Introduction & Objective

- **Introduction:** Modern web applications require secure user authentication, which is often complex and risky to build from scratch.
- **Project Objective:** To develop a functional blog application that delegates all user authentication tasks to a professional, managed cloud service—**IBM App ID**.

The Problem Statement

Building a custom authentication system involves significant challenges:

- Securely hashing and storing user passwords.
- Managing user sessions and preventing hijacking.
- Implementing features like "forgot password" and email verification.
- Adding social logins (Google, etc.) requires separate, complex integrations.

This project aims to solve these problems by using an Identity as a Service (IDaaS) platform.

The Solution: IBM App ID

This project uses **IBM App ID** to handle all aspects of user identity.

- **Enhanced Security:** Leverages IBM's security expertise to protect user credentials.
- **Rapid Development:** Eliminates the need to write complex code for login, registration, and user management.
- **Scalability & Features:** Easily handles a growing number of users and supports features like social logins and multi-factor authentication with simple configuration changes.

System Architecture

The application follows a standard and secure authentication flow.

1. **User Request:** A user tries to access a protected page (e.g., "Create a New Post").
2. **Redirect to App ID:** The application redirects the user to a secure login page hosted by IBM.
3. **User Authentication:** The user signs up or logs in on the App ID page.
4. **Token Issuance:** App ID verifies the user and sends an authorization token back to the application's `/appid/callback` route.
5. **Access Granted:** The application validates the token and grants the user access.

Technology Stack

This project was built using a modern set of tools and services.

- **Cloud Service: IBM Cloud App ID**
- **Backend: Node.js** with the **Express.js** framework
- **Authentication Middleware: Passport.js**
- **Version Control: Git** & **GitHub**
- **Image Hosting: ImageBB**

Core Implementation

The connection between the Node.js app and IBM App ID is configured with a small block of code in `server.js`.

- This configures the **Passport.js** middleware with the application's unique credentials.
- It tells our app exactly how to communicate securely with the App ID service.

```
// server.js
passport.use(new WebAppStrategy({
    tenantId: "YOUR_TENANT_ID",
    clientId: "YOUR_CLIENT_ID",
    secret: "YOUR_SECRET",
    oauthServerUrl: "YOUR_OAUTH_URL",
    redirectUri: "http://localhost:3000/appid/callback"
}));

🚀 Project Features
Secure User Authentication: Full sign-up, login, and logout functionality handled by IBM App ID.

Protected Routes: The "Create a New Post" page is only accessible to logged-in users.

Dynamic Blog Content: Authenticated users can create new posts which are displayed on the homepage.

Custom Styled UI: A visually appealing interface with a custom logo and background image.

Live Demo Flow
A demonstration of the application's user journey.

Homepage: User lands on the public homepage and sees existing posts.

Login: User clicks "Login to Create a Post" and is redirected to the App ID login screen to sign up or log in.

Authentication: After successful login, the user is redirected back to the homepage and sees a personalized welcome message.

Create Post: User clicks "Create a New Post," accesses the protected form, writes a post, and submits it.
```