

# Capture the Flag

## Firestore Usage Report

COMP2003 Group 6

### 1. Overview

This document summarises how Firestore is integrated into the Capture the Flag web prototype. Firestore provides the backend infrastructure for authentication and real-time data storage, removing the need to manage a custom server for these responsibilities.

Firestore project ID: ctf-game-06

Firestore SDK version: 10.14.1 (modular)

### 2. Firestore Services Used

Service	Purpose	Status
Firestore Authentication	Google OAuth Sign-In	Active
Cloud Firestore	Game session & score storage	Active
Firestore App (core)	Initialises all Firestore services	Active
Firestore Analytics	Usage tracking (measurementId)	Configured

### 3. Firestore Authentication

#### 3.1 Method: Google OAuth (signInWithPopup)

Originally the project used email and password authentication via `createUserWithEmailAndPassword` and `signInWithEmailAndPassword`. This was replaced with Google Sign-In for the following reasons:

- No password management required for students
- Player display name is automatically retrieved from the Google account
- Firestore SDK v10 deprecated the old `auth/user-not-found` and `auth/wrong-password` error codes, causing login failures
- Google OAuth is more reliable and user-friendly in a classroom environment

#### 3.2 Implementation

Both `signin.html` and `login.html` now use the same Google Sign-In flow:

```
import { GoogleAuthProvider, signInWithPopup } from "firebase/auth";

const provider = new GoogleAuthProvider();
const result = await signInWithPopup(auth, provider);

// Player name auto-populated from Google account
const playerName = result.user.displayName;
localStorage.setItem("playerName", playerName);
```

### 3.3 Authorised Domains

Firebase Authentication restricts OAuth to authorised domains only. During local development the site must be accessed via localhost rather than 127.0.0.1, as Firebase does not recognise the IP address as an authorised domain.

- localhost – authorised by default in Firebase Console
- 127.0.0.1 – NOT supported for OAuth, causes auth/unauthorized-domain error

### 3.4 Files Involved

File	Firebase Auth Usage
signin.html	Google Sign-In (signInWithPopup) – new user or returning
login.html	Google Sign-In (signInWithPopup) – identical flow to signin
firebase.js	Central config – initialises Firebase app and auth instance

## 4. Cloud Firestore

### 4.1 Purpose

Cloud Firestore is used to persist game session data and player scores. This allows multiple players to contribute scores to a shared session in real time without overwriting each other's data.

### 4.2 Database Structure

Data is organised into the following Firestore collections and documents:

```
sessions/                                ← collection
  {sessionPin}/                          ← document (keyed by 6-digit PIN)
    teamAScore: number                  ← Team A total score
    teamBScore: number                  ← Team B total score
    players/                            ← sub-collection
      {playerName}/                    ← document per player
        score: number                  ← individual score
        team: 'A' | 'B'                ← player's team
```

### 4.3 Score Saving Logic

When a player completes the Unity hacking mini-game, their score is sent to Firestore using `increment()` to safely add to the team total without overwriting other players' contributions:

```
import { doc, setDoc, increment } from 'firebase/firestore';

const sessionRef = doc(db, 'sessions', sessionPin);

// Safely add score to the correct team
await setDoc(sessionRef,
  { teamAScore: increment(finalScore) },
  { merge: true }
);

// Save individual player stats
const playerRef = doc(db, 'sessions', sessionPin, 'players', playerName);
await setDoc(playerRef,
  { score: increment(finalScore), team: team },
  { merge: true }
);
```

## 4.4 Files Involved

File	Firestore Usage
game.js	Writes team scores and player stats using <code>setDoc</code> + <code>increment</code>
firebase.js	Exports <code>db</code> instance and Firestore helper functions

## 5. Firebase Configuration

The Firebase project is initialised using the following configuration, present in `firebase.js` and also inlined into `signin.html` and `login.html` for pages that load Firebase directly:

```
const firebaseConfig = {
  apiKey: "AIzaSyAYQeZzdYUqKYiiwcpFp8BNZQsJNHtoMXo",
  authDomain: "ctf-game-06.firebaseio.com",
  projectId: "ctf-game-06",
  storageBucket: "ctf-game-06.firebaseio.com",
  messagingSenderId: "509318018925",
  appId: "1:509318018925:web:a91ddc9f7fc130dfe54c64",
  measurementId: "G-L4WP0E0V6S"
};
```

Note: The Firebase config API key is safe to include in client-side code as Firebase security is enforced through Firestore Security Rules and Authentication, not by keeping the key secret.

## 6. Known Issues & Fixes Applied

Issue	Cause	Fix Applied
Login failed error on every attempt	Firebase SDK v10 changed error code from	Updated catch block to handle new error code

Issue	Cause	Fix Applied
	auth/user-not-found to auth/invalid-credential	
auth/unauthorized-domain error	Firebase blocks OAuth on 127.0.0.1	Access site via localhost:5500 instead
Scores overwriting other players	setDoc without merge replaces entire document	Used increment() with merge:true

## 7. Potential Next Steps

- Add Firestore Security Rules to restrict score writes to authenticated users only
- Use onSnapshot() for real-time live leaderboard updates during a session
- Store session metadata (start time, topic, teacher ID) in the session document
- Add Firebase Analytics events to track game completion rates