

# FIT5032 Assessed Lab 7 Submission

## Firestore Authentication Implementation

Student Name: [Du Daoan]  
Student ID: [35523166]

### 1 EFOLIO TASK 7.1 - Firestore Authentication Basic Implementation

#### 1.1 Screenshot Set 1: Registration Page

##### 1.1.1 Registration Page in Browser

Home (Week 7) About Firestore Auth FireRegister Login Contact us

### Firestore Registration

Email Address

Password

Password must be at least 6 characters long.

Confirm Password

Create Account

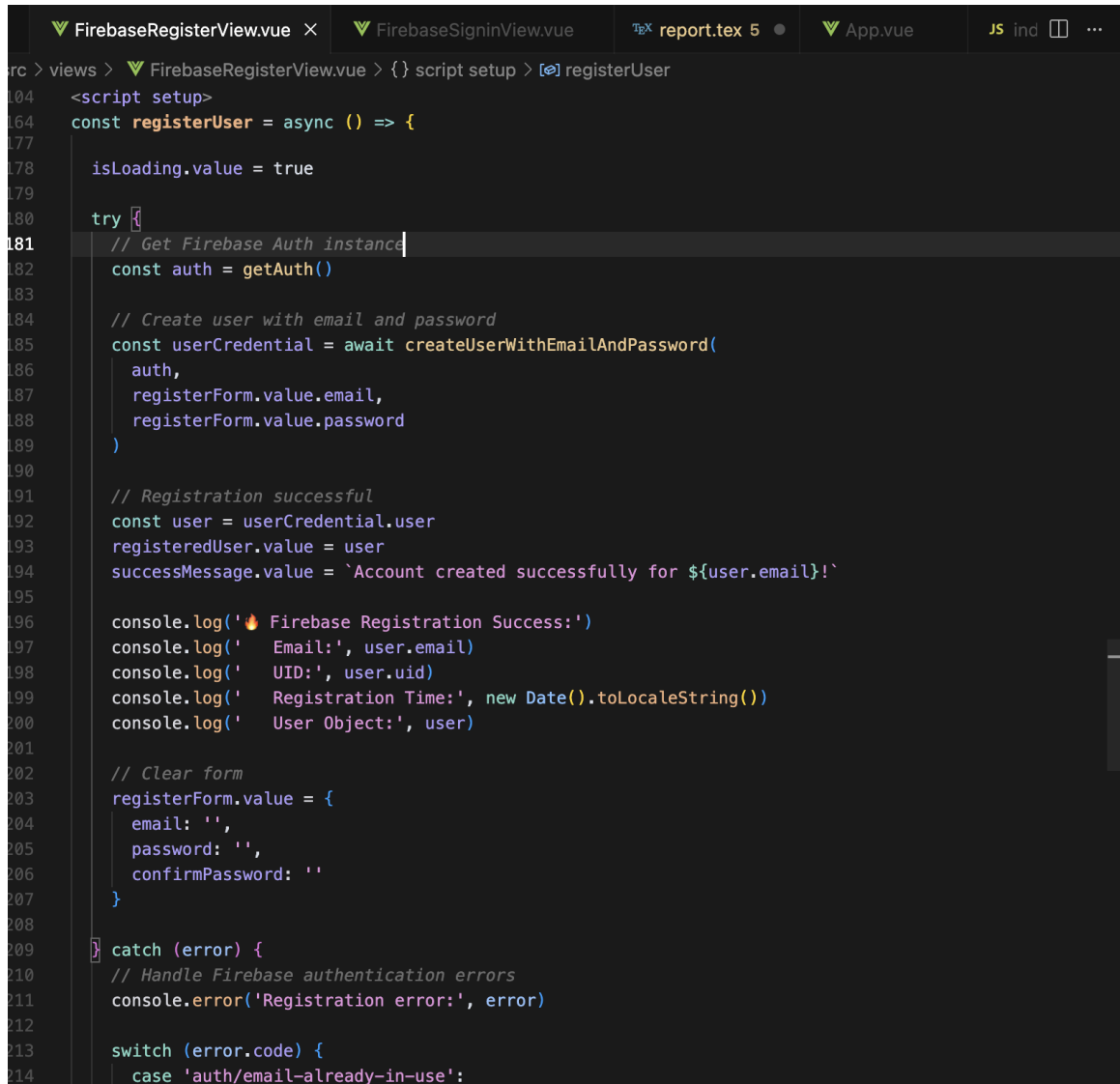
Already have an account?

Sign In Instead

Figure 1: FireRegisterView.vue registration page displayed in browser

**Required:** Screenshot of registration page in browser showing the form with email, password, and confirm password fields.

### 1.1.2 Registration Page in Visual Studio Code



```

104 <script setup>
164 const registerUser = async () => {
177
178   isLoading.value = true
179
180   try {
181     // Get Firebase Auth instance
182     const auth = getAuth()
183
184     // Create user with email and password
185     const userCredential = await createUserWithEmailAndPassword(
186       auth,
187       registerForm.value.email,
188       registerForm.value.password
189     )
190
191     // Registration successful
192     const user = userCredential.user
193     registeredUser.value = user
194     successMessage.value = `Account created successfully for ${user.email}!`
195
196     console.log('🔥 Firebase Registration Success:')
197     console.log('  Email:', user.email)
198     console.log('  UID:', user.uid)
199     console.log('  Registration Time:', new Date().toLocaleString())
200     console.log('  User Object:', user)
201
202     // Clear form
203     registerForm.value = {
204       email: '',
205       password: '',
206       confirmPassword: ''
207     }
208
209   } catch (error) {
210     // Handle Firebase authentication errors
211     console.error('Registration error:', error)
212
213     switch (error.code) {
214       case 'auth/email-already-in-use':

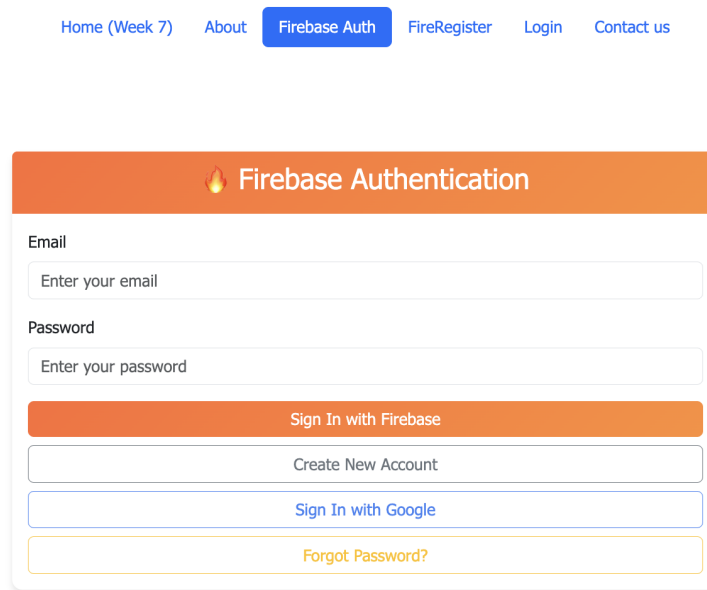
```

Figure 2: FirebaseRegisterView.vue source code in Visual Studio Code

**Required:** Screenshot of FirebaseRegisterView.vue file open in VS Code showing the Vue component code.

## 1.2 Screenshot Set 2: Login Page with Console

### 1.2.1 Login Page in Browser



Home (Week 7) About **Firestore Auth** FireRegister Login Contact us

**Firestore Authentication**

Email

Enter your email

Password

Enter your password

Sign In with Firestore

Create New Account

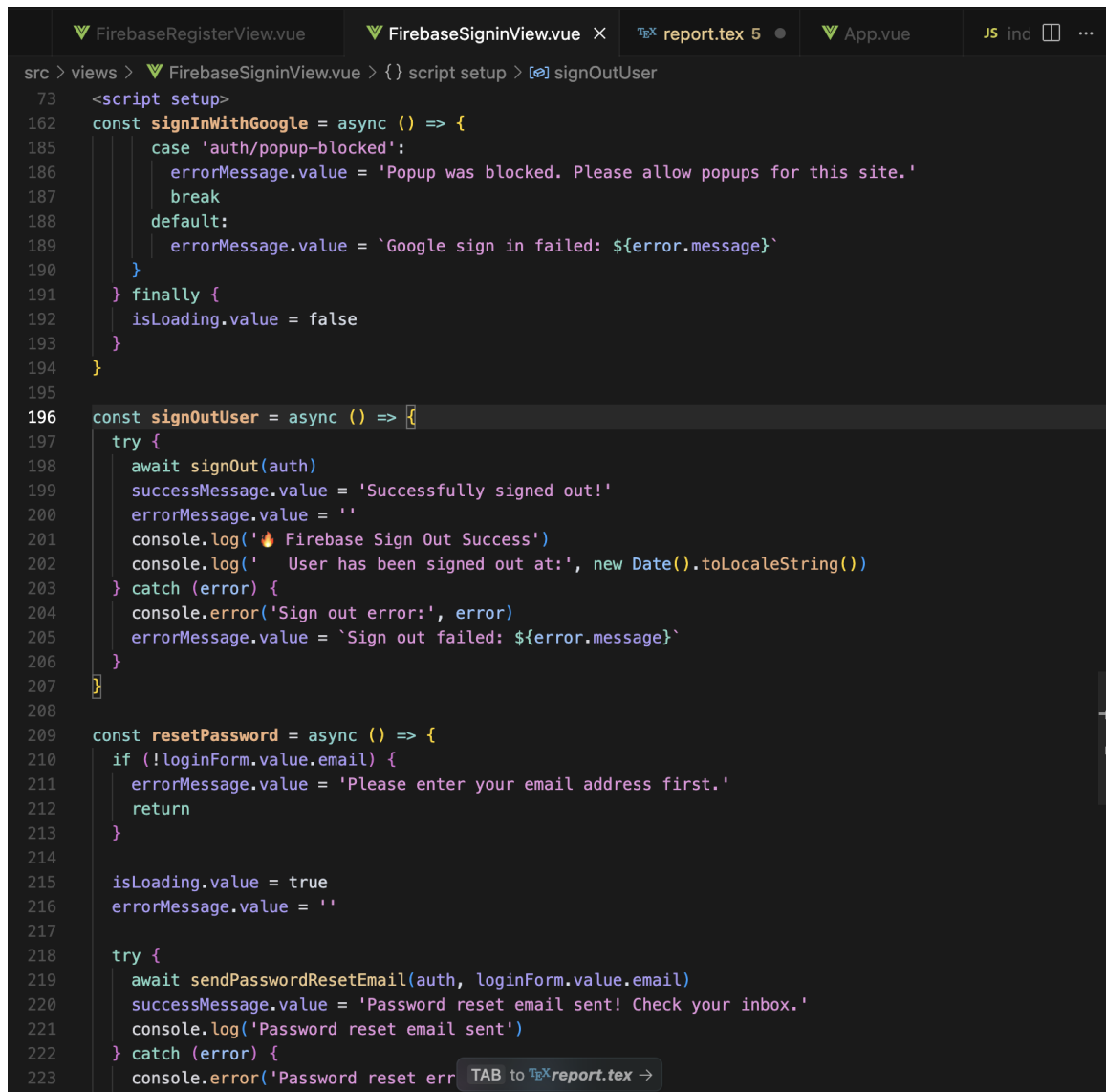
Sign In with Google

Forgot Password?

Figure 3: FirestoreSignInView.vue login page displayed in browser

**Required:** Screenshot of login page in browser with email and password fields.

### 1.2.2 Login Page in VS Code with Console Panel



```

src > views > FirebaseSignInView.vue > {} script setup > [e] signOutUser
73  <script setup>
162  const signInWithGoogle = async () => {
185      case 'auth/popup-blocked':
186          errorMessage.value = 'Popup was blocked. Please allow popups for this site.'
187          break
188      default:
189          errorMessage.value = `Google sign in failed: ${error.message}`
190      }
191  } finally {
192      isLoading.value = false
193  }
194  }
195
196  const signOutUser = async () => {
197      try {
198          await signOut(auth)
199          successMessage.value = 'Successfully signed out!'
200          errorMessage.value = ''
201          console.log('🔥 Firebase Sign Out Success')
202          console.log('  User has been signed out at:', new Date().toLocaleString())
203      } catch (error) {
204          console.error('Sign out error:', error)
205          errorMessage.value = `Sign out failed: ${error.message}`
206      }
207  }
208
209  const resetPassword = async () => {
210      if (!loginForm.value.email) {
211          errorMessage.value = 'Please enter your email address first.'
212          return
213      }
214
215      isLoading.value = true
216      errorMessage.value = ''
217
218      try {
219          await sendPasswordResetEmail(auth, loginForm.value.email)
220          successMessage.value = 'Password reset email sent! Check your inbox.'
221          console.log('Password reset email sent')
222      } catch (error) {
223          console.error('Password reset err
  
```

Figure 4: FirebaseSignInView.vue in VS Code with browser console showing current user information

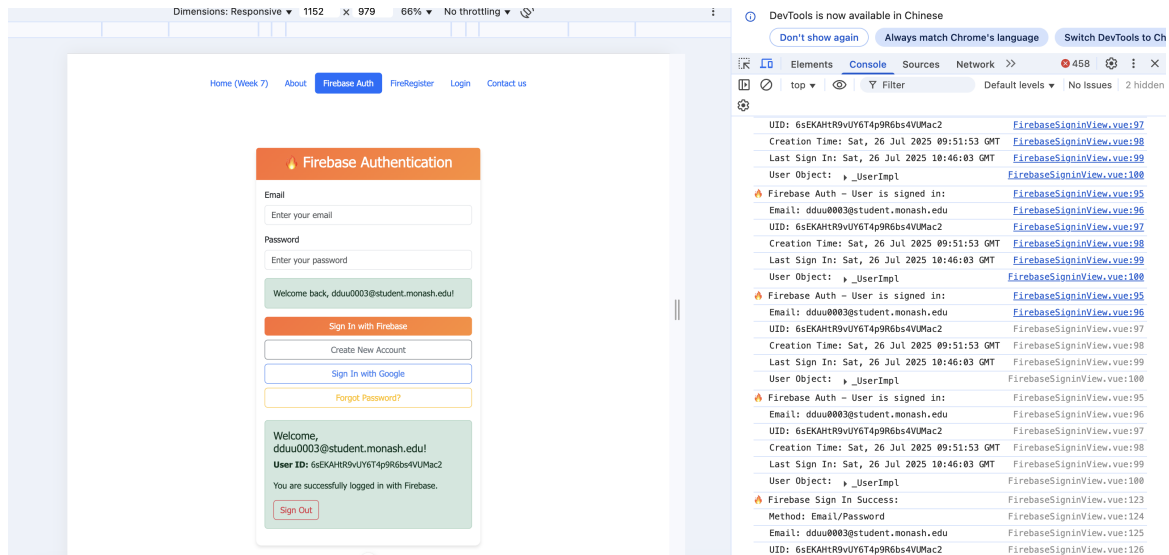


Figure 5: Firebase Authentication login page with developer console showing detailed user authentication logs

**Required:** Screenshot showing VS Code with FirebaseAuthSignInView.vue open AND browser console panel displaying current user information after successful login.

### 1.3 Screenshot 3: Registered User in Firebase Console

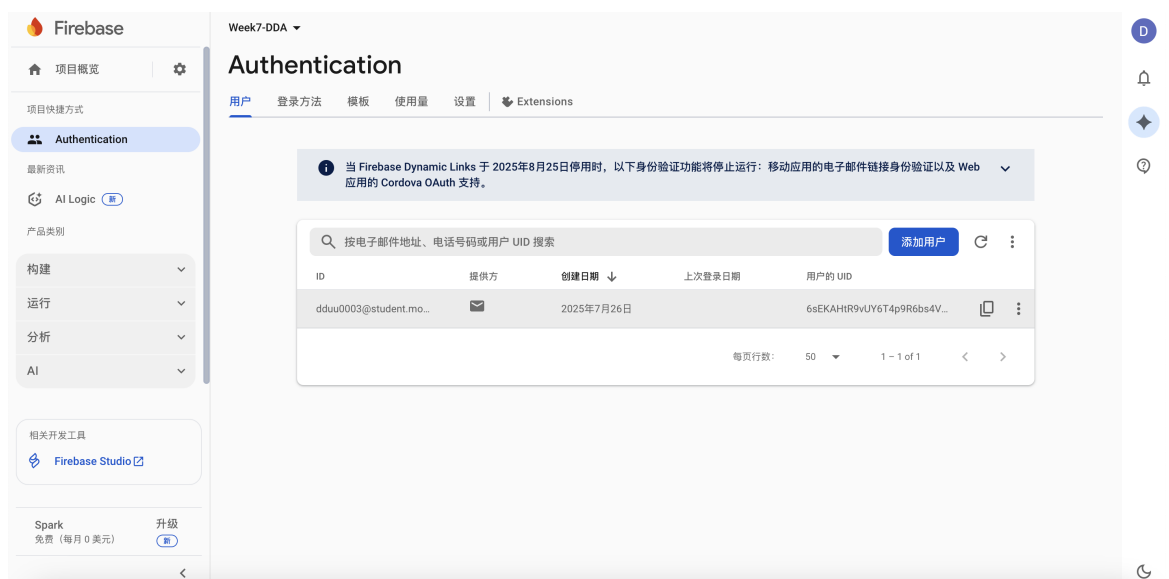


Figure 6: Firebase Console showing registered users in Authentication > Users section

**Required:** Screenshot of Firebase Console Authentication > Users section showing the registered user entries.

## 2 EFOLIO TASK 7.2 - Advanced Firestore Authentication Features

### 2.1 Screenshot Set 1: Multiple Authentication Methods

#### 2.1.1 Multiple Sign-in Methods in Browser

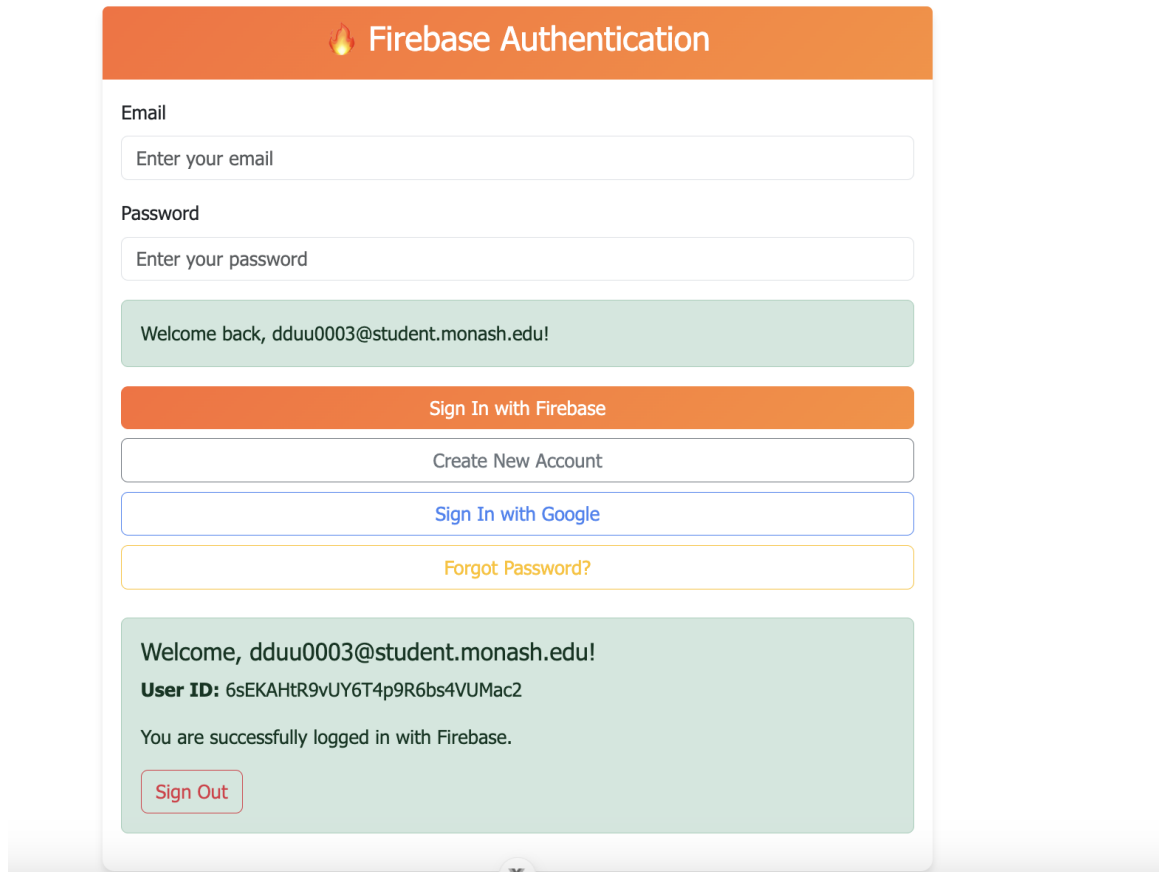
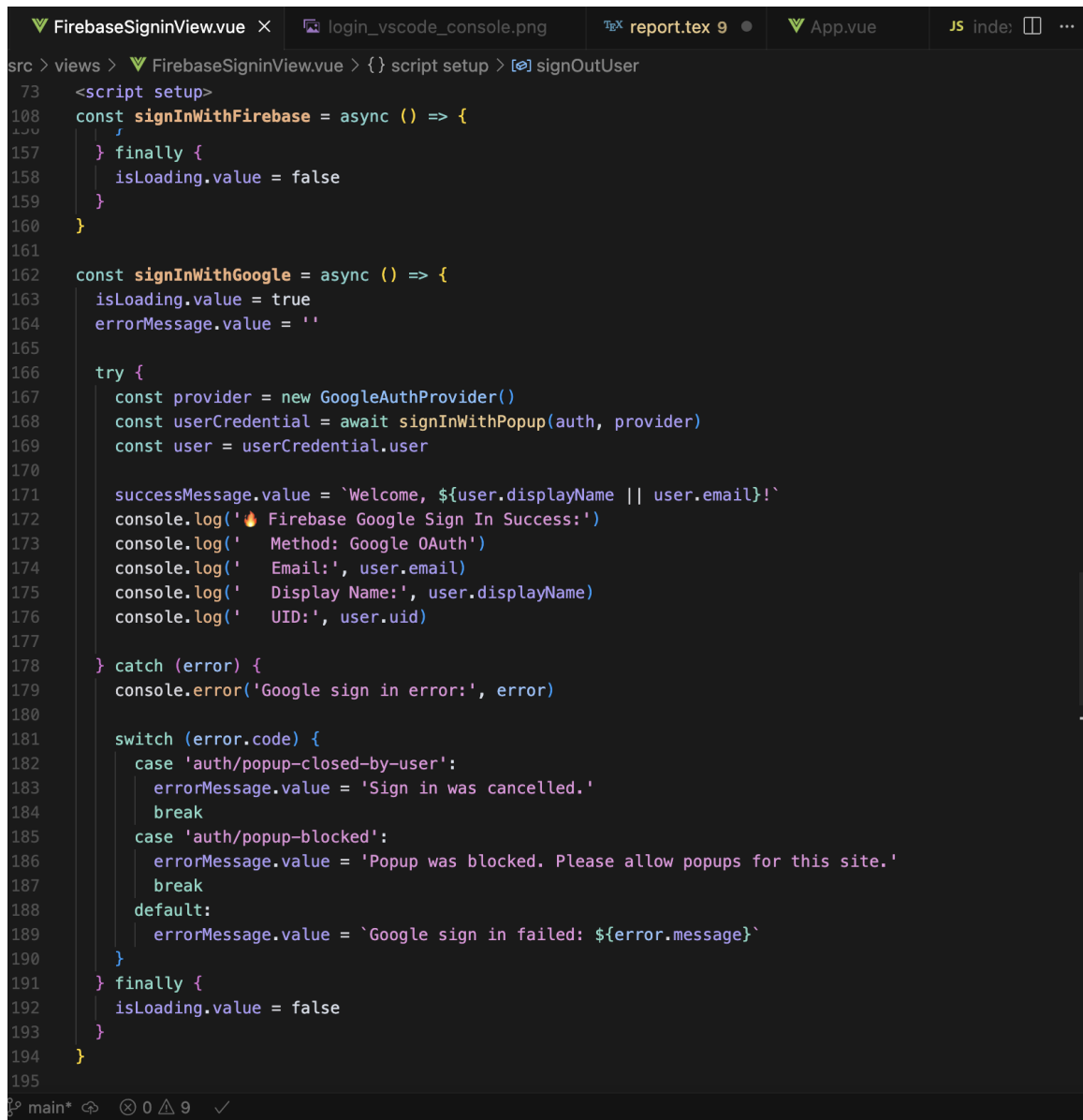


Figure 7: Browser showing multiple authentication options (Email/Password, Google OAuth)

**Required:** Screenshot of browser showing multiple sign-in options (Email/Password and Google Sign-in buttons).

### 2.1.2 Multiple Authentication Implementation in VS Code



```

src > views > FirebaseSignInView.vue > {} script setup > signOutUser
73  <script setup>
108  const signInWithFirebase = async () => {
157    } finally {
158      isLoading.value = false
159    }
160  }
161
162  const signInWithGoogle = async () => {
163    isLoading.value = true
164    errorMessage.value = ''
165
166    try {
167      const provider = new GoogleAuthProvider()
168      const userCredential = await signInWithPopup(auth, provider)
169      const user = userCredential.user
170
171      successMessage.value = `Welcome, ${user.displayName || user.email}!`
172      console.log('🔥 Firebase Google Sign In Success:')
173      console.log('  Method: Google OAuth')
174      console.log('  Email:', user.email)
175      console.log('  Display Name:', user.displayName)
176      console.log('  UID:', user.uid)
177
178    } catch (error) {
179      console.error('Google sign in error:', error)
180
181      switch (error.code) {
182        case 'auth/popup-closed-by-user':
183          errorMessage.value = 'Sign in was cancelled.'
184          break
185        case 'auth/popup-blocked':
186          errorMessage.value = 'Popup was blocked. Please allow popups for this site.'
187          break
188        default:
189          errorMessage.value = `Google sign in failed: ${error.message}`
190      }
191    } finally {
192      isLoading.value = false
193    }
194  }
195

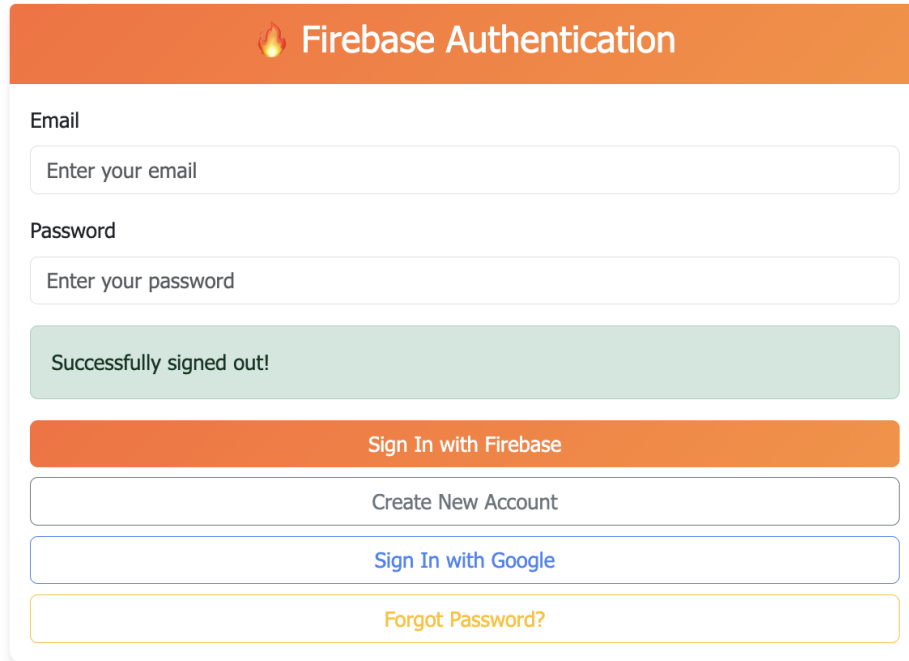
```

Figure 8: VS Code showing implementation of multiple authentication methods

**Required:** Screenshot of VS Code showing the implementation of multiple authentication methods in the source code.

## 2.2 Screenshot Set 2: Logout Functionality

### 2.2.1 Logout Page in Browser



The screenshot shows the Firebase Authentication login interface. At the top is an orange header with the Firebase logo and the text "Firebase Authentication". Below the header, there are two input fields: "Email" with the placeholder "Enter your email" and "Password" with the placeholder "Enter your password". A green message box in the center states "Successfully signed out!". Below this, there are four buttons: "Sign In with Firebase" (orange), "Create New Account" (white with orange border), "Sign In with Google" (white with blue border), and "Forgot Password?" (white with orange border).

Figure 9: Browser showing logout functionality and user authentication state

**Required:** Screenshot of browser showing logout functionality with user information displayed.



## 2.2.2 Developer Console Showing Current User

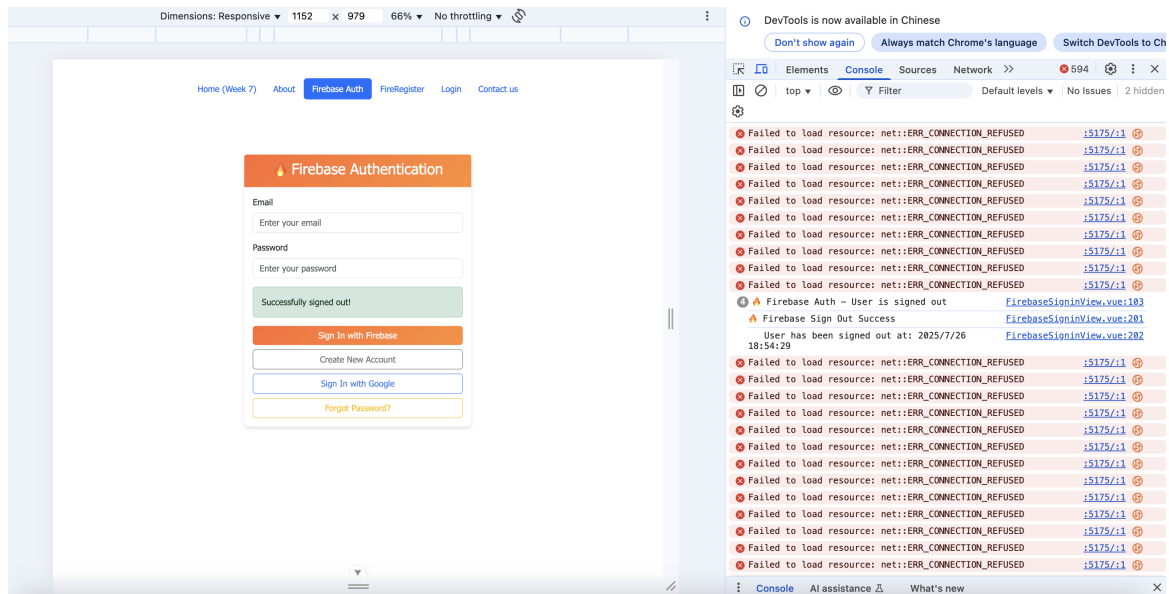


Figure 10: Developer console showing current user information and logout process

**Required:** Screenshot of browser developer console showing current user information and the logout process with detailed console logs.

## 3 Appendix: Key Implementation Code

### 3.1 Firebase Configuration (main.js)

Listing 1: Firebase initialization in main.js

```

1 // Firebase configuration
2 import { initializeApp } from "firebase/app";
3
4 const firebaseConfig = {
5   apiKey: "AIzaSyBNRlwHmMrzr0BzLzfo-dlH3tEY53DuF7k",
6   authDomain: "week7-dda.firebaseio.com",
7   projectId: "week7-dda",
8   storageBucket: "week7-dda.firebaseio.com",
9   messagingSenderId: "56980101562",
10  appId: "1:56980101562:web:addd00ffa74363d48a2583"
11 };
12
13 // Initialize Firebase
14 const firebaseApp = initializeApp(firebaseConfig);

```

### 3.2 Registration Implementation

Listing 2: Key registration function from FirebaseRegisterView.vue

```

1 import { getAuth, createUserWithEmailAndPassword } from 'firebase/auth'
2
3 const registerUser = async () => {
4   try {
5     const auth = getAuth()
6     const userCredential = await createUserWithEmailAndPassword(
7       auth,
8       registerForm.value.email,
9       registerForm.value.password
10    )
11
12    const user = userCredential.user
13    console.log('Firebase Registration Success:', user)
14
15   } catch (error) {
16     console.error('Registration error:', error)
17   }
18 }

```

### 3.3 Sign-in Implementation

Listing 3: Key sign-in function from FirebaseSignInView.vue

```

1 import { getAuth, signInWithEmailAndPassword, onAuthStateChanged } from 'firebase/
  auth'
2
3 const signInWithFirebase = async () => {
4   try {
5     const auth = getAuth()
6     const userCredential = await signInWithEmailAndPassword(
7       auth,
8       loginForm.value.email,
9       loginForm.value.password
10    )
11

```

```
12     const user = userCredential.user
13     console.log('Firebase Sign In Success:', user)
14
15   } catch (error) {
16     console.error('Sign in error:', error)
17   }
18 }
19
20 // Monitor authentication state
21 onAuthStateChanged(auth, (currentUser) => {
22   if (currentUser) {
23     console.log('User is signed in:', currentUser)
24   } else {
25     console.log('User is signed out')
26   }
27 })
```

## 4 Screenshot Capture Guide

### 4.1 Task 7.1 Screenshots

#### Screenshot Set 1 - Registration Page:

- **registration\_browser.png:** Navigate to `http://localhost:5175/FireRegister`, capture full browser window
- **registration\_vscode.png:** Open `src/views/FirebaseRegisterView.vue` in VS Code, capture full window

#### Screenshot Set 2 - Login Page:

- **login\_browser.png:** Navigate to `http://localhost:5175/firebase-signin`, capture browser window
- **login\_vscode\_console.png:** Split screen showing VS Code with `FirebaseSignInView.vue` AND browser with DevTools console open (F12) after successful login

#### Screenshot 3 - Firebase Console:

- **firebase\_registered\_user.png:** Go to Firebase Console › Authentication › Users, capture user list

### 4.2 Task 7.2 Screenshots

#### Screenshot Set 1 - Multiple Authentication:

- **multiple\_signin\_browser.png:** Capture browser showing Email/Password + Google Sign-in buttons
- **multiple\_signin\_vscode.png:** Show VS Code with Google OAuth implementation code

#### Screenshot Set 2 - Logout Functionality:

- **logout\_browser.png:** Show browser with logged-in user and logout button
- **logout\_console.png:** Browser DevTools console showing detailed user info and logout logs

## 5 Technical Achievement Summary

This implementation demonstrates:

- **Firebase Integration:** Complete setup with Vue.js application
- **User Registration:** Email/password registration with validation
- **User Authentication:** Sign-in with detailed console logging
- **State Management:** Real-time authentication state monitoring
- **Multiple Auth Methods:** Email/password and Google OAuth options
- **Security Features:** Proper error handling and user feedback
- **Development Tools:** Console logging for debugging and verification