```
module: 36
title: Mobile-App (Capacitor-Wrapper)
phase: 1
status: detail-spec-draft
last_updated: 2026-05-08
session_source: brainstorming-session-2026-05-07-2206.md
related_lps: [LP-2, LP-8, LP-9, LP-10, LP-12, LP-13]
spec_style: requirements-pur
```

# Modul 36 — Mobile-App (Capacitor-Wrapper)

> ***Detail-Spec gemäß LP-9.*** *Native Mobile-App für iOS und Android als Capacitor-Wrapper um die mobile-first responsive Web-App.* ***Kein*** *echtes Native (LP-8). Pflicht-Native-Features (Push, Face-ID/Touch-ID, Kamera) erfüllen Apple-Guideline 4.2 ("Minimum Functionality").* ***Offline-Modus ist Phase-1-Pflicht*** *(User-Korrektur — kritisch für Baustellen-Use-Case). Apple-Compliance strikt: keinerlei Pricing/Upgrade-UI in der App (LP-8, Guideline 3.1.3(c) Enterprise-Services-Ausnahme).*

---

## Themenblöcke

```
A. Datenmodell (NotificationDevice [Verweis M22], MobileAppSession, OfflineSyncQueue)
B. Build-Strategie und Code-Sharing
C. Native-Features (Pflicht für Apple-Compliance)
D. Offline-Modus (Phase-1-Pflicht)
E. Sync-on-Reconnect (Strategien + Konflikt-Resolution)
F. Apple-Compliance (Pricing-Block, SIWA-Umgehung, Demo-Account, Privacy-Manifest)
G. Account-Löschung in der App (Verweis M35)
H. Permissions (App-Level, OS-Level)
I. Routes / Tiefen-Verlinkung (Deeplinks)
J. Update-Mechanik (Web-Code-Live + Native-Wrapper)
K. Submission-Strategie (Google Play zuerst, dann Apple)
L. Klartext-Hinweise / Fehler
M. Permissions-Anwendung (Tenant-Permissions in der App)
N. Phase 2 / spätere Features
O. Architektur-Vorschriften (Querschnitt)
```

---

## Block A — Datenmodell

### A.1 — `NotificationDevice` (in M22 vollständig spec'd)

Modul 36 ist **Konsument** dieser Entität, nicht Owner. Verweis auf Modul 22, Block A. Anforderungen aus Sicht der Mobile-App:

- Beim ersten Login der App auf einem Gerät: Push-Token holen (APNs-Token auf iOS, FCM-Token auf Android), via `/api/v1/notifications/mobile-device` registrieren
- Pro Tenant eigene Registrierung (User mit mehreren Tenants → mehrere Device-Einträge mit gleichem Push-Token, unterschiedlichem `tenant_id`)
- Bei Logout: Device wird in M22 revoked
- Bei App-Reinstall (neuer Push-Token): alter Eintrag wird automatisch `revoked` gesetzt (Trigger: APNs/FCM `InvalidToken`-Response — siehe M22 Block H)

### A.2 — `MobileAppSession`

Tenant-scoped (RLS aktiv). Erweitert die generelle `Session`-Entität aus Modul 27 um Mobile-spezifische Felder.

- `id` (UUID)
- `tenant_id`, `user_id`
- `session_id` (FK auf zentrale `Session`-Tabelle aus M27)
- `device_kind` (Enum: `ios | android`)
- `device_label` (vom OS oder User benannt)
- `os_version`
- `app_version` (Capacitor-Wrapper-Version, z.B. `1.4.2`)
- `web_bundle_version` (Web-Code-Version, kann durch Live-Update vom Wrapper abweichen)
- `biometric_enabled` (Boolean — User hat Face-ID/Touch-ID-Login aktiviert auf diesem Gerät)
- `biometric_kind` (Enum: `face_id | touch_id | android_biometric` — gemäß OS-Verfügbarkeit, NULL falls deaktiviert)
- `last_foreground_at` (Timestamp — wann war App das letzte Mal aktiv)
- `last_sync_at` (Timestamp — wann lief der letzte erfolgreiche Sync)
- `created_at`, `updated_at`

**Indizes:** `(tenant_id, user_id)`, `(session_id)` unique.

**Constraint:** Bei aktivem Biometrie-Login Speicherung des Auth-Secrets im **OS-Keychain** (iOS Keychain Services, Android Keystore) — kein Server-Persistenz des Secrets.

### A.3 — `OfflineSyncQueue` (lokal in IndexedDB, NICHT Cloud)

**Wichtig:** Diese Entität existiert ausschließlich client-seitig in IndexedDB. Wird in Phase 1 **nicht** auf den Server synchronisiert (keine Cross-Device-Resume-Funktion).

Pro Eintrag:
- `id` (lokale UUID, stabil bis erfolgreicher Sync)
- `tenant_id` (für Multi-Tenant-Trennung im lokalen Storage)
- `user_id`
- `created_at` (lokale Zeit)
- `entity_type` (Enum: `task | acceptance | photo | note | checklist_item | file_upload | activity_create`)
- `entity_id` (lokale UUID falls Append-Only-Create, oder Server-UUID falls Update)
- `operation` (Enum: `create | update | delete | append`)
- `payload` (JSON — vollständige Operation-Daten)
- `binary_blob_ref` (Lokal-Storage-Pfad zu Binary, z.B. Foto-File; NULL falls Text-Only)
- `status` (Enum: `pending | syncing | synced | failed | conflict`)
- `attempt_count` (Anzahl Sync-Versuche)
- `last_attempt_at`
- `last_error_message` (Klartext bei `failed`)
- `conflict_resolution_required` (Boolean — User-Sichtbarkeit nötig)
- `synced_server_id` (Server-UUID nach erfolgreichem Sync — für lokale ID-Auflösung)

**Indizes (IndexedDB):** `tenant_id + status`, `entity_type`, `created_at`.

**Lokale Aufbewahrung:**
- `synced`-Einträge: 7 Tage Aufbewahrung für User-Sichtbarkeit („*Diese 12 Änderungen wurden synchronisiert*"), dann lokal gelöscht
- `failed`-Einträge mit attempt_count >= 5: User-Aufmerksamkeit erforderlich (Sync-Fehler-Liste)
- `conflict`-Einträge: bleiben bis User-Resolution

### A.4 — Lokaler Cache (IndexedDB-Stores)

**Read-Cache pro Tenant:**

- `cache.assigned_projects` — heute zugewiesene Projekte (vorgeladen beim Login)
- `cache.assigned_tasks` — heute fällige + offene Tasks
- `cache.contacts` — Stammdaten (Kontakte mit Beziehung zu aktiven Projekten)
- `cache.checklists` — Checklisten-Definitionen + aktueller Stand
- `cache.acceptance_protocols` — Acceptance-Templates
- `cache.user_profile` — eigenes User-Profil + Notification-Preferences
- `cache.tenant_meta` — Tenant-Branding, Module-Verfügbarkeit, Permissions-Snapshot

**Cache-Metadaten pro Store:**
- `last_synced_at`
- `etag` (für Server-side Diff-Sync)
- `tenant_id` (für Multi-Tenant-Trennung)

**Cache-Größenlimit:** soft-Limit 100 MB pro Tenant; hard-Limit 250 MB Total. Bei Überschreitung: ältester nicht-essentieller Cache wird verworfen.

---

## Block B — Build-Strategie und Code-Sharing

### B.1 — Ein Code-Base

- **Web-App (RR7) ist Single Source of Truth.** Capacitor-Wrapper ist dünner Native-Container.
- Web-App ist mobile-first responsive (PWA-fähig)
- Capacitor packt die Web-App für iOS/Android mit nativem Hülle drumherum
- Native-Plugins werden via Capacitor-API aufgerufen, abstrakte Wrapper im Web-Code (`@/services/native/...`)
- Im Browser: Native-Wrapper geben `null` oder Polyfill zurück; im Capacitor-Container: echter Plugin-Aufruf

### B.2 — Build-Tooling

- **Capacitor CLI** als primäres Build-Tool
- Web-Build wird in Capacitor-Asset-Verzeichnis kopiert
- iOS-Build: Xcode-basiert, Apple-Developer-Signing-Zertifikat
- Android-Build: Android-Studio (oder Gradle-CLI), Signing-Keystore
- Build-Konfiguration:
- `capacitor.config.ts` (App-ID, App-Name, Web-Server-URL für Live-Reload im Dev, Plugins-Liste)
- `Info.plist` (iOS: Permission-Strings für Kamera, Face-ID, Push)
- `AndroidManifest.xml` (Android: Permissions, Intent-Filter für Deeplinks)
- `PrivacyInfo.xcprivacy` (iOS: Privacy-Manifest — siehe Modul 35 Block I)

### B.3 — CI/CD

- **Auto-Build bei Releases:**
- Tag `mobile-vX.Y.Z` triggert Multi-Pipeline
- Web-App-Build → Asset-Bundle
- iOS-Pipeline: Build, Sign, TestFlight-Upload
- Android-Pipeline: Build, Sign, Google-Play-Internal-Testing-Upload
- **Web-only-Releases:** triggern keinen Native-Build, sondern nur Live-Update (siehe Block J)
- **Versionierung:**
- `app_version` (Native-Wrapper-Version, semantic versioning, im App-Store sichtbar)
- `web_bundle_version` (Web-Code-Version, separat)

- `min_supported_app_version` (Server-side gepflegt — bei zu alter App: Force-Update-Screen)

### B.4 — Lokal-Dev

- Capacitor-CLI Live-Reload (Web-App läuft auf Dev-Server, Capacitor-Container lädt von dort)
- Push-Notifications mit Test-Tools (APNs-Tester, FCM-Console)
- Biometrie-Test im iOS-Simulator (Face-ID-Simulation) und Android-Emulator

---

## Block C — Native-Features (Pflicht)

### C.1 — Push-Notifications

**Hintergrund:** Apple-Compliance (Guideline 4.2 — minimaler Native-Mehrwert) und kritischer UX-Bedarf (Monteur bekommt Task-Zuweisung sofort).

**Anforderungen:**
- Capacitor-Push-Plugin (`@capacitor/push-notifications` oder gleichwertig)
- Beim Login auf Mobile-Device:
- User-Pre-Prompt-Erklärung *„HiveGrid sendet dir Benachrichtigungen, wenn dir eine Aufgabe zugewiesen wird oder wichtige Updates passieren — möchtest du das aktivieren?"* mit Funktionen *„Ja, aktivieren"* und *„Später"* (Apple-Compliance LP-8: Pre-Prompt vor System-Dialog)
- Bei *„Ja"*: nativer Permission-Dialog
- Bei *„Erlaubt"*: Token holen, an `/api/v1/notifications/mobile-device` senden (siehe M22 Block H)
- Push-Empfang:
- App im Vordergrund: visualisiert in App (Toast oder In-App-Banner)
- App im Hintergrund: OS-Notification-Center
- Klick auf Push: Deeplink in App (siehe Block I)
- Bei Permission-Denied: Klartext-Hinweis im `/settings/notifications/` (*„Push ist auf diesem Gerät deaktiviert. [Geräte-Einstellungen öffnen]"*)
- iOS: Notification-Permission ist Pflicht-Permission ab iOS 13 (Apple-konform)
- Android: ab API 33 ähnlicher Permission-Flow nötig
- Versand-Pipeline: Server-Side, Modul 22 — Mobile-App ist nur Empfänger

**Provider:** Novu (Modul 22 Block J, EU-Stack LP-13).

### C.2 — Face-ID / Touch-ID Login

**Anforderungen:**
- Capacitor-Plugin: `capacitor-biometric-auth` (oder gleichwertig — wrappt iOS `LocalAuthentication.framework` und Android `BiometricPrompt`)
- **Setup-Flow** in `/{tenant}/settings/security/biometric/`:
- Toggle *„Face-ID/Touch-ID-Login auf diesem Gerät aktivieren"*
- Bei Aktivieren: User muss aktuelles Passwort eingeben → wird im OS-Keychain mit Biometrie-Schutz hinterlegt
- Klartext-Hinweis: *„Beim nächsten Login auf diesem Gerät kannst du dich mit Face-ID statt Passwort anmelden. Funktioniert nur auf diesem Gerät."*
- 2FA bleibt **trotzdem aktiv** (Biometrie ersetzt nur Passwort, nicht 2FA — siehe Modul 27 Block B.4)
- **Login-Flow:**
- App-Start: erkennt registriertes Biometrie-Login → Bietet *„Mit Face-ID anmelden"* an
- Bei Erfolg: Auth-Secret aus OS-Keychain → reguläre Session-Erstellung wie Pwd-Login
- Bei Biometrie-Fehler 3x in Folge: Fallback auf Passwort-Eingabe
- Bei Biometrie-Lockout (OS-Level): Klartext-Hinweis + Fallback
- **Deaktivieren:**
- Toggle in Settings ausschalten → Auth-Secret aus Keychain löschen

- Bei User-Logout (regulär): Biometrie-Token bleibt für nächsten Login (User-Convenience)
- Bei User-Logout (forced — Pwd-Reset, 2FA-Reset etc.): Auch Biometrie-Token entfernt
- **Pflicht-Strings:**
- iOS `Info.plist`: `NSFaceIDUsageDescription` — Klartext: *„HiveGrid nutzt Face-ID für schnellen, sicheren Login auf deinem Gerät — dein Passwort bleibt nur in dir und im sicheren Apple-Speicher."*
- Android: `useBiometric` Permission

### C.3 — Kamera (Foto-Upload)

**Use-Cases:**
- Foto-Upload für Acceptance-Protokoll (Modul 26)
- Foto-Checklisten (Modul 4 / 26)
- Profilbild-Upload (Modul 27)
- Lead-Beleg-Foto (Modul 2)
- Inbox-Anhang-Foto (Modul 9)

**Anforderungen:**
- Capacitor-Plugin: `@capacitor/camera`
- Erst-Use-Permission-Pre-Prompt: *„HiveGrid braucht Zugriff auf deine Kamera, um Fotos für Acceptance-Protokolle und Checklisten aufzunehmen — möchtest du das erlauben?"*
- Foto-Capture-Optionen:
- Direkt aufnehmen (Camera)
- Aus Galerie auswählen (Photo-Picker)
- User-Wahl pro Use-Case oder kombiniert
- Foto-Verarbeitung **lokal** vor Upload:
- Resize (max. 2400x2400, JPEG-Qualität 85)
- EXIF-Daten beibehalten (für Geo-Tag wenn aktiviert) — Phase 2 Geo
- Optionale Komprimierung bei großen Dateien
- Upload-Strategie:
- Online: direkter Upload via `/api/v1/files/upload` (Modul 23)
- Offline: Foto in Lokal-Storage, OfflineSyncQueue-Eintrag mit `binary_blob_ref` (siehe Block D)
- Bei Permission-Denied: Klartext im UI (*„Kamera-Zugriff verweigert — bitte in den Geräte-Einstellungen erlauben."*)
- Pflicht-Strings:
- iOS `Info.plist`: `NSCameraUsageDescription`, `NSPhotoLibraryUsageDescription`
- Android: `CAMERA`-Permission, `READ_MEDIA_IMAGES` (API 33+)

---

## Block D — Offline-Modus (Phase-1-Pflicht)

> ***User-Korrektur:*** *Offline-Modus ist DRINGEND in Phase 1, nicht Phase 2. Kritisch für Baustellen-Use-Case (Monteur ohne Empfang).*

### D.1 — Architektur-Komponenten

- **Service-Worker** (Workbox-basiert oder gleichwertig):
- Cached statische Web-App-Assets (HTML/JS/CSS/Icons)
- Offline-Fallback-Page bei Navigation auf nicht-gecachte Routen
- Cache-First-Strategie für Static, Network-First für API-Calls (mit Cache-Fallback)
- **IndexedDB** (via Dexie.js oder gleichwertige Abstraktion):
- Read-Cache (siehe A.4)
- OfflineSyncQueue (siehe A.3)
- Binary-Storage für Fotos (Blob-Storage im IDB)

- **Network-Status-Detection:**
- Capacitor-Plugin `@capacitor/network` für native OS-Level-Erkennung
- Browser-API `navigator.onLine` als Fallback
- Reachability-Check (Server-Ping alle 30s im Online-Modus, alle 5s wenn unsicher)

## D.2 — Pre-Caching beim Login

**Beim Login (online) wird vorgeladen:**

- **Heutige Aufträge / Projekte** für eingeloggten User:
- Alle Projekte, in denen User Member ist UND die heute aktiv sind (Status `active`, Schedule-Eintrag ≤ heute+1)
- **Heute fällige Tasks** des Users
- **Stammdaten:**
- Kontakte (verknüpft mit aktiven Projekten)
- Tenant-Branding, Permissions-Snapshot, User-Profil
- **Acceptance-Templates** des Tenants
- **Foto-Checklisten** für aktive Projekte

**Pre-Cache-Klartext beim Login:** *„Lade Daten für offline-Nutzung — bitte einen Moment Geduld..."* mit Progress-Indikator (Anzahl geladener Entitäten / Gesamt).

**User-Konfiguration:** in `/{tenant}/settings/mobile/` Toggle *„Daten für Offline-Nutzung vorladen"* (Default: an für Mobile-User, aus für Büro-User — Heuristik nach UI-Modus aus Modul 27).

## D.3 — Offline-Read

**Was funktioniert offline:**

- Anzeige der vorgeladenen Daten (Projekte, Tasks, Kontakte, Checklisten, Acceptance-Templates)
- Navigation in App (alle Routen, die im Cache sind)
- Anzeige von Fotos in Cache
- Lesen eigener offline-erzeugter Einträge

**Was nicht offline funktioniert:**

- Suche über nicht-gecachte Daten (z.B. CRM-Vollsuche) — Klartext: *„Suche nicht verfügbar offline. Wieder verbunden? [Erneut versuchen]"*
- Server-side-Renderings (Reports, KI-Antworten)
- Email-Inbox (Phase 2 erweitert)
- Chat-Real-Time (Read von gecachten Threads ja; senden geht in Queue)

**Klartext-Indikator:** prominente Sichtbarkeit des Offline-Status in der App (z.B. Banner oder Status-Marker — Designer-Wahl, kein Layout-Vorschlag).

## D.4 — Offline-Write-Queue

**Folgende Operationen können offline ausgeführt werden, landen in OfflineSyncQueue:**

| Operation | Modul | `entity_type` | Konflikt-Strategie |
|---|---|---|---|
| Foto-Upload zu Acceptance | 26 | `photo` | Append-only, kein Konflikt |
| Foto-Upload zu Projekt-Files | 23 | `photo` | Append-only |
| Foto-Upload zu Task | 4 | `photo` | Append-only |
| Note schreiben | div. | `note` | Append-only |
| Task-Status ändern (Open → In Progress → Done) | 4 | `update` | Last-Write-Wins mit User-Sichtbarkeit |
| Checklist-Item abhaken | 4/26 | `checklist_item` | Last-Write-Wins (selten Konflikt) |
| Acceptance-Protokoll erstellen | 26 | `create` | Append-only |
| Acceptance-Protokoll abschließen | 26 | `update` | Last-Write-Wins, mit Konflikt-Anzeige |
| Activity erzeugen (manueller Eintrag) | 5 | `activity_create` | Append-only |
| Termin-Status setzen (Erschienen/Nicht-Erschienen) | 6 | `update` | Last-Write-Wins |
| Datei-Anhang | 23 | `file_upload` | Append-only |
| Chat-Nachricht in Projekt-Channel | 13 | `create` | Append-only mit Reihenfolge-Garantie pro Author (siehe E.4) |

**Was bewusst NICHT offline:**

- Anlegen neuer Kontakte / Companies (CRM-Stammdaten — zu hohes Konflikt-Risiko)
- Lead-Bearbeitung (Vertriebs-Use-Case ist online-zentriert)
- Permissions-/User-Aktionen (Sicherheits-relevant)
- Email-Send (Phase 2)
- Booking-Bestätigungen (Phase 2)
- Automation-Trigger (zu komplex für Phase 1)

## D.5 — Sync-Indikator UI

**Benötigte UI-Elemente:**

- Sichtbarer Badge mit Anzahl pending Queue-Einträge (z.B. *„3 Änderungen warten auf Upload"*)
- Klick öffnet Queue-Übersicht mit:
- Liste aller pending Einträge (entity_type, kurze Zusammenfassung, created_at)
- Pro Eintrag: Status-Marker ( `pending | syncing | failed | conflict` )
- Bei `failed` : Klartext-Fehlermeldung + manueller Retry-Button
- Bei `conflict` : Resolution-Funktion (siehe Block E)
- Globaler Sync-Status (Online + Synced / Online + Syncing / Offline / Sync-Fehler)
- Force-Sync-Funktion (User-getriggert)

**Klartext:**

- Online + Queue leer: *„Alle Änderungen synchronisiert"*
- Online + Syncing: *„Synchronisiere 3 Änderungen..."*
- Offline: *„Du bist offline. Deine Änderungen werden gespeichert und später hochgeladen."*
- Sync-Fehler: *„2 Änderungen konnten nicht hochgeladen werden. [Details anzeigen]"*

---

# Block E — Sync-on-Reconnect

## E.1 — Trigger

- Network-Status-Wechsel: `offline → online` (über Capacitor `@capacitor/network`)
- App-Foreground-Resume (User öffnet App nach Hintergrund)
- Manueller Force-Sync vom User
- Periodisch alle 60s wenn online + Queue nicht leer
- Beim regulären Login

## E.2 — Sync-Reihenfolge

1. **Read-Cache-Refresh** (etag-basiert): aktualisiere lokale Stores für aktive Module
2. **Write-Queue-Drain** in `created_at`-Reihenfolge:
   - Pro Queue-Eintrag: API-Call mit Idempotency-Key (lokale UUID)
   - Bei Erfolg: `synced_server_id` schreiben, Status `synced`
   - Bei Server-Fehler 5xx oder Timeout: Retry mit exponentiellem Backoff (max. 5 Versuche, dann `failed`)
   - Bei Server-Fehler 409 (Konflikt): Status `conflict`, User-Resolution erforderlich
   - Bei Server-Fehler 4xx (Validation, Permission etc.): Status `failed`, Klartext-Fehlermeldung
3. **Read-Cache-Refresh** post-Sync (eigene Änderungen sind jetzt server-side, lokaler Cache wird konsistent)

## E.3 — Konflikt-Strategien (zentral)

**Append-only-Daten** (Fotos, Notes, neue Activities, neue Chat-Messages, Acceptance-Protokoll-Erstellung):
- **Keine Konflikte möglich** (jeder Eintrag wird neu angelegt mit eigener UUID)
- Eindeutigkeit garantiert durch Idempotency-Key (lokale UUID)
- Bei Doppel-Sync (Network-Hick-Up): Server erkennt Idempotency-Key, antwortet mit existierendem Server-ID

**Status/Felder-Updates** (Task-Status, Checklist-Items, Acceptance-Abschluss, Termin-Status):
- **Last-Write-Wins** mit User-Sichtbarkeit
- Bei Sync: Server vergleicht lokale `updated_at` mit Server-`updated_at`
- Wenn Server-Stand neuer als lokale Basis → Konflikt-Marker setzen, lokale Änderung wird aber **trotzdem übernommen** (Last-Write-Wins)
- User bekommt **eigene Konflikt-Resolution-Anzeige** (siehe E.4) mit Klartext: *„Es gab einen Konflikt — deine Änderung wurde übernommen, der vorherige Stand war: [Klartext]. [OK] [Vorherigen Stand wiederherstellen]"*
- Audit-Trail dokumentiert beide Stände (siehe Modul 21 + 5)

## E.4 — Konflikt-Resolution-Anzeige (für seltene Konflikt-Fälle)

**Benötigte UI-Elemente:**

- Liste der Konflikte mit Anzahl-Indikator
- Pro Konflikt:
- Entity-Typ + Subject-Klartext (z.B. *„Aufgabe ‚Ventil tauschen' — Status"*)
- Lokaler Stand (was wurde offline geändert)
- Server-Stand (was wurde parallel von wem geändert) inkl. Actor + Timestamp
- Auflösungs-Funktionen: *„Meinen Stand behalten"*, *„Server-Stand übernehmen"*, *„Manuell zusammenführen"* (Phase 2)

**Default bei No-User-Action nach 7 Tagen:** Lokaler Stand bleibt (Last-Write-Wins-Default), Konflikt-Marker wird automatisch entfernt.

**Chat-Message-Reihenfolge:**
- Innerhalb gleichem Author: lokale Reihenfolge bleibt (created_at-Order)
- Cross-Author: Server-Reihenfolge nach Server-Ankunft (kein Konflikt-Recovery — Chat ist append-only)

### E.5 — Edge-Cases

- **Sync läuft, App wird geschlossen:** Sync wird beim nächsten App-Start fortgesetzt (Status `syncing` → `pending` Reset)
- **Sync-Token expired during long offline:** Auth-Refresh nötig vor Sync-Drain; bei Refresh-Fehlschlag → Re-Login erforderlich, Queue bleibt erhalten bis Login
- **Tenant-Wechsel während Queue nicht leer:** Klartext-Warnung *„3 Änderungen für Tenant X warten noch — bitte erst synchronisieren oder bewusst verwerfen."*
- **App-Update während Offline-Queue nicht leer:** Queue-Schema bleibt rückwärts-kompatibel; Migration falls nötig (Phase 2: konkrete Migrations-Strategie)
- **Permission-Wechsel server-side während Offline:** lokale Aktion wird beim Sync abgelehnt mit 403 → Klartext im Failed-Eintrag (*„Du hast keine Berechtigung mehr für diese Aktion. [Eintrag verwerfen]"*)

### E.6 — Storage-Limits + Cleanup

- Lokale Queue: max. 1.000 Einträge — danach Klartext *„Lokale Warteschlange voll — bitte mit Internet verbinden, um zu synchronisieren."*
- Binary-Storage (Fotos): max. 500 MB lokal — bei Überschreitung Klartext, blockiert weiteren Upload bis Sync
- Sync-Erfolg: lokale Foto-Datei nach erfolgreichem Server-Empfang lokal gelöscht (Cache wird Server-Pfad)

---

## Block F — Apple-Compliance

### F.1 — Pricing/Upgrade-UI strikt verboten

**Konkrete Verbote in der App** (LP-8, Apple Guideline 3.1.3(c) Enterprise-Services-Ausnahme):

- **Keine** Pricing-Page
- **Keine** *„Upgrade auf Cloud+"*- oder *„Auf Pro wechseln"*-Buttons
- **Keine** Links auf `hivegrid.de/pricing`, `app.hivegrid.de/admin/billing` etc. (Subscription-Self-Service nur via Web-Browser auf Desktop)
- **Keine** Modals mit Klartext *„Diese Funktion ist im Cloud+-Plan verfügbar — jetzt upgraden"*

**Stattdessen:** Klartext bei Feature-Gate-Block in der App: *„Diese Funktion ist im Cloud+-Plan verfügbar. Bitte deinen Workspace-Admin kontaktieren oder hivegrid.de besuchen."* — **kein** Klick-Link.

**Tenant-Subscription-Status anzeigen** ist erlaubt (read-only Information):
- Status-Badge (Trial / Active / Frozen) — ohne Action
- Trial-Verbleibend-Text — ohne CTA-Button

**Implementierung:** In M28-Subscription-Modul gibt es einen Build-Flag `IS_MOBILE_APP=true` der alle Pricing-/Upgrade-UI ausblendet (vom Capacitor-Wrapper gesetzt).

### F.2 — Sign-in-with-Apple-Umgehung

**Strategie:** Login-only-Framing als „Enterprise SSO".

- Login-Screen-Text: *„Mit deinem HiveGrid-Firma-Account anmelden"* (Klartext positioniert als Enterprise-Login)
- App-Store-Listing: Kategorie *„Business"* mit Beschreibung als *„Enterprise-CRM für Handwerk"*
- App-Review-Notes: Klartext *„This is an enterprise B2B CRM application. Authentication is exclusively via the customer organization's account, per Guideline 4.8 enterprise/business apps exception."*
- **Phase 1 nur Email/Pwd + Magic-Link** (umgeht SIWA-Trigger automatisch — siehe Modul 27 B.1, B.2)
- **Microsoft-OAuth ist Phase 2 backloggt** (siehe Modul 27 B.3) — wenn implementiert: als „Enterprise SSO" in App-Login-Screen klar gekennzeichnet, **kein** generisches Drittanbieter-Login

**Falls Reviewer SIWA-Pflicht durchsetzt** (Reject-Risiko):
- Fallback-Implementation: SIWA als zusätzliche Login-Option vorbereitet (~1 Tag Aufwand) — in Phase 1 nicht aktiv, kann bei Reject schnell eingebaut werden

## F.3 — Reviewer-Demo-Account

**Pflicht für App-Submit (40% aller Rejects sind „Reviewer kommt nicht rein").**

- Dedizierter Tenant `apple-review` mit Mitarbeiter-Account
- Beispieldaten: 5-10 Kontakte, 2-3 aktive Projekte, ein paar Tasks, Beispiel-Acceptance-Protokoll
- Demo-Permissions: Vertriebler-Rolle (deckt die meisten Funktionen ab)
- Login-Credentials in App-Review-Notes:
- Tenant: `apple-review`
- Email: `reviewer@hivegrid.de`
- Pwd: stabiles Test-Pwd (rotiert nicht zu oft, Audit-Eintrag bei Login)
- Demo-Account ist **read-only-write** für Reviewer (nicht öffentlich exponiert)
- Plattform-Admin pflegt den Demo-Tenant in Modul 37
- Klartext-Hinweis in App-Review-Notes: *„Test credentials provided. The app is a B2B SaaS CRM — login required by enterprise organization. Test data is in the apple-review tenant."*

## F.4 — Privacy-Manifest

**Verweis auf Modul 35 Block I — Implementations-Detail liegt in Modul 36:**

- `PrivacyInfo.xcprivacy` wird bei jedem Native-Build aus Master-Liste (Modul 37) generiert
- Pflicht-Inhalt:
- Datentypen: User-Account-Info (Email, Name), User-Content (Notes, Photos, Activities), Identifiers (Device-Token, User-ID), Diagnostics (Crash-Logs, Performance), Usage-Data (Feature-Usage)
- Tracking-Domains: keine Drittanbieter-Tracker in Phase 1 (LP-13 EU-Stack)
- Required-Reason-APIs: `UserDefaults` (für lokale Settings), `FileTimestamp` (für Sync-Logik), `SystemBootTime` (Phase 2 falls genutzt)
- Klartext-Beschreibungen pro Datentyp im Apple-Format

**Android Data-Safety-Form:** analog für Google Play Console.

## F.5 — Account-Löschung in der App (Apple-Pflicht 2022)

**Verweis Modul 35 Block E + Modul 27 D.3** — komplette Spec dort. Mobile-App-spezifische Anforderungen:

- Settings-Route `/{tenant}/settings/account/delete` ist in Mobile-App **identisch** sichtbar wie im Web (gleicher Code-Pfad)
- Multi-Step-Bestätigung funktioniert in der Mobile-App vollständig (keine Web-Redirect-Workarounds)
- Bei Löschung in Mobile-App:
- Lokale Daten werden komplett gelöscht (IndexedDB-Stores leeren, OS-Keychain-Einträge entfernen, Push-Token revoken via M22)
- Biometrie-Login deaktiviert
- App startet mit leerem Login-Screen
- Apple-Compliance-Check: Settings-Screen mit *„Konto löschen"* auf gleicher Ebene wie *„Profil"* sichtbar (Apple-Strenge: Reviewer prüft Sichtbarkeit)

---

## Block G — Account-Löschung in der App (Verweis M35)

Siehe F.5. Vollständige Spec in Modul 35 Block E und Modul 27 Block D.3. Mobile-spezifisch:

- Pre-Lösch-Optional: User-Daten-Export anbieten (Modul 35 Block C) — auf Mobile kann ZIP zu groß sein, Klartext: *„Export wird per Email an dich gesendet — Download bitte am Computer."*
- Lokal-Cleanup nach Lösch-Bestätigung (siehe oben)

---

## Block H — Permissions

### H.1 — App-Level (OS-Permissions, nicht Tenant-Permissions)

| Permission | Wann gefragt | Klartext (Pre-Prompt) |
|---|---|---|
| Push-Notifications | beim Onboarding (nach erstem Login) | *„HiveGrid sendet dir Benachrichtigungen, wenn dir eine Aufgabe zugewiesen wird oder wichtige Updates passieren."* |
| Kamera | beim ersten Foto-Use (Acceptance, Checklist, Profil) | *„HiveGrid braucht Zugriff auf deine Kamera, um Fotos für Acceptance-Protokolle und Checklisten aufzunehmen."* |
| Photo-Library | beim ersten Galerie-Pick | *„HiveGrid braucht Zugriff auf deine Galerie, um vorhandene Fotos auszuwählen."* |
| Face-ID / Touch-ID | beim Aktivieren in Settings | *„HiveGrid nutzt Face-ID für schnellen, sicheren Login auf deinem Gerät."* |
| Geolocation | **Phase 2** (für Einsatzort-Erkennung) | nicht in Phase 1 |
| Bluetooth | **Phase 2** (Beacon-basierter Standort) | nicht in Phase 1 |
| NFC | **Phase 2** (Zeiterfassung) | nicht in Phase 1 |
| Mikrofon | **nicht in Phase 1** | — |

**Konvention bei Permission-Denied:**
- Klartext-Hinweis im UI mit Funktion *„Geräte-Einstellungen öffnen"* (öffnet OS-Settings via Capacitor `@capacitor/app`)
- App bleibt funktional ohne diese Permission (Graceful Degradation)
- Bei kritischer Funktion (z.B. Foto-Pflicht in Acceptance): Klartext-Block + Settings-Link

### H.2 — Tenant-Permissions in der App

- App nutzt **identisches** Permission-System wie Web (Modul 27 Block C)
- Permission-Snapshot wird beim Login geladen und im IndexedDB-Cache abgelegt (siehe A.4)
- Routen-Schutz funktioniert offline auf Basis des Snapshots (Klartext bei Permission-Mismatch nach Sync)
- UI-Modus aus Modul 27 C.2 wird respektiert: Mobile-App default `Baustelle`-Modus (auf Smartphone), `Büro`-Modus auf Tablet möglich (User-Override)

---

## Block I — Routes / Tiefen-Verlinkung (Deeplinks)

### I.1 — Universal Links (iOS) / App Links (Android)

- Domain `app.hivegrid.de` ist als Universal-Link / App-Link registriert
- Apple `apple-app-site-association` File auf `hivegrid.de/.well-known/`

- Android Asset-Links File analog
- Push-Klick mit `deeplink`-URL öffnet App auf Ziel-Route (statt Browser)

### I.2 — Deeplink-Schemas

- `hivegrid://` als Custom-URL-Scheme (Fallback)
- `https://app.hivegrid.de/{tenant}/...` (Universal Link, Default)

### I.3 — Deeplink-Routing-Anforderungen

- App parst eingehenden Deeplink, prüft Auth-Status
- Bei kein Login: parkt Deeplink, zeigt Login → nach Login navigiert auf Deeplink-Ziel
- Bei Login + falscher Tenant: Tenant-Switch-Prompt (*„Diese Notification gehört zu Tenant ‚X' — wechseln?"*)
- Bei Permission-Mismatch (User darf Ziel nicht sehen): Klartext + Fallback auf Tenant-Dashboard

### I.4 — Mobile-spezifische Routen-Anpassungen

Web-App-Routen funktionieren in der Mobile-App identisch (LP-2 mobile-first responsive). Mobile-spezifische Sub-Routen:

- `/{tenant}/settings/security/biometric/` — Biometric-Setup (existiert auch im Web, dort aber kein-op auf Desktop)
- `/{tenant}/settings/mobile/` — Mobile-spezifische Settings (Pre-Cache-Toggle, Sync-Verhalten, Storage-Übersicht)
- `/{tenant}/settings/mobile/storage/` — lokaler Storage-Verbrauch + Reset-Funktion
- `/{tenant}/settings/mobile/queue/` — Sync-Queue-Übersicht (siehe D.5)

### I.5 — Tab-Bar / Native-Navigation (Apple-Compliance)

**Apple-Guideline 4.2 verlangt native Navigations-Elemente** (kein reines Hamburger-Menü).

**Anforderungen:**
- Native-Tab-Bar im Capacitor-Wrapper für die wichtigsten Top-Level-Funktionen
- Tab-Konfiguration abhängig vom UI-Modus:
- **Baustelle-Modus** (Default Mobile): Heutige Aufträge / Projekte / Tasks / Chat / Mehr
- **Büro-Modus**: Dashboard / CRM / Pipeline / Inbox / Mehr
- Tab-Bar wird vom Web-Code via Capacitor-API gesteuert (kein echtes Native)
- Hamburger-Menü als Sekundär-Navigation (für selten genutzte Funktionen) erlaubt

---

## Block J — Update-Mechanik

### J.1 — Web-Code-Live-Update

**Phase 1: Web-Bundle wird als statisches Asset aus Capacitor-Container geladen.**

- App-Start prüft Server-Version ( `/api/v1/mobile/version-check` )
- Bei neuer Web-Bundle-Version: Download im Hintergrund, Klartext-Hinweis *„Update verfügbar — beim nächsten App-Start aktiviert"*
- Beim nächsten Cold-Start: neues Bundle wird geladen (Atomic-Swap, kein User-Disruption)
- **Kein** Capacitor Live Updates in Phase 1 (Phase 2: ggf. CodePush-ähnliche Lösung)
- Server-Endpoint mit `min_supported_app_version`: bei zu alter Wrapper-Version → Force-Update-Screen mit App-Store-Link

### J.2 — Native-Wrapper-Updates

- Regulär über App-Stores (TestFlight für Beta, Production via App Store / Google Play)
- Wrapper-Updates sind selten (nur bei Native-Plugin-Änderungen, OS-Compatibility, Privacy-Manifest-Updates)
- Frequenz: 1-2 Native-Releases pro Quartal Default; Web-Bundle-Updates beliebig häufig
- Bei Native-Plugin-Änderung mit API-Bruch: Wrapper-Version-Bump nötig + Web-Code-Compat-Check

### J.3 — Force-Update-Logik

**Trigger:**
- Web-Bundle inkompatibel mit Wrapper-Version (server-side `min_supported_app_version`)
- Sicherheits-kritisches Update (Plattform-Admin-getriggert)
- iOS/Android-OS-Mindestversion erhöht

**UX:**
- Modal-Screen ohne Schließen-Funktion: *„Bitte aktualisiere die HiveGrid-App — diese Version wird nicht mehr unterstützt. [App-Store öffnen]"*
- Klartext: *„Grund: [Sicherheits-Update | Neue Funktionen | OS-Kompatibilität]"*

---

## Block K — Submission-Strategie

### K.1 — Reihenfolge: Google Play zuerst, dann Apple

**Begründung:** niedrigere Hürde bei Google Play, schnellerer Feedback-Loop, geringere Reject-Risk.

**Phasen:**

1. **Google Play Internal Testing** (sofort, max. 100 Tester ohne Review)
2. **Google Play Closed Testing** (mit Review, ~Tage)
3. **Google Play Production** (mit Review, ~Tage)
4. **Apple TestFlight Internal** (sofort, 100 Tester)
5. **Apple TestFlight External** (mit Review, ~1-2 Tage)
6. **Apple App Store Production** (mit Review, mehrere Tage)

### K.2 — Pre-Submission-Checkliste

**Vor Apple-Submit:**

- App-Review-Notes mit Demo-Account-Credentials (siehe F.3)
- Privacy-Policy-URL gesetzt (`hivegrid.de/privacy`)
- Privacy-Manifest aktuell (siehe F.4)
- Account-Löschung sichtbar getestet (Reviewer prüft das aktiv)
- Keine Pricing-/Upgrade-UI im Build (Build-Flag `IS_MOBILE_APP=true` aktiv)
- Native-Features funktional getestet (Push, Face-ID, Kamera) — Reviewer testet das
- App-Icons + Splash-Screens für alle Device-Größen
- Screenshots zeigen ausschließlich Funktionen, die in der App tatsächlich da sind (Guideline 2.3)

**Vor Google-Play-Submit:**

- Data-Safety-Form ausgefüllt
- Account-Löschung in App + externe Lösch-URL als Backup
- Privacy-Policy-URL
- Permissions-Erklärungen klar

### K.3 — Reject-Recovery

**Bei Apple-Reject:**
- Klartext-Antwort des Reviewers analysieren
- Im Brainstorming dokumentierte Strategien:
- 4.2 (Minimum-Functionality): Native-Feature-Liste vorzeigen, Demo-Video
- 3.1.1 (IAP): kein Pricing-UI nachweisen, ggf. Build mit zusätzlichen Pricing-Hidden-Toggles
- 4.8 (SIWA): SIWA als Backup-Implementation aktivieren (~1 Tag)
- 5.1.1 (Account-Löschung): sichtbarer Pfad zeigen
- 2.1 (kann nicht einloggen): Demo-Account neu setzen, Klartext in Notes
- Re-Submit mit erweiterten Klartext-Notes

---

## Block L — Klartext-Hinweise / -Fehler

**Push:**
- Permission verweigert: *„Push ist auf diesem Gerät deaktiviert. Du verpasst möglicherweise wichtige Updates. [Geräte-Einstellungen]"*
- Token-Registrierung fehlgeschlagen: *„Push-Setup fehlgeschlagen — bitte App neu starten oder Support kontaktieren."*

**Biometrie:**
- Face-ID-Fehler: *„Face-ID nicht erkannt — bitte erneut versuchen oder Passwort eingeben."*
- 3x Fehlschlag: *„Face-ID temporär gesperrt. Bitte mit Passwort anmelden."*
- Hardware nicht verfügbar: *„Dieses Gerät unterstützt kein Face-ID/Touch-ID."*
- Hardware vorhanden, nicht eingerichtet: *„Bitte zuerst Face-ID in den Geräte-Einstellungen einrichten."*

**Kamera:**
- Permission verweigert: *„Kamera-Zugriff verweigert. [Geräte-Einstellungen öffnen]"*
- Storage voll: *„Lokaler Speicher voll — bitte Sync starten oder lokalen Cache leeren."*

**Offline:**
- Banner offline: *„Du bist offline — gespeicherte Änderungen werden später hochgeladen."*
- Sync-Erfolg: *„Alle 7 Änderungen wurden synchronisiert."*
- Sync-Fehler: *„2 Änderungen konnten nicht hochgeladen werden. [Details]"*
- Konflikt: *„Es gab einen Konflikt — deine Änderung wurde übernommen, der vorherige Stand ist noch sichtbar. [Anzeigen]"*
- Queue voll: *„Lokale Warteschlange voll (1.000 Einträge) — bitte mit Internet verbinden."*
- Storage voll (Fotos): *„Foto-Speicher voll (500 MB) — bitte Sync starten oder Fotos später aufnehmen."*

**Pre-Cache:**
- Beim Login: *„Lade Daten für offline-Nutzung — bitte einen Moment Geduld..."*
- Bei Fehler: *„Pre-Caching fehlgeschlagen — App funktioniert online normal, offline möglicherweise eingeschränkt."*

**Update:**
- Update verfügbar (Web-Bundle): *„Ein Update wurde geladen — beim nächsten App-Start aktiv."*
- Force-Update (Wrapper): *„Bitte aktualisiere die HiveGrid-App — diese Version wird nicht mehr unterstützt. [App-Store]"*

**Tenant-Wechsel mit Queue:**
- *„3 Änderungen für Tenant ‚Geerkens GmbH' warten noch auf Upload — bitte erst synchronisieren."*

**Permission-Mismatch nach Sync:**
- *„Du hast inzwischen keine Berechtigung mehr für diese Aktion. Eintrag wurde verworfen."*

**Apple-Compliance-Boundary (Pricing):**
- Bei Feature-Block in App: *„Diese Funktion ist im Cloud+-Plan verfügbar. Bitte Workspace-Admin kontaktieren oder am Computer aktivieren."* (kein Klick-Link)

---

# Block M — Permissions-Anwendung

## M.1 — Permission-IDs

| Permission | Erlaubt |
|---|---|
| (Self-Scope) | Eigene Mobile-Settings, Biometrie-Setup, Sync-Queue, Push-Devices — kein Permission-Bundle nötig |
| `mobile.admin` | Mobile-spezifische Tenant-Settings (z.B. Pre-Cache-Default für Tenant, Force-Update-Trigger pro Tenant) — Phase 2 |

## M.2 — Bundle-Mapping (Variante 2)

| Bundle | Atomare Permissions |
|---|---|
| `mobile.user` (jeder User implizit) | Self-Scope-Mobile-Funktionen |
| `mobile.admin` (Phase 2) | `mobile.admin` |

## M.3 — Anwendung in Routes

- `/{tenant}/settings/security/biometric/` → eingeloggter User
- `/{tenant}/settings/mobile/*` → eingeloggter User
- `/api/v1/notifications/mobile-device` → eingeloggter User (Self-Scope)
- `/api/v1/mobile/version-check` → public (Auth optional, App-Version-Check ohne Login)

## M.4 — App-Level-Permission-Snapshot

- Beim Login: Permission-Set des Users in IndexedDB-Cache `cache.tenant_meta`
- Bei Permission-Wechsel server-side: nächster Sync-Trigger lädt aktualisiertes Set
- Routen-Schutz offline: lokal anhand Cache; bei Mismatch nach Sync → Klartext + Fallback

---

# Block N — Phase 2 / spätere Features

- **NFC** für Zeiterfassung (Tap-to-Clock-In)
- **Bluetooth** für Beacon-basierte Standort-Erkennung (Baustellen-Auto-Check-In)
- **Geolocation** mit User-Consent für Einsatzort-Auto-Tagging
- **CodePush / Capacitor Live Updates** für Web-Code-Live-Updates ohne App-Store-Roundtrip
- **CRDT-basierte Konflikte** für Multi-User-Co-Editing (z.B. gemeinsames Acceptance-Protokoll)
- **Erweiterte Offline-Funktionen:**
- Komplette Tenant-Daten lokal indexierbar (CRM-Offline-Suche)
- Offline-Email-Lesen (Inbox-Cache)
- Offline-Booking-Bestätigungen
- Offline-Lead-Bearbeitung
- **Silent-Push-Capability** für Hintergrund-Sync (Modul 22 Phase 2)
- **Browser-Push-Channel** als 4. Notification-Kanal für Web-Nutzer ohne Mobile-App
- **Watch-App-Companion** (Apple Watch / Wear OS) für Quick-Actions
- **Mikrofon-Permission** für Voice-Notes / Diktier-Funktion
- **Tablet-spezifische UI-Optimierung** (Side-by-Side-Layout im Büro-Modus)
- **Dark-Mode-Native-Sync** (Phase 1 nutzt Web-Theme; Phase 2 native OS-Theme-Detection)
- **Cross-Device-Read-Sync** für Notifications (M22 Phase 2)

- **Subprocessor-Disclosure** im App-Onboarding (M35 Block H)
- **Tenant-Admin-konfigurierbare Pre-Cache-Strategien** (welche Daten pro Rolle vorgeladen werden)
- **App-Store-In-App-Rating-Prompt** an strategischen Stellen (z.B. nach erfolgreichem Acceptance-Abschluss)
- **App-Onboarding-Tour** für Neu-Installs (Native-Tutorial-Screens)

---

## Block O — Architektur-Vorschriften (Querschnitt)

- **LP-2 — Mobile-First-Responsive:** Web-App ist primär; Mobile-App ist Native-Wrapper
- **LP-8 — App-Store-Strategie:** Capacitor + 3 Native-Features (Push, Face-ID, Kamera); strikt kein Pricing/Upsell in der App; Account-Löschung in App; Privacy-Manifest; Reviewer-Demo-Account; Login-only-Framing
- **LP-9 — Spec-Tiefe:** keine UI-Layouts; Anforderungen-pur
- **LP-10 — Klartext-Diagnostik:** alle Push-/Sync-/Permission-Fehler mit menschenlesbarem Text + Lösungs-Hinweis (Block L)
- **LP-12 — i18n:** Klartext-Strings aus Locale-Files (Phase 1 nur DE)
- **LP-13 — EU-Stack:** Push-Provider Novu auf Scaleway; APNs/FCM unausweichlich aber nur Token + minimaler Payload (kein PII-Datenleck)
- **AV #1, LP-10:** Klartext überall
- **AV #5 — Self-Scope-Default:** User sieht nur eigene Devices, eigene Queue, eigene Biometrie-Settings
- **AV #6 — Audit-Trail:** Mobile-spezifische Aktionen (Biometrie-Setup, Account-Löschung, Sync-Konflikt-Resolution) werden geloggt
- **AV #16 — Rate-Limiting:** Sync-Endpoints rate-limited (gegen Loop bei kaputtem Client)
- **AV #20 — KI-Action-Boundary:** keine KI-Aktionen ohne User-Bestätigung in der App

---

## Detail-Spec-Status: DRAFT

Erste vollständige Detail-Spec für Modul 36. **Vor Bau:** Verifizierung mit Noah (gemeinsam mit Pflichtenheft-Sweep), Cross-Check mit Modul 22 (Push-Token-Flow), Modul 27 (Biometrie + Session), Modul 35 (Account-Löschung + Privacy-Manifest), Modul 28 (Subscription-UI-Hide-Flag).

**Offene Detail-Punkte (nicht-blockierend, beim Bau zu klären):**

- Konkrete Capacitor-Plugin-Auswahl ( `@capacitor/push-notifications` vs. Novu-eigenes SDK; `capacitor-biometric-auth` vs. Alternativen)
- Privacy-Manifest konkrete Apple-API-Reasons (mit Apple-Doku-Cross-Check beim Bau)
- IndexedDB-Library: Dexie.js (default) vs. native IDB-API (Implementierungs-Detail)
- Service-Worker-Strategie: Workbox vs. custom (Performance-Trade-off, Build-Komplexität)
- Konkretes Schema der Idempotency-Keys serverseitig (mit allen Producer-Modulen abgleichen)
- Foto-Komprimierungs-Default (Quality 85 / max-Dim 2400 sind Heuristik — beim Bau mit Real-User-Tests validieren)
- Pre-Cache-Größenlimits (100 MB / 250 MB / 500 MB Foto) sind Heuristik; mit Geerkens-Use-Case validieren
- Tab-Bar-Konfiguration pro UI-Modus konkret festlegen (mit Subframe-Designs)
- Apple-App-Review-Notes-Text-Vorlage (rechtlicher Final-Slug mit Anwalt validieren)
- Re-Submit-Strategy bei spezifischen Reject-Grounds (Playbook beim Bau dokumentieren)

**Abhängige Module:**

- Modul 22 (Notifications) — Push-Token-Registrierung, NotificationDevice-Entity, Push-Versand-Pipeline
- Modul 27 (Auth/Multi-Tenancy) — Email/Pwd + Magic-Link-Login, Session-Management, Permissions, Account-Löschung Block D.3
- Modul 35 (DSGVO/Privacy) — Privacy-Manifest, Account-Löschung-Konsolidierung, Subprozessoren-Liste
- Modul 28 (Subscription) — `IS_MOBILE_APP=true`-Build-Flag zur Pricing-UI-Ausblendung

- Modul 21 (Audit-Log) — Mobile-spezifische Audit-Einträge
- Modul 23 (Files) — Foto-Upload, Storage-Pfad-Konvention
- Modul 26 (Acceptance) — Foto-Pflicht für Acceptance-Protokoll
- Modul 4 (Tasks) — Task-Status-Update offline
- Modul 13 (Chat) — Chat-Nachrichten offline-fähig
- Modul 30 (i18n) — Klartext-Strings für Mobile-spezifische Hinweise
- Modul 31 (Branding) — App-Icons, Splash-Screens, Tenant-Branding (Phase 2 dynamisch)
- Modul 37 (Plattform-Admin) — Privacy-Manifest-Master-Liste, Demo-Account-Pflege, `min_supported_app_version`-Steuerung