# User Authentication and Management

Options for user authentication and user management:

1. I would consider using the scaffolded Identity for .Net Core – this can be easily integrated into the project and has a lot of key functionality that is very effective. Once it has been implemented then I would add an ApplicationUsers class to do any customisation and can add new fields to the User interface. I would configure the user to work based on Roles and use their roles to authorise them throughout the system and allow or disallow them access to certain modules of the system.
2. Alternatively I would create my own user authentication and management module which would allow users to set up their own account and create a password and enter necessary details for their account. Once created, an email would be sent to the user’s email address and they would need to click on the link and activate their account before it is ‘Live’. I have used the BCrypt package before which I found really helpful in creating a hashed password which would then be saved in the database for user account. This hashed password would then be used to authenticate user when they login, along with their username and password.
3. If I was developing a Web Api I would consider using Bearer tokens and would use JSON Web Tokens for user authentication. So if a user is accessing the Api they would first need to authenticate themselves with a username and password that would be Posted to the authentication endpoint. If username and password is a match on the system, then a Bearer token would be sent to user in response which they would then send when accessing any other endpoints of the API (eg. Adding a new Product to the system).

User management:

* Can use a Roles based authorisation standard for the E-Commerce site and possibly have roles for:

1. Administrator

2. Employee

3. Customer

4. Supplier