

# Phase 8: FCM Push-Benachrichtigungen - Implementierungs-Übersicht

---

## Ziele von Phase 8

---

- 1. FCM-Integration (Firebase Cloud Messaging)**
    - Client-seitige Registrierung für Push-Nachrichten
    - FCM-Token im User-Profil speichern
    - Service Worker für Background-Push-Nachrichten
  - 2. Erweiterung des Notification-Dispatchers**
    - `dispatchNotification` sendet Push-Nachrichten über Cloud Function
    - Nur wenn `pushEnabled=true` und `fcmTokens` vorhanden
  - 3. Cloud Function für Push-Versand**
    - Authentifizierung prüfen
    - Tokens sammeln und Push-Nachrichten senden
    - Ungültige Tokens automatisch bereinigen
  - 4. Benutzereinstellungen**
    - Neue Settings-Seite in der Club-App
    - Push aktivieren/deaktivieren
    - Einstellungen im Firestore speichern
- 

## Neue/Angepasste Dateien

---

### Shared Types

**packages/shared-types/src/user.ts (Erweiterung)**

```
export interface PlatformUser {  
  // ... bestehende Felder ...  
  
  // Phase 8 Extensions: Push Notifications (FCM)  
  fcmTokens?: string[];           // Array von FCM-Tokens für verschiedene Geräte  
  pushEnabled?: boolean;          // true, wenn Push-Nachrichten aktiviert sind  
  (Standard: true)  
}
```

### Core

**packages/core/src/firebase/init.ts (Erweiterung)**

- Neue Funktion: `getMessagingInstance()` für Firebase Messaging
- Import: `getMessaging` von `firebase/messaging`

**packages/core/src/notifications/fcm-token-manager.ts (NEU)**

```

export async function registerForPushNotifications(
  messaging: Messaging | null,
  userId: string
): Promise<string | null>

export async function unregisterFromPushNotifications(
  messaging: Messaging | null,
  userId: string
): Promise<void>

export async function setPushNotificationStatus(
  userId: string,
  enabled: boolean
): Promise<void>

export async function isPushNotificationEnabled(userId: string): Promise<boolean>

```

### packages/core/src/notifications/notification-dispatcher.ts (Erweiterung)

- Erweitert `dispatchNotification` um Push-Versand
- Neue interne Funktion: `sendPushNotificationIfEnabled`
- Ruft Cloud Function `sendPushNotification` auf (httpsCallable)

### packages/core/src/index.ts (Erweiterung)

```
export * from './notifications/fcm-token-manager'; // Phase 8: FCM
```

## Cloud Functions

### functions/package.json (NEU)

- Dependencies: `firebase-admin`, `firebase-functions`, `typescript`
- Scripts: `build`, `deploy`, `logs`

### functions/tsconfig.json (NEU)

- TypeScript-Konfiguration für Cloud Functions

### functions/.gitignore (NEU)

- Ignoriert `lib/`, `node_modules/`, Logs, etc.

### functions/src/index.ts (NEU)

```

export const sendPushNotification = functions.https.onCall(async (data, context) => {
  // 1) Authentifizierung prüfen
  // 2) Parameter validieren (targetUserIds, title, body, data)
  // 3) Tokens für alle Ziel-User sammeln
  // 4) Push-Nachricht mit admin.messaging().sendEachForMulticast() senden
  // 5) Ungültige Tokens bereinigen (arrayRemove)
});

```

## Club-App

### apps/club-app/public/firebase-messaging-sw.js (NEU)

- Service Worker für Background-Push-Nachrichten
- Empfängt Nachrichten via `messaging.onBackgroundMessage`
- Zeigt Benachrichtigungen mit `self.registration.showNotification`
- WICHTIG: Firebase Config muss manuell eingefügt werden!

**apps/club-app/src/app/settings/page.tsx (NEU)**

- Einstellungs-Übersichtsseite
- Karten für: Benachrichtigungen, Profil, Privatsphäre (Placeholder)
- Navigation zu `/settings/notifications`

**apps/club-app/src/app/settings/notifications/page.tsx (NEU)**

```
// Features:
// - Echtzeit-Abonnement von user.pushEnabled
// - Toggle-Button für Push-Benachrichtigungen
// - Ruft registerForPushNotifications / unregisterFromPushNotifications auf
// - Info-Card mit Hinweisen
```

**i18n****packages/ui/src/locales/de.json (Erweiterung)**

```
"settings": {
  "title": "Einstellungen",
  "notifications": "Benachrichtigungen",
  "pushNotifications": "Push-Benachrichtigungen",
  "pushEnabled": "Push-Benachrichtigungen aktiviert",
  "pushDisabled": "Push-Benachrichtigungen deaktiviert",
  "loading": "Lade Einstellungen..."
}
```

**packages/ui/src/locales/en.json (Erweiterung)**

- Englische Übersetzungen für alle neuen Settings-Keys

## Firestore Schema

**platform/users/{uid} (Erweiterung)**

```
{
  // ... bestehende Felder ...

  // Phase 8: FCM
  fcmTokens: string[], // Array von FCM-Tokens
  pushEnabled: boolean, // Push aktiviert/deaktiviert (Standard: true)
}
```

## Code-Beispiele

**1) FCM-Token registrieren**

```
import { registerForPushNotifications, getMessagingInstance } from '@nightlife-os/core';

const messaging = getMessagingInstance();
const token = await registerForPushNotifications(messaging, user.uid);
```

## 2) Push-Benachrichtigung senden

```
// notification-dispatcher.ts
import { getFunctions, httpsCallable } from 'firebase/functions';

const functions = getFunctions();
const sendPushNotification = httpsCallable(functions, 'sendPushNotification');

await sendPushNotification({
  targetUserIds: [userId],
  title: 'Neue Nachricht',
  body: 'Du hast eine neue Direktnachricht erhalten',
  data: { type: 'NEW_DIRECT_MESSAGE', chatId: 'xyz' },
});
```

## 3) Cloud Function: Push senden

```
// functions/src/index.ts
export const sendPushNotification = functions.https.onCall(async (data, context) => {
  if (!context.auth) {
    throw new functions.https.HttpsError('unauthenticated', 'User must be authenticated');
  }

  // Tokens sammeln
  const tokensToSend = [];
  for (const userId of data.targetUserIds) {
    const userDoc = await admin.firestore().doc(`platform/users/${userId}`).get();
    const userData = userDoc.data();
    if (userData?.pushEnabled && userData?.fcmTokens) {
      tokensToSend.push(...userData.fcmTokens);
    }
  }

  // Push senden
  const message = {
    notification: { title: data.title, body: data.body },
    data: data.data,
    tokens: tokensToSend,
  };
  const response = await admin.messaging().sendEachForMulticast(message);

  // Ungültige Tokens bereinigen
  // ...

  return { success: true, sentCount: response.successCount };
});
```

## 4) Service Worker: Background-Push

```
// firebase-messaging-sw.js
messaging.onBackgroundMessage((payload) => {
  const notificationTitle = payload.notification?.title || 'Nightlife OS';
  const notificationOptions = {
    body: payload.notification?.body,
    icon: '/icon-192.png',
    data: payload.data,
  };
  self.registration.showNotification(notificationTitle, notificationOptions);
});
```

## Testanleitung

### 1. VAPID Key ersetzen

**Wo:** packages/core/src/notifications/fcm-token-manager.ts

```
const VAPID_KEY = 'YOUR_VAPID_KEY_HERE'; // Aus Firebase Console holen!
```

**Wie:** Firebase Console → Project Settings → Cloud Messaging → Web Push certificates → Generate key pair

### 2. Service Worker Config anpassen

**Wo:** apps/club-app/public/firebase-messaging-sw.js

```
const firebaseConfig = {
  apiKey: 'YOUR_API_KEY', // Aus .env kopieren
  authDomain: 'YOUR_AUTH_DOMAIN',
  // ...
};
```

### 3. Cloud Function deployen

```
cd functions
pnpm install
pnpm run build
firebase deploy --only functions
```

### 4. Club-App testen

```
cd apps/club-app
pnpm dev
```

#### Test-Schritte:

1. Login
2. Zu /settings navigieren
3. "Benachrichtigungen" öffnen
4. Push aktivieren (Berechtigung erteilen)

5. Neue Nachricht senden (in anderem Browser/Tab)
  6. Push-Benachrichtigung erhalten
  7. Push deaktivieren
  8. Keine Push-Benachrichtigung mehr erhalten (In-App weiterhin aktiv)
- 

## Bekannte Limitierungen

---

### 1. VAPID Key Placeholder:

- Muss manuell aus Firebase Console geholt werden
- Aktuell nur Platzhalter `'YOUR_VAPID_KEY_HERE'`

### 2. Service Worker Config:

- Firebase Config muss manuell in `firebase-messaging-sw.js` eingefügt werden
- Kann nicht aus `.env` geladen werden (Service Worker läuft außerhalb Next.js)

### 3. Cloud Function Autorisierung:

- Aktuell nur Basic-Auth-Check ( `context.auth` )
- Für Produktion sollte Admin-Check erweitert werden

### 4. Browser-Support:

- Push-Benachrichtigungen funktionieren nur in modernen Browsern
  - Safari iOS hat eingeschränkten Support
- 

## Nächste Schritte (Phase 9+)

---

### 1. E-Mail-Benachrichtigungen:

- Super-Admin Mailing-System
- Segment-basiertes Targeting

### 2. AI-Übersetzung:

- Automatische Übersetzung basierend auf `user.language`
- Template-System für mehrsprachige Notifications

### 3. Erweiterte Push-Features:

- Action-Buttons in Notifications
  - Rich Media (Bilder, Videos)
  - Silent Notifications für Background-Updates
- 

## Status

---

### ✅ Implementiert:

- FCM-Token-Management
- Push-Benachrichtigungs-Versand über Cloud Function
- Benutzereinstellungen für Push
- Service Worker für Background-Push
- i18n (Deutsch & Englisch)
- Dokumentation

**⚠️ Manuell erforderlich:**

- VAPID Key aus Firebase Console holen
- Service Worker Config anpassen
- Cloud Function deployen

**🔄 Getestet:**

- TypeScript-Kompilierung ✓
- Lint-Checks ✓
- Firestore-Schema ✓
- UI-Integration ✓

## Technische Details

### FCM-Token-Lifecycle

#### 1. Registrierung:

- User gibt Berechtigung
- `getToken()` holt FCM-Token
- Token wird in `platform/users/{uid}/fcmTokens` gespeichert (arrayUnion)

#### 2. Push-Versand:

- `dispatchNotification` prüft `pushEnabled` und `fcmTokens`
- Ruft Cloud Function auf
- Cloud Function sammelt alle Tokens
- `admin.messaging().sendEachForMulticast()` sendet Push

#### 3. Token-Bereinigung:

- Cloud Function prüft `response.failureCount`
- Ungültige Tokens ( `registration-token-not-registered` ) werden aus Firestore entfernt (arrayRemove)

#### 4. Deregistrierung:

- `deleteToken()` löscht Token bei FCM
- Token wird aus Firestore entfernt

### Service Worker

- **Pfad:** `/firebase-messaging-sw.js` (muss im Root liegen)
- **Scope:** Gesamte App
- **Background Messages:** `onBackgroundMessage` empfängt Push
- **Notification Display:** `self.registration.showNotification`
- **Click Handler:** `notificationclick` Event öffnet App

Phase 8 erfolgreich implementiert! 🎉