Lab	Practical
1	Database: **CoffeeShop** 1. Add Country, State, City Tables in it. **(A)** • **Country** (CountryID, CountryName) [*CountryID - Primary Key*] • **State** (StateID, CountryID, StateName, StateCode) [*StateID - Primary Key, CountryID – Foreign Key*] • **City** (CityID, StateID, CityName, PinCode) [*CityID - Primary Key, StateID – Foreign Key*] 2. Create Required Stored Procedures for City Table for CRUD. **(A)** 3. Do CRUD for City Table. **(A)** 4. Do CRUD of Country & State Table. **(B)**
2	**Fill Cascade Dropdown** 1. Do fill State Dropdown based on Country. **(A)** 2. Do fill City Dropdown based on States. **(A)** 3. Do fill Product Dropdown based on Orders. (Use Order Detail Table) **(B)**
3	**Counts | URL Encrypt & Decrypt** 1. Do add one more column in state list page as City Count. **(A)** 2. Do add one more column in Customer list page as Order Count. **(A)** 3. When Any user Click on City Count or Order Count column values, he/she will be redirect to their list page. **(B)** 4. Apply Encryption & Decryption in your Project (Edit Record) **(C)**
4	**Dashboard Design** 1. Design a dashboard of your project which having below functionalities. a. Quick Navigation to all Pages of Project **(A)** b. Counts for Total Customers, Total Products, Total Orders, Total Bills **(A)** c. Recent 10 Orders, Recent 10 Newly Added Products **(A)** d. Maximum Ordered Customer List (Top 10) **(B)** e. Maximum Selling Products List (Top 10) **(C)**
5	**Create | Web API – [Part – 1]** 1. SelectAll, SelectByPK, Delete (Use Table City) **(A)** 2. SelectAll, SelectByPK, Delete (Use Product Table) **(B)**
6	**Create | Web API – [Part – 2]** 1. Insert, Update, DDL (Use Table City) **(A)** 2. Insert, Update, DDL (Use Product Table) **(B)** 3. Check for Exist, Duplicate product cannot be entered (Use Product Table) **(C)**

7	**Consume | Web API** 1. Use the Product table & perform CRUD operations with the help of Consuming API **(A)** 2. Use the City table & perform CRUD operations with the help of Consuming API **(B)** 3. Use the User table & perform CRUD operations with the help of Consuming API **(C)**
8	**Fluent Validations** 1. Apply Fluent Validations on below mentioned tables. a. User Table **(A)** b. Product Table **(A)** c. Customer Table **(A)** d. Order Detail Table **(B)** e. Bills Table **(C)**
9	**jQuery** 1. Do apply all jQuery Selectors with <p>, <ul>, <li>, <href>, <a>, <tr> & <button> tags. **(A)** 2. Do apply all jQuery Form Events **(A)** 3. Do apply all Mouse & Key Board Events **(B)**
10	**Basic LINQ Operators** 1. Projection operators and Filtering operators **(A)** 2. Aggregate operators **(B)** 3. Sorting operators **(C)**
11	**Set up EF Core, Basic CRUD Operations with Data Annotations** 1. Create a Web API project and create model classes as follows and add appropriate relationships between the models. Use data annotations to define table names and columns and also add appropriate navigation properties. **(A)** a. Course- CourseId, CourseName b. Student - StudentId, Name, Enrollment, Semester c. StudentCourse - StudentCourseId, StudentId, CourseId, EnrollDate, Grade 2. Create ApplicationDbContext class and implement the DbContext class to represent the database context. Use migrations commands to apply the models to the database. **(B)** 3. Implement Create, Read, Update, and Delete (CRUD) operations using a Web API controller. **(C)**
12	**Repository Pattern | Web API** 1. Define a generic repository interface that outlines the basic operations. Implement a generic repository class that handles common database operations using EF Core. **(A)** 2. Implement specific repositories for entities like Student, Course, and StudentCourse for additional, entity-specific methods. (e.g., complex queries, filtering). **(B)** 3. Use Dependency Injection (DI) to inject repositories into the Web API controllers to decouple business logic from data access logic. And redesign controllers to interact with the repositories instead of directly interacting with EF Core DbContext. **(C)**

13	**Custom Tag Helpers, Partial Views** 1. Create all alerts as Custom tag helper for Success, Warning & Info. **(A)** 2. Build a tag helper that will render HTML that generates a link that allows the user to send an email to the owner of project. **(A)** 3. Create Header & Footer's Partial Views. **(A)** 4. Create Customer's Partial View which involves Customer Name, Address, Email, Mobile No, GST No, City. **(B)** 5. Create a Partial View for Your Projects Filters of all List Pages. **(C)**
14	**JWT, OAuth** 1. JWT Authentication **(A)** 2. Token-based Authentication (OAuth) **(B)**
15	**File Upload with Validation** 1. Add a file upload feature to the project, allowing users to upload profile pictures, with validation on file size and type. **(A)** 2. Implement file upload progress bar using jQuery as display the progress of file uploads in real-time. **(B)**