```
// --------------------------- GLOBAL VARIABLES --------------------
-------

let currentUser = null;

let loginAttempts = 0;

const MAX_ATTEMPTS = 3;

const LOGOUT_TIMER = 5 * 60 * 1000; // 5 minutes auto logout


// --------------------------- UTILITY FUNCTIONS -------------------
--------


/**

 * Generate SHA-256 hash for passwords (to avoid plain text storage)

 * @param {string} text

 * @returns {Promise<string>} hash

 */

async function hashPassword(text) {

  const msgBuffer = new TextEncoder().encode(text);

  const hashBuffer = await crypto.subtle.digest("SHA-256",
msgBuffer);

  const hashArray = Array.from(new Uint8Array(hashBuffer));

  return hashArray.map(b => b.toString(16).padStart(2,
"0")).join("");

}


/**

 * Simple email validation using regex

 * @param {string} email

 * @returns {boolean}

 */

function validateEmail(email) {

  const regex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
```

```
  return regex.test(email);
}

/**
 * Helper function to show alert and log
 */
function showMessage(msg, type = "info") {
  console.log(`[${type.toUpperCase()}] ${msg}`);
  alert(msg);
}

/**
 * Fetch user by email from LocalStorage
 */
function getUser(email) {
  const data = localStorage.getItem(email);
  return data ? JSON.parse(data) : null;
}

/**
 * Save user in LocalStorage
 */
function saveUser(user) {
  localStorage.setItem(user.email, JSON.stringify(user));
}

// --------------------------- SIGNUP FUNCTION ----------------------
------

async function signupUser(name, email, password) {
```

```
  if (!name || !email || !password) {
    showMessage("All fields are required!", "error");
    return;
  }

  if (!validateEmail(email)) {
    showMessage("Invalid email format!", "error");
    return;
  }

  const existingUser = getUser(email);
  if (existingUser) {
    showMessage("User already exists! Please login.", "warning");
    return;
  }

  const hashed = await hashPassword(password);
  const user = {
    name,
    email,
    password: hashed,
    createdAt: new Date().toISOString(),
    lastLogin: null,
  };

  saveUser(user);
  showMessage("Signup successful! You can now login.", "success");
}
```

```
// -------------------------- LOGIN FUNCTION ----------------------
-----


async function loginUser(email, password) {
  if (!email || !password) {
    showMessage("Please fill in all fields!", "error");
    return;
  }

  const user = getUser(email);
  if (!user) {
    showMessage("No user found. Please sign up first.", "error");
    return;
  }

  const hashed = await hashPassword(password);
  if (user.password !== hashed) {
    loginAttempts++;
    if (loginAttempts >= MAX_ATTEMPTS) {
      showMessage("Account locked due to 3 failed attempts!",
"error");
      return;
    }
    showMessage(
      `Invalid password! Attempts left: ${MAX_ATTEMPTS -
loginAttempts}`,
      "warning"
    );
    return;
  }
```

```
  // Successful login
  loginAttempts = 0;
  user.lastLogin = new Date().toISOString();
  saveUser(user);
  localStorage.setItem("currentUser", JSON.stringify(user));
  currentUser = user;

  showMessage(`Welcome ${user.name}! You are now logged in.`,
"success");

  // Start auto logout timer
  startLogoutTimer();

  console.table(user);
}

// --------------------------- AUTO LOGOUT -------------------------
--

let logoutTimeout = null;

function startLogoutTimer() {
  if (logoutTimeout) clearTimeout(logoutTimeout);

  logoutTimeout = setTimeout(() => {
    showMessage("Session expired! Logging out automatically.",
"info");
    logoutUser();
  }, LOGOUT_TIMER);
}
```

```
/**
 * Reset timer if user performs an action
 */
function resetLogoutTimer() {
  if (logoutTimeout) {
    clearTimeout(logoutTimeout);
    startLogoutTimer();
  }
}

// --------------------------- LOGOUT FUNCTION ---------------------
------

function logoutUser() {
  if (currentUser) {
    console.log(`User ${currentUser.email} logged out.`);
  }
  currentUser = null;
  localStorage.removeItem("currentUser");
  showMessage("Logged out successfully!", "info");
}

// --------------------------- SESSION CHECK ----------------------
----

function checkSession() {
  const storedUser = localStorage.getItem("currentUser");
  if (storedUser) {
    currentUser = JSON.parse(storedUser);
    showMessage(`Session active: ${currentUser.name}`, "info");
    startLogoutTimer();
```

```
  } else {
    showMessage("No active session. Please login.", "warning");
  }
}

// -------------------------- ADMIN MODE -------------------------
-

/**
 * View all registered users in console
 * (for admin use only)
 */
function viewAllUsers() {
  const keys = Object.keys(localStorage);
  console.log("===== Registered Users =====");
  keys.forEach(k => {
    if (k.includes("@")) {
      const user = JSON.parse(localStorage.getItem(k));
      console.log(`Name: ${user.name}, Email: ${user.email}`);
    }
  });
  console.log("============================");
}

/**
 * Delete a user (admin only)
 */
function deleteUser(email) {
  if (!getUser(email)) {
    showMessage("User not found!", "error");
```

```
    return;
  }
  localStorage.removeItem(email);
  showMessage(`User ${email} deleted successfully!`, "success");
}

// --------------------------- PASSWORD RESET ----------------------
-----

async function resetPassword(email, newPassword) {
  const user = getUser(email);
  if (!user) {
    showMessage("User not found!", "error");
    return;
  }

  const hashed = await hashPassword(newPassword);
  user.password = hashed;
  saveUser(user);
  showMessage("Password reset successful!", "success");
}

// --------------------------- AUTO EVENTS -------------------------
--

/**
 * Monitor user activity (to reset logout timer)
 */
document.addEventListener("mousemove", resetLogoutTimer);
document.addEventListener("keypress", resetLogoutTimer);
```

```
/**
 * On page load, check if session exists
 */
window.onload = function () {
  const activeUser = localStorage.getItem("currentUser");
  if (activeUser) {
    currentUser = JSON.parse(activeUser);
    console.log(`Welcome back, ${currentUser.name}`);
    startLogoutTimer();
  }
};

// --------------------------- DEMO ACTIONS ------------------------
---

// You can test the functions in browser console, for example:
// signupUser("Vibin", "vibin@gmail.com", "12345");
// loginUser("vibin@gmail.com", "12345");
// viewAllUsers();
// resetPassword("vibin@gmail.com", "newpass");
// logoutUser();

console.log("IBM-FE Login Authentication System JS loaded
successfully!");
```