

7. Firebase - Realtime Database



Ziele dieser Unterrichtseinheit

- Sie kennen die grundlegenden Funktionen von Firebase
- Sie erstellen eine Realtime DB
- Sie erstellen Regeln für den Zugriff
- Sie können Daten speichern, anzeigen, löschen und updaten

Inhaltsverzeichnis

1. Einführung
2. Quick Start

7.1 – Firebase Infos

Firestore

https://www.youtube.com/watch?v=s8H_HrBJb44

7.2 – Quick Start

Firestore – QuickStart – Beispiel von Modle

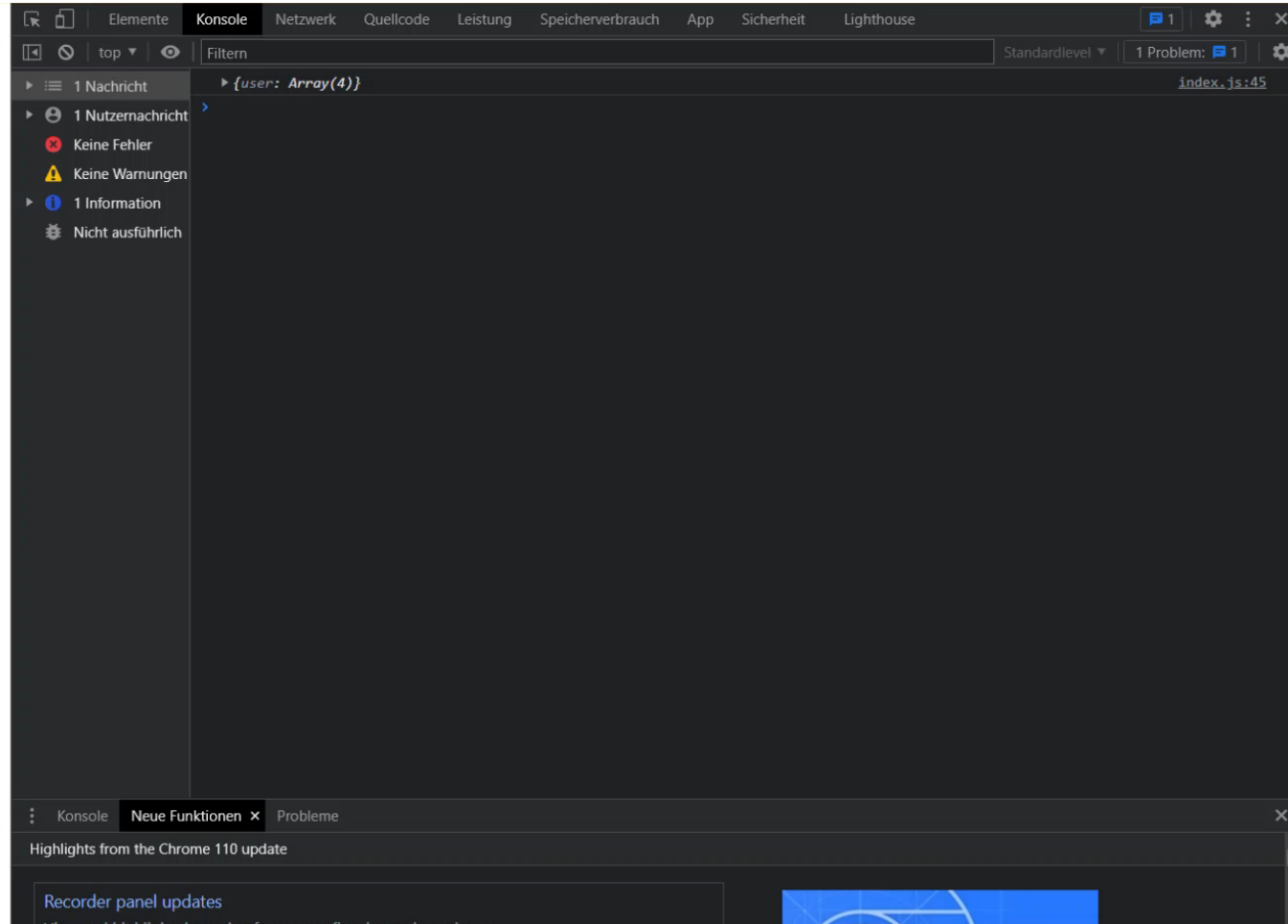
🏠

Nr.: 0 / Michi

Nr.: 1 / Paul

Nr.: 2 / Harald

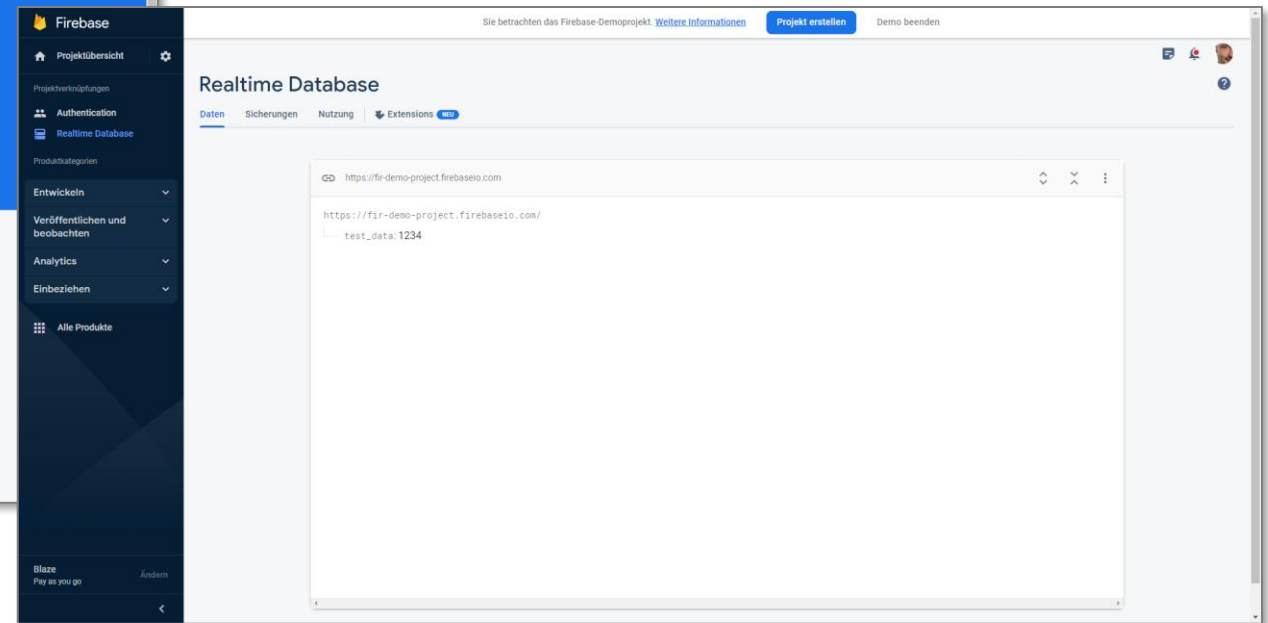
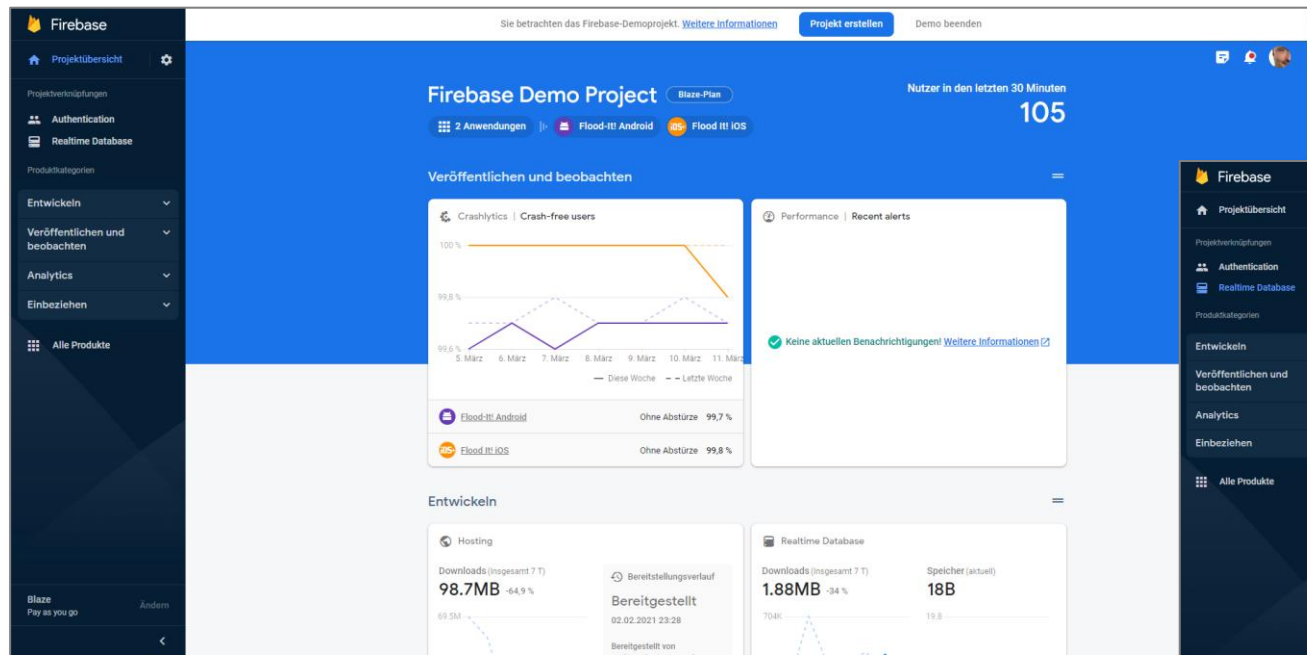
Nr.: 3 / Kurt



Firebase anmelden

<https://firebase.google.com/>

<https://console.firebase.google.com/project/fir-demo-project/overview>



Projekt erstellen

× Projekt erstellen(Schritt 1 von 3)

Geben Sie zuerst einen Namen für Ihr Projekt ein[?]

Projektname eingeben

Eine eindeutige Kennung für Ihr Projekt

Weiter

× Projekt erstellen(Schritt 1 von 3)

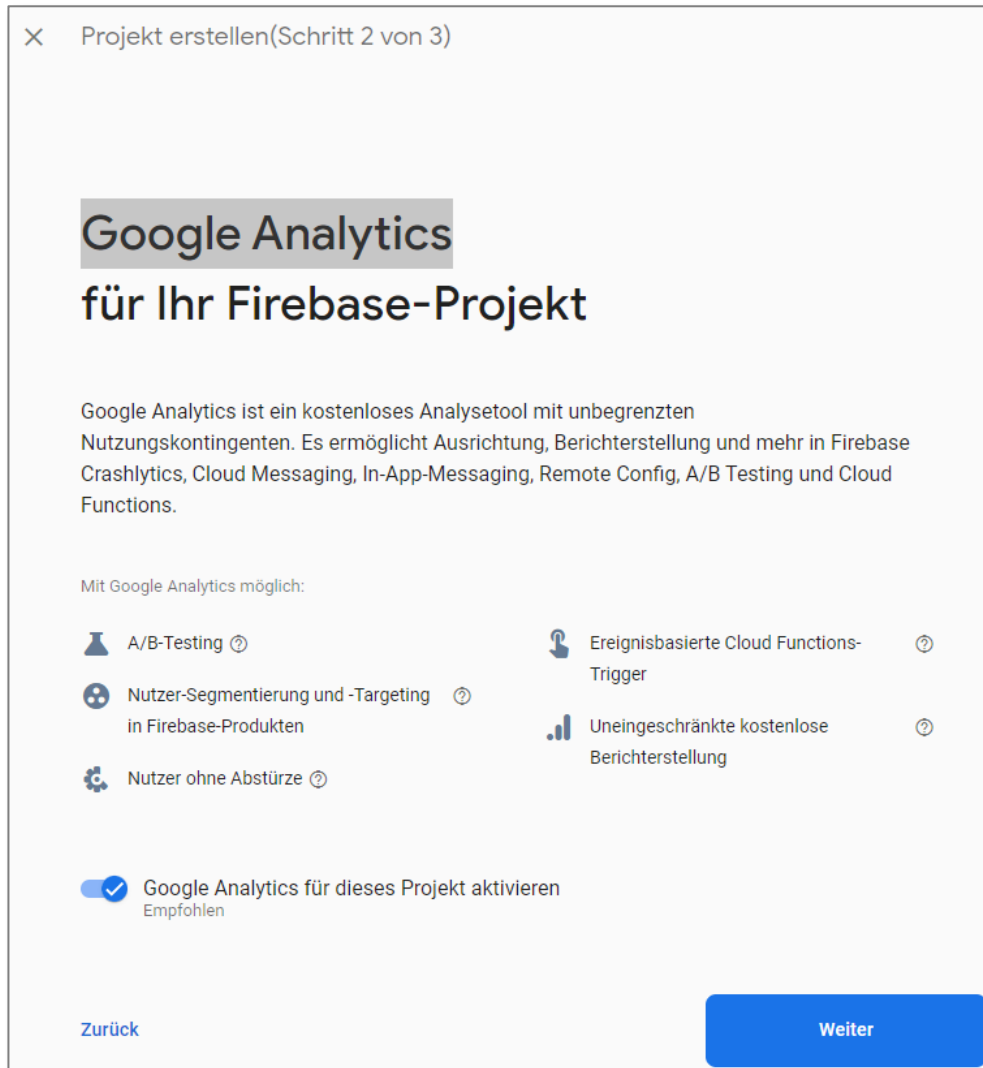
Geben Sie zuerst einen Namen für Ihr Projekt ein[?]

Projektname

 temp001-db53e

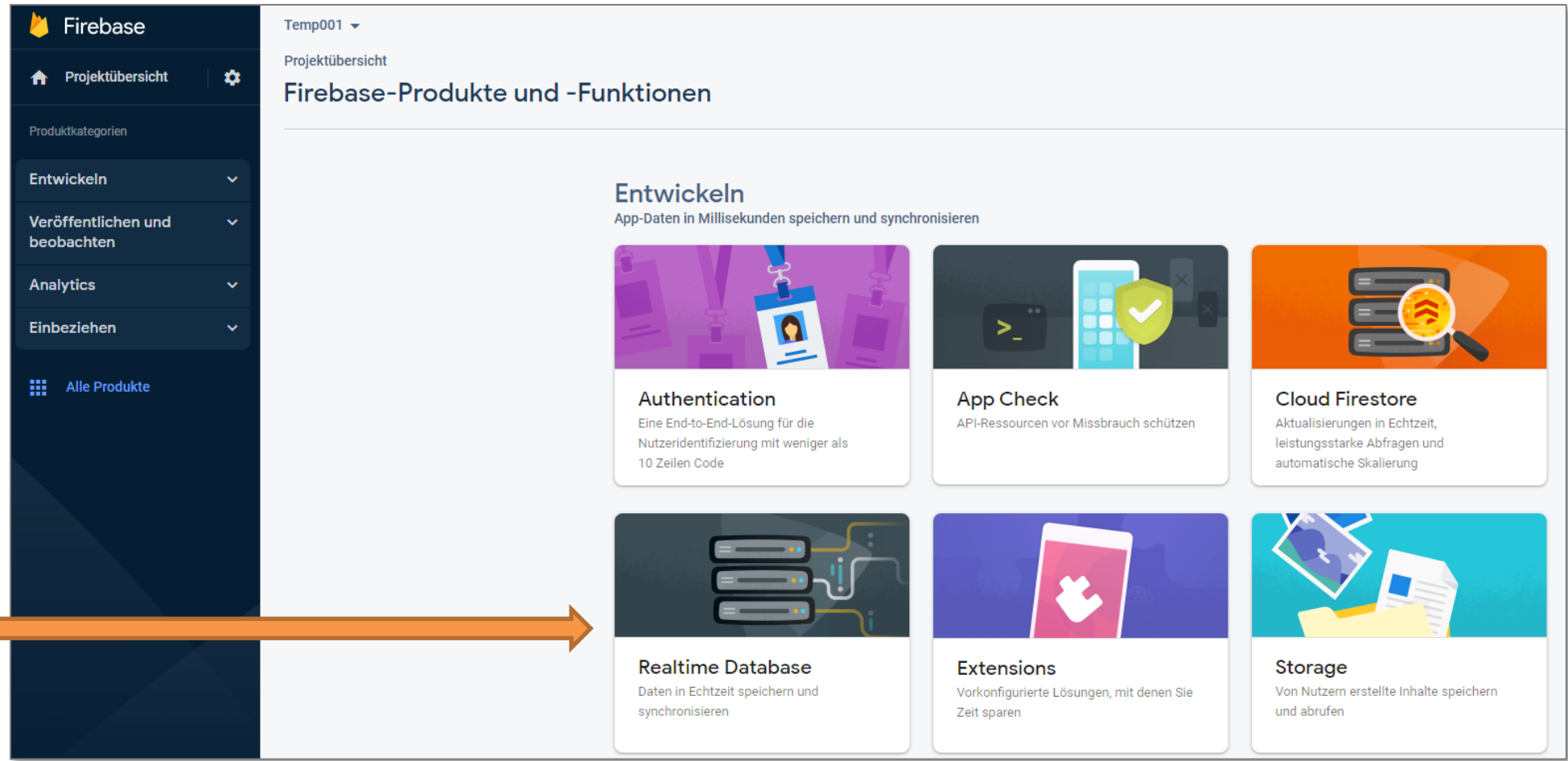
Weiter

Google Analytics ?



Google Analytics ist ein kostenloses Webanalyse-Tool, das von Google entwickelt wurde und Website-Besitzern und -Administratoren hilft, Daten über den Traffic auf ihrer Website zu sammeln und zu analysieren. Das Tool ermöglicht es Benutzern, verschiedene Informationen über ihre Website-Besucher zu verfolgen, wie zum Beispiel die Anzahl der Besucher, ihre Herkunft, die Seiten, die sie besucht haben, wie lange sie auf der Website verbracht haben und vieles mehr.

Realtime – Database erstellen

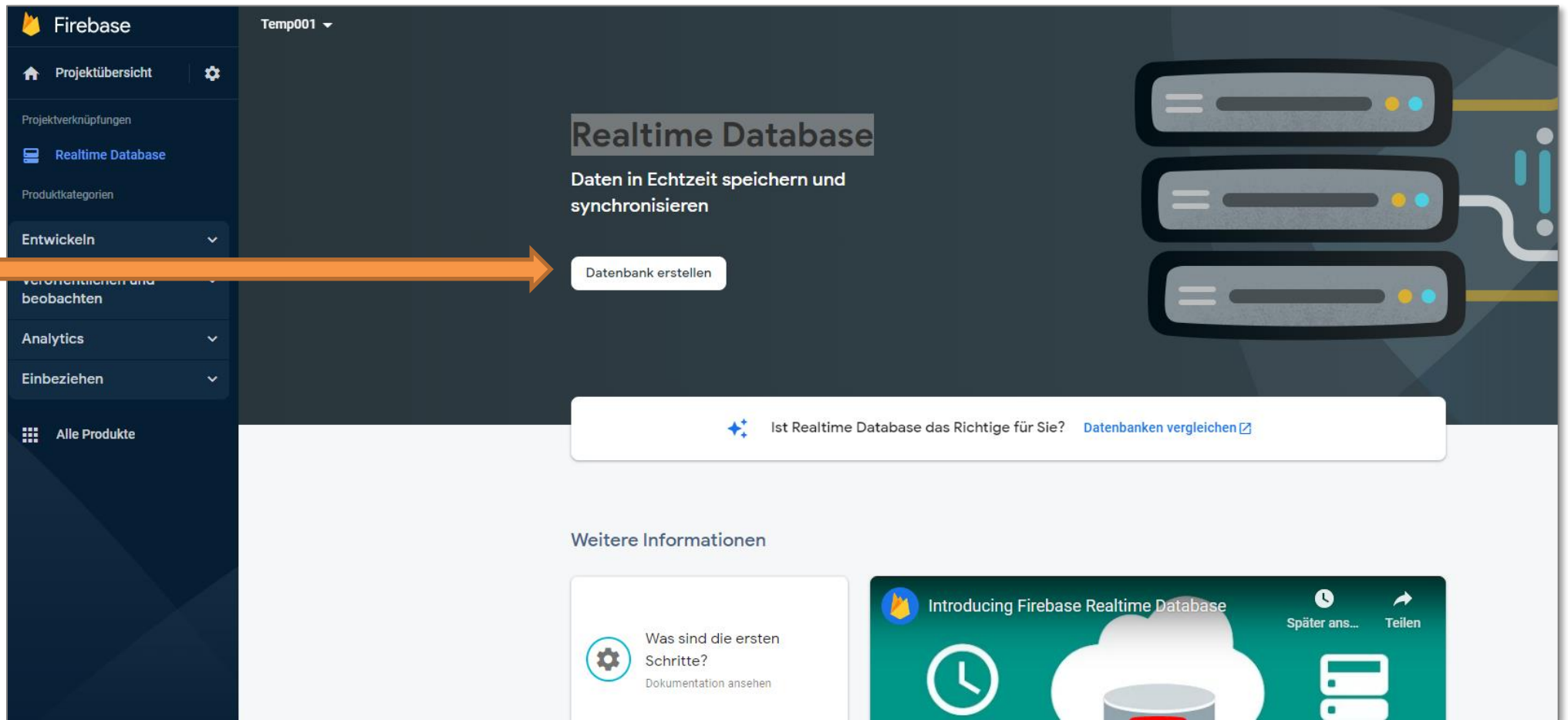


The screenshot shows the Firebase console interface. On the left is a dark sidebar with the 'Firebase' logo and navigation links: 'Projektübersicht', 'Entwickeln', 'Veröffentlichen und beobachten', 'Analytics', 'Einbeziehen', and 'Alle Produkte'. The main area is titled 'Temp001' and 'Projektübersicht', with a subtitle 'Firebase-Produkte und -Funktionen'. Under the 'Entwickeln' section, there are six service cards: Authentication, App Check, Cloud Firestore, Realtime Database, Extensions, and Storage. Each card includes an icon, a title, and a brief description.

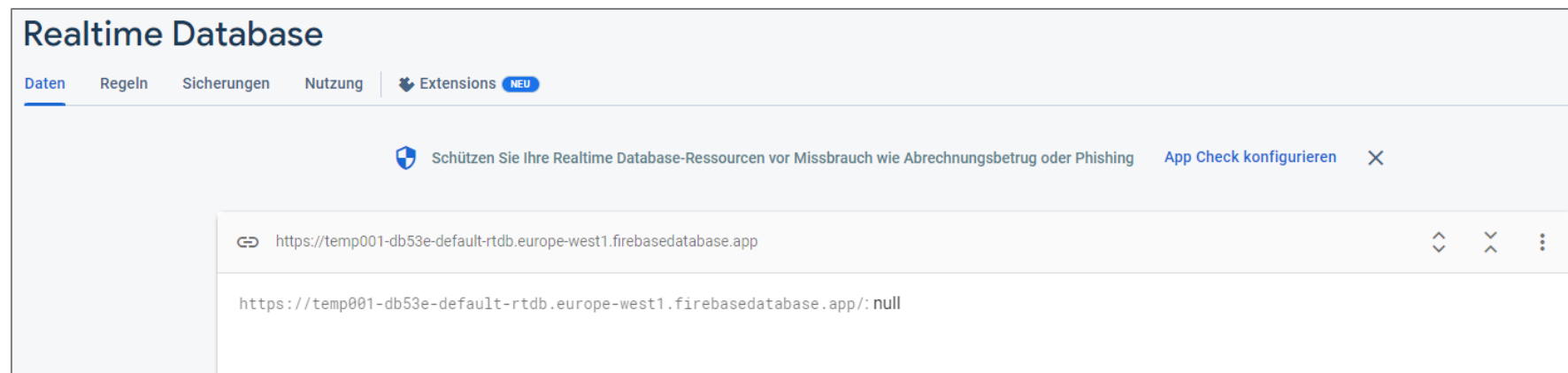
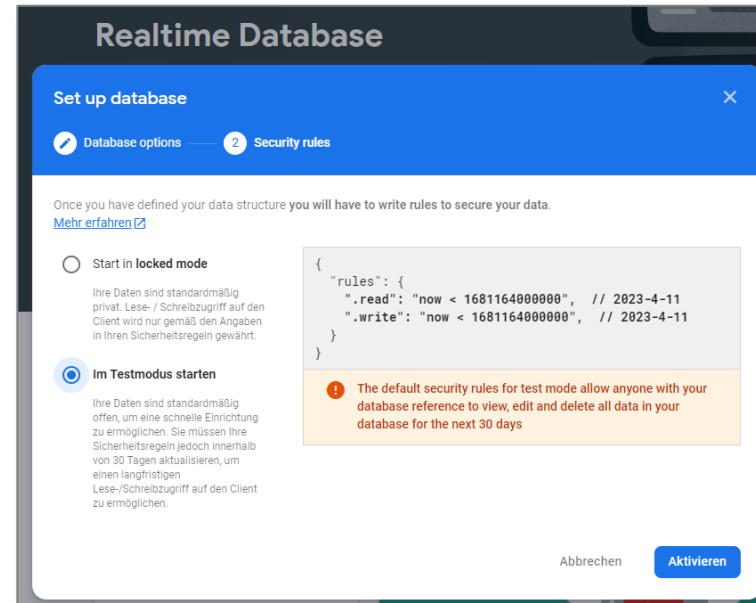
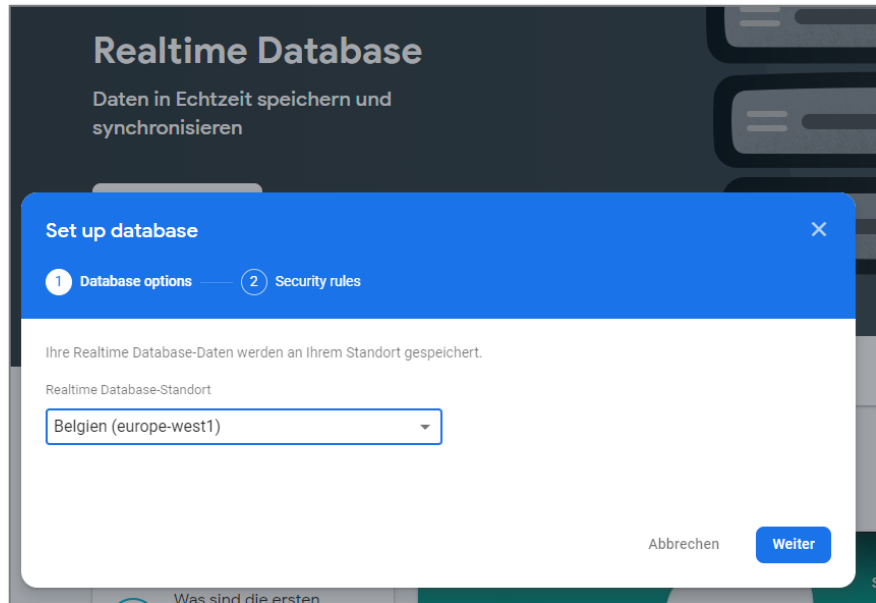
Entwickeln
App-Daten in Millisekunden speichern und synchronisieren

- Authentication**
Eine End-to-End-Lösung für die Nutzeridentifizierung mit weniger als 10 Zeilen Code
- App Check**
API-Ressourcen vor Missbrauch schützen
- Cloud Firestore**
Aktualisierungen in Echtzeit, leistungsstarke Abfragen und automatische Skalierung
- Realtime Database**
Daten in Echtzeit speichern und synchronisieren
- Extensions**
Vorkonfigurierte Lösungen, mit denen Sie Zeit sparen
- Storage**
Von Nutzern erstellte Inhalte speichern und abrufen

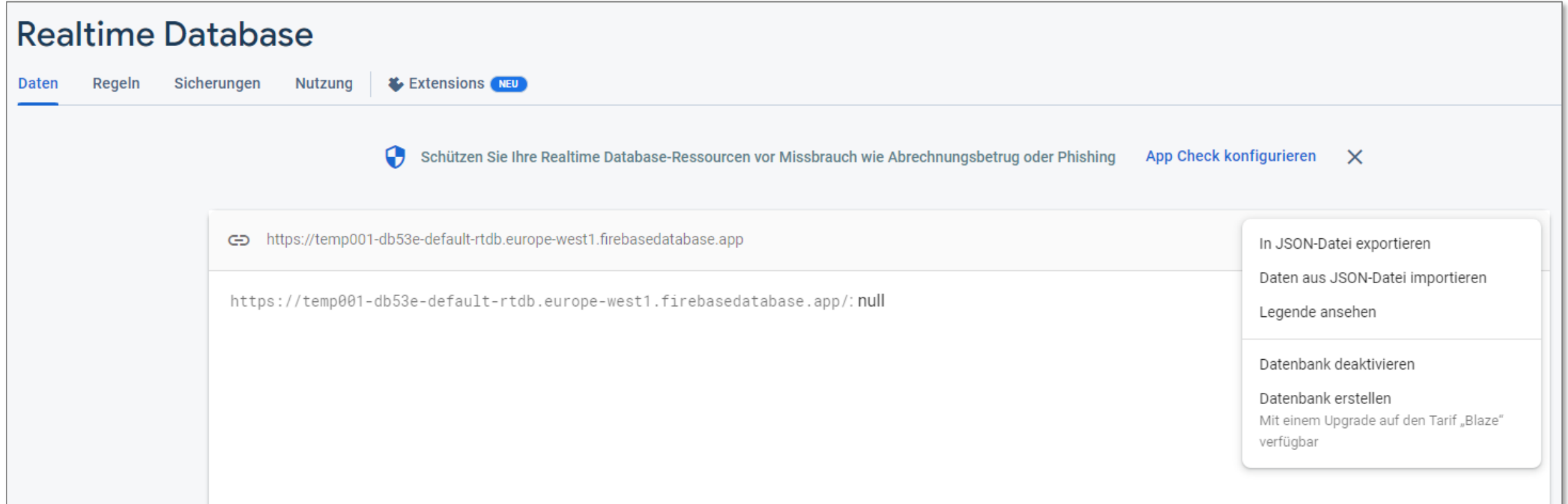
Realtime – Database erstellen



Realtime – Database erstellen



Realtime – Database JSON importieren



The screenshot shows the Firebase Realtime Database console interface. At the top, there's a navigation bar with tabs: **Daten** (selected), Regeln, Sicherungen, and Nutzung. To the right of these tabs is an **Extensions** button with a 'NEU' (New) badge. Below the navigation bar, a security warning message is displayed: 'Schützen Sie Ihre Realtime Database-Ressourcen vor Missbrauch wie Abrechnungsbetrug oder Phishing', followed by a link to 'App Check konfigurieren' and a close button. The main content area shows the database URL: `https://temp001-db53e-default-rtdb.europe-west1.firebaseio.com`. Below this, there's a text input field containing `https://temp001-db53e-default-rtdb.europe-west1.firebaseio.com/: null`. On the right side of the console, a context menu is open, listing several actions: 'In JSON-Datei exportieren', 'Daten aus JSON-Datei importieren', 'Legende ansehen', 'Datenbank deaktivieren', and 'Datenbank erstellen'. The last two options have additional text: 'Mit einem Upgrade auf den Tarif „Blaze“ verfügbar'.

JSON von Moodle – QuickStart – Beispiel laden

Realtime Database – Regeln (timestamp)

Temp001 ▼

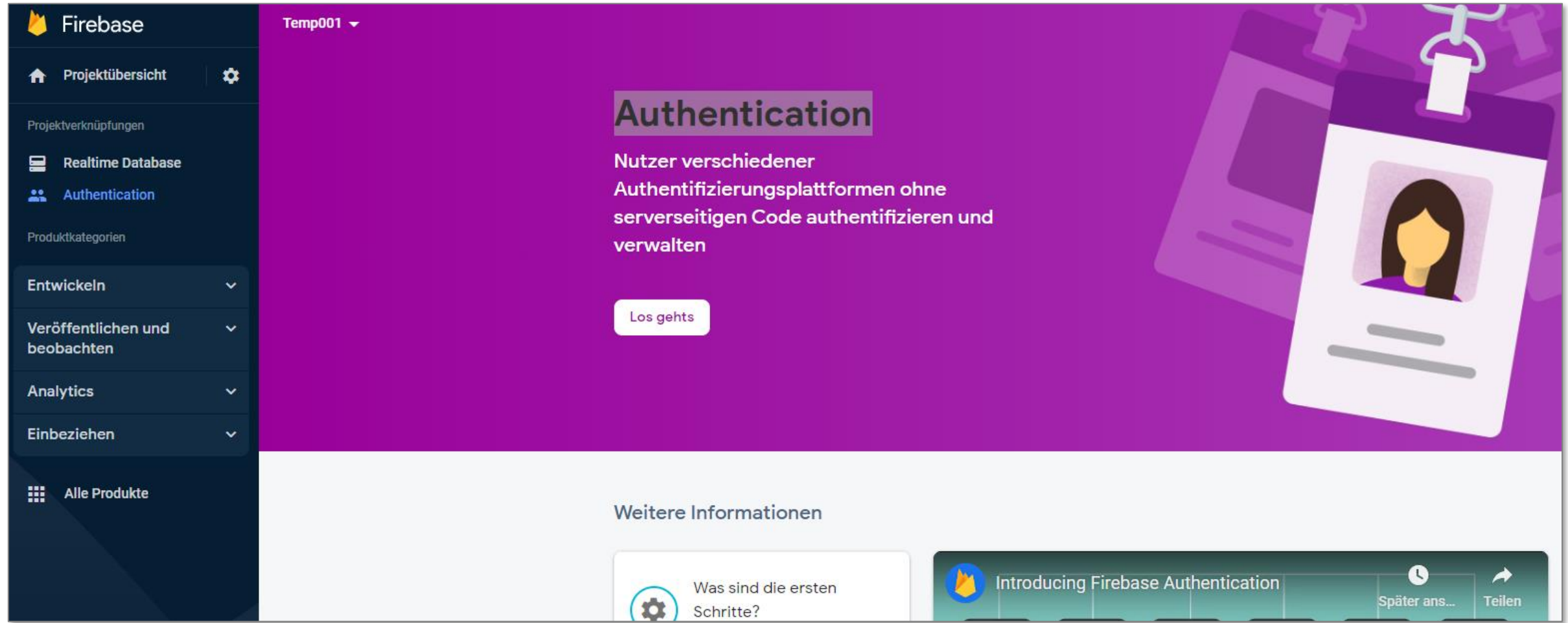
Realtime Database

Daten Regeln Sicherungen Nutzung Extensions **NEU**

Regeln bearbeiten Regeln überwachen

```
1 {  
2   "rules": {  
3     ".read": "now < 1681164000000", // 2023-4-11  
4     ".write": "now < 1681164000000", // 2023-4-11  
5   }  
6 }
```

Authentication



The screenshot shows the Firebase Authentication console. On the left is a dark sidebar with the 'Authentication' link highlighted. The main area has a purple header with the title 'Authentication' and a description: 'Nutzer verschiedener Authentifizierungsplattformen ohne serverseitigen Code authentifizieren und verwalten'. Below this is a 'Los gehts' button. The bottom section, titled 'Weitere Informationen', contains a card for 'Was sind die ersten Schritte?' and a video player for 'Introducing Firebase Authentication'.

Authentication

Nutzer verschiedener Authentifizierungsplattformen ohne serverseitigen Code authentifizieren und verwalten

Los gehts

Weitere Informationen

Was sind die ersten Schritte?

Introducing Firebase Authentication

Später ans... Teilen

Authentication

The screenshot shows the Firebase Authentication console for project 'Temp001'. The left sidebar contains navigation links: Projektübersicht, Realtime Database, Authentication, Produktkategorien, Entwickeln, Veröffentlichen und beobachten, Analytics, Einbeziehen, and Alle Produkte. The main content area is titled 'Authentication' and has tabs for Users, Sign-in method (selected), Templates, Usage, Settings, and Extensions (marked 'NEU'). Under the 'Sign-in method' tab, the section 'Anbieter für Anmeldungen' displays a list of providers categorized into 'Native Anbieter', 'Zusätzliche Anbieter', and 'Individuelle Anbieter'. The 'Anonym' provider under 'Native Anbieter' is highlighted with an orange arrow pointing to a modal dialog.

Erste Schritte mit Firebase Auth: Erste Anmeldemethode hinzufügen

Native Anbieter	Zusätzliche Anbieter	Individuelle Anbieter
E-Mail-Adresse/Passwort	Google	Facebook
Telefon	Game Center	Play Spiele
Anonym	Apple	GitHub
	Microsoft	Twitter
		Yahoo!
		OpenID Connect
		SAML

Anonym ☒ Aktivieren

Aktivieren Sie in Ihrer Anwendung anonyme Gastkonten, mit denen Sie nutzerspezifische Sicherheits- und Firebase-Regeln durchsetzen können, ohne dass dazu Anmeldedaten Ihrer Nutzer erforderlich sind. [Weitere Informationen](#)

Abbrechen **Speichern**

Projekteinstellungen

Projekteinstellungen

Temp001

Projektübersicht | Projekteinstellungen | Nutzer und Berechtigungen | Tagging | Integrationen | Dienstkonten | Datenschutz | Nutzer und Berechtigungen | Nutzung und Abrechnung

Ihr Projekt

Projektname	Temp001
Projekt-ID	temp001-db53e
Projektnummer	984671453000
Standardmäßiger GCP-Ressourcenstandort	Noch nicht ausgewählt
Web-API-Schlüssel	AlZaSyDoYmsdeVXoQHwgc47IUC56IFqoBXABeZs

Umgebung

Mit dieser Einstellung wird Ihr Projekt für verschiedene Phasen des App-Lebenszyklus angepasst.

Meine Apps

In Ihrem Projekt befinden sich keine Anwendungen

Wählen Sie eine Plattform aus, um zu beginnen

iOS+ Android **Web** Electron Flutter

WEB – APP registrieren

×

Firestore zu meiner Web-App hinzufügen

1

App registrieren

App-Nickname ⓘ

Test001

☐

Firestore Hosting für diese App einrichten. [Weitere Informationen](#)

Das Hosting kann auch später eingerichtet werden. Dafür fallen nie Kosten an.

App registrieren

2

Firestore SDK hinzufügen

×

Firestore zu meiner Web-App hinzufügen

✓

App registrieren

2

Firestore SDK hinzufügen

☒ npm verwenden ☐ <script>-Tag verwenden

Wenn Sie schon [npm](#) und einen Module Bundler wie [Webpack](#) oder [Rollup](#) verwenden, können Sie mit dem folgenden Befehl das neueste SDK installieren ([weitere Informationen](#)):

\$ npm install firebase

Initialisieren Sie anschließend Firestore und verwenden Sie die SDKs für die gewünschten Produkte.

```
// Import the functions you need from the SDKs you need
import { initializeApp } from "firebase/app";
import { getAnalytics } from "firebase/analytics";
// TODO: Add SDKs for Firebase products that you want to use
// https://firebase.google.com/docs/web/setup#available-libraries

// Your web app's Firebase configuration
// For Firebase JS SDK v7.20.0 and later, measurementId is optional
const firebaseConfig = {
  apiKey: "AIzaSyDoYmsdeVXoQHwgc47lUC56IFqo8XABeZs",
  authDomain: "temp001-db53e.firebaseio.com",
  databaseURL: "https://temp001-db53e-default-rtdb.europe-west1.firebaseio.com",
  projectId: "temp001-db53e",
  storageBucket: "temp001-db53e.appspot.com",
  messagingSenderId: "984671453000",
  appId: "1:984671453000:web:bcafd8abaeaa7b09553d7d",
  measurementId: "G-H59W8WGN4F"
};

// Initialize Firebase
const app = initializeApp(firebaseConfig);
const analytics = getAnalytics(app);
```

Hinweis: Bei dieser Option wird das [modulare JavaScript SDK](#) verwendet, da dieses SDK kleiner ist.

























Hier finden Sie weitere Informationen zu Firestore für das Web: [Startleitfaden](#), [Web SDK API-Referenz](#), [Beispiele](#)

Weiter zur Konsole

Moodle – QuickStart nun mit Ihrer APP verbinden

```
const firebaseConfig = {  
  apiKey: "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx",  
  authDomain: "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx",  
  databaseURL: "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx",  
  projectId: "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx",  
  storageBucket: "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx",  
  messagingSenderId: "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx",  
  appId: "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"  
};
```

Auftrag: Auto – Liste mit Firebase – DB darstellen (45')

id	name	kraftstoff	farbe	bauart	tank	action
1	BMW	Diesel	 green	SUV	60	 tanken  edit  delete
2	Mercedes	Benzin	 grey	Cabrio	60	 tanken  edit  delete
3	Opel	Diesel	 red	Transporter	60	 tanken  edit  delete
4	BMW	Diesel	 black	PW	60	 tanken  edit  delete
5	Mercedes	Benzin	 grey	SUV	60	 tanken  edit  delete
6	Opel	Diesel	 red	Lastwagen	60	 tanken  edit  delete

7.3 – Firebase Realtime DB

Firestore import

```
import {
  initializeApp
} from "https://www.gstatic.com/firebasejs/9.10.0/firebase-app.js";
import {
  getDatabase,
  ref,
  child,
  get,
  set,
  push,
  update
} from 'https://www.gstatic.com/firebasejs/9.10.0/firebase-database.js';
```

firebase.database()

Diese Funktion gibt Ihnen Zugriff auf die Firebase-Datenbank. Sie können sie verwenden, um eine Verbindung zur Datenbank herzustellen und Daten zu lesen oder zu schreiben.

```
// Initialisierung der Firebase-App
firebase.initializeApp(config);

// Zugriff auf die Datenbank
const database = firebase.database();
```


ref()

Die Funktion `ref()` wird verwendet, um eine Referenz auf einen bestimmten Pfad in der Datenbank zu erhalten. Sie können damit auf bestimmte Daten oder Unterpfade in der Datenbank zugreifen.

```
const dataRef = ref(database, 'path/to/data');
```

child()

Die Funktion `child()` wird verwendet, um eine Referenz auf einen untergeordneten Pfad relativ zu einem übergeordneten Pfad zu erstellen. Sie wird normalerweise zusammen mit der `ref()`-Funktion verwendet.

```
import { ref, child } from 'firebase/database';  
  
const parentRef = ref(database, 'path/to');  
const childRef = child(parentRef, 'subpath');
```

get()

Die Funktion `get()` wird verwendet, um Daten von der Datenbank abzurufen. Sie gibt ein Promise-Objekt zurück, das die abgerufenen Daten enthält.

```
const dataRef = ref(database, 'path/to/data');
get(dataRef).then((snapshot) => {
  const data = snapshot.val();
  // Hier können Sie die abgerufenen Daten verarbeiten
}).catch((error) => {
  // Fehlerbehandlung, falls das Abrufen der Daten fehlschlägt
  console.error(error);
});
```

push()

Diese Funktion wird verwendet, um neue Daten in der Datenbank zu speichern. Sie generiert einen eindeutigen Schlüssel für den neuen Datensatz.

push wird verwendet, um eine neue Datenreferenz innerhalb eines Knotens zu erstellen und neue Daten an diese Referenz zu schreiben. Jede neue Referenz erhält **eine neue eindeutige ID** und die neuen Daten werden unter dieser Referenz gespeichert.

```
const newRecordRef = database.ref('path/to/data').push();  
newRecordRef.set({ name: 'John', age: 25 });
```

set()

Mit dieser Funktion können Sie Daten in der Datenbank speichern oder vorhandene Daten aktualisieren. Sie überschreibt die vorhandenen Daten unter dem angegebenen Pfad.

set wird verwendet, um Daten an einen bestimmten Knoten zu schreiben. Wenn Sie set aufrufen, überschreiben Sie alle Daten an diesem Knoten mit den neuen Daten, die Sie angeben. Wenn der Knoten nicht existiert, wird er erstellt.

```
const newRecordRef = database.ref('path/to/data').push();
newRecordRef
  .set({ name: 'John', age: 25 })
  .then(() => {
    // Data saved successfully!
  })
  .catch((error) => {
    // The write failed...
  });
```

update()

Diese Funktion wird verwendet, um bestimmte Teile der Daten in der Datenbank zu aktualisieren, ohne den Rest der Daten zu überschreiben.

```
database.ref('path/to/data').update({ age: 26 });
```

remove()

Mit dieser Funktion können Sie Daten aus der Datenbank löschen.

```
database.ref('path/to/data').remove();
```

on(), onValue()

Diese Funktion ermöglicht es Ihnen, einen **Event-Listener** an einen bestimmten Pfad in der Datenbank anzuhängen, um **Änderungen in Echtzeit** zu verfolgen.

```
database.ref('path/to/data').on('value', snapshot => {  
  const data = snapshot.val();  
  // Hier können Sie die Daten verarbeiten  
});
```

```
const dataRef = ref(database, 'path/to/data');  
onValue(dataRef, (snapshot) => {  
  const data = snapshot.val();  
  // Hier können Sie die Daten verarbeiten, wenn sich etwas ändert  
});
```


child_added

Der Event-Typ "child_added" wird ausgelöst, wenn ein neues untergeordnetes Element zu einem bestimmten Pfad hinzugefügt wird. Dies kann beim Initialisieren der Listener-Funktion oder immer dann nützlich sein, wenn Sie an neuen Daten interessiert sind, die zu einem bestimmten Pfad hinzugefügt werden.

```
const dataRef = ref(database, 'path/to/data');
onChildAdded(dataRef, (snapshot) => {
  const newChildData = snapshot.val();
  // Hier können Sie die neu hinzugefügten Daten verarbeiten
});
```

child_changed

Der Event-Typ "child_changed" wird ausgelöst, wenn ein vorhandenes untergeordnetes Element unter einem bestimmten Pfad geändert wird. Dies kann nützlich sein, wenn Sie Änderungen an bestehenden Daten verfolgen möchten.

```
const dataRef = ref(database, 'path/to/data');
onChildChanged(dataRef, (snapshot) => {
  const changedChildData = snapshot.val();
  // Hier können Sie die geänderten Daten verarbeiten
});
```

child_removed

Der Event-Typ "child_removed" wird ausgelöst, wenn ein untergeordnetes Element unter einem bestimmten Pfad entfernt wird. Dies ist nützlich, um zu erfahren, wenn Daten gelöscht wurden.

```
const dataRef = ref(database, 'path/to/data');
onChildRemoved(dataRef, (snapshot) => {
  const removedChildData = snapshot.val();
  // Hier können Sie die gelöschten Daten verarbeiten
});
```

INFO

Diese Event-Typen werden normalerweise zusammen mit der **on()**-Funktion verwendet, um **Event-Listener** an einen bestimmten Pfad in der Firebase Realtime Database anzuhängen. Sie können die entsprechende Event-Funktion (z. B. `onChildAdded()`, `onChildChanged()`, `onChildRemoved()`) verwenden und die Daten innerhalb der Listener-Funktion verarbeiten.

Es ist wichtig zu beachten, dass diese Event-Typen nur für untergeordnete Elemente gelten und nicht für die übergeordnete Struktur. Wenn Sie Änderungen an **der übergeordneten Struktur** verfolgen möchten, sollten Sie stattdessen den **Event-Typ "value"** verwenden.

Serverseitige Anwendung mit dem Firebase Admin SDK

Das Firebase Admin SDK bietet erweiterte Funktionen und Berechtigungen für serverseitige Anwendungen. Es ermöglicht Ihnen die Verwaltung und Konfiguration von Firebase-Projekten sowie den Zugriff auf verschiedene Firebase-Dienste, darunter die Firebase Realtime Database, das Cloud Firestore, die Authentifizierung, das Cloud Messaging und vieles mehr.

Serverseitige Anwendung mit dem Firebase Admin SDK

1. Datenbankregeln umgehen
2. Benutzerkonten verwalten
3. Anmeldeoptionen einrichten
4. Projekte konfigurieren
5. Cloud Messaging-Benachrichtigungen an Geräte

Serverseitige Anwendung mit dem Firebase Admin SDK

Das Admin SDK bietet erweiterte Funktionen und Berechtigungen für serverseitige Operationen, wie das Verwalten von Firebase-Projekten, das Lesen/Schreiben von Datenbankinhalten und das Authentifizieren von Benutzern.

Die Funktion **getDatabase()** wird im Firebase Admin SDK für serverseitige Anwendungen verwendet, um eine Instanz der Firebase Realtime Database zu erhalten.

```
const admin = require('firebase-admin');  
// Initialisierung der Firebase-App im Admin SDK  
admin.initializeApp({  
  credential: admin.credential.applicationDefault(),  
  databaseURL: 'https://your-project-id.firebaseio.com'  
});  
// Zugriff auf die Datenbank mit getDatabase()  
const database = admin.getDatabase();
```

Serverseitige Anwendung mit dem Firebase Admin SDK

Es ist wichtig zu beachten, dass das Admin SDK eine **privilegierte Verbindung zur Firebase-Infrastruktur** darstellt und spezielle Berechtigungen erfordert. Es sollte normalerweise **nicht im clientseitigen Webbrowser-Code** verwendet werden, da es zu Sicherheitsproblemen führen kann. Für clientseitige Anwendungen wird stattdessen das Firebase JavaScript SDK verwendet, das über `firebase.database()` eine Verbindung zur Realtime Database herstellt.

Serverseitige Anwendung mit dem Firebase Admin SDK

Es ist wichtig zu beachten, dass das **Firebase Admin SDK** mit privilegierten Berechtigungen und Funktionen kommt, die für den **serverseitigen Einsatz** gedacht sind. Es sollte sorgfältig verwendet und angemessene Sicherheitsvorkehrungen getroffen werden, um sicherzustellen, dass es nur auf vertrauenswürdigen Servern und in sicheren Umgebungen verwendet wird.

Weitere Informationen und detaillierte Anleitungen zur Verwendung des Firebase Admin SDK finden Sie in der offiziellen Firebase-Dokumentation unter:

<https://firebase.google.com/docs/admin/setup?hl=de>

Auftrag: CRUD – Funktionen umsetzen (70')

- insert umsetzen mit push (20')
- update umsetzen mit set (25')
ACHTUNG: die ID (der Key) muss eventuell ins Formular mit übergeben werden, damit dieser Wert bei speichern vorhanden ist.
- Delete und Tanken umsetzen mit set (25')
Frage: Muss immer der ganze Datensatz geschrieben werden?
- Zusatzaufgaben
 - Datumsfeld
 - Beschreibung