Tiếng Anh:  
1. Giới thiệu bản thân và mock project.

Mockproject làm về cổng thông tin điện tử Malaysia, Trong mockproject có sử dụng kiến trúc MVC, N-Layer, Repository pattern, single ton, dependency injection,…

2. Trong mock project thì em chịu trách nhiệm các module nào.

Trong mock project em làm về phần quản lý các thông tin bài viết

3. Em làm như thế nào để import file excel (do đã trình bày thực hiện module import file excel)

Để Import file trong excel thì em dùng thư viện EppPlus trong excel

1. **public** IList<Customers> ImportCustomer()
2. {

5. string rootFolder = \_hostingEnvironment.WebRootPath;
6. string fileName = @"ImportCustomers.xlsx";
7. FileInfo file = **new** FileInfo(Path.Combine(rootFolder, fileName));
9. using (ExcelPackage **package** = **new** ExcelPackage(file))
10. {
11. ExcelWorksheet workSheet = **package**.Workbook.Worksheets["Customer"];
12. **int** totalRows = workSheet.Dimension.Rows;
14. List<Customers> customerList = **new** List<Customers>();
16. **for** (**int** i = 2; i <= totalRows; i++)
17. {
18. customerList.Add(**new** Customers
19. {
20. CustomerName = workSheet.Cells[i, 1].Value.ToString(),
21. CustomerEmail = workSheet.Cells[i, 2].Value.ToString(),
22. CustomerCountry = workSheet.Cells[i, 3].Value.ToString()
23. });
24. }
26. \_db.Customers.AddRange(customerList);
27. \_db.SaveChanges();
29. **return** customerList;
30. }
31. }
33. [HttpGet]
34. [Route("ExportCustomer")]
35. **public** string ExportCustomer()
36. {
37. string rootFolder = \_hostingEnvironment.WebRootPath;
38. string fileName = @"ExportCustomers.xlsx";
40. FileInfo file = **new** FileInfo(Path.Combine(rootFolder, fileName));
42. using (ExcelPackage **package** = **new** ExcelPackage(file))
43. {
45. IList<Customers> customerList = \_db.Customers.ToList();
47. ExcelWorksheet worksheet = **package**.Workbook.Worksheets.Add("Customer");
48. **int** totalRows = customerList.Count();
50. worksheet.Cells[1, 1].Value = "Customer ID";
51. worksheet.Cells[1, 2].Value = "Customer Name";
52. worksheet.Cells[1, 3].Value = "Customer Email";
53. worksheet.Cells[1, 4].Value = "customer Country";
54. **int** i = 0;
55. **for** (**int** row = 2; row <= totalRows + 1; row++)
56. {
57. worksheet.Cells[row, 1].Value = customerList[i].CustomerId;
58. worksheet.Cells[row, 2].Value = customerList[i].CustomerName;
59. worksheet.Cells[row, 3].Value = customerList[i].CustomerEmail;
60. worksheet.Cells[row, 4].Value = customerList[i].CustomerCountry;
61. i++;
62. }
64. **package**.Save();
66. }
68. **return** " Customer list has been exported successfully";
69. }

3. Ai chịu trách nhiệm về cấu trúc của project (trình bày cấu trúc của project - MVC, N-layer,...)

4. Vì sao lại phải xây dựng layer repository khi dbcontext đã làm được CRUD rồi.

Do repository được tách làm một tầng riêng, nó như là lớp trung gian giữa tầng Business Logic và Data Access, giúp cho việc truy xuất dữ liệu được chặt chẽ và bảo mật hơn. Nó đóng vai trò kết nối giữa tầng Business và Model của ứng dụng. Khi tách ra thì phần Repository Layer sẽ đảm nhận một vai trò duy nhất thay đổi quyền truy cập dữ liệu cũng như xử lys dữ liệu, một nơi chịu trách nhiệm duy nhất cho việc mapping cá bảng vào object. Giúp cho việc testing dễ dàng hơn. Nếu không tách thì khi thao tách tầng khác lại gắn chặt với tầng Model dẫn tới việc sau này khi ta chỉnh sửa lại thì lại phải sửa lại một loạt.

5. Phân biệt Overload và Override, để override thì cần điều kiện gì.

6. Nếu như ở lớp cha có 1 phương thức không xác định virtual hay abstract, lớp con cũng có 1 phương thức giống hệt như vậy thì nó sẽ được gọi thế nào. (hidden method)

7. Statíc trong C#. Thuộc tính static có dùng chung cho tất cả các đối tượng của lớp đó được không.

8. Giả sử có 1 lớp Học Sinh, em nghĩ thuộc tính nào trong lớp đó có thể để static, tại sao?

9. Nếu như muốn ngăn kế thừa tới 1 lớp thì dùng cái gì.

10. Nếu như ta có 1 dll của bên thứ ba đã ngăn kế thừa, làm thế nào để mình tự viết thêm hàm cho nó.

11. Các Access modifier trong C#.

12. Em có phân biệt được const và read only không?

13. Em đã từng dùng ViewModel trong project chưa? Nó khác Model như thế nào? Em dùng ViewModel trong trường hợp nào.

14. Trong MVC thì Controller đóng vai trò gì. Có những cách nào để truyền dữ liệu từ Controller lên View.

15. Vòng đời của 1 DbContext.

16. Khi mình viết các lệnh LINQ như Select, Where,... thì nó đã được thực thi chưa, nếu chưa thì thực thi khi nào?

17. Em có phân biệt được Ajax POST và GET không?

18. (Em đã từng làm Web API thì) em có phân biệt được GET, POST, PUT của Ajax với GET, POST, PUT của Web API không.

19. Em có biết về OAuth, RESTful không?

20. Em đã deploy web bao giờ chưa?

21. Lợi ích của View trong SQL. View có phải 1 bảng dữ liệu được lưu trữ trong database để mình truy vấn ra không?

- Lợi ích của View:

+ Giúp cho query trở nên đơn giản hơn khi thao tác.

+ An toàn hơn vì có thể phân quyền.

+ Sử dụng ít bộ nhớ

+ Có thể tái sử dụng