// Import stylesheets

import './style.css';

// Firebase App (the core Firebase SDK) is always required and must be listed first

import \* as firebase from "firebase/app";

// Add the Firebase products that you want to use

import "firebase/auth";

import "firebase/firestore";

import \* as firebaseui from 'firebaseui';

// Document elements

const startRsvpButton = document.getElementById('startRsvp');

const guestbookContainer = document.getElementById('guestbook-container');

const form = document.getElementById('leave-message');

const input = document.getElementById('message');

const guestbook = document.getElementById('guestbook');

const numberAttending = document.getElementById('number-attending');

const rsvpYes = document.getElementById('rsvp-yes');

const rsvpNo = document.getElementById('rsvp-no');

var rsvpListener = null;

var guestbookListener = null;

async function main() {

// Add Firebase project configuration object here

var firebaseConfig = {

apiKey: "AIzaSyAB4fT4eHD69CJ5-yVxrBU8YZ4c83fiQUQ",

authDomain: "fir-web-5a73f.firebaseapp.com",

databaseURL: "https://fir-web-5a73f.firebaseio.com",

projectId: "fir-web-5a73f",

storageBucket: "fir-web-5a73f.appspot.com",

messagingSenderId: "670813848583",

appId: "1:670813848583:web:5bc879f7d872c8810d56a2",

measurementId: "G-SP2N995QKV"

};

firebase.initializeApp(firebaseConfig);

//firebase.analytics();

// FirebaseUI config

const uiConfig = {

credentialHelper: firebaseui.auth.CredentialHelper.NONE,

signInOptions: [

// Email / Password Provider.

firebase.auth.EmailAuthProvider.PROVIDER\_ID

],

callbacks: {

signInSuccessWithAuthResult: function(authResult, redirectUrl) {

// Handle sign-in.

// Return false to avoid redirect.

return false;

}

}

};

//initialize the firebase ui widget using firebase

const ui = new firebaseui.auth.AuthUI(firebase.auth());

//listen to rsvp button clicks

startRsvpButton.addEventListener("click",  
 () => {

if(firebase.auth().currentUser){

firebase.auth().signOut();

}else{

ui.start("#firebaseui-auth-container", uiConfig);

}

});

//해당 firebase.auth()의 상황 바뀌면

firebase.auth().onAuthStateChanged((user) => {

if (user){

startRsvpButton.textContent = "LOGOUT";

guestbookContainer.style.display = "block";

subscribeGuestbook();

}  
 else{

startRsvpButton.textContent = "RSVP";

guestbookContainer.style.display = "none";

unsubscribeGuestbook();

} });

//guestbook add시 guestbook database add 처리

form.addEventListener("submit", (e) => {

e.preventDefault();

firebase.firestore().collection("guestbook").add({

text: input.value,

timestamp: Date.now(),

name: firebase.auth().currentUser.displayName,

userId: firebase.auth().currentUser.uid

})

input.value = "";

return false;

});

//timestamp 순으로 작성하기

function subscribeGuestbook(){

firebase.firestore().collection("guestbook") .orderBy("timestamp","desc") .onSnapshot((snaps) => {

guestbook.innerHTML = "";

snaps.forEach((doc) => {

const entry = document.createElement("p");

entry.textContent = doc.data().name + ": " + doc.data().text;

guestbook.appendChild(entry);

});

});

}

function unsubscribeGuestbook(){

if (guestbookListener != null)  
 {

guestbookListener();

guestbookListener = null;

} };

}

main();