.Net Core BootCamp Program Outline

1. **Program Kickstart**
    a. Student Onboarding, introductions
    b. High-level overview of the Software Product being developed in the program
    c. Overview of Technology and Tools covered and used in the program
    d. Introduction to the methodologies for training and development in this program
    e. Illustration of the outcome of this program
2. **Project Initiation and Planning**
    a. Tools Sign-up
        I. Trello Introduction
        II. Github Intro
    b. Project scope of work & brainstorming
    c. Product backlog creation and estimation
    d. Project Milestones and Delivery Plan
3. **Software Engineering Fundamentals**
    a. Software Systems Overview
    b. Operating Systems & Networking concepts
    c. Software Engineering Roles & Responsibilities
    d. SDLC - Software Development Life Cycle
    e. Agile Methodologies
    f. Git Fundamentals
4. **Requirements Engineering**
    a. Agile Requirements
    b. Creating Use Case Diagrams for the System
    c. Creating User Stories for the System
    d. Creating SRS - Software Requirements Specification
    e. Product Backlog Grooming
    f. Creating a Release & Sprint Plan for the System
5. **UI/UX Designing**
    a. Intro to UI/UX Process & Figma
    b. Building low-fidelity wireframes using Figma
    c. Building high-fidelity mockups using Figma
6. **System Analysis and Design**
    a. System Analysis & Design Overview
    b. Domain Modeling: Create high-level Domain Model for the system
    c. System Architecture: Create high-level architecture of the system
7. **Overview of Front-End Development**
    a. Web development and the different types of web applications
    b. Basic concepts of web servers and web clients
    c. Understanding HTTP and HTTPS protocols
    d. Understanding the basic structure of the web, including HTML & CSS
    e. Front-end programming using JavaScript and JQuery
    f. Overview of popular web development frameworks and libraries
    g. Building the project web-front-end using HTML, CSS & JavaScript
    h. Building responsive web applications using Bootstrap

8. **Introduction to .Net Core**
    a. What is .NET Core?
    b. .NET Core architecture and components
    c. Installing and setting up the development environment
    d. Understanding .NET Standard and .NET Core
9. **C# Programming Language**
    a. Introduction to C# programming language
    b. Data types, variables, and control structures
    c. Methods and classes
    d. Object-oriented programming (OOP) concepts
    e. Inheritance and polymorphism
    f. Delegates, events, and lambda expressions
10. **Introduction to ASP.Net Core Web Development**
    a. Understanding the history and evolution of ASP.Net Core
    b. Overview of the ASP.Net Core Web Development architecture
    c. Setting up the development environment (Visual Studio, Visual Studio Code, .NET Core SDK)
    d. Creating a ASP.Net Core Web Development project for the project back-end services
    e. Choosing an ASP.Net Web UI approach
11. **Building Web UI Using Razor Pages**
    a. Introduction to Razor Pages and its features
    b. Understanding the Page Model
    c. Working with Razor syntax and creating dynamic content
    d. Razor Page Filters
    e. Page Routing, Discovery and processing
    f. Handling form submissions and working with validation
    g. Creating reusable Razor Page components
12. **Building Web Applications with ASP.Net Core MVC**
    a. Introduction to ASP.Net Core MVC and its features
    b. Understanding the MVC design pattern in ASP.Net Core
    c. Building presentation layer using views and partial views
    d. Creating and configuring controllers, views, and models in MVC
    e. Routing and URL management in ASP.Net Core MVC
    f. Working with forms and validation in MVC

13. **Creating Web APIs in ASP.NET Core**
    a. Introduction to Web APIs and their role in modern web development
    b. Overview of RESTful API design principles
    c. Creating a controller-based API
    d. Creating a minimal API
    e. Implementing CRUD operations in Web APIs
    f. Understanding input validation and error handling in Web APIs
    g. Versioning and documentation of APIs
    h. Consuming APIs using Swagger and Postman
14. **Blazor::Getting Started**
    a. Components

b. Hosting Models
c. Data binding
d. Dependency injection
e. Routing & navigation
f. Forms & Validation
g. Authentication & authorization
h. State Management
i. Integrating with back-end Service
j. Testing & Debugging
k. Performance Optimization

15. **Database Concepts**
    a. Introduction to databases and database concepts
    b. Relational database management systems
    c. SQL language fundamentals
    d. Data modeling and design
    e. Normalization and denormalization
    f. Indexes and query optimization
    g. Relational vs non-relational databases
16. **Persisting Data Using Entity Framework Core**
    a. Overview of data access in ASP.Net Core Web Development
    b. Introduction to Entity Framework Core
    c. Creating and configuring models and data contexts
    d. Databinding with Razor Pages
    e. Creating and querying databases with EF Core
    f. Entity Framework Core with ASP.NET Core MVC
    g. Querying and manipulating data using LINQ
    h. Understanding database migrations and seeding
17. **Middleware and Services**
    a. Introduction to Middleware and Services in ASP.NET Core
    b. Creating custom middleware and services
    c. Dependency Injection in ASP.NET Core
    d. Implementing logging and exception handling

18. **Authentication and Authorization**
    a. Understanding authentication and authorization concepts
    b. Implementing authentication using Identity Framework
    c. Authorization using policies, claims and roles
    d. Securing resources and protecting against attacks
    e. Implementing external authentication providers
19. **Test Driven Development**
    a. Understanding the importance of testing in web development
    b. Introduction to xUnit.net testing framework
    c. Writing unit tests for controllers and actions
    d. Using Moq to mock dependencies for unit testing

e. Writing integration tests for Web APIs and database access

f. Implementing test-driven development (TDD) practices in ASP.NET Core Web Development

g. Best practices for testing in ASP.NET Core Web Development

h. Configuring test environments and using test runners

i. Understanding performance and load testing in ASP.NET Core Web Development

j. Using tools like Postman for API testing and Selenium for UI testing.

20. Deployment

a. Introduction to cloud computing and the benefits of using Azure Cloud for deployment

b. Creating an Azure account and setting up Azure resources for deployment

c. Deploying ASP.NET Core applications into Azure using Azure App Service

d. Configuring custom domains and SSL certificates for secure deployment

e. Managing and monitoring applications in Azure using Azure Portal

f. Understanding continuous deployment using Azure DevOps and Github Actions

g. Best practices for deploying and scaling applications in Azure Cloud.