

Lesen und Schreiben in eine Datenbank

Die Komponente **Realtime DB** ermöglicht es aus einer Datenbank zu lesen und zu schreiben. Thinkable verwendet **Firebase** von Google als Datenbank.

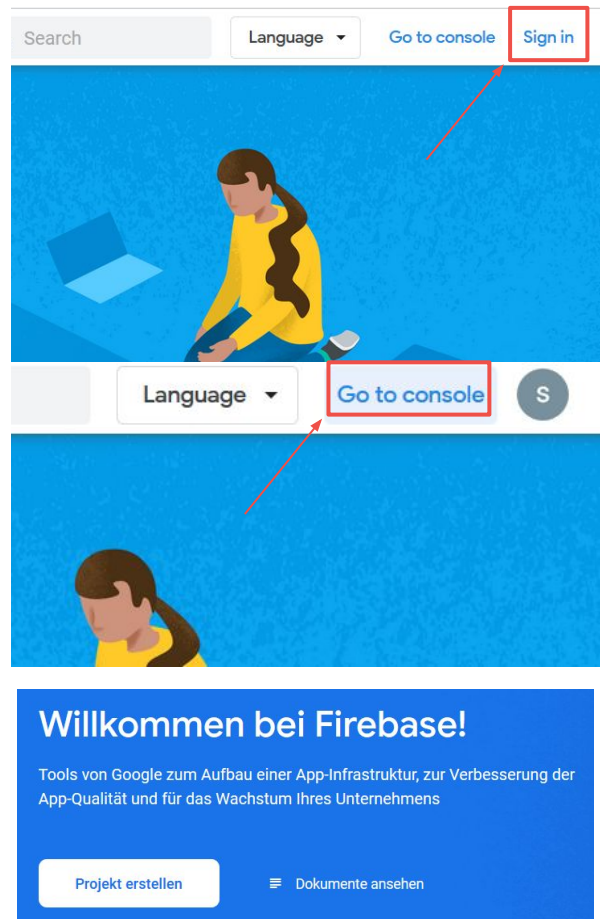


Schritt 1: Anlegen der Datenbank

Gehe auf <https://firebase.google.com/> und melde dich rechts oben mit deiner Mailadresse an.

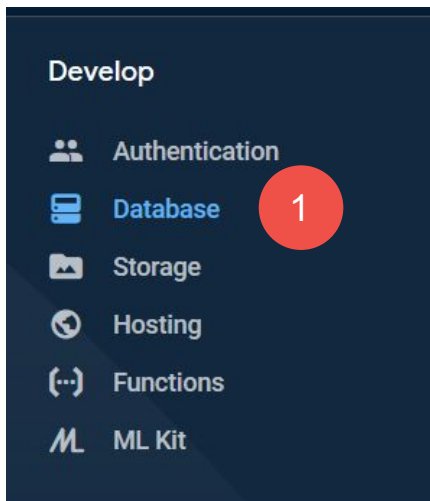
Klicke nach der Anmeldung auf den Button "Go to console".

Lege ein neues Projekt mit dem Button "Projekt erstellen" an und nenne es am besten wie deine App.

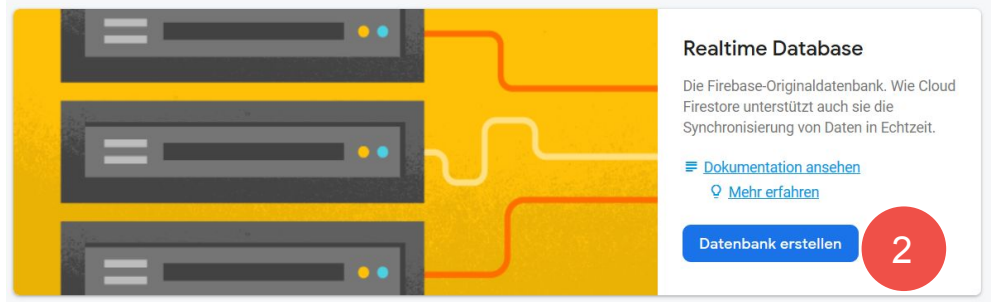


Schritt 2: Datenbank aktivieren

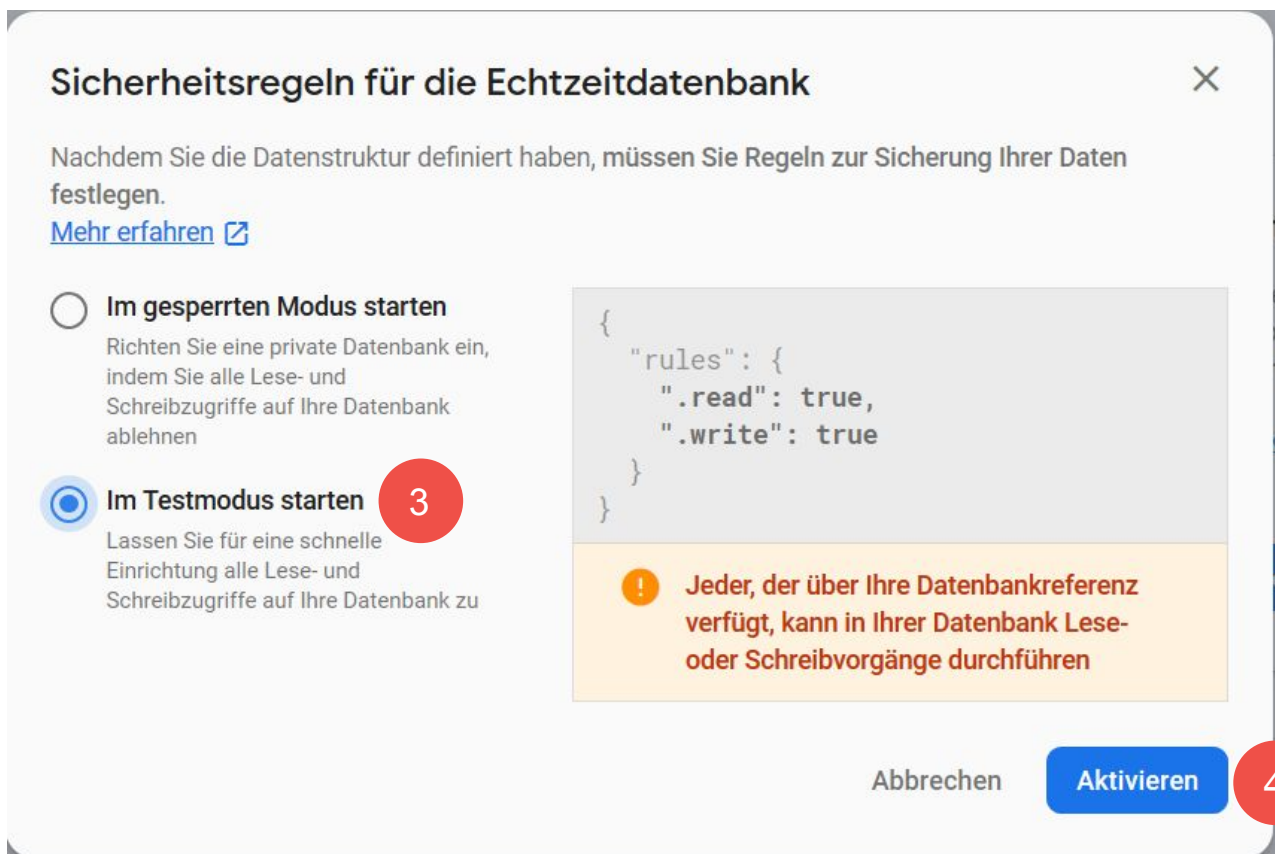
Klicke im Menü links auf "Database". Scrolle im rechten Fenster weiter runter, bis du die "Realtime Database" findest. Klicke auf "Datenbank erstellen".



Oder entscheiden Sie sich für Realtime Database



Klicke als nächsten bei den Sicherheitsregeln auf "Im Testmodus starten" und dann auf "Aktivieren".



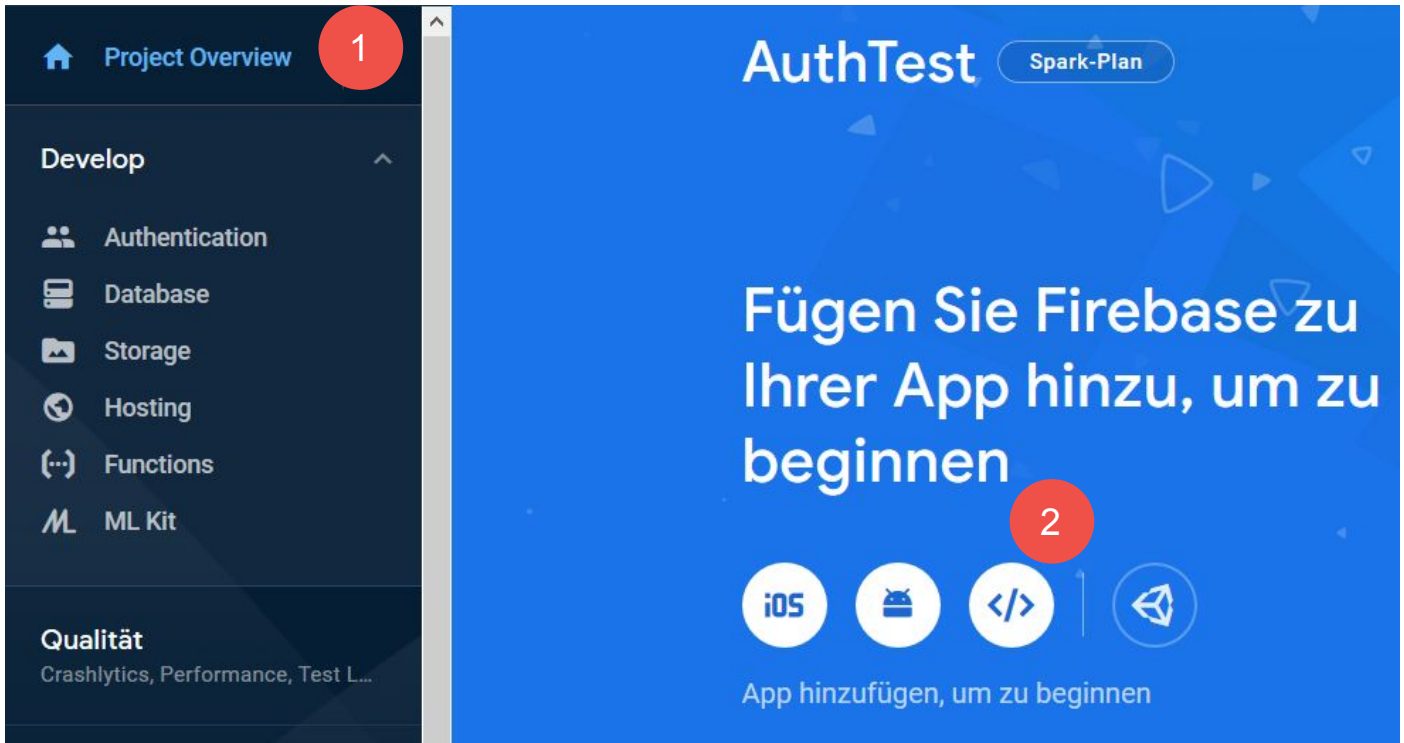
Schritt 3: Daten anlegen

Nun kannst du Daten in deine Datenbank schreiben. Klicke auf das + im Hauptknoten der Datenbank, um einen neuen Datensatz hinzuzufügen.



Schritt 4: Firebase mit Thinkable verknüpfen

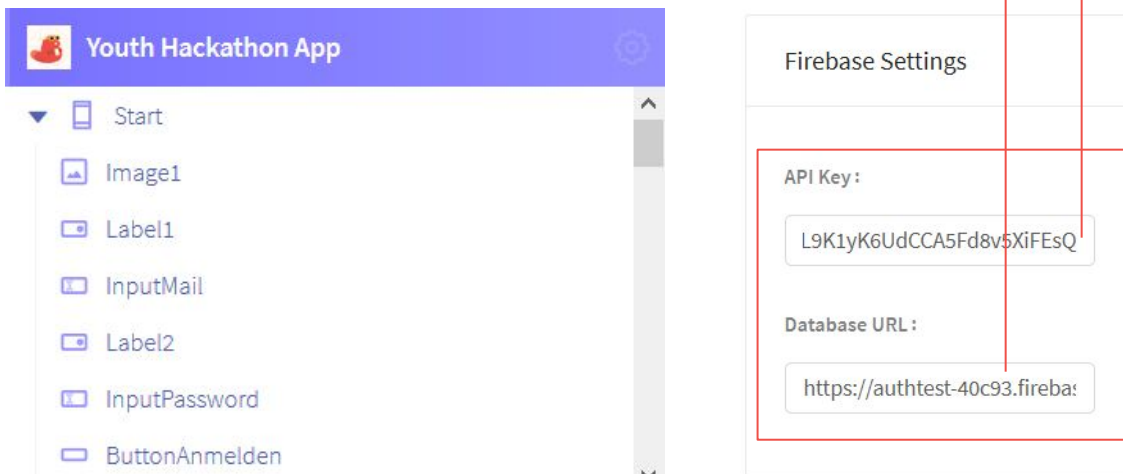
Klick im Menü links auf "Project Overview" und dann auf das Symbol "</>". Registriere deine App.



Nach dem Registrieren erhält du ein Skript. Wichtig ist es, die Werte in "apiKey" und "databaseURL" in die Thinkable-Einstellungen zu laden. Diese findest du unter "Firebase Settings", wenn du auf den Titel deiner App im Abschnitt links drückst.

```
<script>
// Your web app's Firebase configuration
var firebaseConfig = {
  apiKey: "AIzaSyCp17GvhU1L9K1yK6UdCCA5Fd8v5XiFEsQ",
  authDomain: "authtest-40c93.firebaseio.com",
  databaseURL: "https://authtest-40c93.firebaseio.com",
  projectId: "authtest-40c93",
  storageBucket: "authtest-40c93.appspot.com",
  messagingSenderId: "601504874020",
  appId: "1:601504874020:web:3b72077cfbfe67fd70a2b1"
};

```



Schritt 5: Daten lesen

Realtime DB nutzt einige Blöcke, um Daten lesen und zu schreiben. "Key" ist der Name des gespeicherten Werts ("Highscore") und Value der Wert an sich.

```

when Screen1 Opens
do
  in Realtime_DB1 call Get
  key "Highscore"
  with outputs
  error
  value
  then do
    from LabelHighscore set Text to value
  
```

Lesen aus der DB.

```

when ButtonSpeichern Click
do
  in Realtime_DB1 call Save
  key "Highscore"
  value from InputHighscore get Text
  with output
  error
  then do
    when Save is done
  
```

Schreiben in die DB.

Realtime DB erlaubt auch, auf Änderungen in der Datenbank zu reagieren. Dazu muss man vorher einen "Listener" auf den "Key" registrieren. Mit dem Block "DataChanged" kann nun bei Veränderung des Werts das Programm aktualisiert werden, zum Beispiel konstante Aktualisierung eines Labels.

```

when Screen1 Opens
do
  in Realtime_DB1 call AddListener
  key "Highscore"
  with output
  error
  then do
    when AddListener is done
  
```

```

when Realtime_DB1 DataChanged
do
  key
  value
  from LabelHighscore set Text to value
  
```