Lab 2: Firebase Multi-Factor Authentication (MFA) — React Setup & Execution

What we'll build: A React application with Firebase Authentication that supports email/password login, email verification, and SMS-based Multi-Factor Authentication (MFA). Users enroll a phone number as a second factor and are challenged during sign-in.

1) Prerequisites & Requirements

Component	Minimum Requirement	Notes
Node.js + npm	Node 14+	Check with node -v, npm -v
IDE	VSCode	Any IDE that supports React/JS
Firebase Project	Google Firebase	Enable Authentication services
Firebase SDK	Modular v9 or v10	Installed via npm in project
Libraries	firebase, react	Installed inside your project
Authorized domains	localhost, 127.0.0.1	Must be added in Firebase console

2) Firebase Console Setup

1. Go to Firebase Console → Create a new project.
2. Register a Web App (</>) and copy the configuration snippet.
3. Enable Authentication → Providers:
   - Email/Password
   - Phone
4. Enable Multi-Factor Authentication (SMS).
5. Add Authorized domains: localhost, 127.0.0.1.
6. Under Phone Authentication, add test numbers for development.

3) Start a New React Project

```
npx create-react-app mfa-demo-react
cd mfa-demo-react
npm install firebase
npm start
```

Open the app at: http://localhost:3000

4) Project Structure & Files

```
mfa-demo-react/
├─ package.json
└─ src/
   ├─ index.js
   ├─ firebase.js
   └─ App.js
```

4.1 firebase.js

```
import { initializeApp } from "firebase/app";
import { getAuth } from "firebase/auth";

const firebaseConfig = {
  apiKey: "YOUR_API_KEY",
  authDomain: "YOUR_PROJECT.firebaseapp.com",
  projectId: "YOUR_PROJECT",
  storageBucket: "YOUR_PROJECT.appspot.com",
  messagingSenderId: "YOUR_MESSAGING_SENDER_ID",
  appId: "YOUR_APP_ID",
  measurementId: "YOUR_MEASUREMENT_ID"
};

const app = initializeApp(firebaseConfig);
export const auth = getAuth(app);
export default app;
```

4.1 index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import App from './App';

const root =
ReactDOM.createRoot(document.getElementById('root')
);
root.render(<App />);
```

4.3 App.js (core MFA logic) implements:

- Email/Password signup & verification
- Sign-in flow with MFA enrollment (phone number)
- Handling MFA challenge during login

Full code in Appendix section

```
// Signup with Email + Verification
async function handleSignup(email, password) {
  const cred = await createUserWithEmailAndPassword(auth, email, password);
  await sendEmailVerification(cred.user);
  alert("✅ Verify your email before enrolling MFA.");
}

// Signin
async function handleSignin(email, password) {
  await signInWithEmailAndPassword(auth, email, password);
  alert("✅ Signed in!");
}

// Signout
function handleSignout() {
  return signOut(auth);
}

// Send OTP for MFA Enrollment
async function sendOtp(phone) {
  const recaptchaVerifier = new RecaptchaVerifier(
    getAuth(),
    "recaptcha-container",
    {}
  );
  const mfaSession = await multiFactor(auth.currentUser).getSession();
  const provider = new PhoneAuthProvider(auth);
  const vId = await provider.verifyPhoneNumber(
    { phoneNumber: phone, session: mfaSession },
    recaptchaVerifier
  );
  alert("📲 OTP sent!");
  return vId;
}

// Verify OTP & Enroll MFA
async function verifyOtp(verificationId, otp) {
  const cred = PhoneAuthProvider.credential(verificationId, otp);
  const assertion = PhoneMultiFactorGenerator.assertion(cred);
  await multiFactor(auth.currentUser).enroll(assertion, "Phone");
  alert("✅ MFA Enrolled!");
}
```

5) Running the Project

1. Sign up with email/password → check inbox → verify email.
2. Login → MFA enrollment UI is shown.
3. Enter a test phone number, send OTP, verify & enroll.
4. Sign out → sign in again → MFA challenge (OTP) required.
5. Enter OTP → MFA login success.

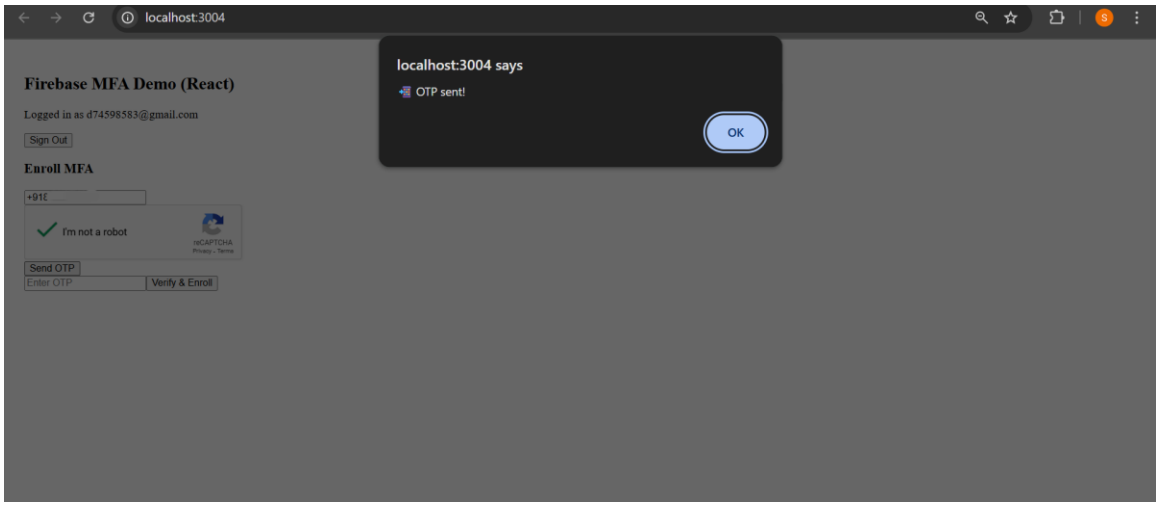
6) Troubleshooting

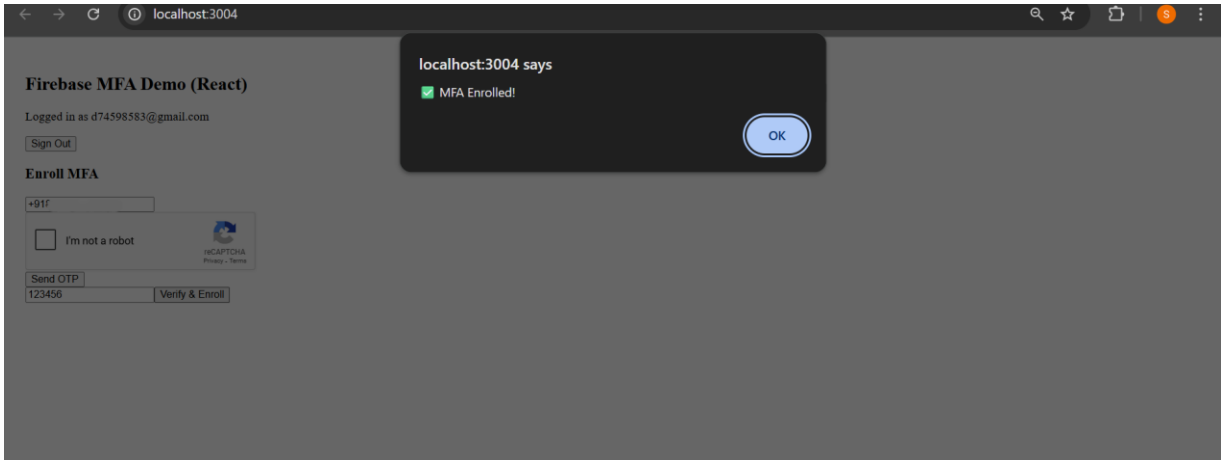
Symptom	Cause	Fix
OTP not delivered	Using real numbers in dev	Use Firebase test numbers
Error: appVerificationDisabledForTesting	Old v8 flag removed in v10	Do not use auth.settings.appVerificationDisabledForTesting
reCAPTCHA not rendering	Missing div	Ensure exists
Login blocked	Missing domain	Add localhost in Firebase Authorized domains

7) Validation & Testing Checklist

✓Sign up, verify email, and enroll phone number.
✓On re-login, OTP challenge is triggered.
✓OTP verification allows access.
✓Test both success and failure flows.

Sample Images:





Appendix A — Full App.js Implementation

```
import React, { useState, useEffect } from "react";
import {
  createUserWithEmailAndPassword,
  signInWithEmailAndPassword,
  sendEmailVerification,
  signOut,
  onAuthStateChanged,
  multiFactor,
  PhoneAuthProvider,
  PhoneMultiFactorGenerator,
  RecaptchaVerifier,
  getAuth
} from "firebase/auth";
import { auth } from "./firebase";

function App() {
  const [user, setUser] = useState(null);
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
  const [phone, setPhone] = useState("");
  const [otp, setOtp] = useState("");
  const [verificationId, setVerificationId] = useState("");

  useEffect(() => {
    const unsubscribe = onAuthStateChanged(auth, (u) => setUser(u));
    return () => unsubscribe();
  }, []);

  // Signup
  const handleSignup = async () => {
    try {
      const cred = await createUserWithEmailAndPassword(auth, email, password);
      await sendEmailVerification(cred.user);
      alert("✅ Signed up! Verify your email before MFA.");
    } catch (e) {
```

```
      alert(e.message);
    }
  };

  // Signin
  const handleSignin = async () => {
    try {
      await signInWithEmailAndPassword(auth, email, password);
      alert("✅ Signed in!");
    } catch (e) {
      alert(e.message);
    }
  };

  // Signout
  const handleSignout = () => signOut(auth);

  // Send OTP
  const sendOtp = async () => {
    try {
      // ✅ Create RecaptchaVerifier instance
      const recaptchaVerifier = new RecaptchaVerifier(
        getAuth(),              // MUST pass getAuth()
        "recaptcha-container",  // div id
        {}
      );

      const mfaSession = await multiFactor(auth.currentUser).getSession();
      const phoneAuthProvider = new PhoneAuthProvider(auth);

      const vId = await phoneAuthProvider.verifyPhoneNumber(
        {
          phoneNumber: phone, // user input
          session: mfaSession
        },
        recaptchaVerifier
      );

      setVerificationId(vId);
      alert("□ OTP sent!");
      console.log("verificationId:", vId);
    } catch (e) {
      alert("Send OTP failed: " + e.message);
      console.error(e);
    }
  };

  // Verify OTP and Enroll
  const verifyOtp = async () => {
    try {
      const cred = PhoneAuthProvider.credential(verificationId, otp);
      const multiFactorAssertion = PhoneMultiFactorGenerator.assertion(cred);

      await multiFactor(auth.currentUser).enroll(multiFactorAssertion,
"Personal Phone");
      alert("✅ MFA Enrolled!");
    } catch (e) {
```

```
      alert("Verify OTP failed: " + e.message);
      console.error(e);
    }
  };

  return (
    <div style={{ padding: "20px" }}>
      <h2>Firebase MFA Demo (React)</h2>

      {!user && (
        <div>
          <input placeholder="Email" onChange={(e) => setEmail(e.target.value)}
/><br />
          <input type="password" placeholder="Password" onChange={(e) =>
setPassword(e.target.value)} /><br />
          <button onClick={handleSignup}>Sign Up</button>
          <button onClick={handleSignin}>Sign In</button>
        </div>
      )}

      {user && (
        <div>
          <p>Logged in as {user.email}</p>
          <button onClick={handleSignout}>Sign Out</button>

          {user.emailVerified && (
            <div>
              <h3>Enroll MFA</h3>
              <input
                placeholder="+91XXXXXXXXXX"
                onChange={(e) => setPhone(e.target.value)}
              /><br />
              <div id="recaptcha-container"></div>
              <button onClick={sendOtp}>Send OTP</button>
              <br />
              <input placeholder="Enter OTP" onChange={(e) =>
setOtp(e.target.value)} />
              <button onClick={verifyOtp}>Verify & Enroll</button>
            </div>
          )}
        </div>
      )}
    </div>
  );
}

export default App;
```