Distributed Report

1. An API is a software intermediary that allows to applications to talk to one another, for it to then specify how software components should interact with each other. Simplified an API is essentially the messenger that sends the request and tells the system what request should be completed then sends the response back.

   The API in the project is stateless as the server does not log/store any previous data that was sent from the client making it a stateless API.

   Stateful and stateless API, in a stateless API doesn't store/keep record of previous data whereas stateful applications require backing storage. Stateful protocols are the type of network protocols in which a client sends a request to the server and the server responds back according to the current state. Whereas a stateful protocol if the client sends a given request to the server then it expects some sort of response from the server.

2. Route mapping is the process of handling the given request in the correct manner to ensure it sent to the correct action in the API. An action are methods within the specified controller that the API can perform the requested function. To ensure the client can fully access the correct action on the server side, route mapping is required. For this to occur the format of the resource identifiers need to match up on both the client side in the sent URI and in the API in the controllers. Web API requires us to define the structure of the expected URI's being sent from the client, so it ensures the request is being handled correctly.

   When the given ID parameter is send using the FromQuery in the action signature, the controller uses the model to search for a user with the given ID in the linked database. The model returns the users information linked to the given ID back to the client.

3. The given HTTP protocols are:
   - GET – This request and retrieves the data from the remove resource

```
[ActionName("New")]
[HttpGet]
0 references | 0 requests | 0 exceptions
public IActionResult Get([FromQuery]string Username)
{
    bool UserExists = Models.UserDatabaseAccess.checkUsername(Username);
    if (UserExists == true)
    {
        return Ok("True - User Does Exist! Did you mean to do a POST to create a new user?");
    }
    else
    {
        return Ok("False - User Does Not Exist! Did you mean to do a POST to create a new user?");
    }
}
```

   - PUT – Updates the current data on the remove server
   - POST – Takes a new piece of data and creates a place for it on the server

```
[ActionName("New")]
[HttpPost]
0 references | 0 requests | 0 exceptions
public IActionResult Post([FromBody] string Username)
{
    bool UsernameExists = Models.UserDatabaseAccess.checkUsername(Username);
    if ((UsernameExists == false) && (Username != null))
    {
        Models.User User = Models.UserDatabaseAccess.CreateUser(Username);
        if (_context.Users.Count() == 0)
        {
            User.Role = "Admin";
        }
        else
        {
            User.Role = "User";
        }
        _context.Users.Add(User);
        _context.SaveChanges();
        return Ok(User.apiKey.ToString());
    }
    else if (UsernameExists == true && Username != null)
    {
        return Forbid("Oops. This username is already in use. Please try again with a new username.");
    }
    else
    {
        return BadRequest("Oops. Make sure your body contains a string with your username and your Content-Type is Content-Type:application/json");
    }
}
```

- DELETE – Remove data from the server.

```
[ActionName("removeuser")]
[HttpDelete]
0 references | 0 requests | 0 exceptions
public IActionResult Delete([FromHeader(Name = "ApiKey")]string Apikey,[FromQuery]string Username)
{
    Models.User user = Models.UserDatabaseAccess.checkApiExists(Apikey);
    bool Api_UserExist = Models.UserDatabaseAccess.CheckUserAPIExists(Apikey, Username);

    if (Api_UserExist == true && (user.Role == "Admin" || user.Role == "User"))
    {
        Models.UserDatabaseAccess.DeleteUser(_context, Apikey);
        return Ok(true);
    }
    else
    {
        return Ok(false);
    }
}
```

4. The APIkey is a unique database key and is the API Key for the given user. The APIKey is used to identify whether a given user is within the database, as most users are recognised by their APIKey. The APIKey overall is a good decision to use in this project for identifying users as if there is an attack at the system the hackers wouldn't be able to obtain any personal information about the given user. As in order to obtain personal information about the user the APIKey has first be validated at the server end. Also the use of a single identifier is a simple process which doesn't use much computational power. An APIKey is independatnt of any naming conventions or master credentials as they're created autonomousley. In the given project the APIKey is not safe as if a hacker gains access to the database they have access to all the APIKeys as well as the users private information.

5. The steps involved in the RSA Algorithm include:
   - Chose two prime numbers, p & q
   - Compute the value N, which is the product of P&Q.
   - Compute the value X in which X = (p -1)(q -1)
   - Find the value e (Public Key exponent) should be the co-prime of theta(n).
   - Compute the value of d (private key exponent) where e^{-1} mod theta(n) = d.
   - Do the encryption given by: C = M^e mod(n)
   - Do the decryption given by M = C^d mod(n)

6. AES algorithm is a symetric encryption technique, the algorithm is split up into 3 different stages, inintial round, main round and final round. The given processes within these rounds include:
   - SubBytes – The bytes foreach given item within the 4x4 matrix is calculated from an S-Box.
   - ShiftRows – Each row within the 4x4 matrix is shifter from left to right with the first row having 0 bytes shifted, second row having 1 bytes shifted and the third row having 2 bytes shifted and so-on.
   - MixColumns - The four bytes on the given column are modulo multiplied in Rijndael's Galois field by a given matrix. (The last two steps are the main source of diffusion in AES).
   - AddRoundKey – The Round Keys are calculated from the symmetric keys

7. Entity framework is an open-source framework for .NET applications which enables developers to work at a higher level with data using objects of domain specific classes without focusing on the underlying database table and columns where this data is stored. Entity framework handles the basic CRUD operations and allows you to perfroms queries using LINQ.

   Entity Framework Approaches:
   - Code first modelling targets a database that doesn't exist and the code will create it. During the code first approach the database along with the relationships are created on the fly in an more agile approach.
   - Database First – In this approach we are creating the entity framework from an existing database. In this approach the database is created first then the model then the code.
   - Model first – In this approach the model is created first, once the model with the specific relationships has been created the database will be generated from the given model then the code.

   A migration is a way to keep to all your data during an update to the database schema when the given model changes without losing any existing data or other database objects currently in the database.

8. All tasks are done, I completed them to the fullest, as I meet the specification for all the coding tasks. During the project I stuggled mainly with task 13 and 14 as understanding what needed to be done and how was a major wall that I had to overcome. some of the issues within the project the labs and lectures as well as general internet research was conducted to understand the problem better.