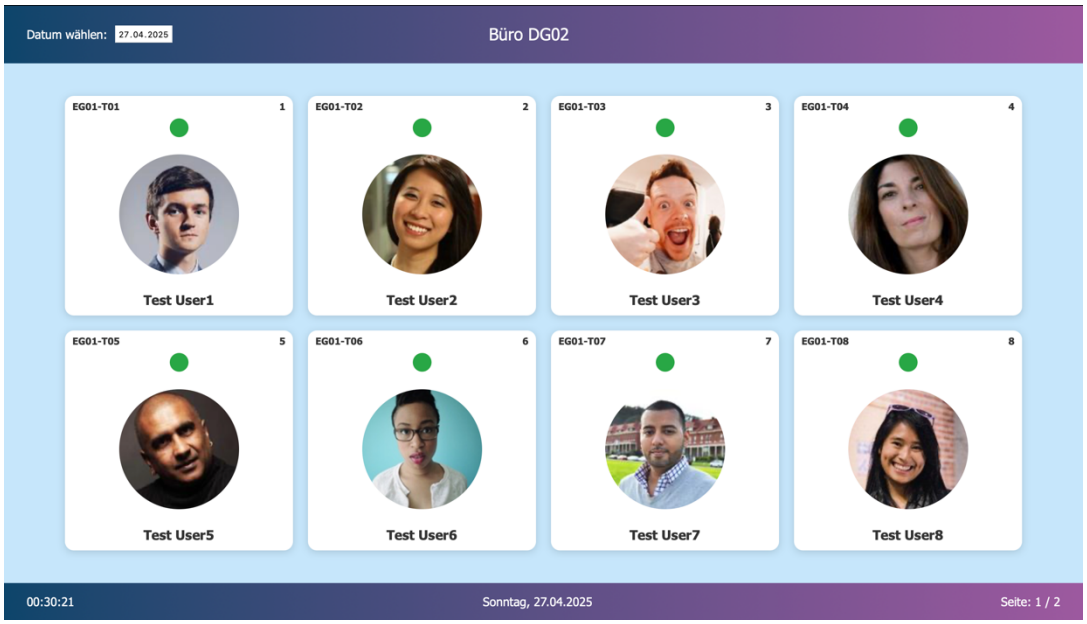


Im Rahmen des Projekts wurde eine GesamtObersicht entwickelt, in der alle derzeit aktiven Mitarbeiter angezeigt werden. Die Darstellung erfolgt abhangig von den gewahlten Parametern Standort und Etage. Diese Auswahl kann entweder Ober zwei Dropdown-MenOs in der linken oberen Ecke der Ansicht oder alternativ durch Obergabe entsprechender Parameter in der URL erfolgen. Die Dropdown-MenOs sind dynamisch and die Standort- und Etagenstruktur von desk.ly angebunden. Neue Standorte oder Etagen, die innerhalb von desk.ly hinzugefOgt werden, stehen somit automatisch in den Auswahloptionen zur VerfOgung, ohne dass zusatzliche manuelle Anpassungen notwendig sind.

Zur Sicherstellung aktueller Daten erfolgt eine automatische Aktualisierung der Obersicht. In konfigurierbaren Zeitintervallen sendet die Anwendung APl-Anfragen an die desk.ly-APl, um den Datenbestand regelmar!ig zu synchronisieren. Eine detaillierte Beschreibung des zugrunde liegenden Prozesses findet sich in Abschnitt 4.2.9, inklusive eines unterstOtzenden Sequenzdiagramms.

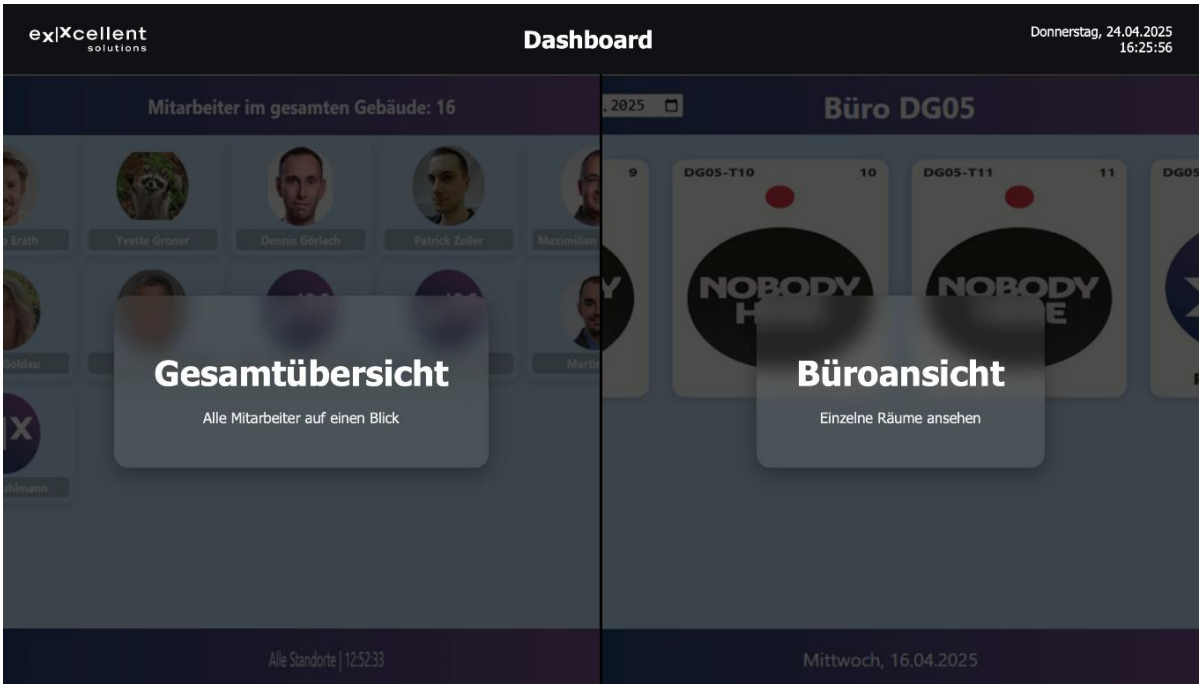
5.1.5 Web-App – Buroansicht

5.1.6



Die BOroObersicht orientiert sich, ebenso wie die Gesamtansicht, an den Farben der Firma eXXcellent solutions. In dieser Ansicht wird, nach der Bestimmung eines BOros Ober die URL, ein einzelnes BOro dargestellt, einschlier!lich aller verfOgbaren Arbeitsplatze. Man erkennt auf einen Blick, welche Platze belegt sind und, falls zutreffend, von wem. Ober die Datumsfunktion oben links lasst sich ein beliebiger Tag auswahlen, wobei ein Zeitraum von bis zu 30 Tagen in die Vergangenheit oder Zukunft abgedeckt wird. Die Aktualisierung der Daten erfolgt taglich um 0:00 Uhr. Der genaue Ablauf dieses Prozesses ist in Abschnitt 4.2.10 anhand eines Sequenzdiagrammes detailliert beschrieben.

5.1.7 Web-App - Dashboard



Zur UnterstOtzung und Vereinfachung der Bedienung wurde zusatzlich ein Dashboard implementiert. Es dient als zentrale Einstiegsmoglichkeit zur Konfiguration und Auswahl der beiden Hauptansichten: GesamtObersicht und BOroansicht. Bei der Auswahl der Gesamtübersicht wird diese mit den Standardwerten “all“ für Standort und Etage geladen. In diesem Fall werden samtliche Mitarbeitenden an allen Standorten angezeigt. Entscheidet man sich fOr die BOroansicht, gelangt man zunachst auf eine Obergangsseite. Dort wird eine vordefinierte URL vorgeschlagen, die individuell angepasst werden kann. Abhangig von der gewahlten URL wird anschlier!end das entsprechende BOro dargestellt. Das Sequenzdiagramm fOr das Dashboard findet man mit Beschreibung im Abschnitt 4.2.11.

5.2 Benutzerdokumentation

In diesem Abschnitt wird auf die von unserem Projekt Team erstellten Dokumentationsunterlagen und Anleitungen fOr unseren Kunden eXXcellent solutions eingegangen.

Die bereits vorhandenen Anleitungen der Mobilanwendung, welche nicht auch urspr0nglich von unserem Team verfasst wurden, werden hierbei aur!envorgelassen.

5.2.1 App-Client - iOS Setup

SetupiOS

Last edited by Lukas Jeckle 4 days ago

SetupiOS

Allgemeine Informationen

Bundle-ID: de.exxcellent.xxapp
Flutter Version: 3.22.3
Android Studio Version: Android Studio Iguana | 2023.2.1 Patch 2

Entwicklungsumfeld einrichten

Flutter

Flutter-SOK der im Projekt verwendeten Flutter Version herunterladen.

Flutter - Version: 3.22.3

Download Link: MacOS (AP.. Rle Silicon)
Download Link: MacOS IApRle Intel)

Als ni:ichstes das Terminal Offnen und folgende Befehle ausfuhren:

```
# In das Verzeichnis in dem die Flutter Dateien liegen navigieren.
cd Downloads/

# Als niichstes ein Verzeichnis fiir die Flutter Dateien anlegen.
mkdir •/Flutter/

If Danach die Flutter Dateien in den neu erstellen Ordner extrahieren.
unzip flutter_macos_arm64_3.22.3-stable.zip -d "'/Flutter
```

Wenn alles richtig ausgefiihrt wurde mUsste die Flutter SOK nun entpackt in ihrem eigenen Ordner liegen.

Flutter zu PATH hinzufugen

Terminal Offnen und folgende Befehle ausfuhren:

```
# Herauszufinden, welche Shell MacDS verwendet:
echo $SHELL

# Neuere /1acDS Versionen verwendet die zsh Snell als default.
# Annahme: Die RUckgabe ist '/bin/zsn'.

# Folgenden 8efehl ausfUhren, um die Umgebungsvariablen zu bearbeiten:
nano ,.,,/.zshenv

# Folgende Umgebungsvariablen in die '. zshenv' Datei einfUgen:
export PATH:"$PATH:$HOME/Flutter/flutter/bin"
export FLUTTER_RODT= "$HOME/Flutter/ fl.utter"


II Nun alle Terminal Instancen einmal schliejlen und eine neue Terminal Instance starten.
# Erste Ausfiihrung kann bis zu 1 Minute dauern. Danach son te eine entsprechende /Wckgabe erscheinen.
flutter --version
```

Flutter Einrichtung abschlieBen

```
II Flutter Doctor Befehl ausfUnren:
flutter doctor
```

Aile noch offenen Punkte der Doctor-Summary nach und nach abarbeiten.

Link: Installation und Einrichtung CocoaPods

XCode

Fiir die Installation und Einrichtung von XCode folgende Anleitungen befolgen:

- XCode Installieren
- XCode iOS Emulator einrichten

Android Studio

> **Important**
> **Android Studio benOtigt fur die iOS-Entwicklung XCode um Builds zu erstellen und ggf. zu signieren.**
>
> Hierbei ist wichtig, dass auch in XCode das hinterlegte Build-Target (Gerat auf das ein Build "gebaut" wird) uberpruft und ggf. angepasst bei Fehlermeldungen in Android Studio angepasst wird.

Installation

Android Studio - Version: 'Android Studio Iguana | 2023.2.1 Patch 2'

- Link: MacOS (Apple Silicon)
- Link: MacOS (AJI. le Intel)

Nach abgeschlossener Installation:

- **Flutter und Dart Plugins in Android Studio installieren.**
- **In den Einstellungen unter "Languages & Frameworks" > "Dart" den Dart-SOK P1ad angeben und fur die eXXcellent App aktivieren.**
  - Beispiel: /Users/< Name>/Flutter/flutter/bin/cache/dart-sdk
- In den Einstellungen unter "Languages & Frameworks" > "Android SOK" > "SOK Tools"
  - o "Android SOK Command-line Tools (latest)" hinzufugen.

Projekt aufsetzen

Das eXXcellent App Projekt uber Gitlab klonen und in Android Studio einbinden. Danach im Android-Studio Terminal folgendes ausfuhren:

```
# Flutter dependencies herunterladen.
flutter pub get

# Localizations erstellen.
flutter gen-llOn

// mocks.mocks.dart Datei erstellen.
flutter pub run build_runner build


# In das ios directory wechseln.
cd ios/

// Pods installieren.
pod install

# Pods aktualisieren.
pod update
```

Physikalischen iOS Release-Build erstellen

Einrichtung:

Provisioning-Profile fur Bundle-ID der eXXcellent App, sowie Apple-Developer Account fur das eXXcellent Entwicklungsteam bei Lisa Lamparter anfragen.

Fur die Einrichtung von XCode fur physikalische Builds folgende Anleitung befolgen:

- XCode gl} sikalisches Zielgerat einrichten

-> **Sollte der Link nicht den richtigen Inhalt anzeigen, dann in der URL-Leiste noch einmal mit Enter bestatigen um die Seite neu zu laden und zum** richtigen Abschnitt der Dokumentation zu gelangen.

Build ausfiihren:

1. **iPhone per Kabel an den fur die Entwicklung verwendeten Mac anschlieBen.**

2. In Android Studio im Terminal nachfolgenden Befehl anpassen und ausführen:

```
flutter run --release -d 'insert-device-id' --flavor 'insert_flavor' --dart-define=CONFIG='insert_config'
```

```
'insert_device_id' = Die DeviceID die verwendet werden soll.
'insert_flavor'    = Das Build-Flavor das verwendet werden soll.
'insert_config'    = Die Konfiguration (e.g.: 'dev.json'/'user.json') die verwendet werden soll.
```

```
Beispiel:
# Release-Build auf dem Ersten verbundenen Gerät installieren und mit dem 'dev' Flavor und der 'dev.json' Konfigurati
flutter run --release -d 0 --flavor dev --dart-define=CONFIG=dev
```

5.2.2 App-Client - NFC-Tags konfigurieren:

Arbeitsplatz NFC-Tag erstellen

Last edited by **Lukas Jeckle** 4 days ago

NFC-Tag für einen Arbeitsplatz erstellen

Allgemeine Informationen

- Die verwendeten NFC-Tags müssen das Datenformat NDEF unterstützen!

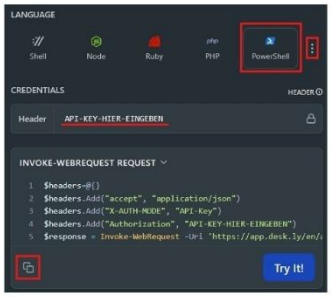
Arbeitsplatz-ID über desk.ly-API abfragen

> **Important**
> Zum Zeitpunkt der Erstellung dieser Dokumentation befindet sich ein Fehler in der offiziellen gelisteten desk.ly-API Abfrage. Nachfolgend wird gezeigt, wie die Anfrage trotz des Fehler ausgeführt werden kann.

1. Auf die desk.ly-API Website gehen und die Resource > List Abfrage auswählen.

   Link: https://deskly.readme.io/reference/resource-list

2. Unter dem Aspekt "Language" PowerShell auswählen.
3. Unter dem Aspekt "Header" den API-Key einfügen.
4. Die aufgelisteten WebRequest Befehle kopieren.



5. PowerShell öffnen und die kopierten Befehle einfügen.

> **Note**
> Der letzte Befehl muss vor der Ausführung noch angepasst werden.

```
# Der von desk.ly im Befehl hinterlegte Endpunkt 'resources' existiert nicht.
# .../v2/resources/list?... -> anpassen zu: .../v2/resource/list?...

# Auch das limit von 50 wollen wir erhöhen, um alle resource-id's auf einmal abzufragen.
# ...[limit]=50... -> anpassen zu: ...[limit]=250...
```

```
# Berichtigte API-Anfrage:
$response = Invoke-WebRequest -Uri 'https://app.desk.ly/en/api/v2/resource/list?page[limit]=250&page[offset]=0' -M
```

6. Ergebnisse der Abfrage in eine JSON-Datei speichern.

```
# Beispiel für Speicherpfad: "C:\Users\<Nutzername>\Downloads\deskly-resources.json".
$response | ConvertFrom-Json | ConvertTo-Json -Depth 100 | Out-File -FilePath "Speicherpfad-hier-angeben!" -Encodi
```

7. Gespeicherte JSON-Datei öffnen.

> **Tip**
> Die am Beginn der JSON-Datei aufgelisteten Arbeitsplätze mit "1.OG-**" sind Arbeitsplätze vom Standort Darmstadt.

Beispiel Eintrag eines Arbeitsplatzes (Standort Ulm):

```
        "id":  "ef15e333-1cb8-4782-9659-53b62b53245d",
        "name":  "DG05-T01",
        "resourceType":  {
                            "name":  "Seat"
                        },
        "vertices":

                            368.84310618067,
                            854.92868462758


    },
```

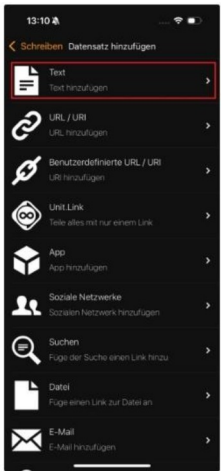

Arbeitsplatz-1D auf NFC-Tag schreiben

Empfehlung: Die App 'NFC-Tools' van 'wakdev' verwenden.
Link (Playstore): Jmps:// y google.com/store/apps/details?id=com.wakdev.wdnfc
Link (AppStore): https://apps.apple.com/us/app/nfc-tools/id1252962749

1. App offnen und 'Schreiben' anwahlen.

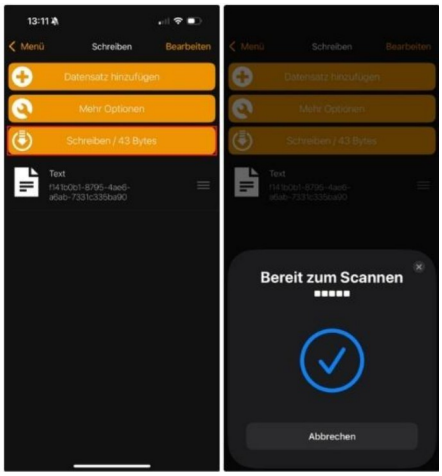
2. **'Oatensatz hinzufugen' anklicken.**

3. **'Text' auswahlen.**



4. Die 'resource.id' des Arbeitsplatzes in den Text-Record eintragen und mit 'OK' best8tigen.

5. 'Schreiben' anwiihlen und den NFC-Tag an das Handy halten.



Einrichtung des NFC-Tags abgeschlossen.

5.2.3 App-Client - Android Berechtigungen

Android Installation Berechtigungen

Last edited by **Lovro Lupis** 1 day ago

Android Berechtigungen

Android Berechtigungen die für die Nutzung der Hintergrundbuchung erforderlich sind.

Bitte Schritt für Schritt diesem Setup folgen.

1. Benachrichtigungen zulassen.

2. Gerätestandort erlauben.

Genauer Standort und während Nutzung der App auswählen.

3. Berechtigung Standort

Bitte hier immer zulassen auswahten.

4. Appimmer im Hintergrund ausfi.ihren.

Zulassen auswiihlen.

Erktarung:

Geratestandort Berechtigung Immer erlauben, damit die Wifi-SSID im Hintergrund gelesen werden kann und einen Service zu starten, falls man sich dabei im exxcellent Wifi befindet.
App immer im Hintergrund ausfi.ihren Berechtigung erlauben, damit ein Service tautt, der jede 5 Minuten UberprOft ob man sich in der exxcellent Wifi-SSID befindet und ggf. Aktionen auszufOhren. Diese Berechtigung ist notwendig, damit der Service Gberhaupt im Hintergrund laufen kann und nicht vom Betriebssystem, durch zum Beispiel Akku Optimierungen beendet werden kann.

5.2.4 Web-Anwendung - 0berblick uber die WebApp

Aus Datenschutzgründen wurde dieser Abschnitt entfernt. In der Originalversion waren hier Screenshots aus dem internen Wiki enthalten, die aus dem privaten GitLab-Account der Firma *exxcellent solutions* stammen. Ein Austausch oder eine Veröffentlichung dieser Bilder ist derzeit leider nicht möglich.

5.2.5 Web-Anwendung - Setup Anleitung

Setup

Last edited by **Lukas Jeckle** 4 days ago

Projekt Setup: deskly-webapp

1. Node.js und npm

- Installiere Node.js (npm ist enthalten):
  https://nodejs.org/en

2. IDE (z. B. Android Studio oder VS Code)

- Du kannst eine beliebige IDE verwenden.
- Navigiere im Terminal in den Projektordner.

3. Git

- Prufen ob Git installiert ist:

```
git --version
```

- Falls nicht installiert: https://git-scm.com/downloads

4. Repository klonen

```
git clone 'REP0_URL'
```

Beispiel-URLs:

- HTTPS:
  `https://gitlab-ext.exxcellent.de/r-and-i/student-apps/deskly/deskly-webapp.git`
- SSH:
  `git@gitlab-ext.exxcellent.de:r-and-i/student-apps/deskly/deskly-webapp.git`

Credentials im geoffneten Fenster eingeben und danach ins Projektverzeichnis wechseln:

```
cd deskly-webapp
```

5. Git Hook fur Commit Messages

```
cp ./commit-msg ./.git/hooks/
```

Uberprufen, ob die Oatei korrekt kopiert wurde:

```
dir ./.git/hooks/
```

6. Abhängigkeiten installieren

```
npm install
```

> Hinweis: CMD mit **Administratorrechten** starten!
> PowerShell kann ggf. Probleme verursachen.

7. `.env` Datei anlegen

Datei `.env` im Projektordner erstellen:

```
API_KEY=…
PORT=80
```

> Achte darauf, dass `.env` in `.gitignore` steht.

8. Webserver starten

```
node Server.js
```

9. Docker

Voraussetzungen

- Docker Desktop für Windows/macOS oder Docker nativ unter Linux.

Container starten:

```
docker-compose up --build
```

> `--build` sorgt dafür, dass bei jeder Ausführung ein neues Image erstellt wird.

Nur Image bauen:

```
docker-compose build
```

5.2.6 Web-Anwendung – Benutzeroberflache & Funktionen

Aus Datenschutzgründen wurde ein Teil dieses Abschnittes entfernt. In der Originalversion waren hier Screenshots aus dem internen Wiki enthalten, die aus dem privaten GitLab-Account der Firma *exxcellent solutions* stammen.
Ein Austausch oder eine Veröffentlichung dieser Bilder ist derzeit leider nicht möglich

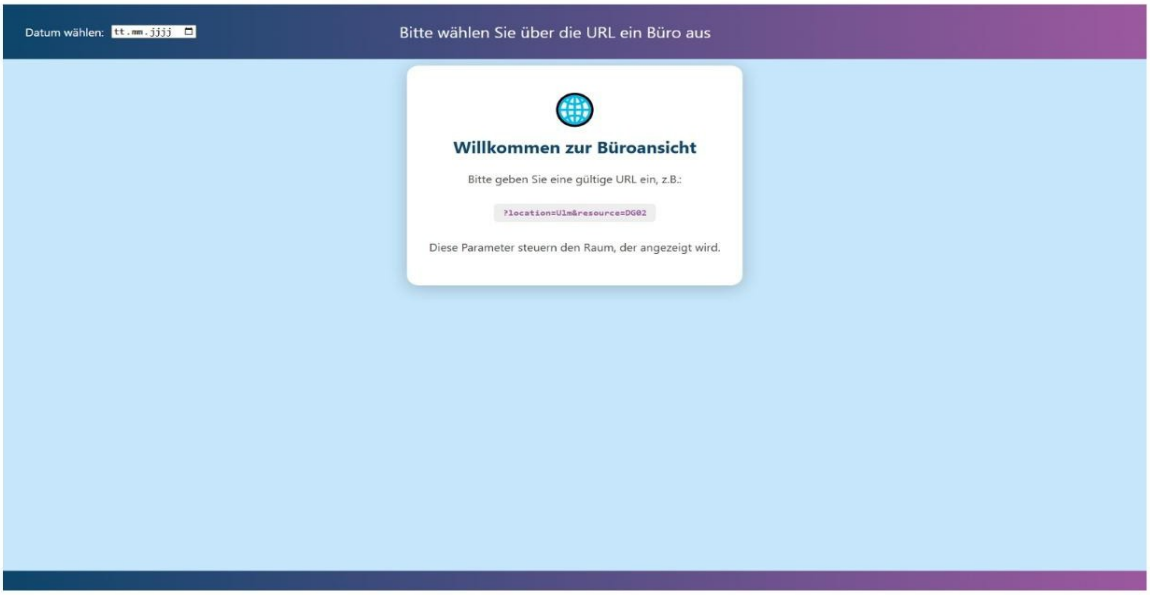
- Über zwei Dropdown-Menüs kann der **Standort** und die **Etage** ausgewählt werden.
- Beide Filter sind **aufeinander abgestimmt**.
- Beispiel: Wählt man einen Standort aus, werden nur die dazugehörigen Etagen angezeigt.
- Alternativ kann die Filterung **direkt über die URL** gesteuert werden:
  - Beispiel:

```
localhost/general-overview?loc=Ulm&floor=DG
```

    - `loc` = Standort (Location)
    - `floor` = Etage (Floor)

Büroansicht

Wählt die einrichtende Person die Büroansicht aus, erscheint eine **Übergangsseite**:



- Auf dieser Seite wird ein **vorgeschlagener Link** angezeigt.
- Der Link muss **an die bestehende URL angehängt** werden. Dabei können die URL Parameter angepasst werden.
- Steuerung über URL-Parameter:
  - Beispiel:

```
localhost/office-overview?location=Ulm&resource=DG02
```

    - `location` = Standort
    - `resource` = Bürozimmer

Büro Detailansicht

Nach Eingabe der richtigen URL wird die Büroansicht geöffnet:
Hier sieht die einrichtende Person die aktuelle Belegung des Büros:

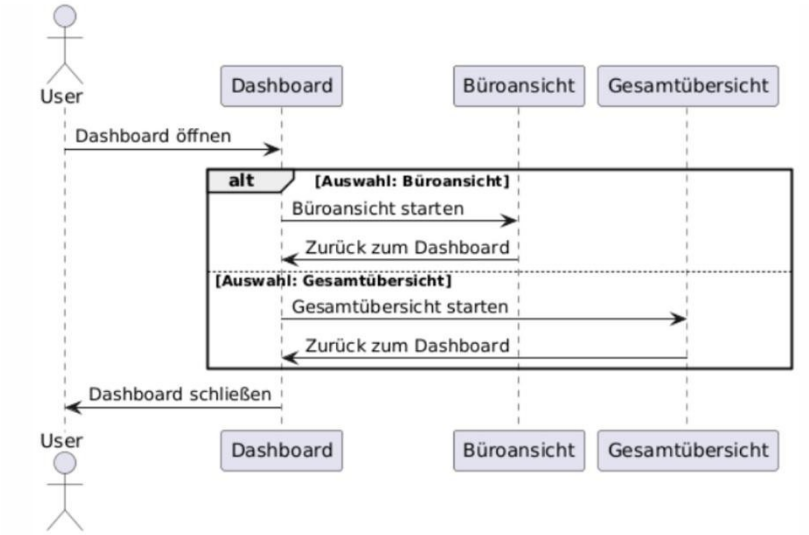
5.2.7 Web-Anwendung - Technische Umsetzung

Technische Umsetzung

Last edited by **Niklas Kuemmel** 1 week ago

📊 Ablaufübersicht der Applikation (Sequenzdiagramme)

Bild 1 – Navigation über das Dashboard



Beschreibung:

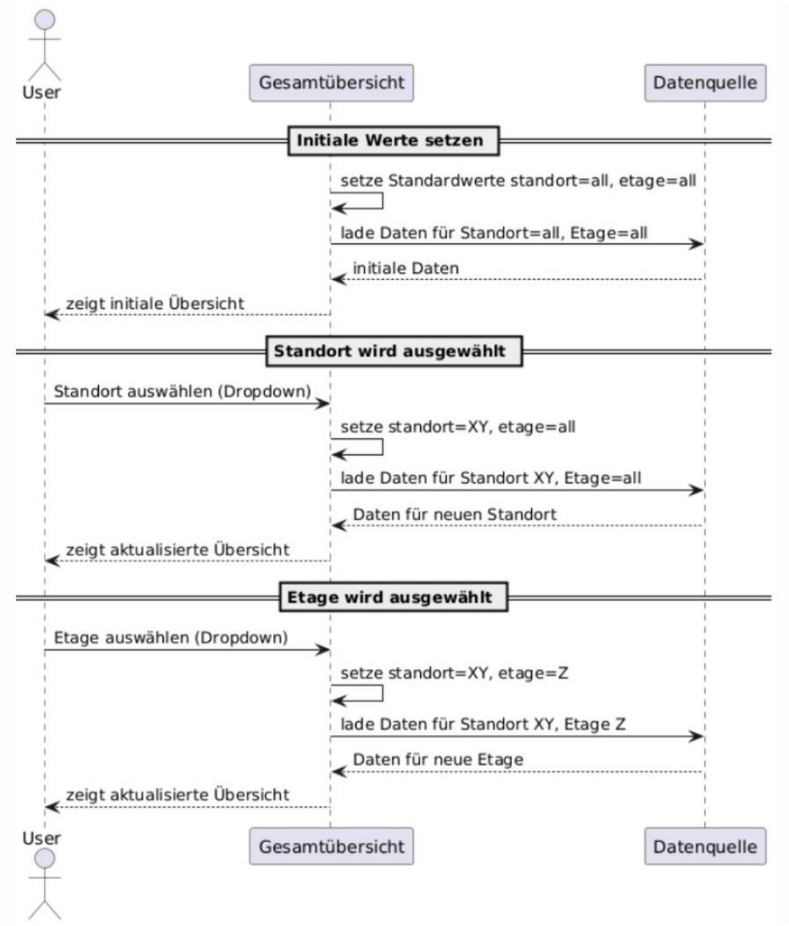
Beim Öffnen des Dashboards bekommt der User zwei Optionen:

```
Die Büroansicht starten

Die Gesamtübersicht starten
```

Diese beiden Varianten sind im Diagramm als alt-Block dargestellt (alternative Abläufe). Nach jeder Auswahl gelangt der User mit der Pfeiltaste wieder zurück zum Dashboard. Am Ende kann es geschlossen werden.

Fazit: Klare, einfache Navigation. Flexibel und benutzerfreundlich gestaltet.

Bild 2 – Interaktion in der Gesamtübersicht

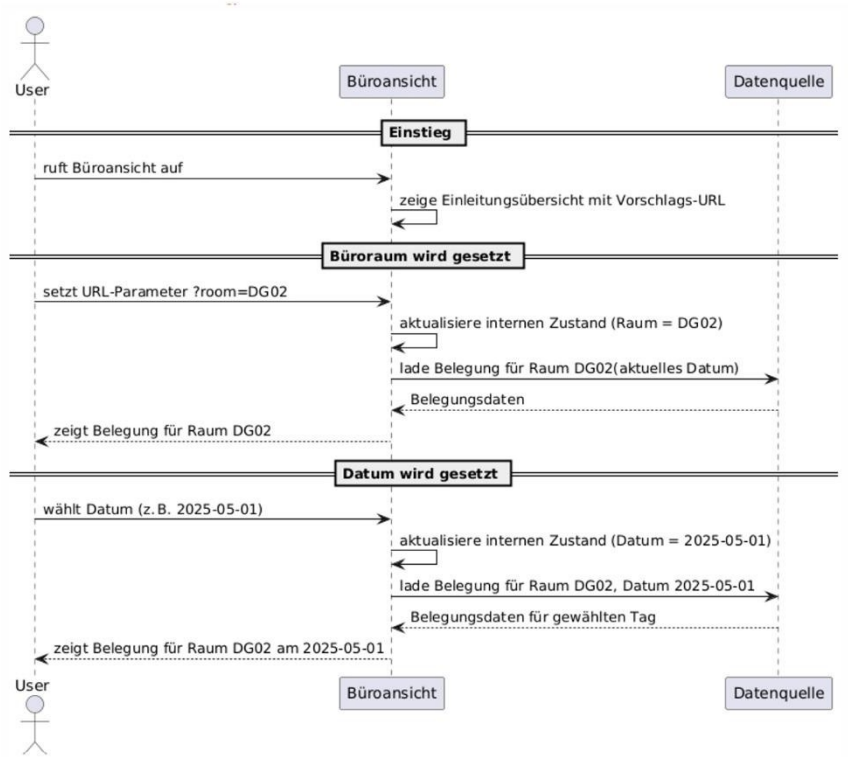


Beschreibung:

```
Beim Aufruf werden Standardwerte für Standort & Etage gesetzt (all).

Der User kann über das Dropdown-Menü einen Standort auswählen, was zu einem Datenabruf führt.

Nun kann er die Informationen, also die Abfrage der Mitarbeiter, einschränken, indem er mit Hilfe des

zweiten Dropdown-Menüs die gewünschte Etage auswählt.

Diese beiden Parameter können auch mit Hilfe der URL gesetzt werden.
```

Fazit: Interaktiv & reaktiv, alle Änderungen wirken sich sofort auf die angezeigten Daten aus.

Bild 3 – Büroansicht mit Raum- und Datumsauswahl



Beschreibung:

Beschreibung:

```
Beim ersten Öffnen bekommt der User eine Einleitungsübersicht mit einer vorgeschlagenen URL.

Anschließend kann er den Raum über die URL setzen (z. B. ?room=DG02).

Danach kann zusätzlich ein Datum gewählt werden, um Buchungen für die Zukunft zu sehen.

Das System lädt anschließend automatisch die Belegung für den entsprechenden Tag und Raum.
```

Fazit: Dynamische Ansicht mit voller Kontrolle über Raum & Datum, perfekt für tagesgenaue Belegungsinfos.

5.2.8 Web-Anwendung - Testing

Testing

Last edited by Alton Bekolli 6 days ago

5. Testing

FUr die eXXcellent Arbeitsplatzllbersicht wurde ein umfassendes automatisiertes Test-Setup mit Jest und Supertest entwickelt. Die Tests decken sowohl die statischen Ressourcen (HTML, CSS, JS, Bilder) als auch die Backend-API-Endpunkte fur:

- das **Dashboard**
- die GeneralUbersicht
- die BUroansicht

ab.

Voraussetzungen

Stelle sicher, dass folgende Pakete im Projekt installiert sind:

```
npm install --save-dev jest supertest
```

Diese sind in der package.json unter devDependencies bereits enthalten:

```
"devDependencies": { "jest": ""'29.7.8", "supertest": ""7.1. 0" }
```

Ausfuhrung der Tests

Die Tests werden direkt Ober das in der package.json definierte Skrip! gestartet: `npm test`

Alternativ mit detaillierter Ausgabe:

```
npm test ---------- verbose
```

Das -- trennt die Jest-Optionen von npm und sorgt fur die Weltergabe an das Test-Framework.

Abgedeckte Bereiche

Dashboard

- HTML-, CSS- und JS-Dateien
- Bilder (z. B. Logos, Obersichts-Grafiken}
- Navigation zur General- und BUroansicht
- Fehlerseiten (z.B. 404)

Generaliibersicht

- Standort- & Etagen-Dalen Ober /api/dropdowndata
- Mitarbeitenden-Daten per /api/general-data?loc=...&floor=...
- Tests fur alle gOltigen und ungUltigen Kombinationen

Buroansicht

- HTML-, CSS- und JS-Dateien
- Bilder und Fallbacks
- API /api/office-data fiir alle Raume in Ulm & Darmstadt
- Test fur alle Raume uber 30 Tage in die Zukunft
- Umgang mit ungUltigen Parametern

Beispiel-Test

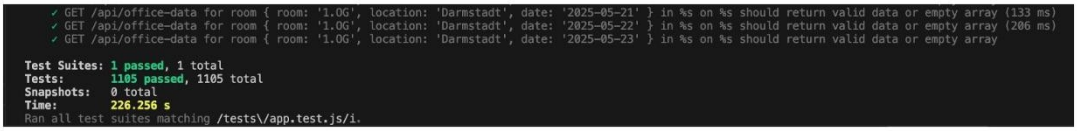
```
test  llllk IIQ 'BurOdr1Slcl1l' ln  ddS'lboaru  s•ioulu  relur11 HlML IGLl /ofl1Ce-ove·v1ew) ' ,,,    (}    {
        rf's = a...alt   request  opp. .get, "/ot 1rf'-o.,lt''rv1f\,'  ;
    expect ı rc$.statusCode .toBe 200) ;
    expec!rt'>.headers  (o1 t(nt  typ1   J.toMatch(      J;
} ;
```

Hinweis zur Laufzeit

Da sehr viele Räume und Datenkombinationen getestet werden (z. B. für 30 Tage in die Zukunft), kann die Ausführung der Tests mehrere Minuten dauern.

Die maximale Laufzeit wurde im Testfile entsprechend angepasst:

```
jest.setTimeout(300000); // 5 Minuten
```



6. Diskussion

Im diesem Abschnitt werden die im Rahmen des Projekts erreichten Ziele detailliert analysiert und reflektiert. Dabei wird auf die ErfOllung der in Abschnitt 3 definierten Projektanforderungen und Zielsetzungen eingegangen. Zudem erfolgt eine retrospektive Betrachtung des Projekts, in der die Umsetzung des Projekts kritisch beleuchtet wird.

6.1 Erreichte Ziele

6.1.1 Erreichte Projektziele

1. Ziel: Abschluss der Projektplanung

Zielsetzung:
Die Projektplanung des Projekts “Automatische Arbeitsplatzübersicht” wird bis zum 18.03.2025 im Rahmen des 1. Projekt-Meilensteins vollstandig abgeschlossen und dokumentiert sein.

Zielerreichung:
Die Projektplanung wurde fristgerecht abgeschlossen. Das Planungsdokument wurde dabei am 17.03.2025 an Hr. Franz per E-Mail gesendet. Das Ziel wurde somit erreicht.

2. Ziel: Erste Demoversion

Zielsetzung:
Bis zum 01.04.2025 soll eine erste Demoversion des Projekts “Automatische Arbeitsplatzübersicht” entwickelt werden. Diese umfasst einen Mobile-App-Client und eine Web-Anwendung mit eingeschrankten Funktionalitaten. Wichtige Funktionalitaten sind das Buchen von Arbeitsplatzen per Mobile-App-Client Ober die desk.ly-API, sowie das Anzeigen einer GesamtObersicht der gebuchten Arbeitsplatze Ober die Web-Anwendung, sowie alle dafOr benotigten Unter-Features. Die Demoversion gilt als Erfolg, wenn diese Funktionalitaten implementiert, getestet und funktionsfahig sind.

Zielerreichung:
Eine Erste Demoversion des App-Clients und der Web-Anwendung des Projekts wurde mit eingeschranktem Funktionalitatsumfang bis zum 01.04.2025 realisiert und im dortigen Sprint-Review auch den Betreuern von eXXcellent solutions vorgestellt. Dieses Ziel wurde somit ebenfalls erreicht.

3. Ziel: Abschluss der Entwicklungsarbeiten am 22.04.2025

Zielsetzung:
Der Mobile-App-Client und die Web-Anwendung des Projekts “Automatisierte Arbeitsplatzübersicht” werden bis zum 22.04.2025 vollständig entwickelt und bereitgestellt. Ab diesem Zeitpunkt sollen nur noch Robustheitstests und Bugfixes erforderlich sein, ohne dass gror!ere Entwicklungsarbeiten notwendig sind. Die Fertigstellung ist erreicht, wenn alle geplanten Funktionen implementiert wurden.

Zielerreichung:
Aufgrund mehrerer zunachst unvorhergesehener Umstande konnten die Entwicklungsarbeiten leider nicht vollstandig bis zum 22.04.2025 abgeschlossen werden. Daher wurde dieses Ziel zeitlich nicht ganz erreicht.

4. Ziel: Projektabschluss

Zielsetzung:
Das Projekt “Automatische Arbeitsplatzübersicht”, einschließlich der vollständigen Dokumentation, soll bis zum 29.04.2025 abgeschlossen sein. Ab diesem Zeitpunkt sollen keine weiteren inhaltlichen Arbeiten erforderlich sein, sodass bis zum 02.05.2025 nur noch ausschlier!lich die Projekt-Abschlussprasentation erstellt und eingeObt werden muss. Der Abschluss des Projekts ist erreicht, wenn alle definierten Features implementiert und die Dokumentation finalisiert wurde.

Zielerreichung:
Dieses Ziel wurde nicht erreicht. Aufgrund von Hindernissen und zusatzlichen Aufgaben, welche wahrend des Projektverlaufs auf das Projekt-Team zukamen, war es leider nicht moglich, das Projekt bis zum 29.04.2025 wie im oben beschriebenen Ziel abzuschlier!en.

5. Ziel: Abschlussprasentation einuben und abhalten

Zielsetzung:
Die Abschlusspräsentation des Projekts “Automatische Arbeitsplatzübersicht” wird als zweiter Meilenstein am 02.05.2025 gehalten. DafOr wird die Prasentation bis zu diesem Termin vollstandig erstellt und eingeObt, sodass sie Oberzeugend prasentiert werden kann. Der Meilenstein gilt als erreicht, wenn die Prasentation vorbereitet, intern geObt und termingerecht durchgefOhrt wurde.

Zielerreichung:
Die Abschlussprasentation wurde fristgerecht vor dem 02.05.2025 erstellt und vom Projekt-Team eingeObt. Aur!erdem wurde die Abschlussprasentation am 02.05.2025 Oberzeugend prasentiert. Damit wurde dieses Ziel termingerecht durchgefOhrt.

6. Ziel: Abgabe Quelltexte

Zielsetzung:
Die Abgabe der Projektdokumentation und der Quelltexte stellt den dritten Meilenstein des Projekts “Automatische Arbeitsplatzübersicht” dar und erfolgt bis 04.05.2025. Dieses Ziel gilt als erfolgreich abgeschlossen, wenn die vollstandige Projektdokumentation und alle finalen Quelltexte fristgerecht eingereicht wurden.

Zielerreichung:
Der Abschluss des dritten Projektmeilensteins erfolgte fristgerecht. Die Dokumentation, sowie alle relevanten Quelltexte wurden bis zum 04.05.2025 eingereicht. Damit wurde dieses Ziel erfolgreich erreicht.

6.1.2 Umsetzung der funktionalen Anforderungen

1. Funktionale Anforderung:

Anforderung:
Die Arbeitsplatzbuchung soll mittels eines Mobile-App-Clients durchgefOhrt werden konnen. Hierzu soll ein fOr jeden Arbeitsplatz einzigartiger NFC-Tag ausgelesen werden, anhand dessen Informationen die Arbeitsplatzbuchung entsprechend abgewickelt wird. Vom Mitarbeitenden soll hierfOr keine weitere Interaktion, als das Smartphone Ober den NFC-Tag zu halten, erforderlich sein.

Anforderungsbefriedigung:
Eine komplett interaktionslose Buchung, bei der sich das Smartphone des Nutzers im Sperrmodus befindet, war leider aufgrund von Sicherheitsrestriktionen nicht moglich. Nichtsdestotrotz wurde diese Anforderung durch das Ermoglichen der Hintergrundbuchungen auf Android Geraten bestmoglich erfOllt.

2. Funktionale Anforderung:

Anforderung:
Ober eine Web-Anwendung soll eine GesamtObersicht aller aktuell im Unternehmen anwesenden Mitarbeitenden realisiert werden. Diese GesamtObersicht soll auf einem gror!en Monitor im Eingangsbereich von eXXcellent solutions Ober den Browser dargestellt werden.

Anforderungsbefriedigung:
Wahrend des Projekts wurde in der Web-Anwendung eine GesamtObersicht erstellt, welche alle Anwesenden Mitarbeitenden klar und benutzerfreundlich darstellt. Hierbei kann in der Gesamtansicht sowohl der Standort als auch die Etage, die angezeigt werden soll, einfach Ober ein Dropdown Feld ausgewahlt werden. Die Moglichkeit diese Gesamtansicht auf einem Monitor im Eingangsbereich darzustellen ist gegeben. Diese Anforderung wurde damit ebenfalls bestmoglich erfOllt.

3. Funktionale Anforderung:

Anforderung:
Ober eine Web-Anwendung soll fOr jedes BOro eine detailliertere Ansicht der aktuell im BOro anwesenden Mitarbeitenden ermoglicht werden. Diese BOroObersicht soll jeweils auf kleineren Monitoren vor jedem BOro von eXXcellent solutions Ober den Browser dargestellt werden.

Anforderungsbefriedigung:
Wahrend des Projekts wurde in der Web-Anwendung eine BOroansicht erstellt, in welcher das anzuzeigende BOro Ober URL-Parameter ausgewahlt werden kann. Die BOroansicht zeigt eine visuell ansprechende Obersicht aller im BOro anwesenden Mitarbeitenden. Die Moglichkeit diese Ansicht auf kleinen Monitoren vor jedem BOro bei eXXcellent solutions darzustellen ist gegeben. Somit wurde diese Anforderung ebenfalls bestmoglich erfOllt.

4. Funktionale Anforderung:

Anforderung:
Um einen Arbeitsplatz zu buchen, sollen sich die Mitarbeitenden von eXXcellent solutions im Mobile-App-Client mittels Microsoft Single Sign-On einloggen konnen.

Anforderungsbefriedigung:
Der im App-Client implementierte Microsoft Single Sign-On Login ist funktionsfahig, jedoch handelt es sich dabei um keine direkte OAuth Authentifizierung, sondern um einen Workaround, welcher den Microsoft Access-Token Ober den in der App bestehenden Firebase Authentifizierungsprozess abgreift. Aufgrund vertraglicher Schwierigkeiten mit desk.ly war es leider nicht moglich, einen offiziellen OAuth Zugang zu implementieren. Trotz der gegebenen Umstande wurde das Ziel bestmoglich umgesetzt.

5. Funktionale Anforderung:

Anforderung:
Die Darstellung der Gesamtansicht und der einzelnen BOroansichten Ober den Browser soll keine Runtime Umgebung benotigen, damit sichergestellt ist, dass die Darstellung auf allen Monitoren ohne Probleme erfolgen kann.

Anforderungsbefriedigung:
Die in der Web-Anwendung verwendeten Technologien wurden bereits in Abschnitt 4.1.2 aufgelistet. Diese benotigen keine Runtime Umgebung, wodurch diese Anforderung ebenfalls erfOllt ist.

6. Funktionale Anforderung:

Anforderung:
Im Eingangsbereich von eXXcellent solutions soll direkt neben der EingangstOr ein iBeacon installiert werden, welcher dauerhaft ein Signal zum Ausloggen einer aktuellen Arbeitsplatzbuchung vorbeilaufender Mitarbeiter sendet. Damit sollen Arbeitsplatzressourcen effizient und automatisiert wieder freigegeben werden.

Anforderungsbefriedigung:
Diese Anforderung wurde im Laufe des Projekts fallengelassen.

6.1.3 Umsetzung der nicht funktionalen Anforderungen

1. Nicht funktionale Anforderung:

Anforderung:
Die Arbeitsplatzbuchung muss fOr die Mitarbeitenden moglichst einfach sein, damit diese akzeptiert und angewendet wird.

Anforderungsbefriedigung:
Durch das Scannen der NFC-Tags ist die Arbeitsplatzbuchung deutlich einfacher geworden, eine noch einfachere Umsetzungsmoglichkeit gibt es nicht. Damit ist diese Anforderung bestens erfOllt worden.

2. Nicht funktionale Anforderung:

Anforderung:
Der Quellcode unserer Implementierungen soll hohe Qualitat aufweisen. Hierzu sollen die Architekturen, sowie die Wart- und Erweiterbarkeit der Implementierungen bereits in der Planungsphase berOcksichtig und in der Entwicklung entsprechend umgesetzt werden.

Anforderungsbefriedigung:
Bereits wahrend der Planungsphase wurden Faktoren bezOglich der Wart- und Erweiterbarkeit der Software-Implementationen berOcksichtigt. Damit wurde auch diese Anforderung bestmoglich erfOllt.

3. Nicht funktionale Anforderung:

Anforderung:
Der Mobile-App-Client soll eine intuitive Benutzeroberflache bieten.

Anforderungsbefriedigung:
Die Oberflachen des App-Clients wurden so designt, dass dem Nutzer zu jedem Zeitpunkt auf einen Blick klar ist, ob er aktuell an einem Arbeitsplatz eingebucht ist, oder nicht. Entsprechend ist die Benutzeroberflache fOr den Nutzer intuitiv gestaltet, wodurch auch dieses Ziel erfOllt ist.

4. Nicht funktionale Anforderung:

Anforderung:
Der Mobile-App-Client soll so optimiert sein, dass er auf den ausfOhrenden Mobilgeraten moglichst wenig Akku verbraucht.

Anforderungsbefriedigung:
Nachdem die App zwei Tage lang auf einem Android Gerat mit aktivierten Hintergrunddiensten im taglichen Gebrauch getestet wurde, konnte kein signifikanter Anstieg des Akkuverbrauchs festgestellt werden. Somit ist auch diese Anforderung erfOllt worden.

5. Nicht funktionale Anforderung:

Anforderung:
Die Mobile-App soll auf den Plattformen Android und iOS umgesetzt werden.

Anforderungsbefriedigung:
Durch die Verwendung des Frameworks Flutter im App-Client kann dieser mit einer einzigen gemeinsamen Codebasis fOr beide Plattformen, Android und iOS, umgesetzt werden. Die entsprechenden Build-Workflows sind ebenfalls eingerichtet, wobei der iOS Build Prozess im Laufe des Projekts repariert und um die benotigten Funktionalitaten erweitert wurde. Somit wurde auch diese Anforderung erfOllt.

6.2 Retrospektive

Der Einstieg in das Projekt:

Der Projektstart verlief reibungslos. Zu Beginn haben wir unser Projekt und die Standards definiert. Dies beinhaltete eine erste Vorstellung der Architektur und die Festlegung der Programmiersprache. Alle Teammitglieder wurden Ober die Ziele und Aufgaben informiert. Besonders positiv war, dass wir von Anfang an die gleichen Arbeitszeiten hatten, was die Koordination und Zusammenarbeit sehr erleichtert hat.

Verlauf des Projekts:

- Die Aufgaben wurden zu Beginn klar verteilt, jedes Teammitglied entnahm eigens Aufgaben aus dem Backlog.
- Die Kommunikation im Team war offen und konstruktiv.
- Herausforderungen wurden gemeinsam gelost innerhalb der Projektgruppen, dabei wurde zuerst der Vorrang von Problemen der App innerhalb der App-Gruppe versucht zu losen und in der Web-App Gruppe genauso.
- Der Zeitplan wurde eingehalten, und die erste lauf bare Demo war zum abgesprochenen Zeitpunkt fertig.

Nach Abschluss des Projekts:

- Die Ziele des Projekts wurden erfolgreich erreicht.
- Die Zusammenarbeit im Team hat sich im Laufe der Zeit verbessert.
- Es fanden regelmar!ige Reflexionen und Feedbackrunden statt.
- Verbesserungsvorschlage fOr zukOnftige Projekte wurden gesammelt.

„Lessons learned“:

- Eine gute Planung und eine klare Kommunikation sind entscheidend fOr den Erfolg des Projekts.
- Einheitliche Arbeitszeiten sind ein Plus fOr die Arbeit im Team besonders unter den Teilgruppen innerhalb des Projektes.
- FrOhzeitiges Entwickeln von Kommentaren und Tests fOr bessere Wartbarkeit und Problemlosung.
- Vorhandene Dokumentation lesen und verstehen, um Anfangsprobleme zu vermeiden.

7. Ausblick

lm letzten Abschnitt dieser Projektdokumentation soll ein moglicher Ausblick auf Verbesserungen der im Projekt erstellten Software-lmplementationen gegeben werden. Ziel des Ausblicks ist es, aufzuzeigen, in welchen Bereichen der lmplementationen durch gezielte Anpassungen oder Erweiterungen die Benutzerfreundlichkeit erhoht werden konnte.

Verbesserungspotenzial besteht bei:

1. Microsoft Single Sign-On im App-Client

Die aktuelle lmplementation des im App-Client verwendeten Microsoft Single Sign-On Prozesses verwendet als Grundstein den von eXXcellent solutions bereits vorab implementierten Firebase Authentifizierungsprozess. Hierbei bei Ober einen Microsoft Provider eine Microsoft Authentifizierung durchfOhrt.

Dieser Ansatz hat jedoch einen entscheidenden Nachteil gegenOber einer offiziellen Microsoft OAuth Authentifizierung. Der fOr die Microsoft Authentifizierung verwendete Firebase Microsoft Provider liefert bei der Anmeldung nur einen Microsoft Access-Token, nicht aber einen Refresh-Token zum Erneuern des Access-Token, nachdem dieser ablaufen ist.

Aufgrund dessen lauft der Microsoft Access-Token aktuell nach einer Stunde ab, wodurch sich die Nutzer bei einer Microsoft Authentifizierung im App-Client nach einer Stunde neu authentifizieren mOssen.

Um diesen Prozess zu verbessern, mOsste von Seitens eXXcellent solutions auf desk.ly zugegangen werden, um fOr die eXXcellent App einen offiziellen OAuth Zugang anzufragen. Das Rekonstruieren des auf der desk.ly Homepage implementierten Microsoft OAuth Zugangs ist aufgrund der Redirect-URL, welche auf die Homepage von desk.ly und nicht in die eXXcellent App weiterleitet, nicht moglich gewesen.

Im Rahmen des Projekts „Automatische Arbeitsplatzübersicht“ konnte leider jedoch kein offizieller Microsoft OAuth Zugang fOr die eXXcellent App umgesetzt werden, da der Supportvertrag zwischen desk.ly und eXXcellent solutions von Seitens eXXcellent solutions gekOndigt worden war. In einem direkten Telefonat mit dem desk.ly Kundensupport wurde von Seitens desk.ly mitgeteilt, dass keinerlei Support fOr unser Studentenprojekt gegeben werden mochte.

Sollte der Supportvertrag mit desk.ly wieder aufgenommen werden, bestOnde die Moglichkeit in Kooperation mit diesen einen offiziellen Microsoft OAuth Zugang fOr die eXXcellent App einzurichten. HierfOr konnte zum Beispiel die Flutter Abhangigkeit „msal_auth“ verwendet werden, welche die lmplementierung der Microsoft OAuth Authentifizierung in Flutter, sowie eine automatische Token-Verwaltung ermoglichen wOrde. Dadurch wOrde der Authentifizierungsprozess deutlich Benutzerfreundlicher gestaltet werden, da nicht stOndlich eine neue Authentifizierung benotigt werden wOrde.

2. Alarm-Manager im Foreground-Service des App-Clients

Der Foreground-Service wird gestartet, sobald ein Nutzer sich mit dem exxcellent Wifi verbindet und andere Anforderungen passen. Das Verfahren funktioniert mittels eines Alarm-Managers. Dies ist ein Service auf Android, mit dem es moglich ist, die App alle x Minuten im Hintergrund aufzuwecken und eine Aufgabe zu bearbeiten. Derzeit wird dieser Alarm-Manager verwendet, um die jetzige Wifi-SSID mittels eines WiFi Managers jede 5 Minuten im Hintergrund auszulesen. Statt jede 5 Minuten im Hintergrund zu pollen, gibt es bessere Wege, um der App mittzuteilen das es jetzt in einem bestimmten Wifi ist. Man kann z.B. den Router so konfigurieren, dass er Push-Nachrichten per Firebase verwendet. Man konnte ebenso einen iBeacon am Eingang platzieren und automatisch einen Service starten falls notig, ohne jedes Mal pollen zu mOssen.

3. APl-Modularitat der App-Anwendung

Damit desk.ly nicht nur mit einer spezifischen APl funktioniert, sondern auch alternative Backends unterstOtzt, ist eine modulare APl-Architektur unabdingbar. Dazu mOssen insbesondere die Basis-URLs und Endpunkte flexibel gestaltet und klar von der Geschaftslogik getrennt werden. Dieser Aspekt ist bereits gut umgesetzt, sollte aber weiter ausgebaut werden, um die Austauschbarkeit von APl-Backend zu gewahrleisten.

4. APl-Modularitat der Web-Anwendung

Wie bei der Mobile-App sollte es auch bei der Web-App moglich sein die APl fOr die Buchungsdaten auszutauschen. Hier gestaltet sich dies leider etwas schwieriger. Beispielsweise stellt die APl von desk.ly zum jetzigen Zeitpunkt noch keinen vollstandigen Endpunkt zum Abrufen der Raumdaten bereit. FOr die BOroansicht mussten wir daher die Raume Ober die Namen der Ressourcen zusammenstellen. Dies war nur moglich, da eXXcellent solutions die Ressourcen einheitlich benannt hat, sodass die Etage und der Raum im Namen enthalten ist. Sollte die APl ausgetauscht werden, ware eine Losung Ober einen Endpunkt fOr Raumdaten vorzuziehen. Generell mOssten vor allem 2 Dinge angepasst werden. Die APl-Endpunkte und die Methoden, die die Daten nach Etagen filtern. Die Filterung ist stand jetzt auf die JSON-Responses der desk.ly APl zugeschnitten.

8. Quellenangaben

8.1 Projektquellen

[1] OpenAI. (2025). ChatGPT. https://chatgpt.com/

[2] *PlantUML Web Server*. (n.d.). PlantUML.com. http://www.plantuml.com/

[3] *Newest questions*. (n.d.-b). Stack Overflow. https://stackoverflow.com/questions

[4] *Home I Scrum Guides*. (n.d.). https://scrumguides.org/

[5] Atlassian. (n.d.). *Jira I Software for die Vorgangs- und Projektverfolgung I Atlassian*. https://www.atlassian.com/de/software/jira

[6] Atlassian. (n.d.-a). *Confluence I Der Remote-freundliche Arbeitsbereich for dein Team I Atlassian*. https://www.atlassian.com/de/software/confluence

[7] *Flutter - Build apps for any screen*. (n.d.). https://flutter.dev/

[8] *Flutter learning resources*. (n.d.). https://docs.flutter.dev/reference/learning-resources

[9] Shah Wali. (2024, June 30). *How to install Flutter on Windows 2024 I Setup Android Studio for Flutter Step by Step* [Video]. YouTube. https://www.youtube.com/watch?v=mMeQhLGD-og

[10] *Dart programming language*. (n.d.). Dart. https://dart.dev/

[11] *Android Studio und App-Tools herunterladen – Android-Entwickler*. (n.d.). Android Developers. https://developer.android.com/studio?hl=de

[12] *Create API-Key*. (n.d.-b). desk.ly. https://deskly.readme.io/

[13] desk.ly GmbH. (n.d.). *Desk Sharing Software for flexible Teams I desk.ly*. https://www.desk.ly/de/

[14] *Burp Suite - Application Security Testing Software*. (n.d.). PortSwigger. https://portswigger.net/burp

[15] Wikipedia-Autoren. (2006, April 8). *HTTP-StatusCode*. https://de.wikipedia.org/wiki/HTTP-Statuscode

[17] *An overview of HTTP - HTTP I MDN*. (2025, March 14). MDN Web Docs. https://developer.mozilla.org/en-US/docs/Web/HTTP/Guides/Overview

[18] *Redirections in HTTP - HTTP | MDN*. (2025, April 10). MDN Web Docs. https://developer.mozilla.org/en-US/docs/Web/HTTP/Guides/Redirections

[19] *HTTP authentication - HTTP | MDN*. (2025, March 13). MDN Web Docs. https://developer.mozilla.org/en-US/docs/Web/HTTP/Guides/Authentication

[20] *Navigate with named routes*. (n.d.). Flutter. https://docs.flutter.dev/cookbook/navigation/named-routes

[21] *Future.microtask constructor - Future - dart:async library - Dart API*. (n.d.). https://api.flutter.dev/flutter/dart-async/Future/Future.microtask.html

[22] Kamal, F. (n.d.). *JSToolSet*. https://www.jstoolset.com/jwt

[23] Damienbod. (n.d.). *Konfigurieren der JWT-Bearer-Authentifizierung in ASP.NET Core*. Microsoft Learn. https://learn.microsoft.com/de-de/aspnet/core/security/authentication/configure-jwt-bearer-authentication?view=aspnetcore-9.0

[24] Alizadeh, H. (2021, December 16). Tips on HTTP request in Darts for Beginners - HamidReza Alizadeh - Medium. *Medium*. https://hamidrdeveloper.medium.com/tips-on-http-request-in-darts-for-beginners-21187470f06b

[25] *Simple app state management*. (n.d.). Flutter. https://docs.flutter.dev/data-and-backend/state-mgmt/simple

[26] Block, S. (2024, July 24). *Token zur Authentifizierung - Access Token, Refresh Token und Bearer Token - Rock the Prototype - Softwareentwicklung & Prototyping*. Rock the Prototype - Softwareentwicklung & Prototyping. https://rock-the-prototype.com/it-sicherheit/token-zur-authentifizierung-access-token-refresh-token-und-bearer-token/

[27] Born, T. (2019, December 17). *Was ist ein Bearer Token? Beispiel einer API Autorisierung ober HTTP*. https://www.predic8.de/bearer-token-autorisierung-api-security.htm

[28] *Writing custom platform-specific code*. (n.d.). https://docs.flutter.dev/platform-integration/platform-channels

[29] *Node.js — Run JavaScript everywhere*. (n.d.). https://nodejs.org/en

[30] *Random User Generator | Home*. (n.d.). https://randomuser.me/

[31] *dio | Dart package*. (n.d.). Dart Packages. https://pub.dev/packages/dio

[32] *firebase_auth | Flutter package*. (n.d.). Dart Packages. https://pub.dev/packages/firebase_auth

[33] *msal_auth | Flutter package*. (n.d.). Dart Packages. https://pub.dev/packages/msal_auth

[34] *nfc_manager | Flutter package*. (n.d.). Dart Packages. https://pub.dev/packages/nfc_manager

[35] *json_annotation | Dart package*. (n.d.). Dart Packages. https://pub.dev/packages/json_annotation

[36] *flutter_dotenv | Flutter package*. (n.d.). Dart Packages. https://pub.dev/packages/flutter_dotenv

[37] *build_runner | Dart package*. (n.d.). Dart Packages. https://pub.dev/packages/build_runner

[38] *json_serializable | Dart package*. (n.d.). Dart Packages. https://pub.dev/packages/json_serializable

[39] *mockito | Dart package*. (n.d.). Dart Packages. https://pub.dev/packages/mockito

[40] *Typen von Diensten im Vordergrund sind erforderlich*. (n.d.). Android Developers. https://developer.android.com/about/versions/14/changes/fgs-types-required?hl=de

[41] *Wecker stellen*. (n.d.). Android Developers. https://developer.android.com/develop/background-work/services/alarms/schedule?hl=de

[42] *NFC – grundlagen*. (n.d.). Android Developers. https://developer.android.com/develop/connectivity/nfc/nfc?hl=de

[43] *Benachrichtigung erstellen*. (n.d.). Android Developers. https://developer.android.com/develop/ui/views/notifications/build-notification?hl=de

[44] *Kotlin-Koroutinen unter Android*. (n.d.). Android Developers. https://developer.android.com/kotlin/coroutines?hl=de

[45] *Berechtigung zur Standortermittlung anfordern*. (n.d.). Android Developers. https://developer.android.com/develop/sensors-and-location/location/permissions?hl=de

[46] *WifiManager: API reference: android developers*. Android Developers. (n.d.). https://developer.android.com/reference/android/net/wifi/WifiManager

[47] *HttpURLConnection: API reference: android developers*. Android Developers. (n.d.-a). https://developer.android.com/reference/java/net/HttpURLConnection

[48] Ratliff, S. (2025, May 1). *Docker: Accelerated Container Application Development*. Docker. https://www.docker.com/

[49] *npm: dotenv*. (n.d.). Npm. https://www.npmjs.com/package/dotenv

[50] *npm: express*. (n.d.). Npm. https://www.npmjs.com/package/express

[51] *npm: node-fetch*. (n.d.). Npm. https://www.npmjs.com/package/node-fetch

[52] *npm: jest*. (n.d.). Npm. https://www.npmjs.com/package/jest

[53] *npm: supertest*. (n.d.). Npm. https://www.npmjs.com/package/supertest

8.2 Bilder

Figure 1: https://innosued.de/wp-content/uploads/2019/02/THU_Logo_Standard.jpg

9. Anlagen

- Projekt Poster
- Abschlussprasentation

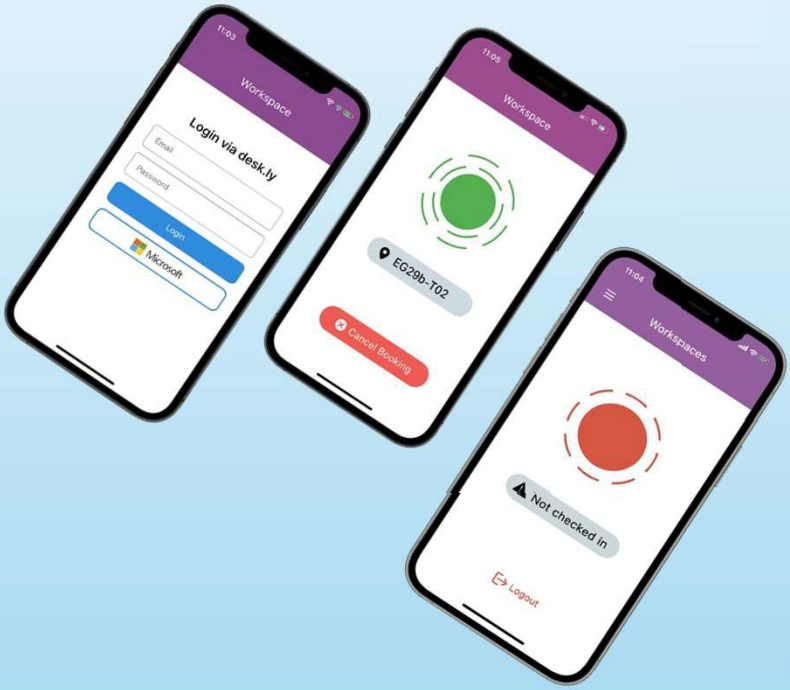
Automatische Platzübersicht
Teamorientiertes Projekt
Sommer ~~Semester~~ 2025

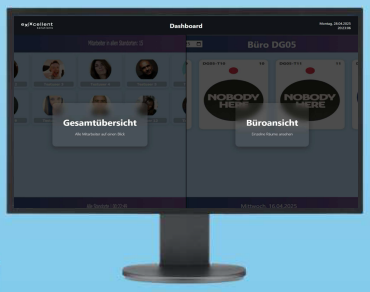
THU Technische Hochschule Ulm

exXcellent solutions

Projektziel

Scanne dein Handy an einem NFC- Tag und buche deinen Arbeitsplatz. Du wirst nun sowohl in der Etage als auch vor dem Büro in dem du Sitzt als anwesend angezeigt.





Features

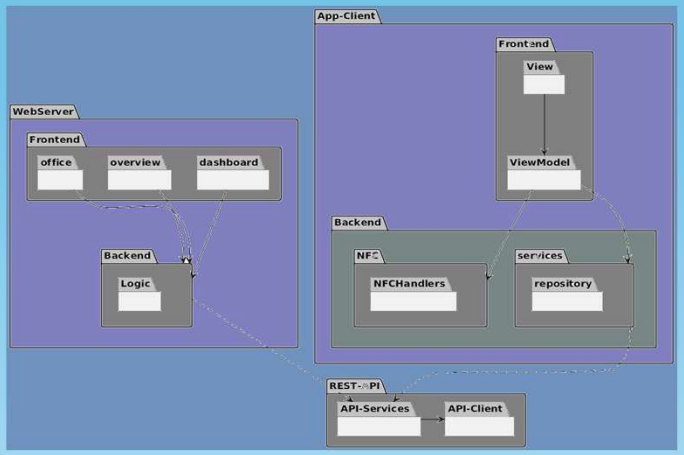
Web-Anwendung

- ✓ Standortfilterung
- ✓ Etagen-/URL-Filterung
- ✓ Bilder- und Namensgenerierung
- ✓ Dashboard
- ✓ Büro- und Gesamtansicht

Mobile-App Client

- ✓ Platzbuchung mittels NFC-Tag
- ✓ Zeigt an, ob (Grün/Rot) und an welchem Platz man gebucht hat

Architecture



Dart Flutter

Alton Bekolli / Felix Kussmann
Niklas Kümmel / Lovro Lupis
Lukas Jeckle / Christian Kasper

exlXcellent
solutions

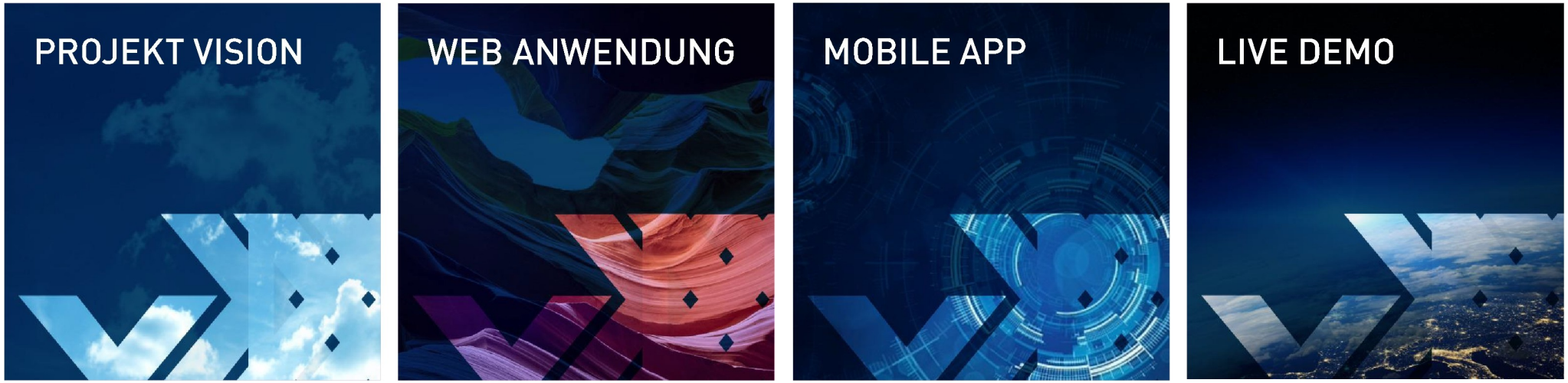
THU
Technische
Hochschule
Ulm

AUTOMATISCHE ARBEITSPLATZUBERSICHT

Teamorientiertes Projekt | Technische Hochschule Ulm

02.05.2025


ALTON BEKOLLI
FELIX KUSSMANN
NIKLAS KÜMMEL
CHRISTIAN KASPER
LOVRO LUPIS
LUKAS JECKLE


PROJEKT VISION
WEB ANWENDUNG
MOBILE APP
LIVE DEMO

PROJEKT VISION

ex|Xcellent solutions

ALLES BEGINNT MIT EINEM GEDANKEN, ALLES BEGINNT MIT EINER IDEE.

- THEODOR FONTANE

exlXcellent solutions

PROJEKT VISION
Automatische Arbeitsplatzübersicht

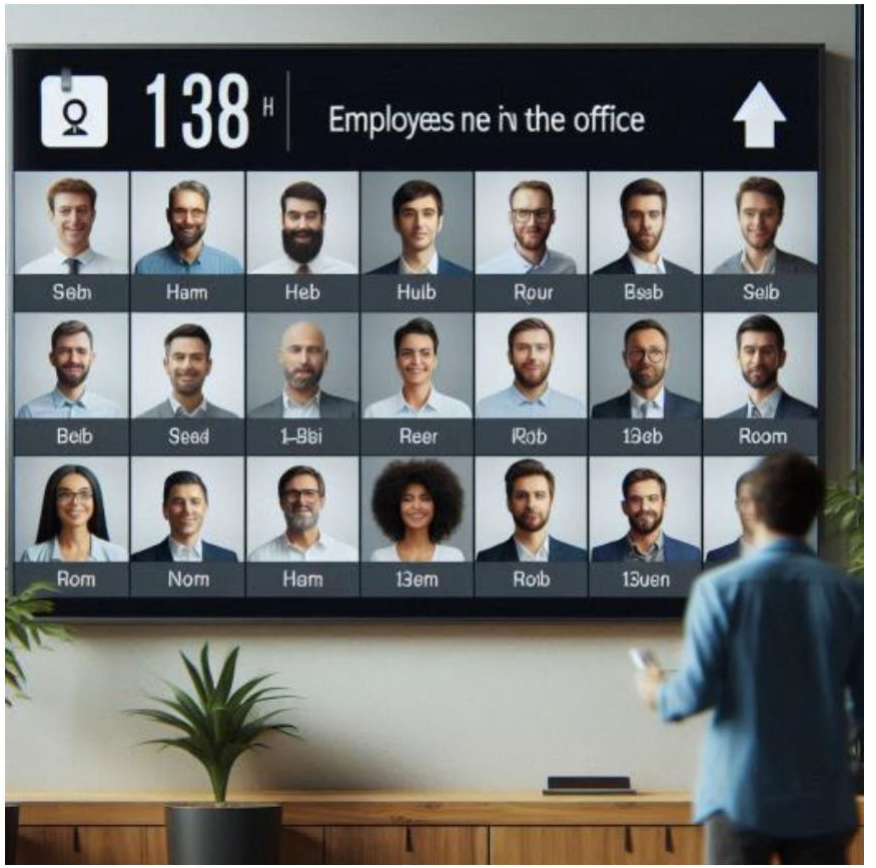
Projektvorgaben:

- Integration und Interaktion mit der bestehenden Raumbuchungssoftware desk.ly.
- Buchungsinformationen sollen klar und benutzerfreundlich visualisiert dargestellt werden.
- Arbeitsplatz Buchung soll so einfach wie möglich, am besten interaktionslos abgewickelt werden können.

Umsetzungsideen:

- Gesamtansichten
- Einzelne Büroübersichten
- Automatische Arbeitsplatzbuchungen

Impressionen



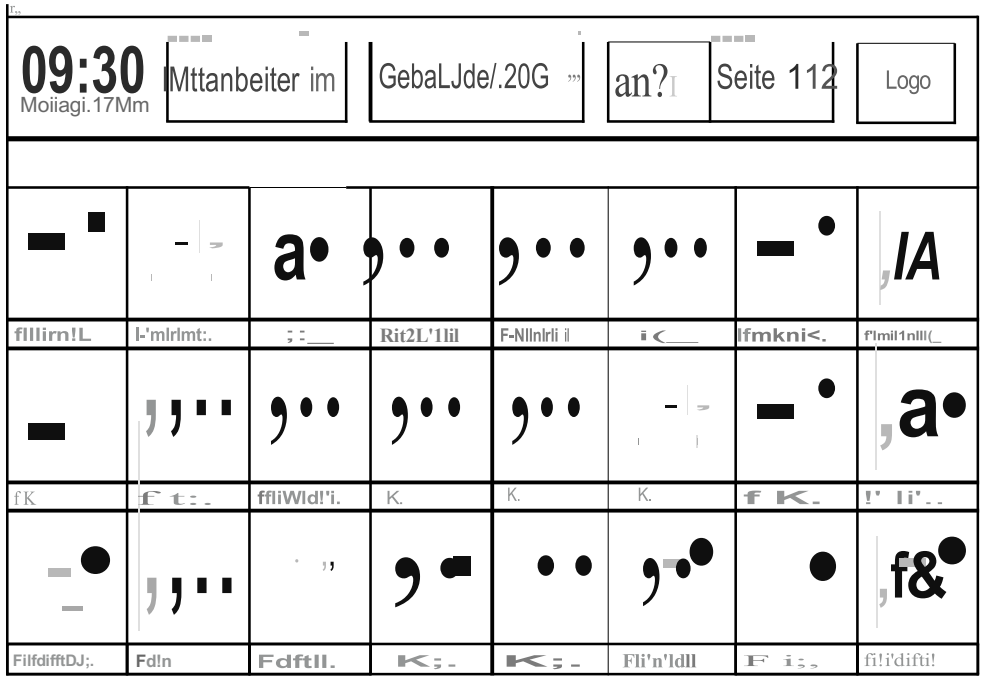
exXcellent solutions

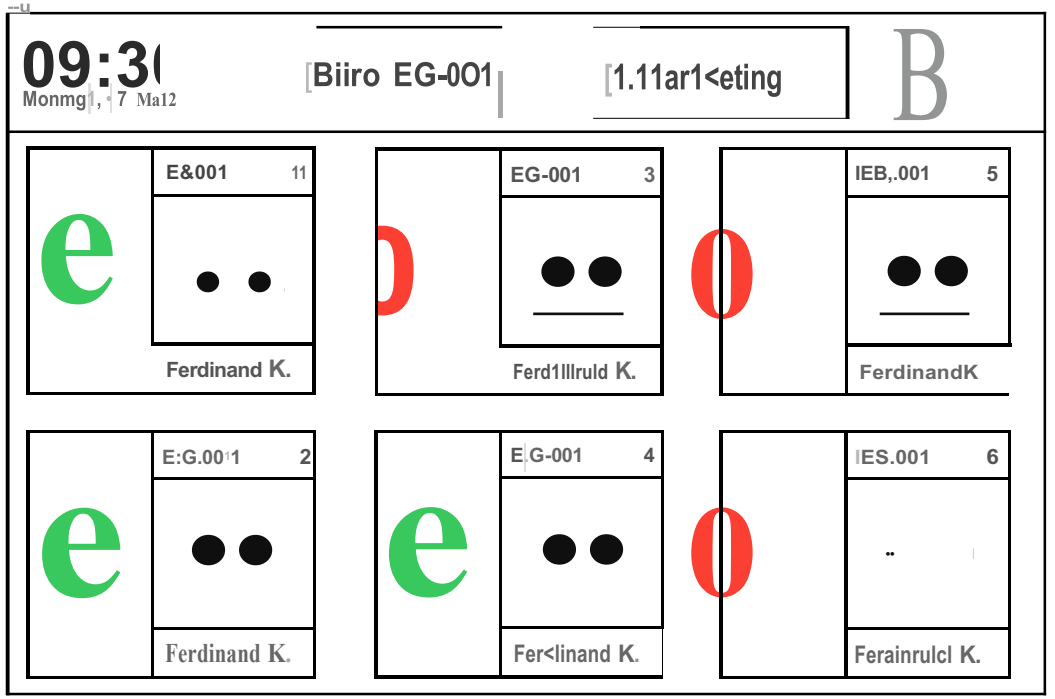
WEB ANWENDUNG

excellent solutions

WEB ANWENDUNG

UrsprUngliche Mockups





exIXcellent solutions

WEB ANWENDUNG

Oberflache – Gesamtübersicht, Büroansicht / Dash

Aus Datenschutzgründen wurde dieser Abschnitt entfernt. In der Originalversion waren hier Screenshots von dem jeweiligen Nutzer enthalten. Ein Austausch oder eine Veröffentlichung dieser Bilder ist derzeit leider nicht möglich.

WEB ANWENDUNG
Technische Umsetzung

1. Frontend

- Designsprachen:
  - HTML
  - CSS

- Programmiersprachen:
  - JavaScript

- Funktionen:
  - Befüllen der Dropdown-Menus
  - Anzeigen der Uhrzeit
  - Automatischer Seitenwechsel bei zu vielen Mitarbeitern
  - Neu laden der Seite nach best. Zeitintervall

Fig. 1. HTML & CSS Logo [1]

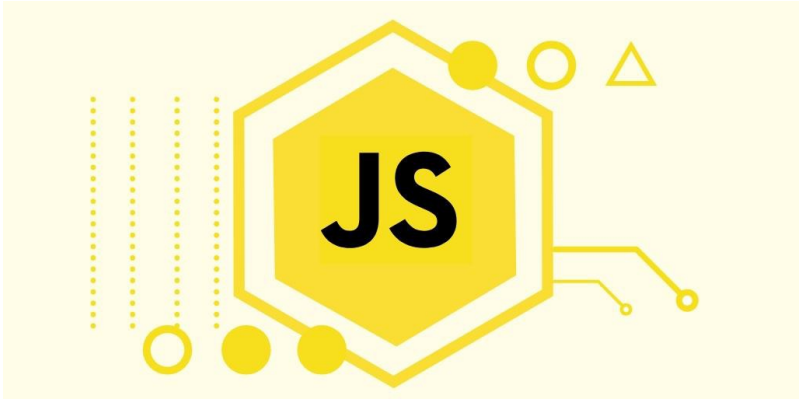


Fig. 2. JavaScript Logo [2]

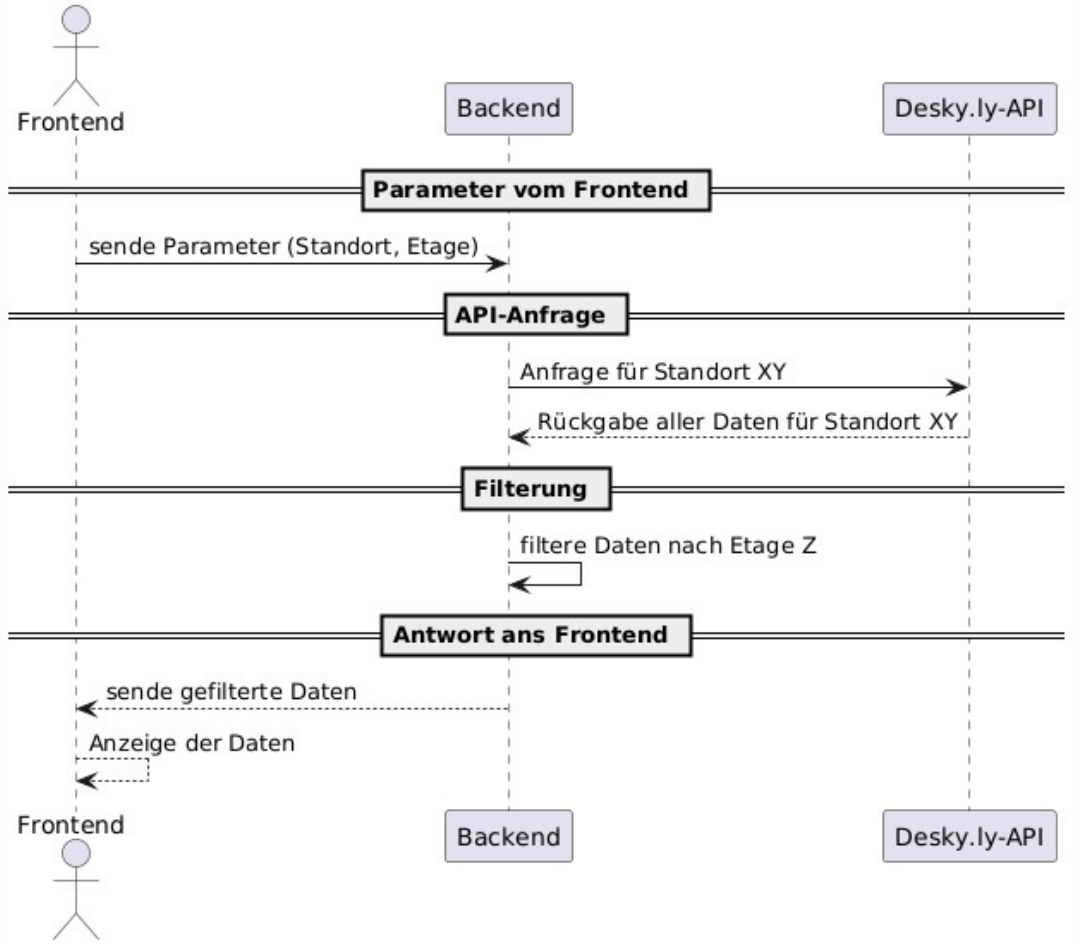
exXcellent solutions

WEB ANWENDUNG
Technische Umsetzung

2. Backend
   - Frameworks:
     - NodeJS
     - Express

Fig. 3. NodeJS + Express Logo [3]

3. Ablauf
   - Parameter für Standort und Etage werden aus dem Frontend übergeben
   - Anfrage zu dem Standort wird an desky.ly API gestellt, die Antwort wird nach Etage gefiltert
   - Finale Daten werden zurück ans Frontend gesendet und angezeigt



WEB ANWENDUNG
Technische Umsetzung

4. **Deployment:**
   - Docker

5. **Tests:**
   - Supertest
   - Jest

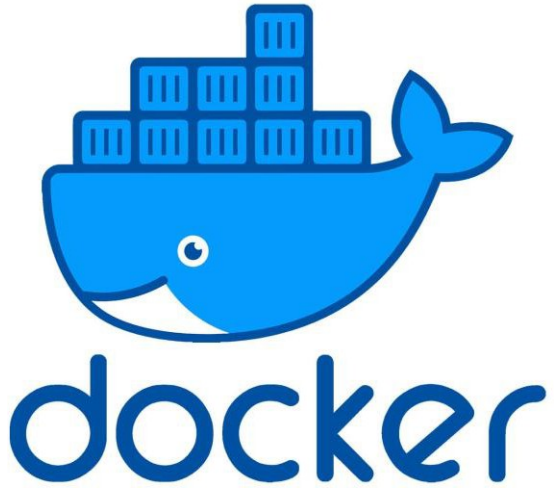


Fig. 4. Docker Logo [4]

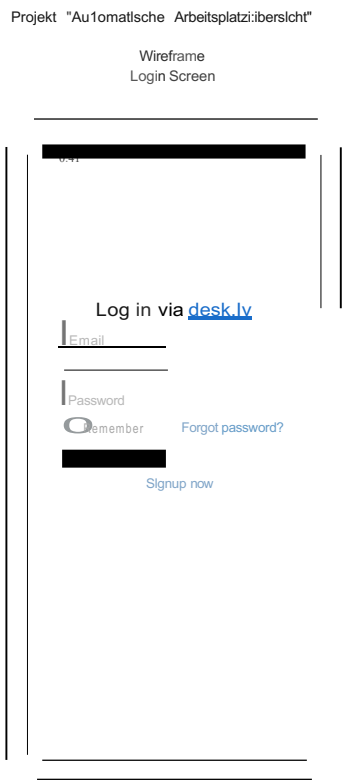
```
Test Suites: 1 passed, 1 total
Tests:       1105 passed, 1105 total
Snapshots:   0 total
Time:        215.454 s
Ran all test suites matching /tests\/app.test.js/i.
altonbekolli@MacBookAir deskly-webapp %
```

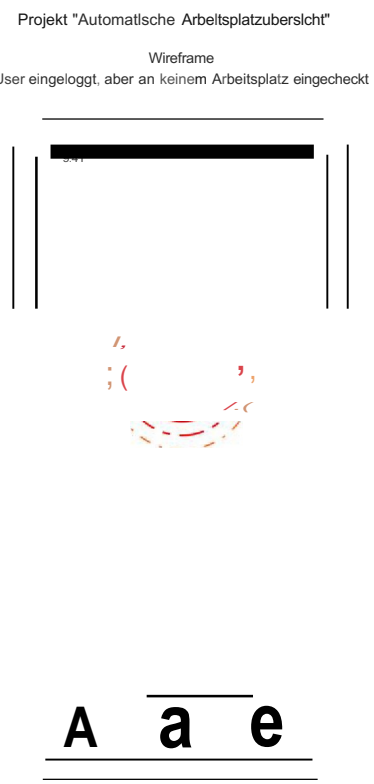
MOBILE APP

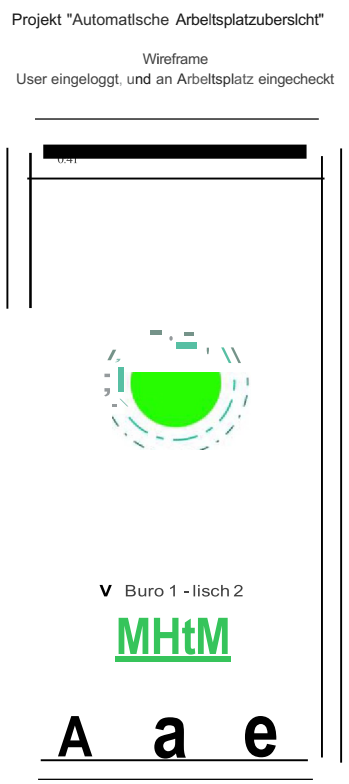
ex|Xcellent solutions

MOBILE APP

UrsprUngliche Mockups

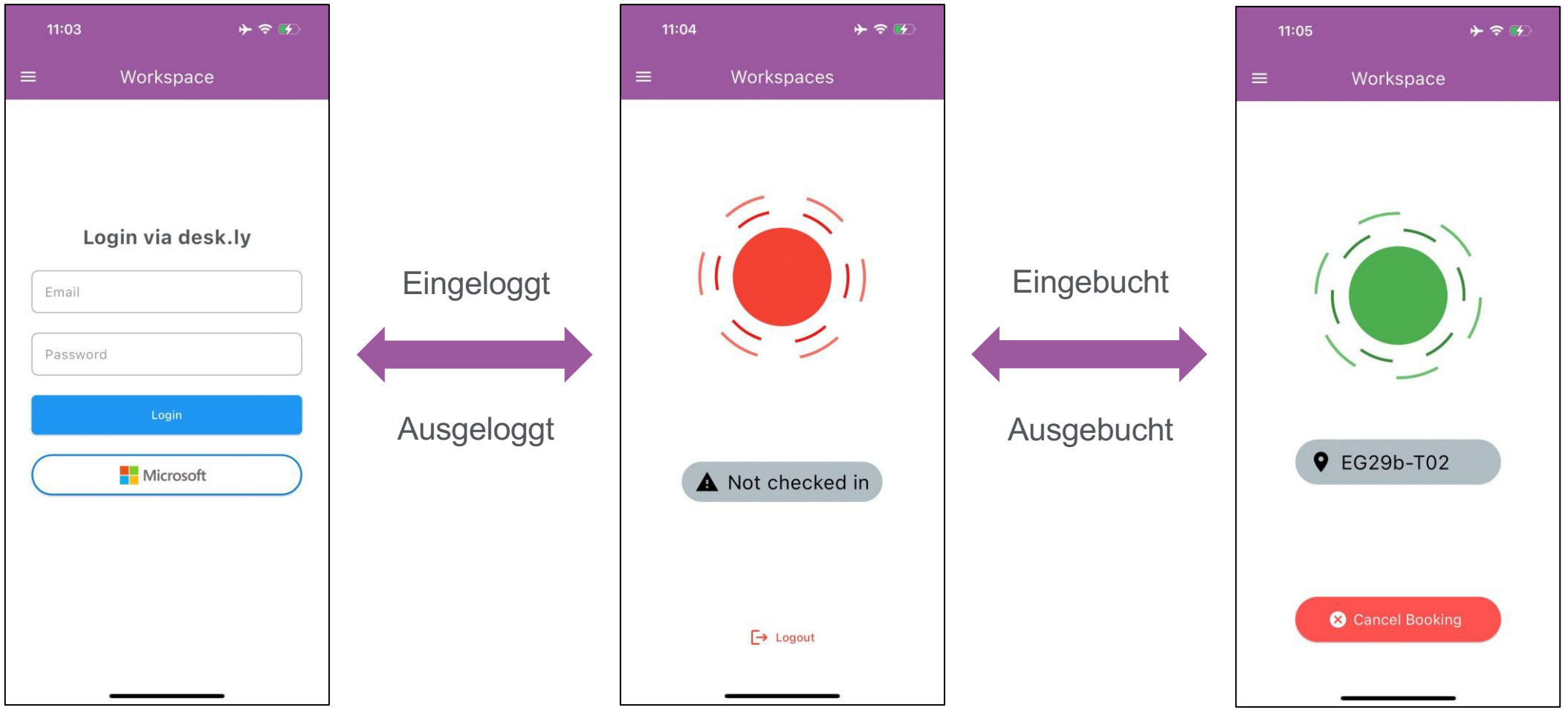






exIXcellent solutions

MOBILE APP
Oberflächen



exXcellent solutions

MOBILE APP
Technische Umsetzung

1. **Programmiersprachen:**
   - Dart
   - Kotlin

2. **Frameworks:**
   - Flutter

3. **Tools:**
   - Android Studio

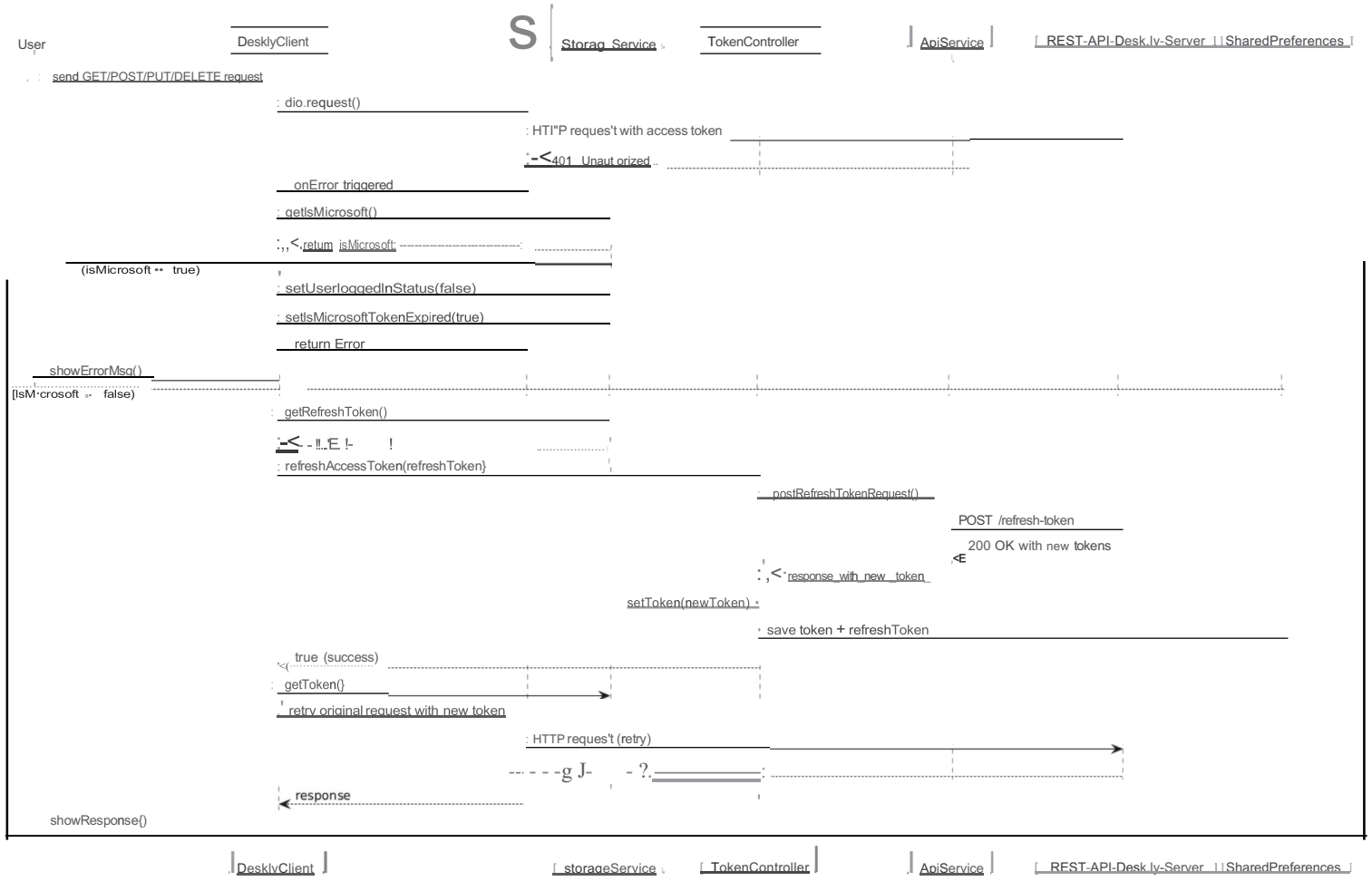
Fig. 5. Kotlin Logo [5]

Fig. 6. Dart Logo [6]

Fig. 7. Flutter Logo [7]

MOBILE APP

Autorisierung und Tokens

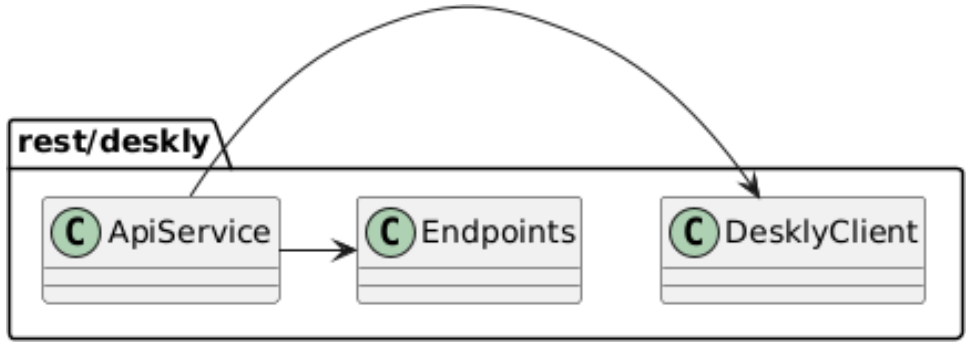


exIXcellent solutions

MOBILE APP
desk.ly API-Anbindung

1. **Verschiedene Clients für folgende Funktionen:**
   - Login
   - Buchungen (erstellen / löschen)
   - API-Anfragen, an der offiziellen API

2. **Workarounds verwendet für:**
   - Login
   - Buchungen (erstellen / löschen)



➢ desk.ly stellt zurzeit keine offiziellen API-Endpunkte für Login und Buchungen bereit.

MOBILE APP
NFC-Tags

1. Buchung mittels auslesen eines NFC-Tags:

- Die verwendeten Tags mussen NDEF unterstutzen.
- Am Gerat muss NFC aktiviert sein.

2. NFC-Tag Inhalt:

- Entspricht der desk.ly ..resource.id".

Beispiel:

f141b0b1-8795-4ae6-a6ab-7331c335ba90

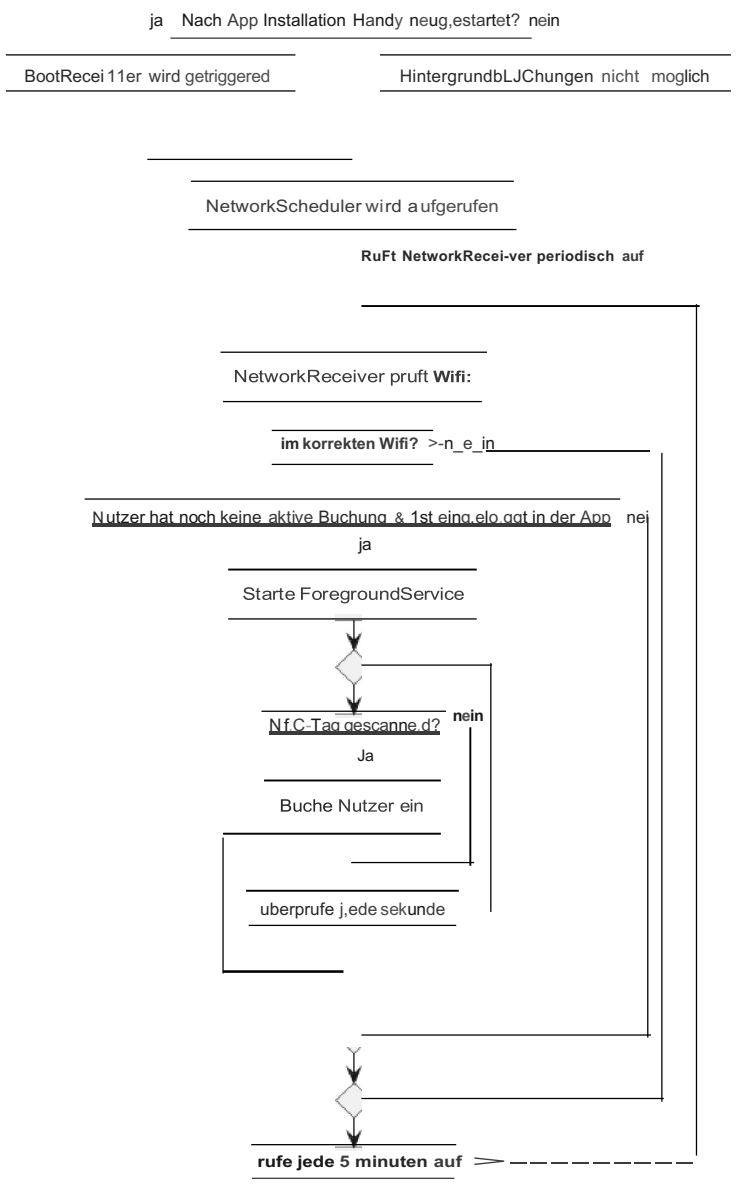
MOBILE APP
Buchungsarten

1. **Buchung innerhalb der App**
   - Default Buchungsart auf iOS
   - Unterstützt von Android und iOS

2. **Hintergrundbasierte Buchung**
   - Arbeitet via ForegroundService
   - Handy muss entsperrt sein
   - Nur von Android unterstützt, da iOS Hintergrund Dienste stark einschränkt.

MOBILE APP

Hintergrundbasierte Buchung



exIXcellent solutions

LIVE DEMO

exXcellent solutions

LIVE DEMO
Live Demo über Teams

Fig. 8. Live-Demo Logo [8]

QUELLEN

Bilder

[1] : https://externaL-content.duckduckgo.com/i u/?u=htt ps%3A%2F%2Fwww.nicepng.com%2Fpng%2FdetaiL%2F985-9857805_htmL5-css3-Logo-png-htmL-and-css-Logo.png&f=1 &nofb=1&ipt=e794647d76d683754a25f0f981aece996d7f7c2bba3054acba41f3b15e671776 (Last Accessed: 01.05.2025]

[2] : https:// externaL-eontent.duckduckgo.com/i u/?u=htt ps%3A%2F%2Fwww.dongee.com%2Ftutori aLes%2Fcontent%2Fimages%2F2022%2F10%2Fimage-83.png&f=1 &nofb=1&ipt=8edb0922b19595f59a4e55a83c486aa7f19035f79e4cf3d676f8e9594a13a01c (Last Accessed: 01.05.2025]

[3] : https://externaL-content.duckduckgo.com/iu/?u=https%3A%2F%2Fmiro.medium.com%2Fv2°/o2Fresize%3Afit⁰/o3A1400%2F1*f7ztMaMM0etsFHpEfkdiwA.png&f=1&nofb=1&ipt=5b6e5e773d8a946a49ddfa1041a4a52f9a66dec1a0d27ab8d0ad80c76859c2d5 (Last Accessed: 01.05.2025]

QUELLEN

Bilder

[4] : https:// externa l-content.duckduckgo.com/iu/?u=https%3A %2F%2Flogas-world.net%2Fwp-content%2Fuploads%2F2021%2F02%2FDocker-Symbol.png&f= 1&nofb=1&ipt=34530ff6fa35fd037a11717f4b3eafdb1a711cbbf6e8e55db80527511a9d261c (Last Accessed: 01.05.2025)

[5] : https:// external-content.duckduckgo.co m/i u/?u=https%3A %2F%2Flogowik.com%2Fcontent%2Fuploads%2Fimages%2 Fkotlin.jpg&f=1&nofb=1&ipt=ab7dd01dce0303a3e98d617ead3912b9a3ab9f9f4d3bc91637f12b852d81da27 (Last Accessed: 01.05.2025)

[6] : https://external-content.duckduckgo.com/iu/?u=https%3A%2F0/o2Feveryday.codes0/o2Fwp-content%2Fuploads0/o2F2019%2F11%2F0-nsblYn7PGj9YK3dB.png&f=1&nofb=1&ipt=444e2917a0a84c1453bca1d1e77bfc5ce8af86d805d4f42aec5eb6400f7c297d (Last Accessed: 01.05.2025)

QUELLEN

Bilder

[7] : https://external-content.duckduckgo.com/iu/?u=https%3A%2F%2Flogospng.org%2Fdownload%2Fflutter%2Fgoogle-flutter-4096.png&f=1 &nofb=1&ipt=937991 b79bb81 eb2cb8d696ee665b071b8c278df53a4c162902d6ebd2a08baca (Last Accessed: 01.05.2025)

[8] : https://static.vecteezy.com/system/resources/previews/019/163/657/non_2x/live-demo-icon-badge-label-icon-design-free-vector.jpg (Last Accessed: 01.05.2025)

QUELLENANGABEN

Sonstiges

PowerPoint Folienmaster: Wurde van den Betreuern van eXXcellent solutions per E-Mail zur Verfi..igung gestellt.

Link: Bilder der Agenda-Kacheln (Benotigt eXXcellent Microsoft Authentifizierung)

Link: Bilder der Kapitelfolien Hintergri..inde (Benotigt eXXcellent Microsoft Authentifizierung)