# Lecture 10

## 1. Unit Test in API

1. In the project *CourseAdminSystem.API*, add a new class *AuthenticationHelper.cs* under Middleware folder and replace the contents of the class *AuthenticationHelper* as below.

```
public class AuthenticationHelper {
    public static string Encrypt(string username, string password) {
        // 1. Concatenate credentials with a ':'
        string credentials = $"{username}:{password}";

        // 2. Retrieve bytes from text
        byte[] bytes = System.Text.Encoding.UTF8.GetBytes(credentials);

        // 3. Base64 encode credentials
        string encryptedCredentials = Convert.ToBase64String(bytes);

        // 4. Prefix credentials with 'Basic' and return
        return $"Basic {encryptedCredentials}";
    }

    public static void Decrypt(string encryptedHeader, out string username, out string
password) {
        // 1. Extract the username and password from the value by splitting it on space,
        // as the value looks something like 'Basic am9obi5kb2U6VmVyeVNlY3JldCE='
        var auth = encryptedHeader.Split(new[] { ' ' })[1];

        // 2. Convert it form Base64 encoded text, back to normal text
        var usernameAndPassword =
System.Text.Encoding.UTF8.GetString(Convert.FromBase64String(auth));

        // 3. Extract username and password, which are separated by a semicolon
        username = usernameAndPassword.Split(new[] { ':' })[0];
        password = usernameAndPassword.Split(new[] { ':' })[1];
    }
}
```

2. Add a new Unit Test project called *CourseAdminSystem.Tests* by clicking on the '+' symbol next to the Solution and choosing **MSTest Test project**. Alternatively, you can use the following commands in the terminal.

```
# From within the Folder containing the solution, i.e. CourseAdminSystemBackend
```

```
dotnet new mstest -n CourseAdminSystem.Tests

# Add the newly created project to the solution
dotnet sln add CourseAdminSystem.Tests

# Add the reference of the CourseAdminSystem.API to AuthenticationHelperTests project
dotnet add AuthenticationHelperTests reference CourseAdminSystem.API
```

3. Add the reference of *CourseAdminSystem.API* project to *CourseAdminSystem.Tests* by right-clicking on the *CourseAdminSystem.Tests* and selecting *Add Project Reference*. Choose the project name. Alternatively, you can use the following command in the terminal.

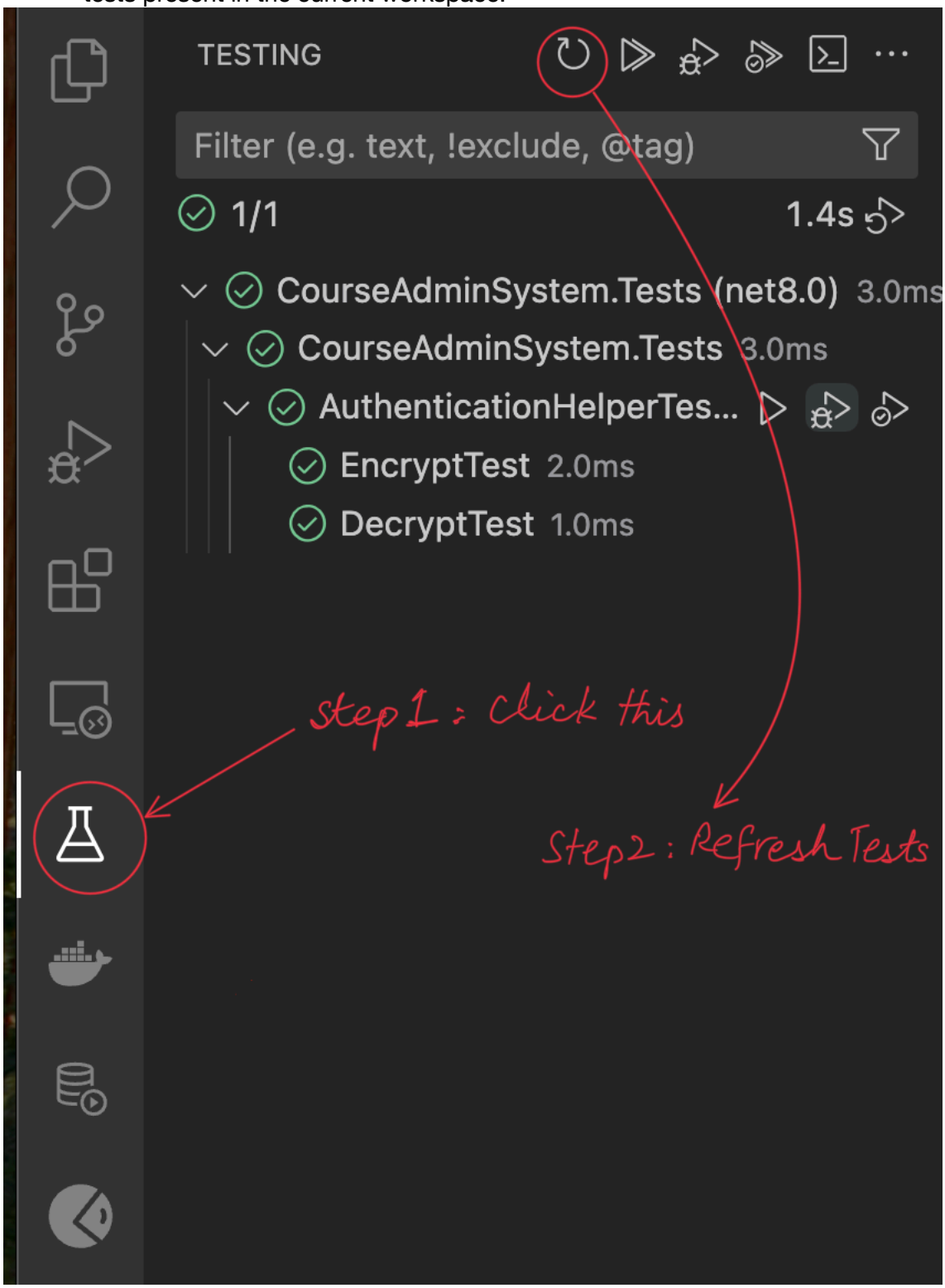
```
dotnet add CourseAdminSystem.Tests reference CourseAdminSystem.API
```

4. Rename the *UnitTests1.cs* to *AuthenticationHelperTests* and replace the contents as below.

```
namespace CourseAdminSystem.Tests;

[TestClass]
public class AuthenticationHelperTest {
   [TestMethod]
   public void EncryptTest() {
      // Arrange
      string username = "john.doe";
      string password = "VerySecret!";

      // Act
      var header = CourseAdminSystem.API.Middleware.AuthenticationHelper.Encrypt(username,
password);

      // Assert
      Assert.AreEqual("Basic am9obi5kb2U6VmVyeVNlY3JldCE=", header);
   }

   [TestMethod]
   public void DecryptTest() {
      // Arrange
      string header = "Basic am9obi5kb2U6VmVyeVNlY3JldCE=";

      // Act
      CourseAdminSystem.API.Middleware.AuthenticationHelper.Decrypt(header, out string
username, out string password);

      // Assert
      Assert.AreEqual("john.doe", username);
```

```
        Assert.AreEqual("VerySecret!", password);
    }
}
```

5. Run the required test by clicking on the green play button on the left side of the method.
    1. If the *play* button is not visible, then first click on the TESTING tab on the left side menu in VS Code as shown in the image below and then click on the *Refresh Tests* button. This will automatically identify all the tests present in the current workspace.




6. Debug the unit tests and verify that they run.

7. Once we are confident it works, we can replace the logic of our *BasicAuthenticationMiddleware* class, (steps 3, 4 & 5) to use our new *AuthenticationHelper*, as shown below

```
...
AuthenticationHelper.Decrypt(authHeaderValue, out string username, out string password);
...
```

8. Replace the logic of *LoginController* (steps 1, 2 & 3) to use our new *AuthenticationHelper* as shown below.

```
...
var headerValue = AuthenticationHelper.Encrypt(credentials.Username, credentials.Password);
...
```

9. Verify that the Angular application works.

# 2. Testing in Angular

We use the *spec.ts* files for writing tests. An example of testing our *AuthService* to make sure that it returns the correct header values, we could do the following.

1. Edit *auth.service.ts* file and add some new tests as follows.

```
import { TestBed } from '@angular/core/testing';

import { AuthService } from './auth.service';
import { provideHttpClient } from '@angular/common/http';

describe('AuthService', () => {
  let service: AuthService;

  beforeEach(() => {
   TestBed.configureTestingModule({
     providers: [provideHttpClient()]
   });
   service = TestBed.inject(AuthService);
 });

  it('should be created', () => {
    expect(service).toBeTruthy();
  });

  it('should return header', (done) => {
   service.authenticate('john.doe', 'VerySecret!').subscribe(login => {
     expect(login.headerValue).toBe('Basic am9obi5kb2U6VmVyeVNlY3JldCE=')
     done();
   });
 })
```

```
 it('should NOT return header', (done) => {
   service.authenticate('wrong', 'password').subscribe({
     error: (err) => {
       expect(err).toBeTruthy()
       done();
     }
   });
 })
});
```

2. Run the test using the following command.

```
ng test --include '**/auth.service.spec.ts'
```

3. Edit *student.service.ts* file to test for the case to retrieve a specific student with an id.

```
import { TestBed } from '@angular/core/testing';

import { StudentService } from './student.service';
import { provideHttpClient } from '@angular/common/http';

describe('StudentService', () => {
   let service: StudentService;

   beforeEach(() => {
      TestBed.configureTestingModule({
         providers: [provideHttpClient()]
      });
      service = TestBed.inject(StudentService);
   });

   it('should be created', () => {
      expect(service).toBeTruthy();
   });

   it('should return student with id 1', (done) => {
      localStorage.setItem('headerValue', 'Basic am9obi5kb2U6VmVyeVNlY3JldCE=')
      service.getStudent(1).subscribe(student => {
         expect(student.firstName).toBe('Bruce');
         done();
      });
   })
});
```

```
```

4. Run the above test using the following command

```
ng test --include '**/student.service.spec.ts'
```