CS385 Lecture 18

An overview of Firebase, lab exam 2 details

Using Firebase: "The CS385 Shipping Company"

Firebase

Firebase is OPTIONAL for your CS385 project

Knowledge about Firebase is VERY GOOD for your CS385 learning

Topic Firebase - Optional Video Lecture on using FIrebase in React

This is a video screencast lecture (50 minutes) explaining the entire process around building a React Application with a Google Firebase (Firestore) database backend. This also includes the use of user authentication from Google Firebase authentication.

The source code for the entire example is provided below. You will need to include Firebase as a dependency in your application. At the time of writing this is 10.6.0 (see https://firebase.google.com/support/releases)

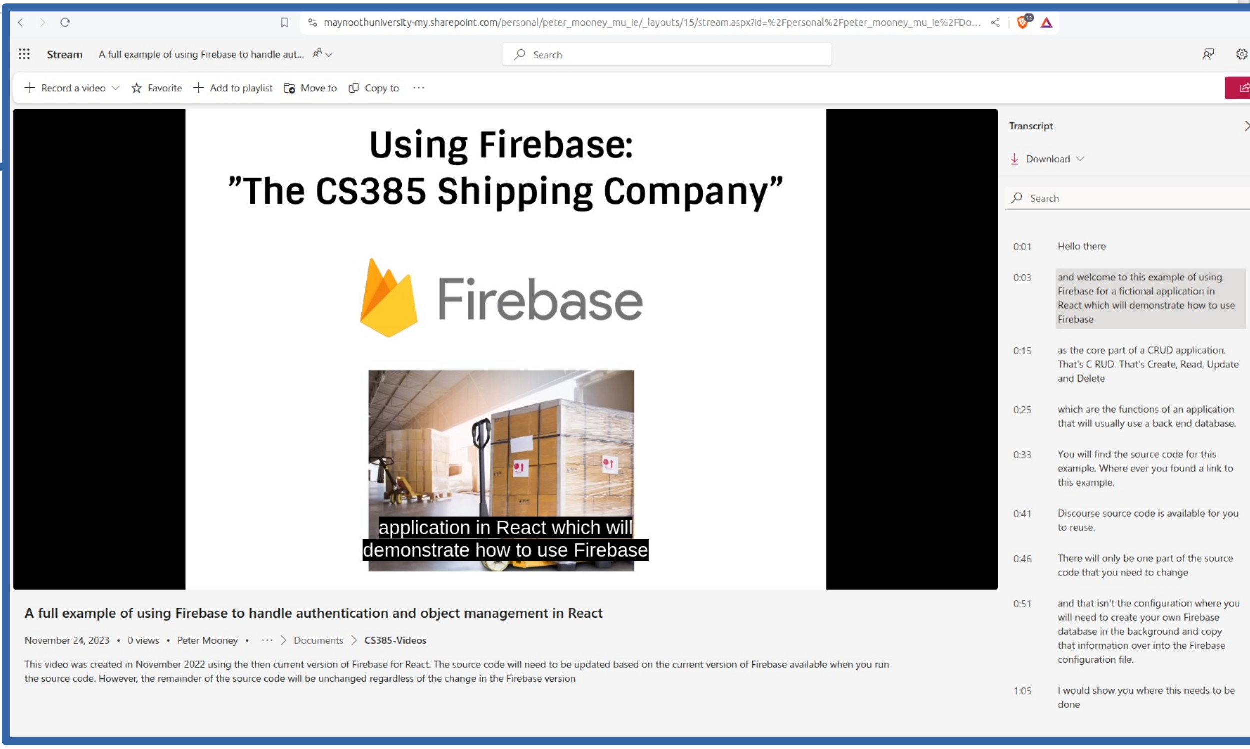
It is very important to note that you will need to create your own Firebase Firestore database on Google before this source code will work properly for you. This is all explained in the video screencast lecture. In the source code the fbconfig.js file is only partially available as the details of your own personal Firestore database must be placed there.

Finally, it is not mandatory that projects use Firebase for any functionality in CS385. This video screencast and source code is provided as an easily accessible but comprehensive example of how Firebase could be used.

Using Firebase to build a database-driven React Application (Screencast lecture 50 minutes)

Firebase-Example-Source-Code-2023-2024

Firebase-Example-Slides-for-Screencast


maynoothuniversity-my.sharepoint.com/personal/peter_mooney_mu_ie/_layouts/15/stream.aspx?id=%2Fpersonal%2Fpeter_mooney_mu_ie%2FDo...
Stream
A full example of using Firebase to handle aut...
Search
Record a video
Favorite
Add to playlist
Move to
Copy to
Using Firebase:
"The CS385 Shipping Company"
Firebase
application in React which will
demonstrate how to use Firebase
Transcript
Download
Search

A full example of using Firebase to handle authentication and object management in React

November 24, 2023 • 0 views • Peter Mooney • ··· > Documents > CS385-Videos

This video was created in November 2022 using the then current version of Firebase for React. The source code will need to be updated based on the current version of Firebase available when you run the source code. However, the remainder of the source code will be unchanged regardless of the change in the Firebase version

0:01	Hello there
0:03	and welcome to this example of using Firebase for a fictional application in React which will demonstrate how to use Firebase
0:15	as the core part of a CRUD application. That's C RUD. That's Create, Read, Update and Delete
0:25	which are the functions of an application that will usually use a back end database.
0:33	You will find the source code for this example. Where ever you found a link to this example,
0:41	Discourse source code is available for you to reuse.
0:46	There will only be one part of the source code that you need to change
0:51	and that isn't the configuration where you will need to create your own Firebase database in the background and copy that information over into the Firebase configuration file.
1:05	I would show you where this needs to be done

The structure of this Firebase example

- **Part A:** Consider the **use of Firebase as a means of providing persistent storage for an application** (without any authentication or user management).
- **Part B:** Consider how to use **Google Firebase as an API for providing authentication functionality** for any React application.
- **Part C: Combination:** Use the code in Part B to offer authentication and user data management for the application in Part A.

Some prerequisites on this Firebase example in CS385 (1)

- **The example we will use today is The CS385 Shipping Company Ltd.**
- The example is designed to be reasonably generic BUT offer you opportunities to re-use the code reasonably easily in your project.
- **The complexity and functionality of the example is calibrated for the general CS385 audience.** Some projects will want MORE Firebase functionality while some projects will want LESS Firebase functionlity (or none at all)

Firebase allows us to synchronize data continuosly across all users of our app

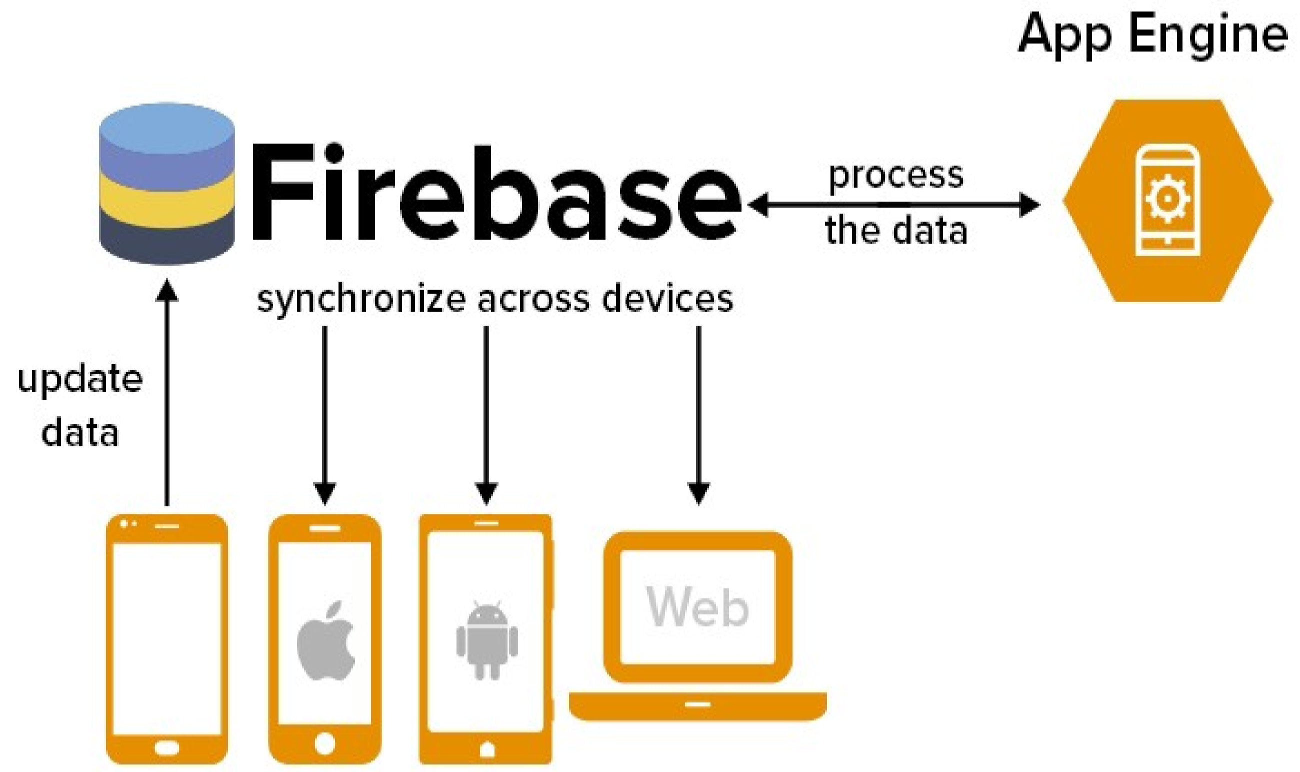
- Firebase has real-time and cloud-based database where you can store data is JSON and synchronized continuously to all connected clients.
- If you want to create an Android, iOS, or Web app which provides real-time updates to users without creating Database or API then you should use firebase.
- Firebase provides the capabilities to manage backend components of applications.
- **The real-time database is the foremost advantage of the Firebase.**

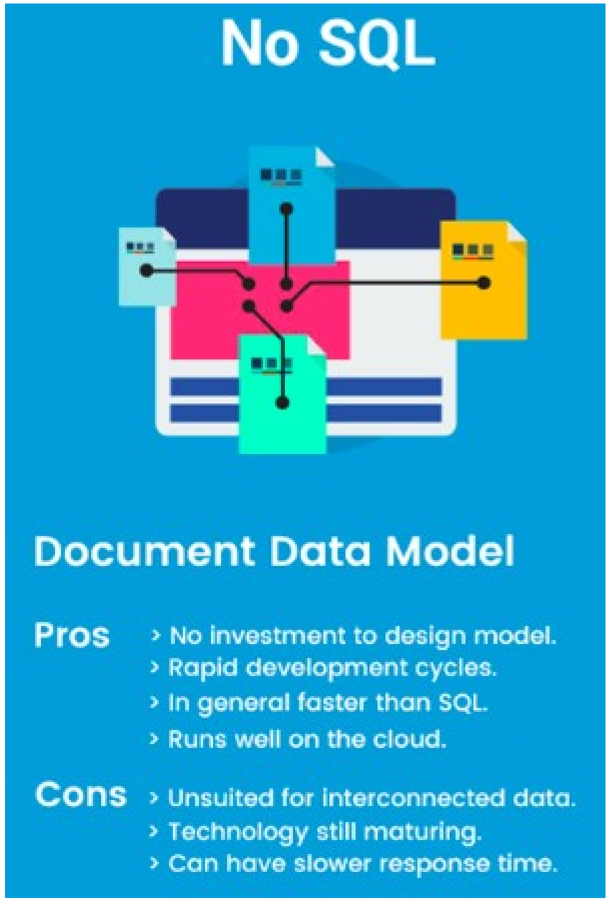
Why use Firebase Authentication?

- **The process of authentication answers the question "Who are you?", while authorization answers the question: "What are you allowed to do?".**
- Firebase provides Oauth which stands for "Open Authorization". While the OAuth flow handles authentication, its main emphasis is on the authorization process.
- OAuth doesn't pass authentication data between consumers and service providers - but acts as an authorization token of sorts.
- We can use this feature of Firebase to help us build our own applications in React (and indeed in many other languages)

Data in your database

- Firebase is a NoSQL database – so it is very different to the Relational Database Management Systems (DBMS) you've seen so far



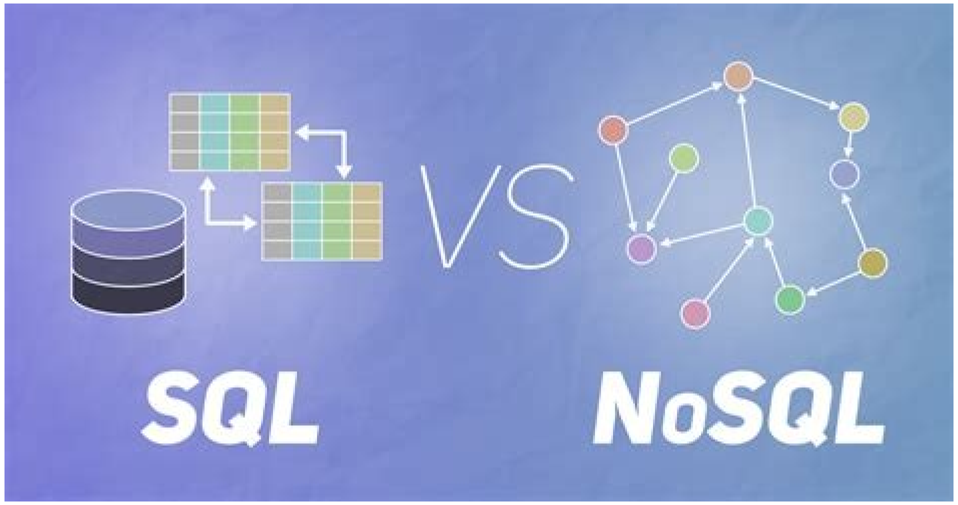


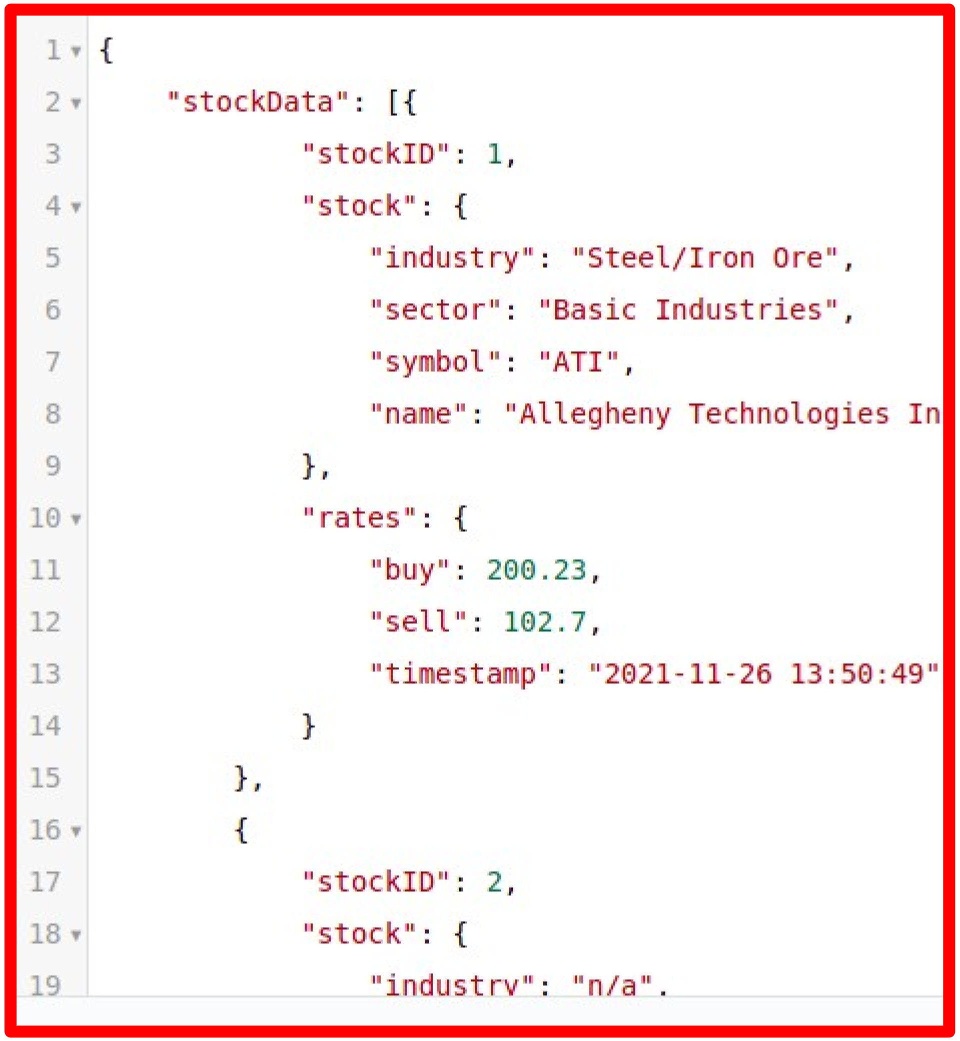
Document Data Model

Pros
- No investment to design model.
- Rapid development cycles.
- In general faster than SQL.
- Runs well on the cloud.

Cons
- Unsuited for interconnected data.
- Technology still maturing.
- Can have slower response time.

We can insert JSON directly into our NoSQL database

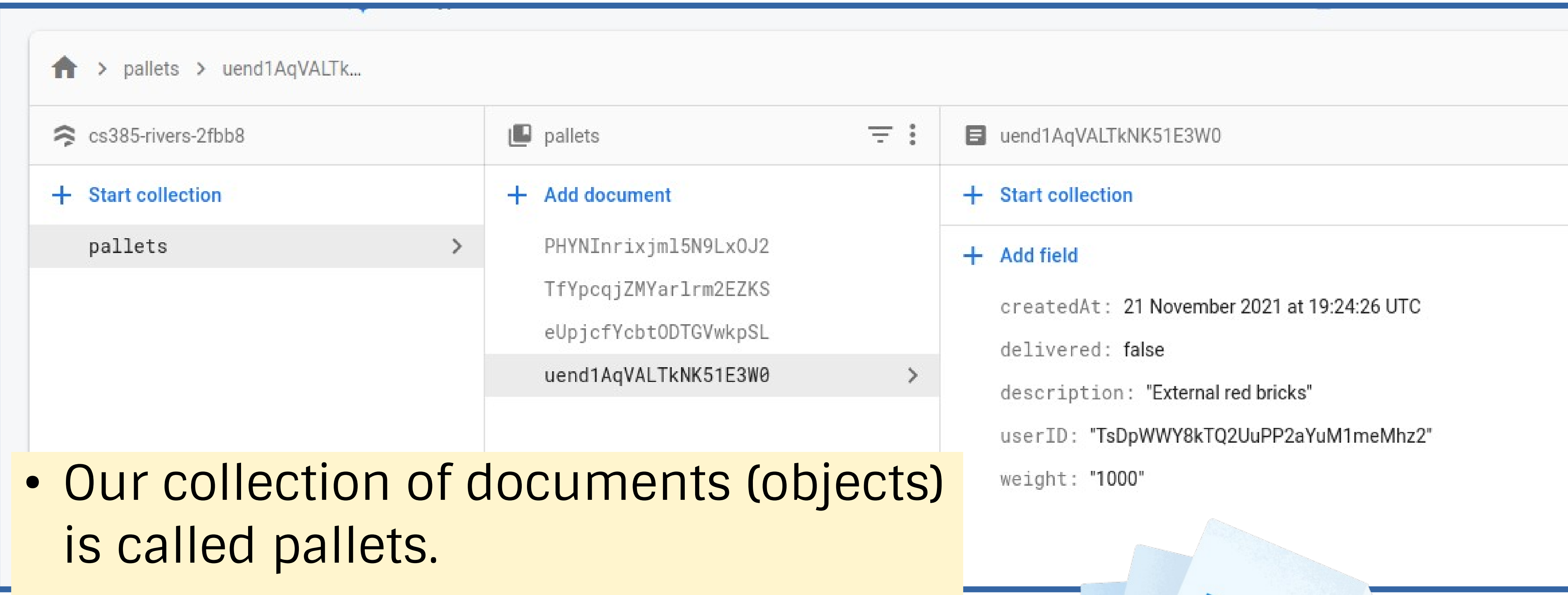



```
{
    "stockData": [{
            "stockID": 1,
            "stock": {
                "industry": "Steel/Iron Ore",
                "sector": "Basic Industries",
                "symbol": "ATI",
                "name": "Allegheny Technologies In
            },
            "rates": {
                "buy": 200.23,
                "sell": 102.7,
                "timestamp": "2021-11-26 13:50:49"
            }
        },
        {
            "stockID": 2,
            "stock": {
                "industry": "n/a",
```

The FireStore Data Model

- Cloud Firestore is a NoSQL, document-oriented database. Unlike a SQL database, there are no tables or rows. Instead, **you store data (JSON objects) in documents, which are organized into collections.**
- **Each document (object) contains a set of key-value pairs.** Cloud Firestore is optimized for storing large collections of small documents. **All documents (objects) must be stored in collections.**
- **Collections and documents are created implicitly in Cloud Firestore.** Simply assign data to a document within a collection. If either the collection or document does not exist, Cloud Firestore creates it.

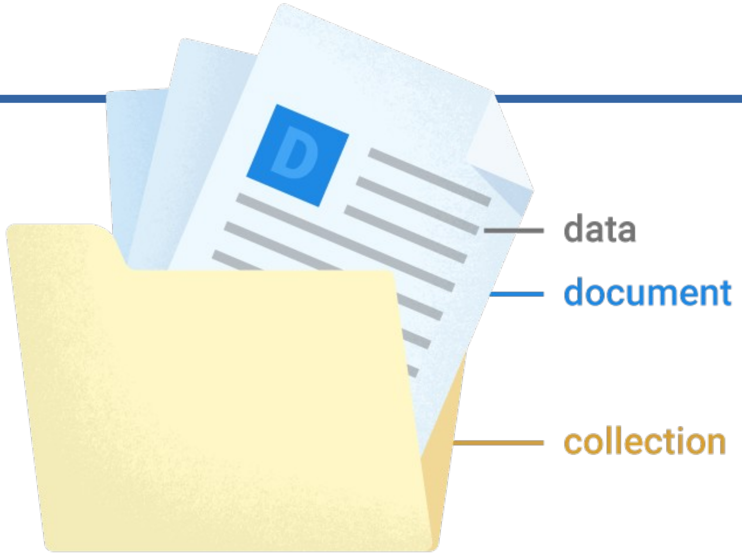
FireStore Data Model - Example



pallets > uend1AqVALTk...

cs385-rivers-2fbb8	pallets	uend1AqVALTkNK51E3W0
Start collection	Add document	Start collection
pallets	PHYNInrixjml5N9LxOJ2	Add field
	TfYpcqjZMYarlrm2EZKS	createdAt: 21 November 2021 at 19:24:26 UTC
	eUpjcfYcbtODTGVwkpSL	delivered: false
	uend1AqVALTkNK51E3W0	description: "External red bricks"
		userID: "TsDpWWY8kTQ2UuPP2aYuM1meMhz2"
		weight: "1000"

- Our collection of documents (objects) is called pallets.
- We have FOUR documents (objects).
- The document (object) shown has FIVE properties (or key-value pairs).
- You, the programmer, supply the document and the properties.



FireStore – React + CRUD

- **All CRUD (Create, Read, Update, Delete)** operations we wish to perform on our FireStore database must be performed using React.
- We have access to the FireStore backend User Interface here in this example. However, this is because we are essentially the database administrators for this example.
- **The only way other users should be able to access your FireStore database is by using your app.**

The structure of this Firebase example

- **Part A:** Consider the **use of Firebase as a means of providing persistent storage for an application** (without any authentication or user management).
- **Part B:** Consider how to use **Google Firebase as an API for providing authentication functionality** for any React application.
- **Part C: Combination:** Use the code in Part B to offer authentication and user data management for the application in Part A.

The CS385 Shipping Company Functionality

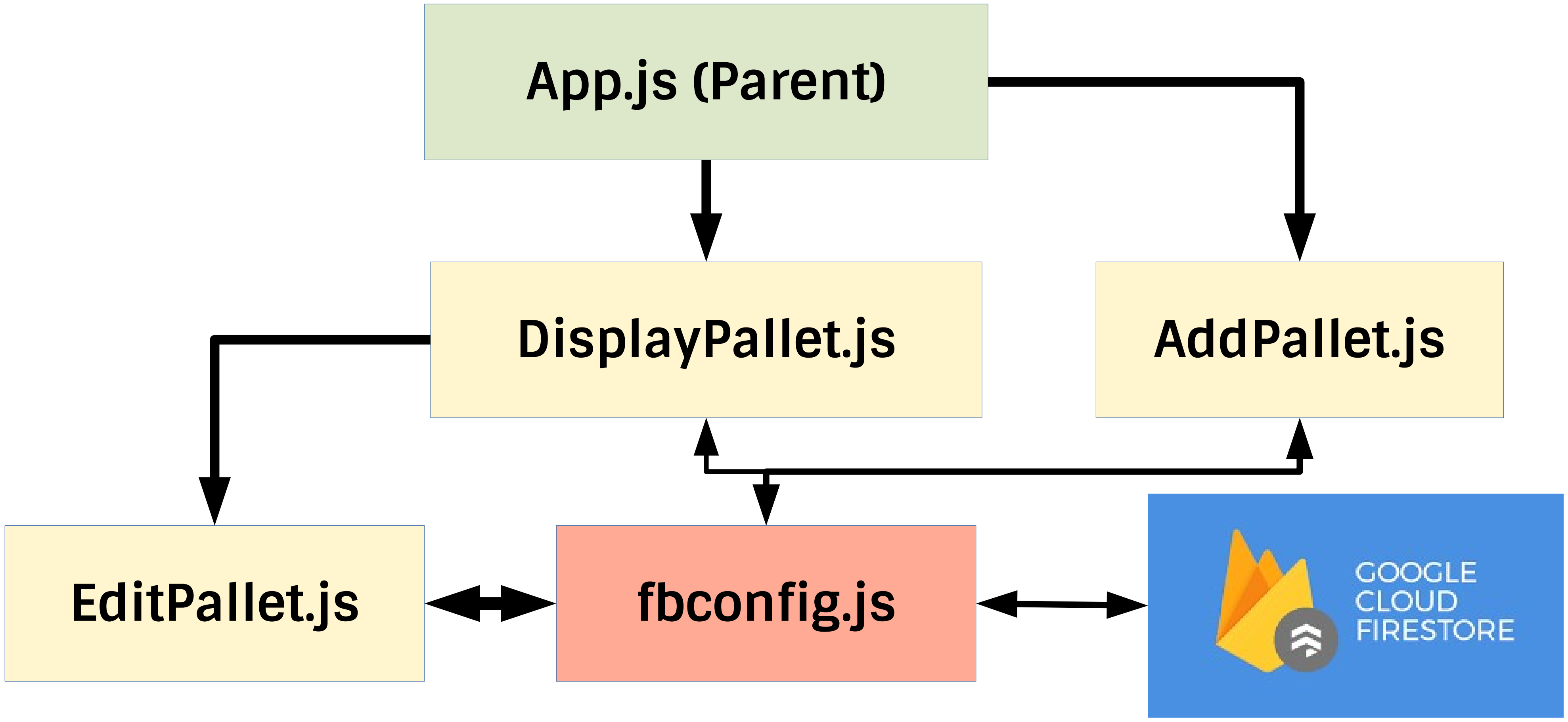
- Allow a user to
  - **Add a pallet for shipping** (add to our Firestore database) – Allow the user to specify the goods on the pallet and the overall weight (KG)
  - **Display all pallets** in our warehouse
  - **Delete a pallet** – remove this pallet from our Firestore database
  - **Edit a pallet** – allow the user to change the details for an existing pallet in our Firestore database.

The CS385 Shipping Company Data Model

- At this stage the shipping pallet will have four properties
  - **Description** – String – a short description of the contents of the pallet.
  - **Weight** – the approximate total weight of the pallet in kilograms.
  - **CreatedAt** – this will be timestamp when the pallet document is stored in Firestore.
  - **Delivered** – this is a boolean value to indicate if the pallet has been delivered.

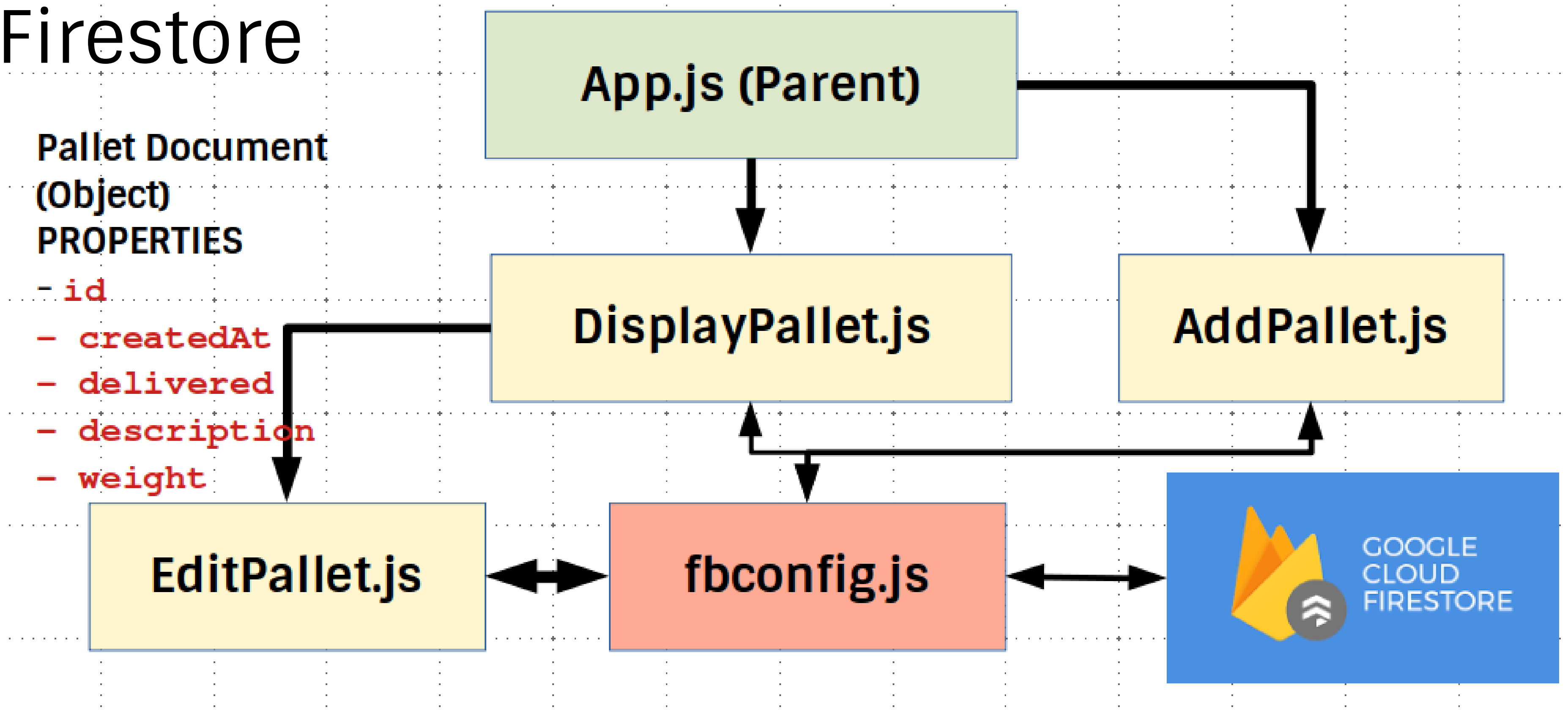
Overall Component Schematic

- The **App.js** component is the parent. The other components are all considered as child components. The **fbconfig.js** file provides the configuration information to allow the components to access the Firestore database for this application.



It is VERY IMPORTANT for you to design your document (object) to satisfy the needs of your application

- **Our entire application was developed around the Pallet document (object)** – see below. The application is essentially driven by the movement of this object in and out of Firestore



The structure of this Firebase example

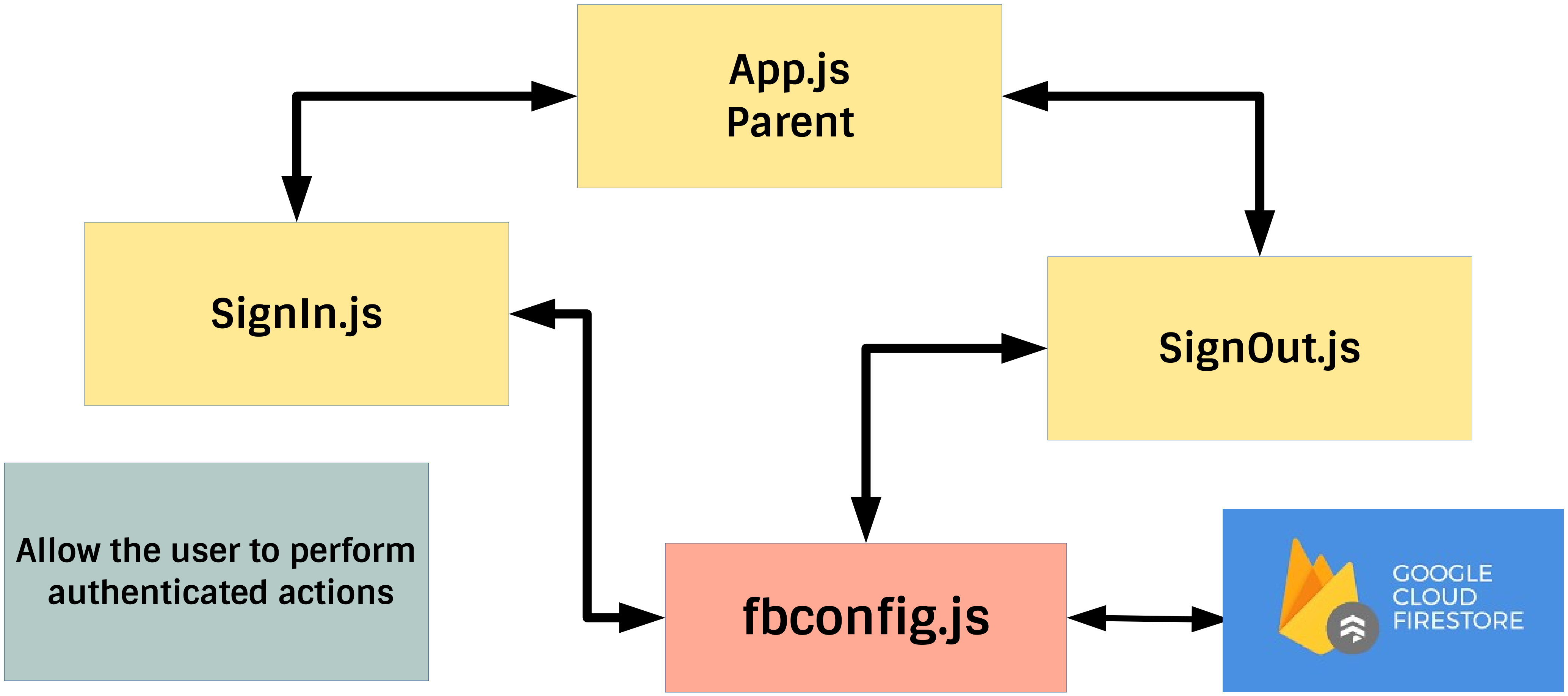
- **Part A:** Consider the **use of Firebase as a means of providing persistent storage for an application** (without any authentication or user management).
- **Part B:** Consider how to use **Google Firebase as an API for providing authentication functionality** for any React application.
- **Part C: Combination:** Use the code in Part B to offer authentication and user data management for the application in Part A.

Think about an authenticated user in an application

- **When a user authenticates in an application your application software code has access to a special object containing authentication details about the user. This is the authenticated user.**
- By using this object your software code can determine if the user is authenticated (logged in) or not. When the user has logged out (not authenticated) then this object is null.
- The complication (solved by Firebase) is to provide a way to establish if a user is authenticated or not.

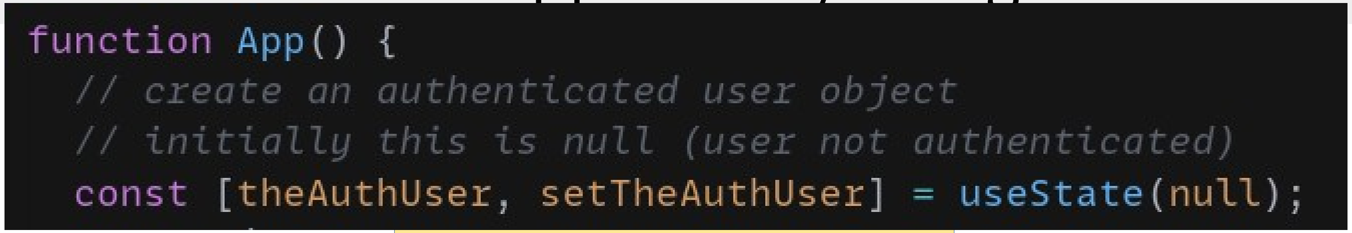
Overall Component Schematic

- The App component (Parent) can offer the user the opportunity to authenticate (via SignIn.js). When the user is authenticated they can then be presented with the opportunity to sign out (via SignOut.js)

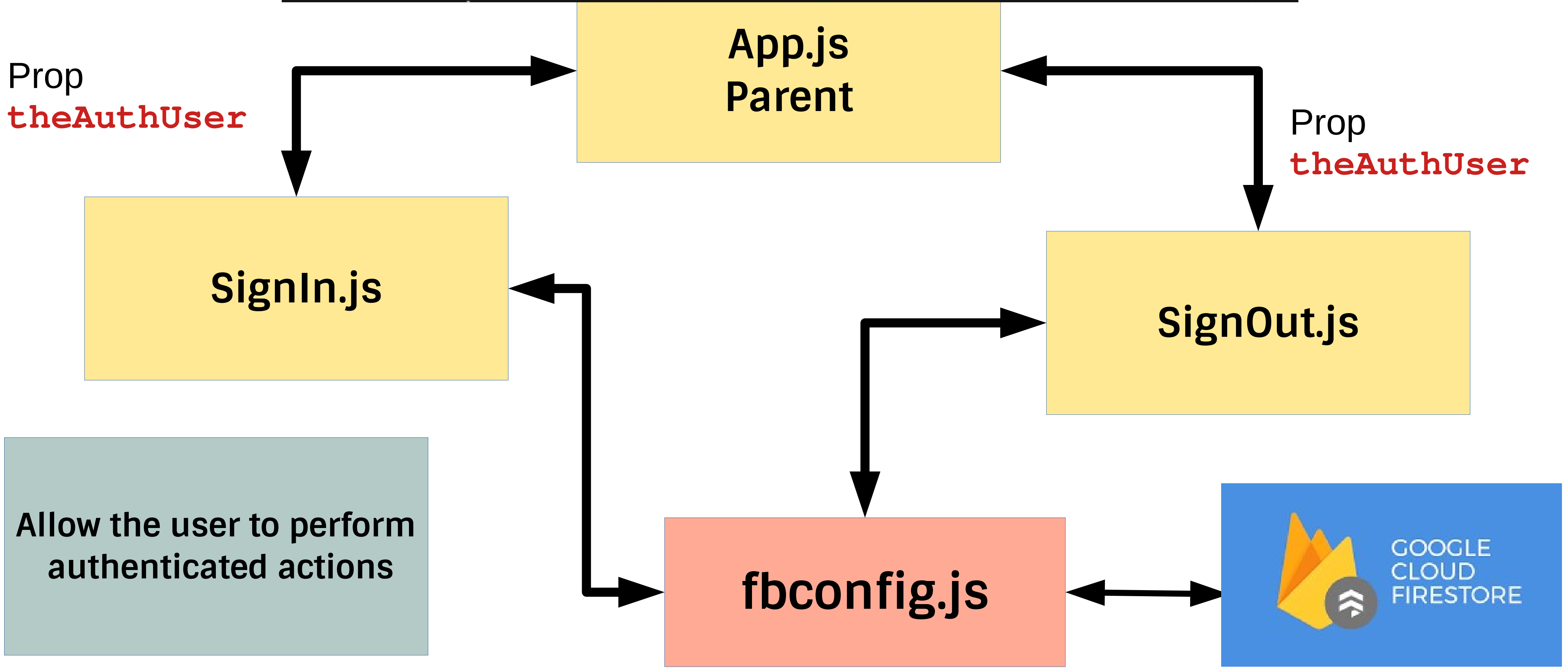


Overall Component Schematic

- The App component (Parent) can offer the user the opportunity to authenticate (via SignIn.js). When the user is authenticated they can then be presented with the opportunity to sign out (via SignOut.js)



```
function App() {
  // create an authenticated user object
  // initially this is null (user not authenticated)
  const [theAuthUser, setTheAuthUser] = useState(null);
```

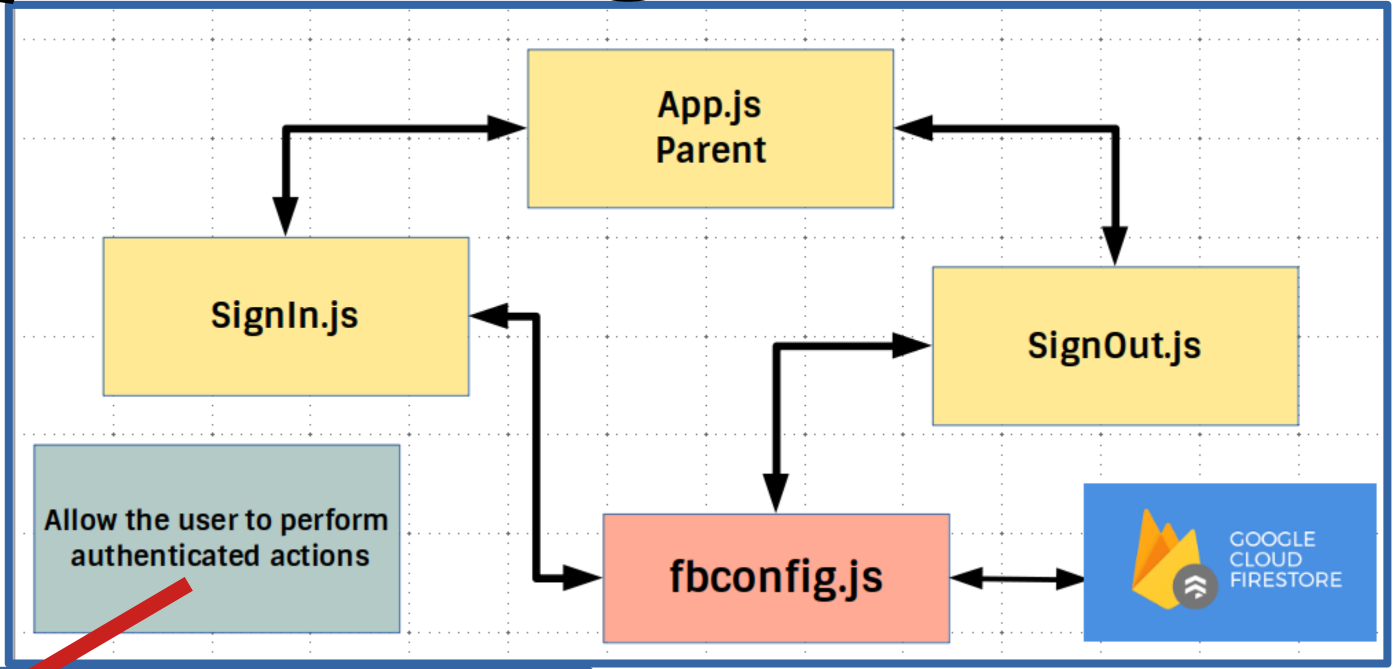


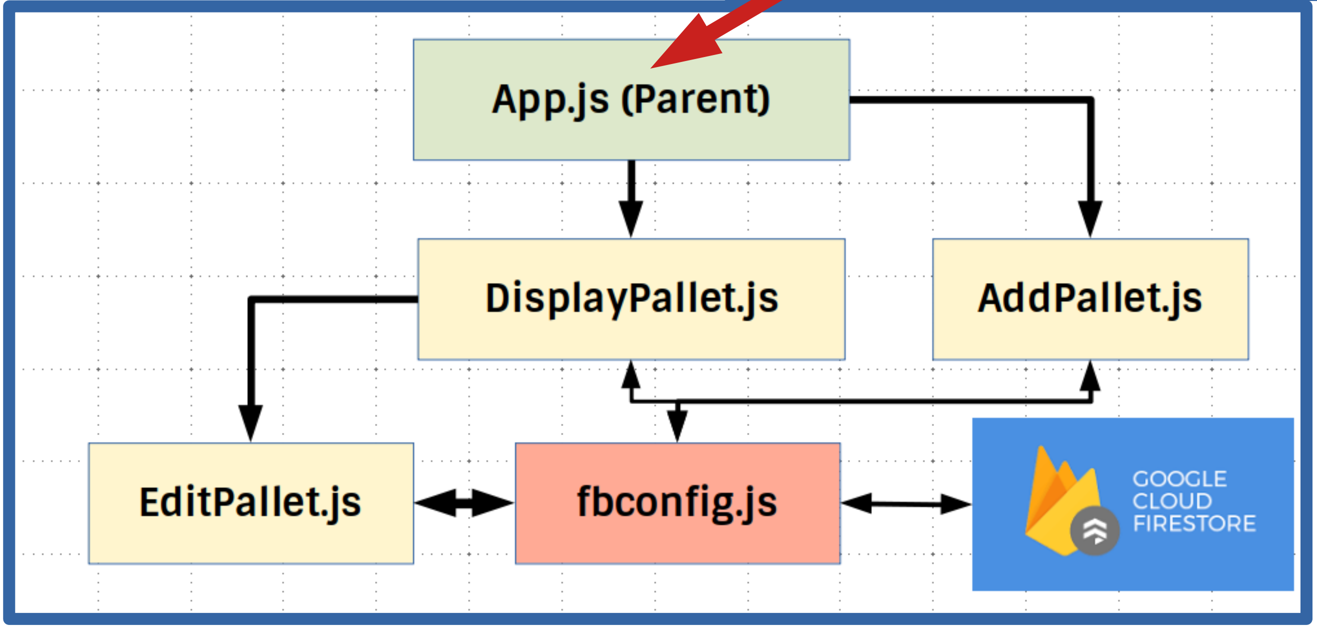
Part C Combination – THE GOAL

- The goal of Part C is to use the authentication from Part B to control access to Part A.
- Informally, when a user logons to our CS385 Shipping Company Ltd application they can CRUD Pallet documents (objects) AKA Shipments. Users will be only able to see their own documents (objects).
- Users will be able to AddPallet, EditPallet, DeletePallet, and so on – but only for the Pallet documents that they own.

The good news …. there are surprisngly few changes to make

Part B





Part A

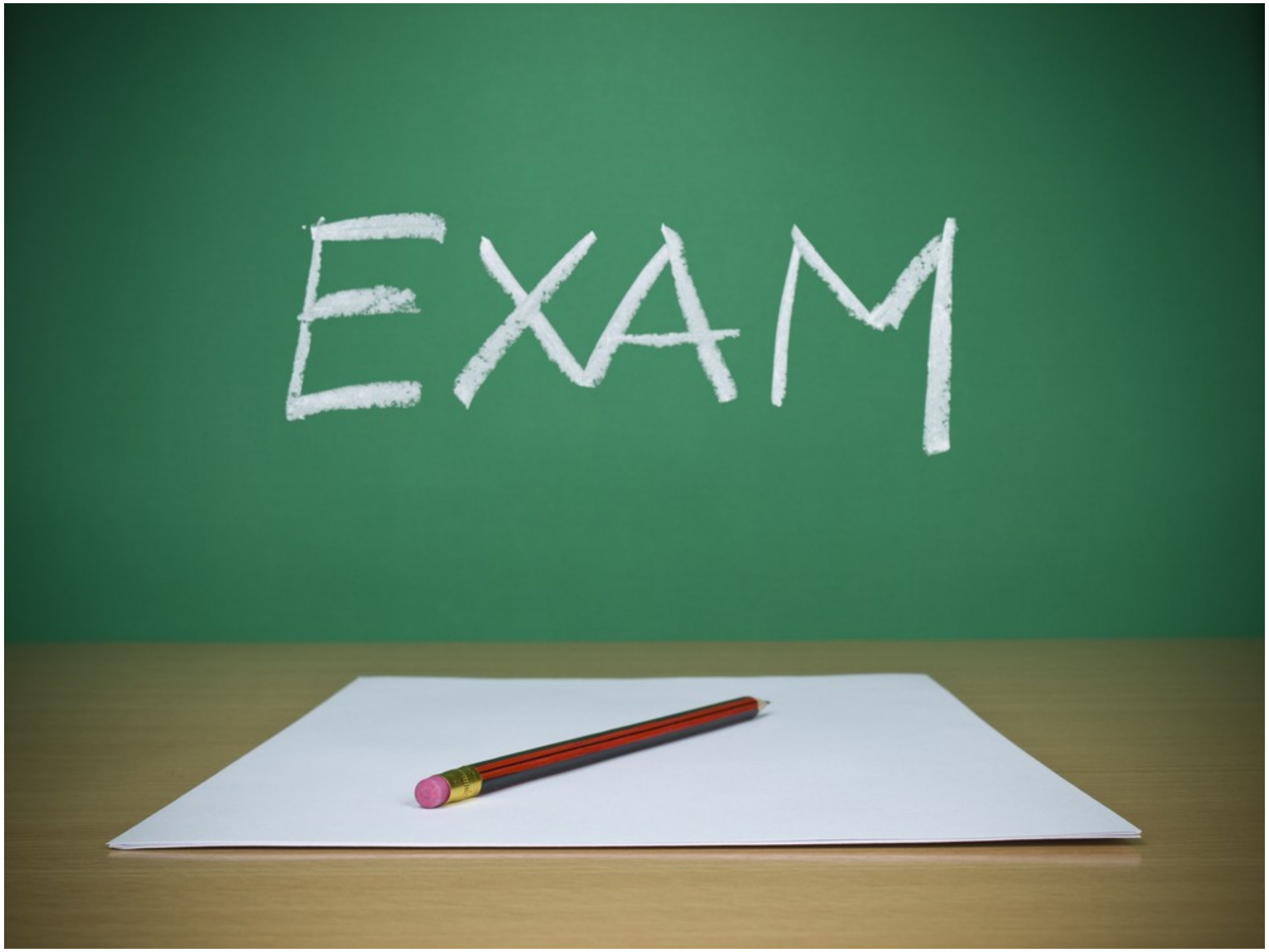
Why use Firebase?

- It’s free, mobile-app focused
- You don’t need to write your own SERVER or backend – Firebase provides the whole solution
- Works seamlessly with JSON objects.
- Provides you with a full C-R-U-D application functionality set

Firebase is OPTIONAL for your CS385 project

Knowledge about Firebase is VERY GOOD for your CS385 learning

CS385 Mobile Application Development
(Lab Exam #2 Guidance)



Lab Exam #2

- **When:** Friday 1st Dec 2023 11:00 – 12:00
- **Where:** Your CS385 lab location
- **How:** OPEN BOOK, using Moodle MCQ quiz
- **Why:** Lab Exam #2 is worth 15% of your C/A

Lab Exam 2 – The rules

- **60 minutes** – Quiz is between 11:00 and 12:00
- **12 MCQ questions – one correct answer per question.**
- You can only attempt the quiz once.
- **You can use ANY of your CS385 materials – Moodle, notes, lecture notes, sample, code etc.**
- **ABSOLUTELY No use of Google, ChatGPT, BingChat, StackOverFlow,Codesandbox, messaging apps, etc.**
- I will supply you with some writing paper (for rough work), if you need it.

Lab Exam 2 – SEATING ARRANGEMENTS

- You will be RANDOMLY assigned to one of the ROWS in the lab. This list will be posted on Moodle and printed in the lab
- There is LOTS of room in the lab.
- Unless it is impossible, PLEASE try to keep ONE computer space between you and the next person.

A demo lab exam is available

- **Demo lab 2 exam quiz will be provided on Moodle.**
- There are NO C/A marks associated with this quiz.
- **This quiz has a duration of 24 hours** – the clock starts counting down when you open the quiz. **You can take the quiz at any time you wish**
- This quiz will help you become more comfortable with the Moodle MCQ environment. You will also see the standard of questions you are likely to be asked.

Lab Exam 2 – HINTS and QUESTIONS

- This will only be shown in the lectures – this slide will not be available on Moodle.

Lab Exam 2 – Make sure that you complete the demo lab exam

CS385 Project Update

- There is a project lab 10:00 – 11:00 on Friday 1st December 2023
- Moodle submission will open at the end of the week

See you all on Friday