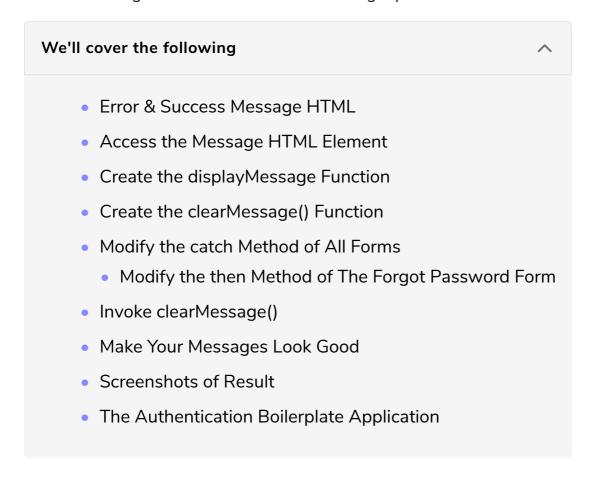# Handling Error and Success Messages

When you try to sign in or create a user, things can go wrong. The user might type in an incorrect email or the wrong password for instance. Firebase returns an error which we can show to the user. This helps them know why they are not getting the results they expected. We will also cover success messages which are relevent to the forgot password form.

We'll cover the following ^

- Error & Success Message HTML
- Access the Message HTML Element
- Create the displayMessage Function
- Create the clearMessage() Function
- Modify the catch Method of All Forms
  - Modify the then Method of The Forgot Password Form
- Invoke clearMessage()
- Make Your Messages Look Good
- Screenshots of Result
- The Authentication Boilerplate Application

Error & Success Message HTML

We create a div with the id of `message`. This div will be just below the forgot password form inside your modal. It's not specifically for that form, though. It will be available as a message center for all of our authentication forms.

```html
<!-- Success and error messages -->
<div id="message"></div>
```

HTML

Access the Message HTML Element

In the JavaScript file, we get access to the HTML element we just created.

```javascript
// Access the message HTML element
```

```javascript
const authMessage = document.getElementById('message')
```

Create the `displayMessage` Function

The function will take two parameters:

1. type
2. message

type will be passed as a string with a value either, `success` or `error`.

message will be passed a string returned by Firebase Authentication.

Lastly, we will create some functionality for hiding the message after seven seconds.

```javascript
// Makes the messageTimeout global so that the clearTimeout method will work when invoked
let messageTimeout

// Error and message handling
displayMessage = (type, message) => {
    if (type === 'error'){
        authMessage.style.borderColor = 'red'
        authMessage.style.color = 'red'
        authMessage.style.display = 'block'
    } else if (type === 'success'){
        authMessage.style.borderColor = 'green'
        authMessage.style.color = 'green'
        authMessage.style.display = 'block'
    }

    authMessage.innerHTML = message
    messageTimeout = setTimeout(() => {
        authMessage.innerHTML = ''
        authMessage.style.display = 'none'
    }, 7000)
}
```

Create the `clearMessage()` Function

Even though our message will clear after seven seconds, we need to clear it when toggling between forms. That's what the following function is for:

```javascript
clearMessage = () => {
  clearTimeout(messageTimeout)
  authMessage.innerHTML = ''
  authMessage.style.display = 'none'
```

```
}
```

Modify the `catch` Method of All Forms

All of our email and password authentication forms need to have the `displayMessage` function invoked from the `.catch()` method of each asynchronous request to Firebase.

Modify the `then` Method of The Forgot Password Form

In the forgot password form, we need to invoke `displayMessage` in the `then` as well because we have a success message we want to show our users.

```javascript
// Create user form submit event
createUserForm.addEventListener('submit', event => {
  event.preventDefault()
  // Grab values from form
  const displayName = document.getElementById('create-user-display-name').value
  const email = document.getElementById('create-user-email').value
  const password = document.getElementById('create-user-password').value
  // Send values to Firebase
  auth.createUserWithEmailAndPassword(email, password)
    .then(() => {
      auth.currentUser.updateProfile({
        displayName: displayName
      })
      createUserForm.reset()
    })
    .catch(error => {
      displayMessage('error', error.message);
    })
})

// Sign in form submit event
signInForm.addEventListener('submit', event => {
        event.preventDefault()
        // Grab values from form
  const email = document.getElementById('sign-in-email').value
        const password = document.getElementById('sign-in-password').value
        // Send values to Firebase
        auth.signInWithEmailAndPassword(email, password)
    .then(() => {
     signInForm.reset()
     hideAuthElements()
    })
    .catch(error => {
      displayMessage('error', error.message)
    })
})

// Forgot password form submit event
forgotPasswordForm.addEventListener('submit', event => {
        event.preventDefault()
        // Grab value from form
```

```javascript
        var emailAddress = document.getElementById('forgot-password-email').value
        // Send value to Firebase
    firebase.auth().sendPasswordResetEmail(emailAddress)

    .then(() => {
      forgotPasswordForm.reset()
      displayMessage('success', 'Message sent. Please check your email')
     })
    .catch(error => {
      displayMessage('error', error.message)
    })
})
```

Invoke `clearMessage()`

As we toggle between our authentication forms, we want to clear any message that may have been showing at that moment.

To do this, we invoke the `clearMessage()` function from inside the `hideAuthElements()` function.

```javascript
// Invoked at the start of auth functions in order to hide everything before selectively showing th
hideAuthElements = () => {
  clearMessage()
  createUserForm.classList.add('hide')
  signInForm.classList.add('hide')
  forgotPasswordForm.classList.add('hide')
  createUserDialog.classList.add('hide')
  signInDialog.classList.add('hide')
  haveOrNeedAccountDialog.classList.add('hide')
}
```

Make Your Messages Look Good

We want to make our program look good for our users, so here is some helpful CSS:

```css
#message{
    display: none;
    border: 1px solid;
    position: relative;
    margin: 40px 20px 20px 20px;
    padding: 10px;
}
```

Screenshots of Result

sdf@fda.com   ••••

                              Forgot?

                Sign In

┌─────────────────────────────────┐
│   There is no user record   │
│ corresponding to this identifier. │
│   The user may have been deleted. │
└─────────────────────────────────┘

        Don't have an account? Create User

Email
_____

            Send Recovery Email

┌─────────────────────────────────┐
│   Message sent. Please check your │
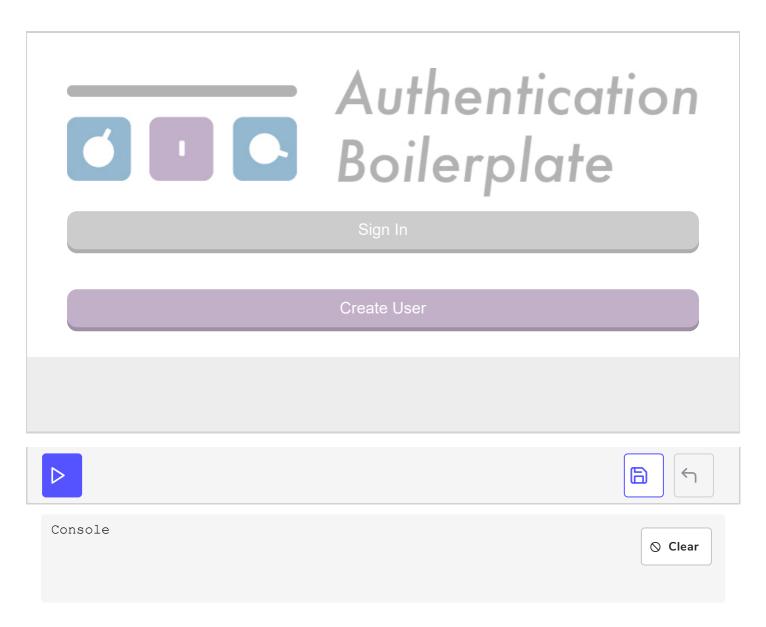│   email   │
└─────────────────────────────────┘

The Authentication Boilerplate Application

Check out the playground below and try to create login errors to see them in action. You might also submit the reset password form so you can see a success message as well.

This code requires the following API keys to execute: ⌃

apiKey   Not Specified...

authDomain   Not Specified...

databaseURL   Not Specified...

projectId   Not Specified...

storageBucket   Not Specified...

messagingSenderId   Not Specified...

appId   Not Specified...

                        Output

                       JavaScript

                         HTML

                       CSS (SCSS)

Authentication Boilerplate

Sign In

Create User

Console

⊘ Clear

In the next lesson, we will add a loading visual so we have some kind of feedback while Firebase is processing a request.