An Exploration of Firebase

Rich Frost & Sal Trupiano

CSI-5450: Database Systems I

Paper Presentation/Demonstration

Overview

- What is Firebase?
- Core Feature Sets of Firebase
- Comparison of Firebase NoSQL vs. SQL RDBMS
- Sample Firebase Queries
- Firebase Research Spotlight: Real-Time Communication Network
- Sample Database Demonstration
- References

What is Firebase?

- Firebase Realtime Database is a NoSQL Cloud-Hosted Database
- Data in Firebase is stored in JSON Format instead of Tables
  - Access to Data is done using Keys that map to Values, which are stored as Objects inside the DB.
  - No querying for CRUD Operations.
- Data is synced in Realtime to each client that is connected to the database
- Works from a numerous number of clients, such as Android, iOS, etc.
- Has the ability to use REST APIs for communication and initiation

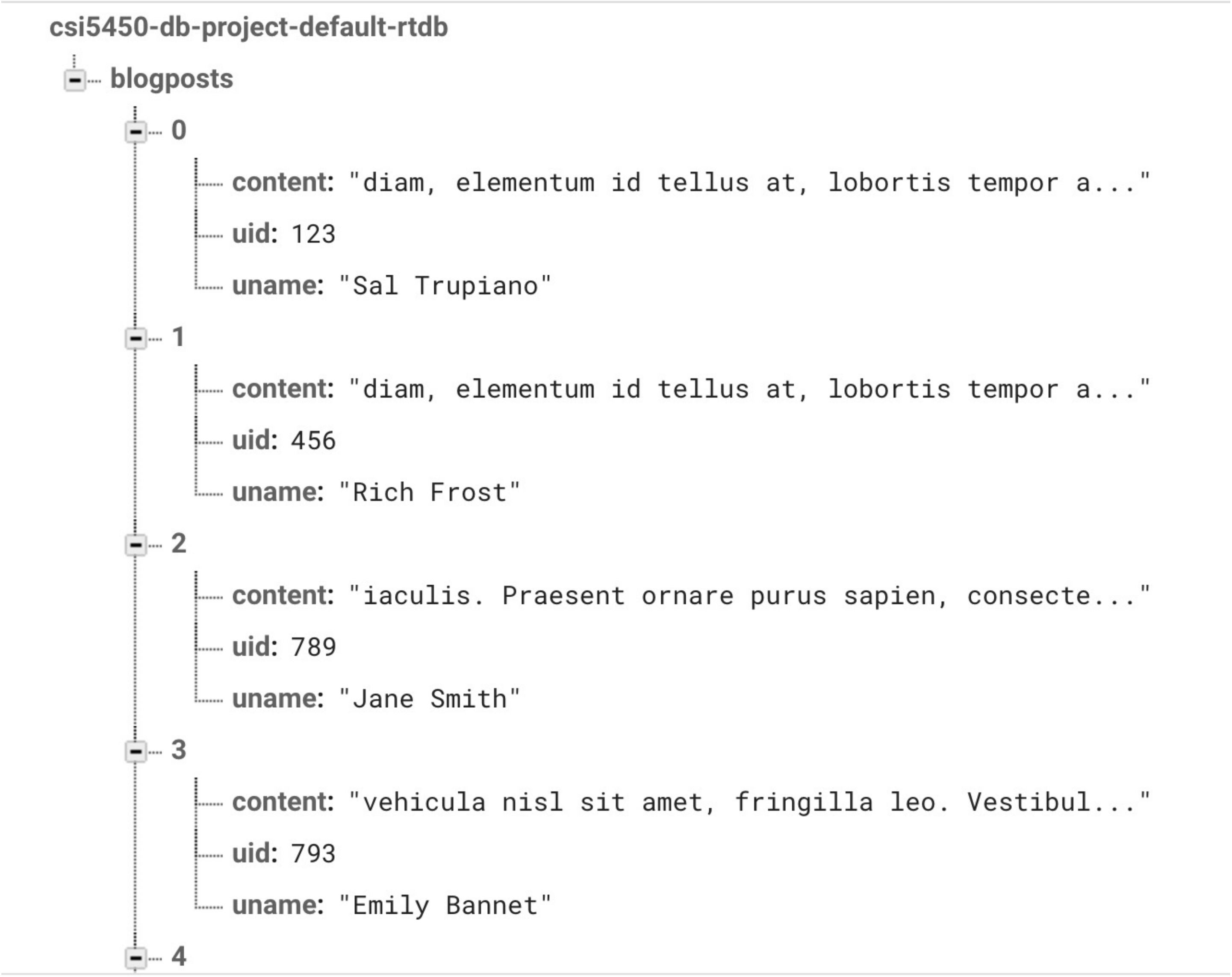
Firebase Core Feature Sets

- Storage
  - Storing large files in the cloud easily, with minimal overhead.
  - Easily-accessible remote storage for each client.
- Authentication
  - Ability to integrate directly with identity providers through APIs.
- Messaging
  - Real-Time Database Functionality allows projects to send/receive notifications and messages in real time; data stored and retrieved almost instantly.
- Offline Availability
  - If someone loses their data connection, SDKs provide utilization of the device’s local cache to provide functionality, and then synchronization takes place whenever the connection is restored.

Firebase vs. SQL: The Differences

Category	Firebase (NoSQL)	SQL (RDBMS)
Data Storage	Stored as a JSON Tree	Stored in a Relational Model (Table) as Rows and Columns
Schema Flexibility	Dynamic Schema: The structure of the Database is changeable at any time, without a predefined structure for the data	Fixed Schema: Data cannot be added, updated, or deleted without resulting in access going offline temporarily
Data Types	Data does not have to be of a specified type or structure	Data must be declared as a known type, and this type must be known to the SQL RDBMS in advance
Data Storage/Retrieval Method	Data is synchronized between server and client continuously	Data is retrieved, updated, inserted, or deleted based on a query provided.
Supported Languages	Java, JavaScript, Swift	C++, Go, Java, PHP, Python, Ruby, VB, Delphi

Firebase Structure



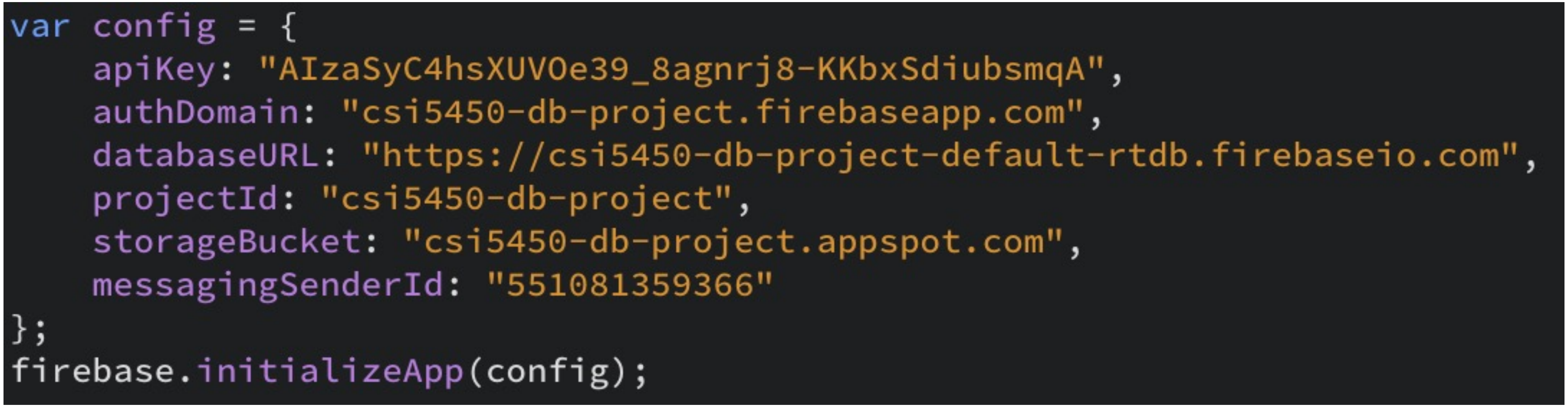
```
csi5450-db-project-default-rtdb
  blogposts
    0
      content: "diam, elementum id tellus at, lobortis tempor a..."
      uid: 123
      uname: "Sal Trupiano"
    1
      content: "diam, elementum id tellus at, lobortis tempor a..."
      uid: 456
      uname: "Rich Frost"
    2
      content: "iaculis. Praesent ornare purus sapien, consecte..."
      uid: 789
      uname: "Jane Smith"
    3
      content: "vehicula nisl sit amet, fringilla leo. Vestibul..."
      uid: 793
      uname: "Emily Bannet"
    4
```

- One JSON file structure
- Queries executed on the Database using JavaScript
- Typical format for NoSQL Database

Firebase Interaction

Firebase Queries

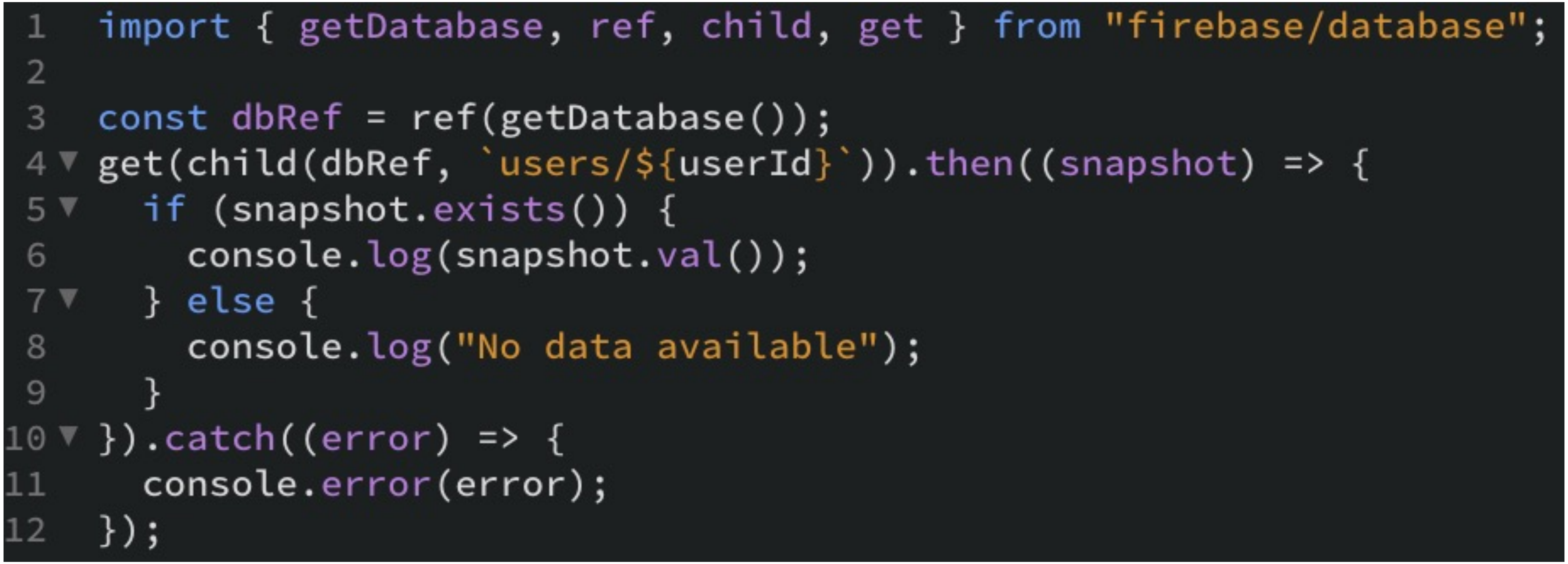
- Connecting to Firebase:



```
var config = {
    apiKey: "AIzaSyC4hsXUVOe39_8agnrj8-KKbxSdiubsmqA",
    authDomain: "csi5450-db-project.firebaseapp.com",
    databaseURL: "https://csi5450-db-project-default-rtdb.firebaseio.com",
    projectId: "csi5450-db-project",
    storageBucket: "csi5450-db-project.appspot.com",
    messagingSenderId: "551081359366"
};
firebase.initializeApp(config);
```

Firebase Queries

- Reading All Data form a Users Database:

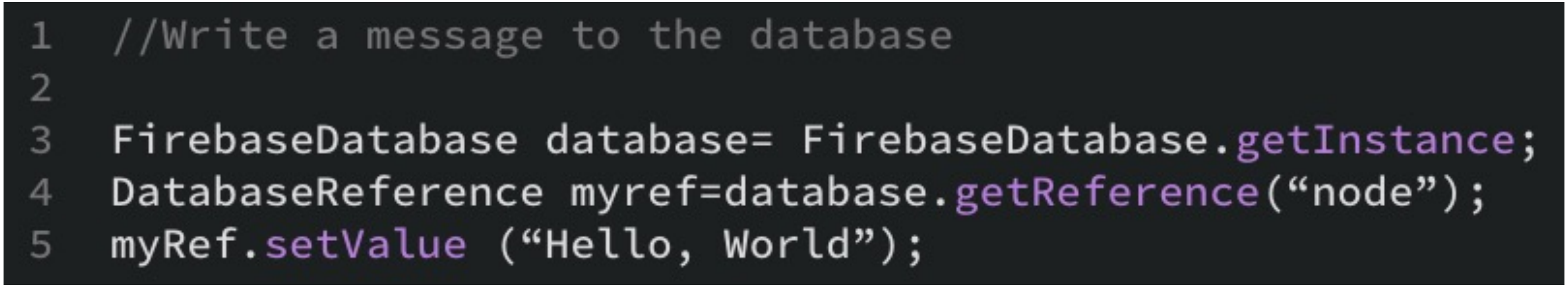


```
import { getDatabase, ref, child, get } from "firebase/database";

const dbRef = ref(getDatabase());
get(child(dbRef, `users/${userId}`)).then((snapshot) => {
  if (snapshot.exists()) {
    console.log(snapshot.val());
  } else {
    console.log("No data available");
  }
}).catch((error) => {
  console.error(error);
});
```

Firebase Queries

- Writing Simple Text to Firebase:

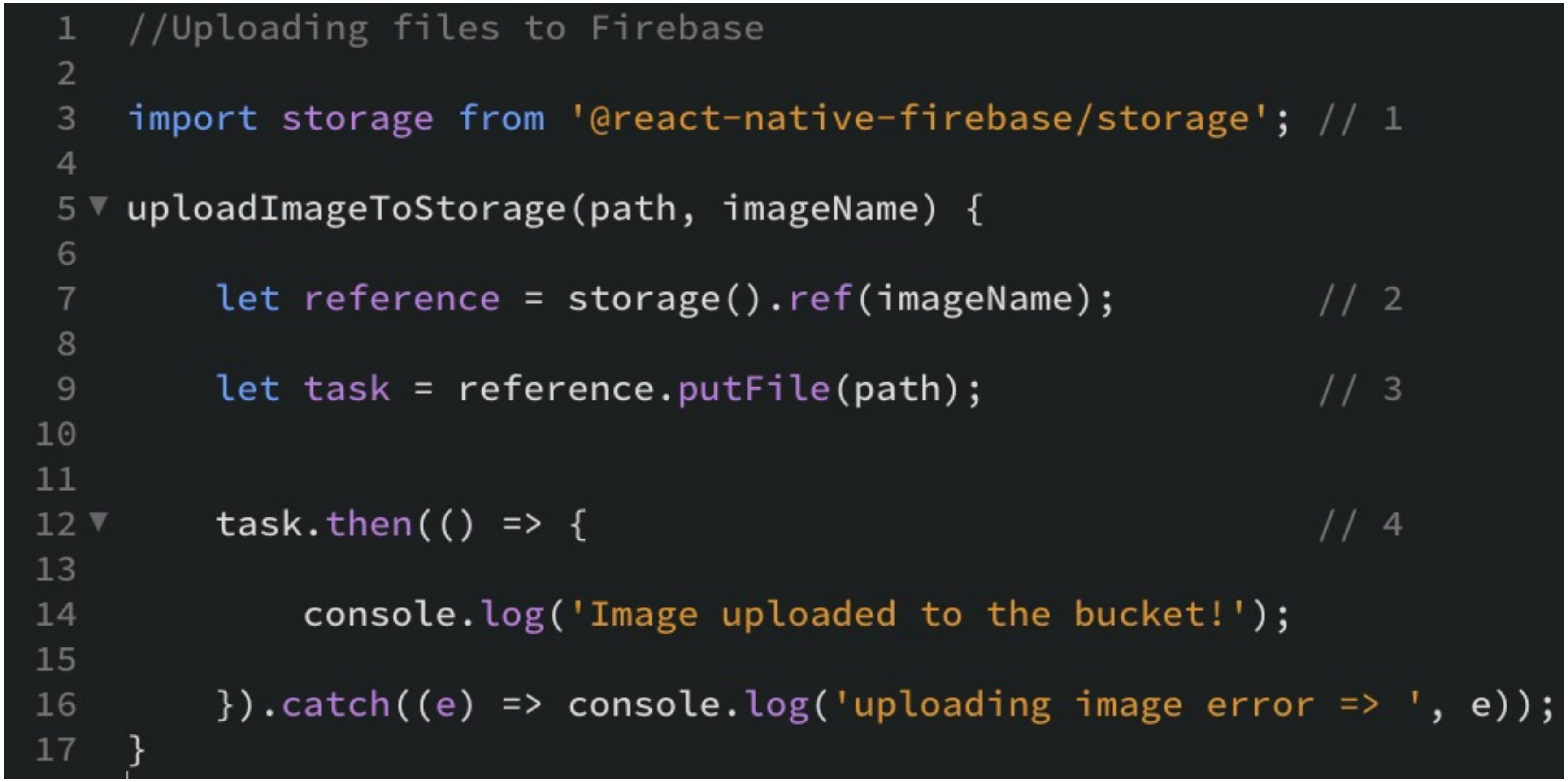


```
//Write a message to the database

FirebaseDatabase database= FirebaseDatabase.getInstance;
DatabaseReference myref=database.getReference(“node”);
myRef.setValue (“Hello, World”);
```

Firebase Queries

- Uploading Files to Firebase Database Storage



```
//Uploading files to Firebase

import storage from '@react-native-firebase/storage'; // 1

uploadImageToStorage(path, imageName) {

    let reference = storage().ref(imageName);          // 2

    let task = reference.putFile(path);                // 3


    task.then(() => {                                  // 4

        console.log('Image uploaded to the bucket!');

    }).catch((e) => console.log('uploading image error => ', e));
}
```

Real-Time Communication Network with Firebase

- In a Qatar hospital, a simulation training machine, extracorporeal membrane oxygenation (EMCO), was created using Firebase.
- The system connected simulation units for: passing patient-related data parameters, ECMO blood-circuit parameters, and related emergencies from the patient to the Firebase System in real-time.
- One requirement satisfied by Firebase: The three systems that capture the above parameters must be continuously synchronized through a real-time database.
  - Firebase satisfies this by providing real-time communication, as discussed above.

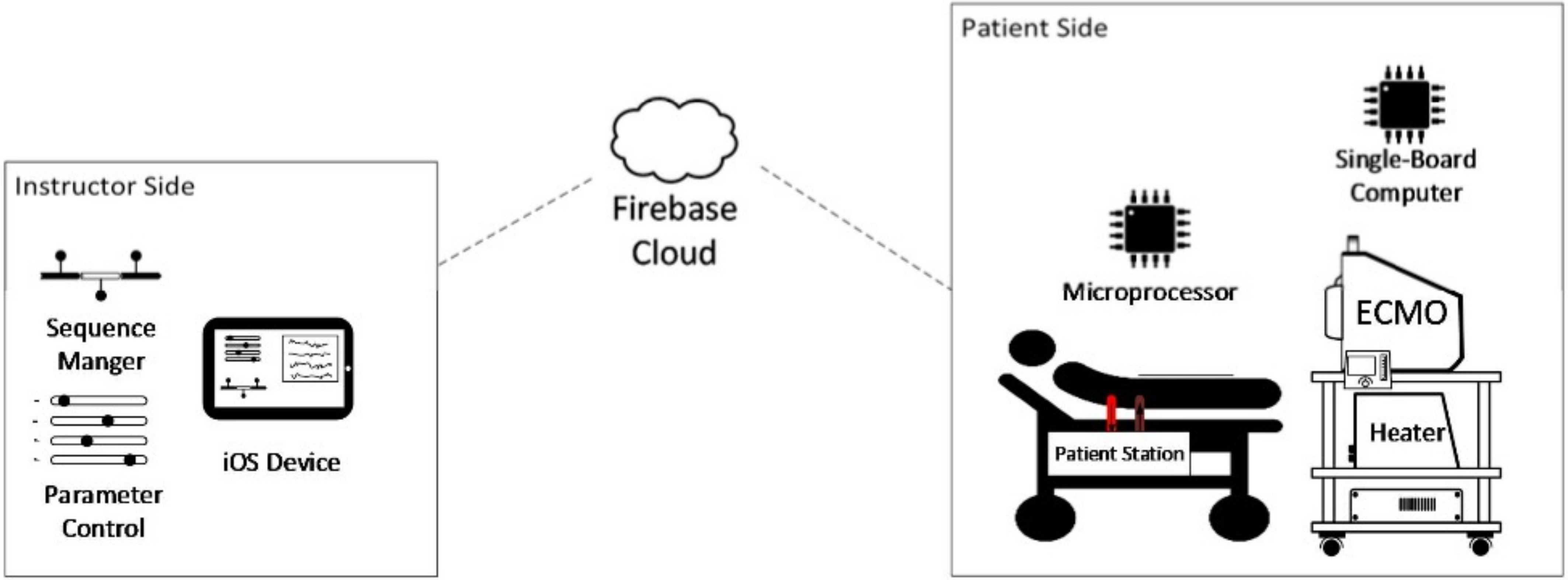
Real-Time Communication Network with Firebase

- Firebase satisfied the requirement of the trainer being able to control the system and parameters through a smartphone app.
    - Firebase made this possible because the data from the smartphone and DB is continuously updated, due to the nature of the Firebase database connection.

Real-Time Communication Network with Firebase

- Communication between System Components:
  - Firebase was justified as the DB of choice for this project due to these factors:
    - Low-Latency
    - Adequately Reliable, Moderately Uninterrupted Connection
    - Instant Changing Control from Smartphone App
  - Firebase DB is stored in a Single JSON file; allowed for highly-responsive data synchronization across system modules. Latency was in the range of milliseconds.
    - Proven in a test: A single parameter was adjusted from one module (Smartphone App) and the change was reflected in a Raspberry Pi running a Python Script.
      - Results: Data retrieval was consistent; 100-500ms at normal operation

Real-Time Communication Network with Firebase



Summary: Real-Time Communication Network with Firebase

- Using the NoSQL properties that were outlined in the previous slides, Firebase was chosen as a ”strong candidate” for the EMCO simulator, due to the low latency, and instant syncing properties inherent to the Database.
- Since Firebase is widely compatible with many devices and interfaces, such as Android, iOS, and Arduino, it made sense as the DB of choice for EMCO, due to a limited time frame for development.
  - Communication between devices is coded using SDKs, instead of coding from scratch.

Firebase Demonstration

Brief Demo:

- What a Firebase Database looks like
- Connecting to, and reading from Firebase using JavaScript

References

- Slide 2: Get from Rich
- Slide 3: Get from Rich
- Slide 4: Khawas, Chunnu & Shah, Pritam. (2018). Application of Firebase in Android App Development-A Study. International Journal of Computer Applications. 179. 49-53. 10.5120/ijca2018917200.
- Slide 5: Read and write data on the web: Read data once . (2021). Retrieved from https://firebase.google.com/docs/database/web/read-and-write#read_data_once
- Slide 6: Khawas, Chunnu & Shah, Pritam. (2018). Application of Firebase in Android App Development-A Study. International Journal of Computer Applications. 179. 49-53. 10.5120/ijca2018917200.
- Slide 7: Singh, P. (2020). Upload images to firebase storage in react native. Retrieved from https://www.pluralsight.com/guides/upload-images-to-firebase-storage-in-react-native
- Slides 8-12: A. Alsalemi *et al*., "Real-Time Communication Network Using Firebase Cloud IoT Platform for ECMO Simulation," *2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, 2017, pp. 178-182, doi: 10.1109/iThings-GreenCom-CPSCom-SmartData.2017.31.
- Project Demonstration: SoftAuthor. (2018). Firebase CRUD Web App with Javascript – Part 1: Get/Read Data. Retrieved from https://softauthor.com/learn-to-build-firebase-crud-app-with-javascript-part01-reading-data/

Thank You!

Questions or Comments? Email strupiano@oakland.edu or rfrost@oakland.edu