# Tilt Maze - Kompetenznachweis Modul 335

RiciYT

Januar 2026

# 1 Planung – Tilt Maze

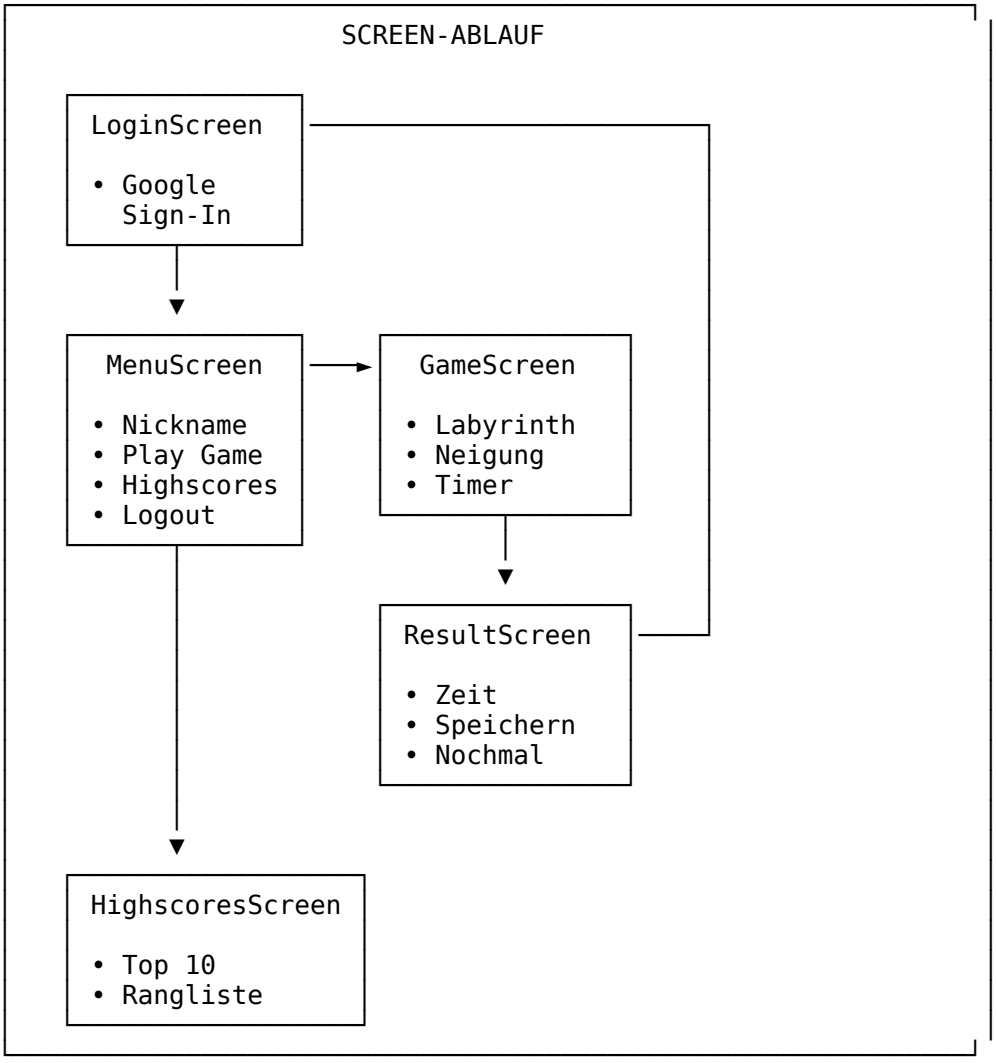
# 1.1 1. App-Idee

**Tilt Maze** ist ein Geschicklichkeitsspiel für mobile Geräte, bei dem die Spielenden eine Kugel durch Neigen des Smartphones durch ein Labyrinth navigieren. Ziel ist es, die Kugel möglichst schnell ins Zielfeld zu manövrieren.

# 1.2 2. Screen-Storyboard

Die Storyboard-Skizzen befinden sich im Ordner assets/storyboard/.

## 1.2.1 Übersicht der Screens




# 1.3 3. Funktionsliste

## 1.3.1 3.1 Benutzerauthentifizierung

| ID | Funktion | Beschreibung |
|---|---|---|
| F01 | Google Sign-In | Anmeldung mit Google-Konto über OAuth 2.0 |
| F02 | Anonyme Anmeldung | Anmeldung ohne Google-Konto (Firebase Anonymous Auth) |

| ID | Funktion | Beschreibung |
|---|---|---|
| F03 | Gastmodus | Spielen ohne Anmeldung, keine Speicherung von Zeiten |
| F04 | Logout | Abmeldung und Rückkehr zum Login-Screen |
| F05 | Nickname | Benutzer können einen Spielernamen festlegen |

### 1.3.2 3.2 Spielfunktionen

| ID | Funktion | Beschreibung |
|---|---|---|
| F04 | Kugelsteuerung | Kugel wird durch Neigen des Geräts (Accelerometer) gesteuert |
| F05 | Physik-Engine | Realistische Kugelbewegung mit matter-js (Reibung, Kollision) |
| F06 | Labyrinth | Spielfeld mit Hindernissen (Wände im Zick-Zack-Muster) |
| F07 | Kollisionserkennung | Erkennung von Kontakt mit Wänden und Zielfeld |
| F08 | Timer | Zeitmessung mit Millisekunden-Genauigkeit |
| F09 | Vibration | Haptisches Feedback bei Spielereignissen |
| F10 | Steuerungseinstellungen | Anpassbare Sensitivität, Deadzone, Glättung |
| F11 | Ziel erreichen | Spiel endet bei Kontakt der Kugel mit dem Zielfeld |

### 1.3.3 3.3 Datenspeicherung

| ID | Funktion | Beschreibung |
|---|---|---|
| F12 | Bestzeit speichern | Speicherung der Bestzeit in Firebase Realtime Database |
| F13 | Nickname speichern | Nickname wird pro Benutzer gespeichert |
| F14 | Bestenliste laden | Abruf der Top 10 Spielzeiten aus der Datenbank |

### 1.3.4 3.4 Bestenliste

| ID | Funktion | Beschreibung |
|---|---|---|
| F15 | Top 10 Anzeige | Sortierte Liste der besten 10 Zeiten |
| F16 | Podium-Hervorhebung | Gold, Silber, Bronze für die Top 3 |
| F17 | Zeitformatierung | Anzeige im Format X.XXs |

# 1.4 4. Verwendete Sensoren und Aktoren

### 1.4.1 4.1 Sensor 1: Accelerometer (Beschleunigungssensor)

- **Bibliothek:** `expo-sensors`
- **Verwendung:** Erfassung der Geräteneigung auf X- und Y-Achse
- **Implementierung:** `src/input/tiltInput.ts` und `src/screens/GameScreen.tsx`
- **Parameter:**
  - Update-Intervall: 50ms
  - Deadzone: 0.05
  - Glättung (Alpha): 0.3
  - Sensitivität: einstellbar (0.3–3.0)

#### 1.4.2 4.2 Aktor 1: Vibration (Haptisches Feedback)

- **Bibliothek:** `react-native` (Vibration API)
- **Verwendung:** Feedback bei Kollisionen oder Spielereignissen
- **Implementierung:** `src/screens/GameScreen.tsx`

## 1.5 5. Persistente Speicherung

- **Technologie:** Firebase Realtime Database
- **Datenmodell:** Siehe 03_loesungskonzept.md
- **Operationen:**
  - Schreiben: Bestzeiten, Nicknames
  - Lesen: Highscores, eigene Bestzeit

## 1.6 6. Authentifizierung

- **Technologie:** Firebase Authentication
- **Methoden:**
  - Google Sign-In (OAuth 2.0)
  - Anonyme Anmeldung (Firebase Anonymous Auth)
  - Gastmodus (ohne Authentifizierung, keine Datenspeicherung)
- **Konfiguration:** `src/config/firebase.ts`

---

*Erstellt im Rahmen des Kompetenznachweises Modul 335*

# 2 Testplan – Tilt Maze

## 2.1 1. Testübersicht

Dieser Testplan beschreibt die funktionalen Testfälle für die App «Tilt Maze». Die Testfälle sind aus den Use-Cases und der Funktionsliste abgeleitet.

## 2.2 2. Testumgebung

| Aspekt | Spezifikation |
|---|---|
| Gerät | Android-Smartphone (physisches Gerät empfohlen) |
| Betriebssystem | Android 10+ |
| Test-App | Expo Go / APK Build |
| Netzwerk | WLAN (für Firebase-Zugriff) |

## 2.3 3. Funktionale Testfälle

#### 2.3.1 3.1 Authentifizierung

| Nr. | Testfall | Vorbedingung | Aktion | Erwartetes Resultat |
|---|---|---|---|---|
| T01 | Google Sign-In | App gestartet, Login-Screen angezeigt | «Mit Google anmelden» tippen, Google-Konto wählen | Erfolgreiche Authentifizierung, Weiterleitung zum Menü |
| T02 | Logout | Benutzer ist eingeloggt, Menü-Screen angezeigt | «Logout» tippen | Benutzer wird abgemeldet, Login-Screen wird angezeigt |
| T03 | Kein Internet beim Login | Internetverbindung deaktiviert | Login-Versuch | Fehlermeldung wird angezeigt |

### 2.3.2 3.2 Navigation

| Nr. | Testfall | Vorbedingung | Aktion | Erwartetes Resultat |
|---|---|---|---|---|
| T04 | Menü zu Spiel | Menü-Screen angezeigt | «Spiel starten» tippen | Game-Screen wird geladen |
| T05 | Menü zu Highscores | Menü-Screen angezeigt | «Highscores» tippen | Highscores-Screen wird angezeigt |
| T06 | Zurück zum Menü | Beliebiger Screen | Zurück-Button tippen | Rückkehr zum Menü |

### 2.3.3 3.3 Spielmechanik

| Nr. | Testfall | Vorbedingung | Aktion | Erwartetes Resultat |
|---|---|---|---|---|
| T07 | Kugelsteuerung links | Spiel gestartet | Gerät nach links neigen | Kugel bewegt sich nach links |
| T08 | Kugelsteuerung rechts | Spiel gestartet | Gerät nach rechts neigen | Kugel bewegt sich nach rechts |
| T09 | Kugelsteuerung oben | Spiel gestartet | Gerät nach vorne neigen | Kugel bewegt sich nach oben |
| T10 | Kugelsteuerung unten | Spiel gestartet | Gerät nach hinten neigen | Kugel bewegt sich nach unten |
| T11 | Kollision mit Wand | Kugel in Bewegung Richtung Wand | Kugel erreicht Wand | Kugel prallt ab, bleibt im Spielfeld |
| T12 | Ziel erreichen | Spiel gestartet | Kugel ins Zielfeld navigieren | «You Won!» wird angezeigt, Timer stoppt |
| T13 | Timer-Funktion | Spiel gestartet | Beobachten | Timer zählt in Echtzeit hoch (0.00s → …) |
| T14 | Vibration bei Ereignis | Vibration aktiviert | Kollision oder Spielende | Gerät vibriert kurz |

### 2.3.4 3.4 Steuerungseinstellungen

| Nr. | Testfall | Vorbedingung | Aktion | Erwartetes Resultat |
|---|---|---|---|---|
| T15 | Sensitivität ändern | Spiel gestartet, Einstellungen öffnen | Sensitivität-Slider bewegen | Kugel reagiert empfindlicher/träger |

| Nr. | Testfall | Vorbedingung | Aktion | Erwartetes Resultat |
|---|---|---|---|---|
| T16 | X-Achse invertieren | Einstellungen öffnen | «Invert X» aktivieren | Steuerung horizontal umgekehrt |

### 2.3.5 3.5 Datenspeicherung

| Nr. | Testfall | Vorbedingung | Aktion | Erwartetes Resultat |
|---|---|---|---|---|
| T17 | Erste Bestzeit speichern | Eingeloggt, noch keine Zeit gespeichert | Spiel abschliessen | «New Personal Best!» wird angezeigt, Zeit in DB gespeichert |
| T18 | Neue Bestzeit (schneller) | Bestzeit existiert | Spiel schneller abschliessen | «New Personal Best!» erscheint, DB aktualisiert |
| T19 | Keine neue Bestzeit (langsamer) | Bestzeit existiert | Spiel langsamer abschliessen | Hinweis zum Weiterüben, alte Bestzeit bleibt |
| T20 | Nickname speichern | Eingeloggt, Menü angezeigt | Nickname bearbeiten und speichern | Nickname wird in DB gespeichert |

### 2.3.6 3.6 Bestenliste

| Nr. | Testfall | Vorbedingung | Aktion | Erwartetes Resultat |
|---|---|---|---|---|
| T21 | Highscores laden | Scores in Datenbank vorhanden | Highscores-Screen öffnen | Top 10 werden angezeigt, sortiert nach Zeit |
| T22 | Leere Bestenliste | Keine Scores in Datenbank | Highscores-Screen öffnen | Hinweis «Keine Scores vorhanden» |
| T23 | Podium-Anzeige | Mind. 3 Scores vorhanden | Highscores-Screen öffnen | Top 3 mit Gold/Silber/Bronze hervorgehoben |

### 2.3.7 3.7 Sonderfälle / Edge Cases

| Nr. | Testfall | Vorbedingung | Aktion | Erwartetes Resultat |
|---|---|---|---|---|
| T24 | App in Hintergrund | Spiel läuft | Home-Button drücken, App wieder öffnen | Spielzustand erhalten oder kontrolliert zurückgesetzt |
| T25 | Schnelle Gerätebewegungen | Spiel läuft | Gerät schnell schütteln | Kugel verhält sich stabil, kein Absturz |
| T26 | Offline-Modus | Keine Internetverbindung | Spiel abschliessen | Lokale Meldung, Speicherung scheitert mit Hinweis |

## 2.4 4. Nicht-funktionale Tests

| Nr. | Testfall | Beschreibung | Erwartetes Resultat |
|---|---|---|---|
| N01 | Performance | Spiel 60 Sekunden spielen | Flüssige Animation (>30 FPS) |
| N02 | Reaktionszeit | Gerät neigen | Verzögerung < 100ms |
| N03 | Firebase-Antwortzeit | Score speichern | < 3 Sekunden |

## 2.5 5. Testprotokoll-Vorlage

```
Testdatum:     _________________
Tester:        _________________
Gerät:         _________________
OS-Version:    _________________
App-Version:   _________________

Testfall | Ergebnis | Bemerkung
---------|----------|----------
T01      | OK/NOK   |
T02      | OK/NOK   |
...      | ...      | ...

Gefundene Fehler:
1.
2.

Fazit:
```

## 2.6 6. Testabdeckung

| Bereich | Anzahl Testfälle | Abdeckung |
|---|---|---|
| Authentifizierung | 3 | Vollständig |
| Navigation | 3 | Vollständig |
| Spielmechanik | 8 | Vollständig |
| Einstellungen | 2 | Exemplarisch |
| Datenspeicherung | 4 | Vollständig |
| Bestenliste | 3 | Vollständig |
| Edge Cases | 3 | Exemplarisch |
| **Total** | **26** | |

---

*Erstellt im Rahmen des Kompetenznachweises Modul 335*

# 3 Lösungskonzept – Tilt Maze

## 3.1 1. Übersicht

Dieses Dokument beschreibt das technische Lösungskonzept für die App «Tilt Maze» – ein Geschicklichkeitsspiel, bei dem die Spielenden eine Kugel durch Neigen des Smartphones durch ein Labyrinth navigieren.

## 3.2 2. Framework und App-Typ

### 3.2.1 2.1 Verwendetes Framework

| Technologie | Version | Zweck |
|---|---|---|
| **React Native** | 0.81.5 | Plattformübergreifende App-Entwicklung |

| Technologie | Version | Zweck |
|---|---|---|
| **Expo** | SDK 54 | Vereinfachter Zugriff auf Native APIs, Build-Prozess |
| **TypeScript** | 5.9.2 | Typsicherheit und bessere Entwicklererfahrung |
| **matter-js** | 0.20.0 | 2D-Physik-Engine für realistische Kugelbewegung |
| **Firebase** | 12.7.0 | Backend-Services (Auth, Database) |

### 3.2.2 2.2 App-Typ

#### Hybrid-App (Cross-Platform)

React Native kompiliert JavaScript/TypeScript-Code in native Komponenten, was folgende Vorteile bietet:

- **Eine Codebasis** für Android und iOS
- **Nativer Look & Feel** durch native UI-Komponenten
- **Schnelle Entwicklung** durch Hot Reloading
- **Expo-Ökosystem** für einfachen Zugriff auf Sensoren und Build-Tools

## 3.3 3. Architektur-Überblick

### 3.3.1 3.1 Projektstruktur

```
App-M335/
├── App.tsx                    # Hauptkomponente, Navigation, Auth-State
├── src/
│   ├── config/
│   │   ├── firebase.ts        # Firebase-Initialisierung
│   │   └── tiltControls.ts    # Steuerungskonfiguration
│   ├── input/
│   │   └── tiltInput.ts       # Accelerometer-Input-Verarbeitung
│   ├── screens/
│   │   ├── LoginScreen.tsx    # Authentifizierung
│   │   ├── MenuScreen.tsx     # Hauptmenü
│   │   ├── GameScreen.tsx     # Spiellogik + Physik
│   │   ├── ResultScreen.tsx   # Ergebnis + Speicherung
│   │   ├── HighscoresScreen.tsx # Bestenliste
│   │   └── SettingsScreen.tsx # App-Einstellungen
│   ├── hooks/
│   │   └── useAppSettings.ts  # Settings-Hook
│   ├── components/
│   │   └── ui/                # Wiederverwendbare UI-Komponenten
│   ├── theme/                 # Styling-Konstanten
│   └── types/
│       └── index.ts           # TypeScript-Typdefinitionen
├── assets/                    # Bilder, Sounds
├── docs/                      # Dokumentation
├── app.json                   # Expo-Konfiguration
├── eas.json                   # EAS Build-Konfiguration
└── package.json               # Abhängigkeiten
```

### 3.3.2 3.2 Komponentenhierarchie

```
App (Root)
│
├─ LoginScreen
│   └─ Google Sign-In Button
│
├─ MenuScreen
│   ├─ Welcome + Nickname
│   ├─ Play Button → GameScreen
│   ├─ Highscores Button → HighscoresScreen
│   └─ Logout Button → LoginScreen
│
├─ GameScreen
│   ├─ HUD (Timer, Back-Button)
│   ├─ Game Area
│   │   ├─ Ball (physics body)
│   │   ├─ Target (sensor body)
│   │   └─ Maze Walls (static bodies)
│   └─ Settings Modal
│
├─ ResultScreen
│   ├─ Zeit-Anzeige
│   ├─ Speicherstatus
│   └─ Navigation-Buttons
│
└─ HighscoresScreen
    └─ FlatList mit Top 10
```

# 3.4 4. Sensor-Nutzung

## 3.4.1 4.1 Accelerometer (Sensor 1)

### 3.4.1.1 Verwendungszweck

Der Beschleunigungssensor erfasst die Neigung des Geräts in 3 Achsen (x, y, z). Für das 2D-Spiel werden nur x- und y-Achse verwendet.

### 3.4.1.2 Technische Umsetzung

**Datei:** `src/input/tiltInput.ts`

```
import { Accelerometer } from 'expo-sensors';

// Sensor-Listener starten
export function startTilt(config: TiltConfig) {
  Accelerometer.setUpdateInterval(config.updateInterval); // 50ms

  subscription = Accelerometer.addListener((data) => {
    const { x, y } = data;

    // Tiefpassfilter für Glättung
    const alpha = config.smoothingAlpha;
    smoothedX = alpha * x + (1 - alpha) * smoothedX;
    smoothedY = alpha * y + (1 - alpha) * smoothedY;

    // Deadzone (kleine Bewegungen ignorieren)
    if (Math.abs(smoothedX) < config.deadzone) smoothedX = 0;
    if (Math.abs(smoothedY) < config.deadzone) smoothedY = 0;

    // Optional: Invertierung
```

```
        if (config.invertX) smoothedX = -smoothedX;

        // Werte für Physik-Engine bereitstellen
        currentTilt = { x: smoothedX, y: smoothedY };
      });
    }
```

**Datei:** `src/screens/GameScreen.tsx`

```
    // In der Game-Loop: Physik-Engine mit Neigungswerten aktualisieren
    const tiltX = getTiltX();
    const force = tiltX * sensitivity * 0.0015;
    Matter.Body.applyForce(ball, ball.position, { x: force, y: 0 });
```

#### 3.4.1.3 Konfigurationsparameter

| Parameter | Standardwert | Beschreibung |
|---|---|---|
| `updateInterval` | 50ms | Abtastrate des Sensors |
| `deadzone` | 0.05 | Schwelle für minimale Bewegung |
| `smoothingAlpha` | 0.3 | Glättungsfaktor (0–1) |
| `sensitivity` | 1.0 | Multiplikator für Steuerungsempfindlichkeit |
| `invertX` | false | X-Achse invertieren |

### 3.4.2 4.2 Vibration (Aktor 1)

#### 3.4.2.1 Verwendungszweck

Haptisches Feedback bei Spielereignissen (Kollisionen, Spielende).

#### 3.4.2.2 Technische Umsetzung

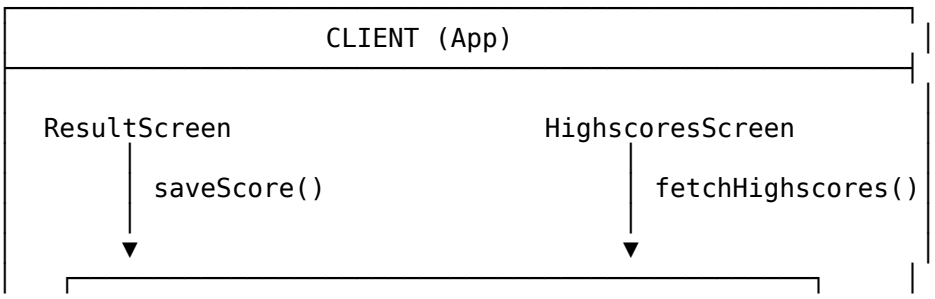
**Datei:** `src/screens/GameScreen.tsx`

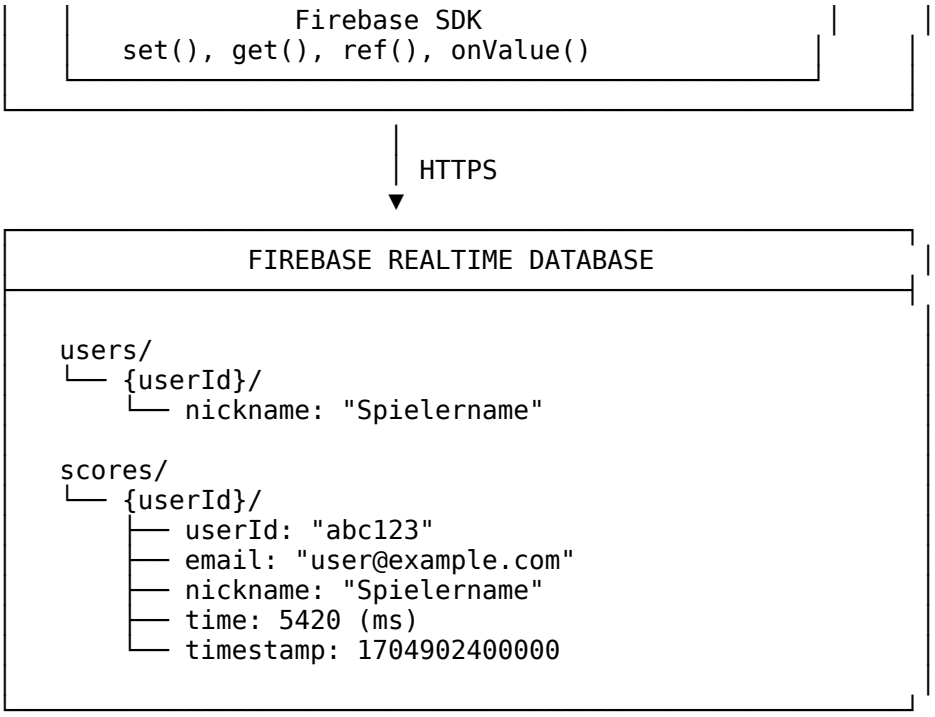
```
    import { Vibration } from 'react-native';

    // Bei Kollision oder Spielende
    if (vibrationEnabled) {
      Vibration.vibrate(100); // 100ms Vibration
    }
```

## 3.5 5. Persistenz (Firebase Realtime Database)

### 3.5.1 5.1 Datenfluss






### 3.5.2 5.2 Gespeicherte Daten

| Pfad | Datentyp | Beschreibung |
|---|---|---|
| `users/{uid}/nickname` | String | Spielername des Benutzers |
| `scores/{uid}/userId` | String | Firebase User ID |
| `scores/{uid}/email` | String | E-Mail-Adresse |
| `scores/{uid}/nickname` | String | Spielername (Kopie) |
| `scores/{uid}/time` | Number | Bestzeit in Millisekunden |
| `scores/{uid}/timestamp` | Number | Unix-Timestamp der Speicherung |

### 3.5.3 5.3 Datenbankoperationen

**Score speichern (ResultScreen.tsx):**

```
const saveScore = async (time: number) => {
  const user = auth.currentUser;
  if (!user) return;

  // Bestehende Zeit prüfen
  const scoreRef = ref(database, `scores/${user.uid}`);
  const snapshot = await get(scoreRef);

  if (!snapshot.exists() || time < snapshot.val().time) {
    // Neue Bestzeit speichern
    await set(scoreRef, {
      userId: user.uid,
      email: user.email,
      nickname: nickname,
      time: time,
      timestamp: Date.now()
    });
    return true; // Neue Bestzeit
  }
```

```
    return false; // Keine neue Bestzeit
  };
```

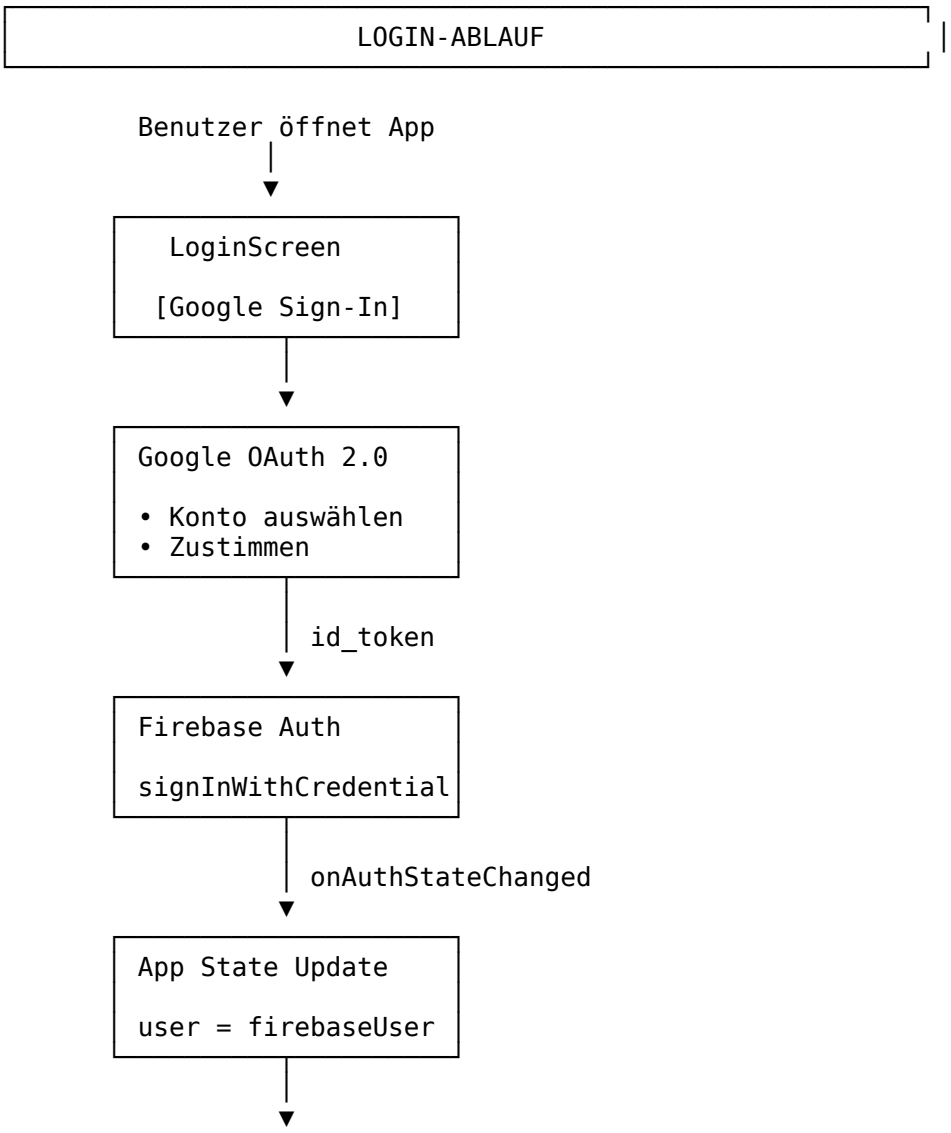
**Highscores laden (HighscoresScreen.tsx):**

```
  const fetchHighscores = async () => {
    const scoresRef = ref(database, 'scores');
    const snapshot = await get(scoresRef);

    if (snapshot.exists()) {
      const data = snapshot.val();
      const scores = Object.values(data)
        .sort((a, b) => a.time - b.time) // Aufsteigend nach Zeit
        .slice(0, 10);                   // Top 10
      return scores;
    }
    return [];
  };
```

# 3.6 6. Authentifizierung (Firebase Auth)

## 3.6.1 6.1 Ablaufdiagramm



```
  MenuScreen

Willkommen, {Name}!
```

### 3.6.2 6.2 Implementierung

**Datei:** `src/screens/LoginScreen.tsx`

```
        import { GoogleSignin } from '@react-native-google-signin/google-signin';
        import { GoogleAuthProvider, signInWithCredential, signInAnonymously } from
'firebase/auth';
        import { auth } from '../config/firebase';

        // Google Sign-In konfigurieren
        GoogleSignin.configure({
          webClientId: 'YOUR_WEB_CLIENT_ID.apps.googleusercontent.com'
        });

        // Google Sign-In
        const handleGoogleSignIn = async () => {
          try {
            await GoogleSignin.hasPlayServices();
            const response = await GoogleSignin.signIn();

            if (response.data?.idToken) {
              const credential = GoogleAuthProvider.credential(response.data.idToken);
              await signInWithCredential(auth, credential);
              onLogin(); // Navigation zum Menü
            }
          } catch (error) {
            // Fehlerbehandlung
          }
        };

        // Anonyme Anmeldung
        const handleAnonymousSignIn = async () => {
          try {
            await signInAnonymously(auth);
            onLogin(); // Navigation zum Menü
          } catch (error) {
            // Fehlerbehandlung
          }
        };

        // Gastmodus (ohne Auth)
        const handleGuestMode = () => {
          // Direkt zum Menü navigieren
          // Zeiten werden nicht gespeichert
          onLogin();
        };
```

**Datei:** `App.tsx` (Auth-State-Listener)

```
        useEffect(() => {
          const unsubscribe = onAuthStateChanged(auth, (user) => {
            if (user) {
              setUser(user);
```

```
            setCurrentScreen('Menu');
          } else {
            setUser(null);
            setCurrentScreen('Login');
          }
          setLoading(false);
        });

        return () => unsubscribe();
      }, []);
```

### 3.6.3 6.3 Fehlerfälle

| Fehler | Ursache | Behandlung |
|---|---|---|
| `auth/network-request-failed` | Keine Internetverbindung | Toast-Meldung anzeigen |
| `auth/cancelled-popup-request` | Benutzer bricht ab | Zurück zum Login-Screen |
| `auth/invalid-credential` | Ungültiges Token | Erneuter Login-Versuch |

## 3.7 7. Physik-Engine (matter-js)

### 3.7.1 7.1 Engine-Konfiguration

```
// Engine ohne Standard-Gravitation erstellen
const engine = Matter.Engine.create({
  gravity: { x: 0, y: 0, scale: 0.001 }
});

// Physik-Bodies erstellen
const ball = Matter.Bodies.circle(x, y, BALL_RADIUS, {
  restitution: 0.7,   // Abprall-Elastizität
  friction: 0.05,     // Reibung
  frictionAir: 0.02,  // Luftwiderstand
  label: 'ball'
});

const target = Matter.Bodies.circle(tx, ty, TARGET_RADIUS, {
  isStatic: true,
  isSensor: true,     // Nur Kollisionserkennung
  label: 'target'
});

// Kollisionserkennung
Matter.Events.on(engine, 'collisionStart', (event) => {
  event.pairs.forEach((pair) => {
    if (pair.bodyA.label === 'ball' && pair.bodyB.label === 'target') {
      setGameWon(true);
    }
  });
});
```

## 3.8 8. Hauptscreens und Komponenten

| Screen | Datei | Beschreibung |
|---|---|---|
| LoginScreen | `src/screens/LoginScreen.tsx` | Google Sign-In |
| MenuScreen | `src/screens/MenuScreen.tsx` | Hauptmenü, Nickname-Bearbeitung |
| GameScreen | `src/screens/GameScreen.tsx` | Spiellogik, Physik, Timer |
| ResultScreen | `src/screens/ResultScreen.tsx` | Ergebnisanzeige, Score-Speicherung |
| HighscoresScreen | `src/screens/HighscoresScreen.tsx` | Top 10 Bestenliste |
| SettingsScreen | `src/screens/SettingsScreen.tsx` | App-Einstellungen |

---

*Erstellt im Rahmen des Kompetenznachweises Modul 335*

# 4 Build APK mit EAS – Tilt Maze

## 4.1 1. Übersicht

Diese Anleitung beschreibt die Schritte zur Erstellung einer Android APK-Datei mit **Expo Application Services (EAS)**. Die APK kann direkt auf Android-Geräten installiert werden (ohne Google Play Store).

## 4.2 2. Voraussetzungen

### 4.2.1 2.1 Installierte Software

- **Node.js** (Version 18+)
- **npm** oder **yarn**
- **EAS CLI** (wird global installiert)

### 4.2.2 2.2 Konten

- **Expo-Konto** (kostenlos): expo.dev
- Optional: **Google Play Developer Account** (für Store-Veröffentlichung)

## 4.3 3. Einrichtung

### 4.3.1 3.1 EAS CLI installieren

```
npm install -g eas-cli
```

### 4.3.2 3.2 Bei Expo anmelden

```
eas login
```

Gib deine Expo-Zugangsdaten ein.

### 4.3.3 3.3 Projekt mit EAS verknüpfen (einmalig)

```
eas build:configure
```

Dies erstellt/aktualisiert die `eas.json`-Datei im Projektverzeichnis.

# 4.4 4. Build-Konfiguration

## 4.4.1 4.1 eas.json

Die Datei `eas.json` im Projektroot definiert die Build-Profile:

```
{
  "build": {
    "preview": {
      "distribution": "internal",
      "android": {
        "buildType": "apk"
      }
    }
  }
}
```

| Eigenschaft | Wert | Beschreibung |
|---|---|---|
| `distribution` | `internal` | APK für interne Verteilung (nicht Play Store) |
| `buildType` | `apk` | Erstellt eine direkt installierbare APK-Datei |

## 4.4.2 4.2 app.json (Auszug)

```
{
  "expo": {
    "name": "TiltMaze",
    "slug": "app-m335",
    "version": "1.0.0",
    "android": {
      "package": "com.riciyt.tiltmaze",
      "permissions": ["android.permission.SENSORS"],
      "adaptiveIcon": {
        "foregroundImage": "./assets/adaptive-icon.png",
        "backgroundColor": "#ffffff"
      }
    },
    "extra": {
      "eas": {
        "projectId": "ea02cbdc-02ce-4529-a804-2cfd1dcc00c9"
      }
    }
  }
}
```

# 4.5 5. APK erstellen

## 4.5.1 5.1 Build-Befehl (Preview-Profil)

```
eas build --platform android --profile preview
```

**Alternativer Kurzbefehl:**

```
eas build -p android --profile preview
```

### 4.5.2 5.2 Build-Prozess

1. **Upload**: Der Code wird zu Expo-Servern hochgeladen
2. **Queue**: Der Build wird in die Warteschlange eingereiht
3. **Build**: Die APK wird auf Expo-Servern kompiliert (~5-15 Minuten)
4. **Download**: Nach Abschluss erhältst du einen Download-Link

### 4.5.3 5.3 Build-Status prüfen

```
eas build:list
```

Oder im Browser: [expo.dev/accounts/[username]/builds](expo.dev/accounts/[username]/builds)

## 4.6 6. APK herunterladen und installieren

### 4.6.1 6.1 Download

Nach erfolgreichem Build erscheint ein Link in der Konsole:

```
✔ Build finished.

🤖 Android build: https://expo.dev/artifacts/eas/xxxxx.apk
```

Klicke auf den Link oder kopiere ihn in den Browser.

### 4.6.2 6.2 Installation auf Android-Gerät

1. **APK auf Gerät übertragen** (USB, Cloud, etc.)
2. **Installation aus unbekannten Quellen erlauben** (in den Einstellungen)
3. **APK-Datei öffnen** und «Installieren» tippen
4. **App starten**

## 4.7 7. Fehlerbehebung

### 4.7.1 7.1 Häufige Fehler

| Fehler | Lösung |
|---|---|
| `eas: command not found` | EAS CLI installieren: `npm install -g eas-cli` |
| `Not logged in` | `eas login` ausführen |
| `Missing projectId` | `eas build:configure` ausführen |
| Build-Timeout | Expo-Serverstatus prüfen: [status.expo.dev](status.expo.dev) |

### 4.7.2 7.2 Credentials

Für den ersten Build generiert EAS automatisch einen Debug-Keystore. Für Produktions-Builds kannst du eigene Keystores verwenden:

```
eas credentials
```

## 4.8 8. Alternative Build-Profile

### 4.8.1 8.1 Development Build (mit Dev Client)

```
{
  "build": {
    "development": {
      "developmentClient": true,
      "distribution": "internal"
    }
  }
}

eas build --platform android --profile development
```

### 4.8.2 8.2 Production Build (AAB für Play Store)

```
{
  "build": {
    "production": {
      "android": {
        "buildType": "app-bundle"
      }
    }
  }
}

eas build --platform android --profile production
```

## 4.9 9. Zusammenfassung der Befehle

| Aktion | Befehl |
|---|---|
| EAS CLI installieren | `npm install -g eas-cli` |
| Bei Expo anmelden | `eas login` |
| Projekt konfigurieren | `eas build:configure` |
| **APK erstellen** | `eas build --platform android --profile preview` |
| Build-Liste anzeigen | `eas build:list` |
| Credentials verwalten | `eas credentials` |

## 4.10 10. Weiterführende Links

- EAS Build Dokumentation
- eas.json Referenz
- Android-spezifische Konfiguration
- Expo Status

---

*Erstellt im Rahmen des Kompetenznachweises Modul 335*

# 5 Testbericht – Tilt Maze

## 5.1 1. Testübersicht

| Eigenschaft | Wert |
|---|---|
| **Testdatum** | Januar 2026 |
| **Tester** | Entwickler |
| **Getestete Version** | 1.0.0 |
| **Testgerät** | Android-Smartphone (physisch) |
| **OS-Version** | Android 14 |
| **Test-Umgebung** | APK Build (EAS Preview) |

# 5.2 2. Testergebnisse

## 5.2.1 2.1 Authentifizierung

| Nr. | Testfall | Ergebnis | Bemerkung |
|---|---|---|---|
| T01 | Google Sign-In | ✅ OK | Anmeldung funktioniert, Weiterleitung zum Menü |
| T02 | Logout | ✅ OK | Abmeldung erfolgreich, Login-Screen erscheint |
| T03 | Kein Internet beim Login | ✅ OK | Fehlermeldung wird angezeigt |

**Fazit Authentifizierung:** Alle Tests bestanden (3/3)

## 5.2.2 2.2 Navigation

| Nr. | Testfall | Ergebnis | Bemerkung |
|---|---|---|---|
| T04 | Menü zu Spiel | ✅ OK | Game-Screen lädt korrekt |
| T05 | Menü zu Highscores | ✅ OK | Highscores werden angezeigt |
| T06 | Zurück zum Menü | ✅ OK | Navigation funktioniert auf allen Screens |

**Fazit Navigation:** Alle Tests bestanden (3/3)

## 5.2.3 2.3 Spielmechanik

| Nr. | Testfall | Ergebnis | Bemerkung |
|---|---|---|---|
| T07 | Kugelsteuerung links | ✅ OK | Reaktion korrekt |
| T08 | Kugelsteuerung rechts | ✅ OK | Reaktion korrekt |
| T09 | Kugelsteuerung oben | ✅ OK | Reaktion korrekt |
| T10 | Kugelsteuerung unten | ✅ OK | Reaktion korrekt |
| T11 | Kollision mit Wand | ✅ OK | Kugel prallt ab |
| T12 | Ziel erreichen | ✅ OK | «You Won!» erscheint, Timer stoppt |
| T13 | Timer-Funktion | ✅ OK | Zählt korrekt in Echtzeit |
| T14 | Vibration bei Ereignis | ✅ OK | Haptisches Feedback funktioniert (wenn aktiviert) |

**Fazit Spielmechanik:** Alle Tests bestanden (8/8)

### 5.2.4 2.4 Steuerungseinstellungen

| Nr. | Testfall | Ergebnis | Bemerkung |
|---|---|---|---|
| T15 | Sensitivität ändern | ✅ OK | Änderung wirkt sich sofort aus |
| T16 | X-Achse invertieren | ✅ OK | Steuerung wird umgekehrt |

**Fazit Einstellungen:** Alle Tests bestanden (2/2)

### 5.2.5 2.5 Datenspeicherung

| Nr. | Testfall | Ergebnis | Bemerkung |
|---|---|---|---|
| T17 | Erste Bestzeit speichern | ✅ OK | «New Personal Best!» wird angezeigt |
| T18 | Neue Bestzeit (schneller) | ✅ OK | Zeit wird aktualisiert |
| T19 | Keine neue Bestzeit | ✅ OK | Alte Bestzeit bleibt erhalten |
| T20 | Nickname speichern | ✅ OK | Nickname erscheint in Highscores |

**Fazit Datenspeicherung:** Alle Tests bestanden (4/4)

### 5.2.6 2.6 Bestenliste

| Nr. | Testfall | Ergebnis | Bemerkung |
|---|---|---|---|
| T21 | Highscores laden | ✅ OK | Top 10 werden korrekt sortiert angezeigt |
| T22 | Leere Bestenliste | ✅ OK | Hinweis erscheint |
| T23 | Podium-Anzeige | ✅ OK | Gold/Silber/Bronze-Hervorhebung funktioniert |

**Fazit Bestenliste:** Alle Tests bestanden (3/3)

### 5.2.7 2.7 Sonderfälle / Edge Cases

| Nr. | Testfall | Ergebnis | Bemerkung |
|---|---|---|---|
| T24 | App in Hintergrund | ✅ OK | Spielzustand wird zurückgesetzt |
| T25 | Schnelle Gerätebewegungen | ✅ OK | Kugel verhält sich stabil |
| T26 | Offline-Modus | ✅ OK | Fehlermeldung bei Speicherversuch |

**Fazit Edge Cases:** Alle Tests bestanden (3/3)

## 5.3 3. Gesamtergebnis

| Bereich | Bestanden | Gesamt | Prozent |
|---|---|---|---|
| Authentifizierung | 3 | 3 | 100% |
| Navigation | 3 | 3 | 100% |
| Spielmechanik | 8 | 8 | 100% |
| Einstellungen | 2 | 2 | 100% |

| Bereich | Bestanden | Gesamt | Prozent |
|---|---|---|---|
| Datenspeicherung | 4 | 4 | 100% |
| Bestenliste | 3 | 3 | 100% |
| Edge Cases | 3 | 3 | 100% |
| **Total** | **26** | **26** | **100%** |

# 5.4 4. Gefundene und behobene Fehler

## 5.4.1 4.1 Fehler während der Entwicklung (behoben)

| Nr. | Beschreibung | Lösung | Status |
|---|---|---|---|
| B01 | Kugel reagierte nicht intuitiv auf Neigung | INVERT_X/Y Optionen implementiert | ✅ Behoben |
| B02 | Jitter bei ruhig gehaltenem Gerät | Deadzone-Funktion implementiert | ✅ Behoben |
| B03 | Ungleichmässige Bewegung | Tiefpassfilter (Smoothing) hinzugefügt | ✅ Behoben |
| B04 | Z-Achse beeinflusste Steuerung | Z-Achse explizit ignoriert | ✅ Behoben |
| B05 | Timer nicht präzise genug | Update-Intervall auf 100ms gesetzt | ✅ Behoben |

## 5.4.2 4.2 Bekannte Einschränkungen

| Nr. | Beschreibung | Workaround |
|---|---|---|
| L01 | Accelerometer funktioniert nicht auf Emulatoren | Physisches Gerät verwenden |
| L02 | Web-Version ohne Sensor-Unterstützung | Nur auf mobilen Geräten testen |

# 5.5 5. Nicht-funktionale Tests

| Nr. | Testfall | Ergebnis | Messwert |
|---|---|---|---|
| N01 | Performance (60s Spielzeit) | ✅ OK | Konstant >30 FPS |
| N02 | Reaktionszeit Sensor | ✅ OK | <100ms |
| N03 | Firebase-Antwortzeit | ✅ OK | ~1-2 Sekunden |

# 5.6 6. Screenshots

## 5.6.1 6.1 Login-Screen

Der Login-Screen zeigt den Google Sign-In Button mit Neon-Cyan Design.

## 5.6.2 6.2 Game-Screen

Das Spielfeld zeigt die Kugel (cyan), das Ziel (grün) und die Maze-Wände.

### 5.6.3 6.3 Highscores-Screen

Die Bestenliste zeigt die Top 10 mit Podium-Hervorhebung.

*Hinweis: Screenshots wurden während der manuellen Tests erstellt und können auf Anfrage bereitgestellt werden.*

## 5.7 7. Fazit

Die App «Tilt Maze» hat alle 26 funktionalen Testfälle bestanden. Die Implementierung entspricht den Anforderungen des Kompetenznachweises:

- ✅ **2 Sensoren/Aktoren**: Accelerometer und Vibration
- ✅ **Persistente Speicherung**: Firebase Realtime Database
- ✅ **Authentifizierung**: Firebase Auth mit Google Sign-In
- ✅ **Funktionale App**: Alle Features arbeiten wie spezifiziert

**Gesamtstatus: BESTANDEN**

---

*Erstellt im Rahmen des Kompetenznachweises Modul 335*