CO1: Basics of Web Forms and Master Page

1) What is a Web Application? How is it different from a Website?

- A **Web Application** is a dynamic app that runs on a web server.
- It requires user interaction and uses a database (e.g., online banking).
- A **Website** can be static or informational (e.g., news site), not always interactive.

2) Explain the purpose of a Registration Form.

- Used to collect user details like name, email, password, etc.
- Helps create user accounts.
- Stores data in the database for login and authentication.

3) What are Advanced Controls in ASP.NET?

- Controls that provide enhanced features like:
  - **GridView** (display records),
  - **DetailsView** (single record),
  - **Calendar** (date selection),
  - **FileUpload**, etc.

4) What is a Master Page?

- A template page that defines a common layout.
- Other pages (content pages) use this layout.
- Helps in maintaining a consistent look across pages.

5) How do you create and apply a Master Page?

- Create Site.master.
- Define common layout (header, footer).
- Add <asp:ContentPlaceHolder> tag.
- Add content pages and use <asp:Content> inside them.

6) What are the benefits of a Master Page?

- Reusability of layout.
- Easy updates (change in one place reflects everywhere).
- Better organization of large projects.

CO2: Connection, Data Controls, and Stored Procedures

7) What is a Connection-Oriented Architecture?

- Maintains continuous connection with the database.
- Example: Using SqlConnection to open and close a database connection manually.

8) What is a Disconnected Architecture?

- Data is fetched and stored in memory (like DataSet).
- Connection is closed after data is fetched.
- Useful in offline scenarios.

9) What are Data-Bound Controls?

- Controls that display data from a source.
- Examples:
  - GridView, DetailsView, Repeater, DropDownList.
- Used with DataSource objects.

10) How does GridView bind data?

- Fetch data using SqlDataAdapter.
- Fill DataTable.
- Bind it to GridView:
- GridView1.DataSource = dt;
- GridView1.DataBind();

11) What is a Stored Procedure?

- A saved SQL query in the database.
- Called from code using a command.
- Improves performance and security.

12) How do you call a Simple Stored Procedure?

- Use SqlCommand:
- SqlCommand cmd = new SqlCommand("GetUsers", con);
- cmd.CommandType = CommandType.StoredProcedure;

13) What is a Parameterized Stored Procedure?

- Stored procedure with input parameters.
- Example:
- CREATE PROCEDURE GetUserByID @ID INT

14) How to pass parameters in code?

cmd.Parameters.AddWithValue("@ID", 1);

15) What is LINQ?

- **Language Integrated Query** to query data using C# syntax.
- Works with arrays, lists, XML, SQL, etc.

16) Example of LINQ:

var result = from s in students where s.age > 18 select s;

CO3: Entity Framework, Sessions, and AJAX

17) What is Entity Framework?

- ORM (Object Relational Mapper) for working with databases using C# objects.
- Avoids writing SQL queries.

18) Code First vs. Database First:

- **Code First**: Create classes first, EF creates DB.
- **Database First**: DB exists first, EF creates classes.

19) What is Client-Side Session Management?

- Data stored on the browser (Cookies, Local Storage).
- Example: Response.Cookies["user"].Value = "admin";

20) What is Server-Side Session Management?

- Data stored on the server (Session object).
- Example: Session["username"] = "admin";

21) Advantages of Session:

- Stores user-specific data.
- Secure (server-side).
- Auto expires after time.

22) What is AJAX?

- Stands for Asynchronous JavaScript and XML.
- Updates part of a page without reloading the full page.

23) Examples of AJAX Controls:

- UpdatePanel

- ScriptManager
- Timer
- Used to improve user experience.

CO4: Web Services and MVC

24) What is a Web Service?

- A software service that allows apps to communicate over the web.
- Example: Weather API.

25) SOAP vs REST Web Services:

- **SOAP**: XML-based, heavier, more secure.
- **REST**: Lightweight, uses JSON, faster.

26) What is MVC?

- **Model View Controller** architecture.
- Separates logic (Model), UI (View), and input control (Controller).

27) Request Flow in MVC:

- User request → Controller → Model → View → Browser

28) Difference between MVC and Web Forms:

- MVC: Clean separation, testable.
- Web Forms: Event-driven, tightly coupled.

29) Role of Entity Framework in MVC:

- Manages database operations (CRUD).
- Maps DB tables to C# classes.

30) What is CRUD in MVC?

- C: Create
- R: Read
- U: Update
- D: Delete
- Implemented using EF and Razor Views.

Let me know if you want **code examples, diagrams, or simplified handouts** for these.