MERN

Introduction

This document outlines the additional topics to be incorporated into the current 7 months course. These topics are designed to enhance the learning experience and provide students with essential skills relevant to today's technological landscape.

Overview of Additional Topics

Total Duration: 11 Weeks

Detailed Breakdown of Each Topic

1. **OOAD - (1 Week)**
   a. Topics: SDLC, programming paradigms, OOP advantages, core concepts (Abstraction, Encapsulation, Polymorphism, Inheritance), design patterns (Creational, Structural, Behavioral), UML diagrams.
   b. Hands-on: Create UML diagrams, map them to Python code.
   c. Project: Analyze, design, and implement using UML and Python.
   d. Testing Basics: Overview of testing OOP.
2. **AI Integration & Testing - (1 Week)**
   a. AI Tools: GitHub Copilot, TabNine, CodeWhisperer for auto-completion and code generation.
   b. Testing: Introduction to TDD (Red, Green, Refactor), unit, integration, functional, and E2E testing.
   c. Backend Testing: Mocha, Chai, Sinon for Node.js.
   d. Frontend Testing: Jasmine, Karma, Cypress for Angular, Jest for React
   e. API Testing: Mocking backend in E2E tests.
3. **Microservices & Mini Project - (4 Weeks)**
   a. Topics: Microservices, communication styles, message brokers.
   b. Tools: Docker, Kubernetes for deployment.
   c. Micro Frontends: Using single-spa and module federation.
   d. Project: Dockerize and deploy a microservice.
4. **GraphQL - (2 Weeks)**
   a. Topics: Writing queries and mutations, setting up a GraphQL server, MongoDB integration, client-side basics with Angular or React.
   b. Features: CRUD operations, subscriptions, performance optimization.

5. **Next.js - (1 Week)**
    a. Topics: Page-based routing, static site generation, SSR, API routes, image optimizations, dynamic routing, error handling, deployment.
6. **System Design and DB Design - (2 Week)**
    a. Topics: High-level architecture, workflow diagrams, scalability, caching, security planning, CI/CD strategy.

Cost Structure

The following outlines the payment options available for the additional topics:

Payment Option	**Total Cost (Excluding GST)**
Upfront Payment (₹3000 × 11)	₹33,000
PAP/ISA Payment (₹4800 × 11)	₹52,800

Features Comparison

Compare the features of different course options:

Features	**MEAN and MERN**		
	7 Months Course	**1 Year Course**	**Addon: Advanced Topics**
Duration	29 Weeks	52 Weeks	11 Weeks
Repeat Count	12 Weeks	24 Weeks	3 Weeks
Number of Main Projects	2	3	0
Expert Mentorship	❌	✅	❌
Money Back Guarantee	❌	✅	❌
Fitness Activities	❌	✅	❌
Termination	✅	❌	✅
Group Project	❌	✅	❌

Python Full Stack

Introduction

This document outlines the additional topics to be incorporated into the current 7 months course. These topics are designed to enhance the learning experience and provide students with essential skills relevant to today's technological landscape.

Overview of Additional Topics

Total Duration: 11 Weeks

Detailed Breakdown of Each Topics

1. **Unix - (1 Week)**
    a. Unix Fundamentals
    b. Unix Commands
    c. Python Virtual Environment
    d. IDE Setup
2. **OOAD - (1 Week)**
    a. Topics: SDLC, programming paradigms, OOP advantages, core concepts (Abstraction, Encapsulation, Polymorphism, Inheritance), design patterns (Creational, Structural, Behavioral), UML diagrams.
    b. Hands-on: Create UML diagrams, map them to Python code.
    c. Project: Analyze, design, and implement using UML and Python.
    d. Testing Basics: Overview of testing OOP.
3. **Web Security - (1 Week)**
    a. Web Vulnerabilities
    b. Security Mechanisms
    c. Authentication & Authorization
    d. Input Sanitization
    e. Network Security
    f. Logging & Monitoring
4. **AI Integration & Testing - (1 Week)**
    a. AI Tools: GitHub Copilot, TabNine, CodeWhisperer for auto-completion and code generation.
    b. Testing: Introduction to TDD (Red, Green, Refactor), unit, integration, functional, and E2E testing.
    c. Backend Testing: Pytest.
    d. Frontend Testing: Jest

    e. API Testing: Mocking backend in E2E tests.

5. **Microservices & Mini Project - (3 Weeks)**
    a. Topics: Microservices, communication styles, message brokers.
    b. Tools: Docker, Kubernetes for deployment.
    c. CI/CD: Pipelines, automation, code quality, environment management, monitoring, rollback strategies.
    d. Learn One: Redis/Memcached, RabbitMQ/Kafka, Celery, Elasticsearch/Solr, WebSocket.
    e. Project: Build and deploy microservices with Docker and Kubernetes. Implement CI/CD pipeline.
6. **Next.js - (1 Week)**
    a. Topics: Page-based routing, static site generation, SSR, API routes, image optimizations, dynamic routing, error handling, deployment.

7. **FastAPI - (1 Week)**
    a. Topics: Async programming, path and query parameters, response models, dependency injection, ORM with SQLAlchemy, background tasks, data validation and serialization with Pydantic, middleware, and comparison with Django.

8. **System Design and DB Design- (2 Week)**
    a. Topics: High-level architecture, workflow diagrams, scalability, caching, security planning, CI/CD strategy.

Cost Structure

The following outlines the payment options available for the additional topics:

Payment Option	**Total Cost (Excluding GST)**
Upfront Payment (₹3000 × 11)	₹33,000
PAP/ISA Payment (₹4800 × 11)	₹52,800

Features Comparison

Compare the features of different course options:

Features	MEAN and MERN		
	7 Months Course	1 Year Course	Addon: Advanced Topics
Duration	29 Weeks	52 Weeks	11 Weeks
Repeat Count	12 Weeks	24 Weeks	3 Weeks
Number of Main Projects	2	3	0
Expert Mentorship	❌	✅	❌
Money Back Guarantee	❌	✅	❌
Fitness Activities	❌	✅	❌
Termination	✅	❌	✅
Group Project	❌	✅	❌

Flutter Mobile app Development

Introduction

This document outlines the additional topics to be incorporated into the current 7 months course. These topics are designed to enhance the learning experience and provide students with essential skills relevant to today's technological landscape.

Overview of Additional Topics

Total Duration: 9 Weeks

Detailed Breakdown of Each Topics

1. **Testing Fundamentals - (1 Week)**
    a. Intro to Unit Testing : Importance, Flutter testing framework, basic test structure.
    b. Writing Unit Tests : Test sync, async, void functions. Understand test coverage.
    c. Mocking with Mockito : Mock external services, simulate APIs/databases.
    d. Advanced Techniques : TDD, edge cases, error handling, complex methods.
2. **Widget and Integration Testing - (1 Week)**
    a. Widget Testing
        i. Writing Tests: Test individual widgets, user interactions, widget states.
    b. Integration Testing
        i. Writing Tests: Test app flows, multi-widget interactions.
        ii. Mocking: Use Mockito/Mocktail for mocking services.
3. **Test-Driven Development (TDD) - (1 Week)**
    a. Intro to TDD
        i. TDD Cycle: Red-Green-Refactor.
        ii. Test-First: Write tests before implementation.
    b. Implementing Features with TDD
        i. Develop Code: Make failing tests pass, refactor.
        ii. TDD in Flutter: Apply TDD to widgets, services, business logic.
    c. TDD Best Practices
        i. Effective Tests: Avoid over-specifying, maintain tests.
        ii. Testing Best Practices: Clean, modular tests, improve coverage.
4. **Advanced State Management and Performance Optimization**
    a. Riverpod & GetX: Setup, basic apps, comparison.
    b. Advanced Use Cases: Dependency injection, navigation.
    c. Performance: Reduce rebuilds, lazy loading, memory.

   d. Best Practices: Profiling, optimization.

5. **Behavior-Driven Development (BDD)**
   a. Understand BDD principles and workflow.
   b. Write human-readable specifications using Gherkin.
   c. Set up and use BDD frameworks in Flutter.
   d. Collaborate effectively with non-developers using BDD.
6. **App Architecture and Clean Code**
   a. Understand Clean Architecture and SOLID principles.
   b. Apply MVC and MVVM patterns in Flutter.
   c. Write clean, maintainable, and scalable code.
   d. Refactor code and write unit tests effectively.
7. **Dependency Injection and Provider for DI**
   a. Understand Dependency Injection (DI) and its importance.
   b. Implement DI using Provider and GetIt in Flutter.
   c. Apply best practices for organizing and testing DI.
   d. Refactor apps to use DI for loose coupling and testability.
8. **Firebase Advanced Features**
   a. Firebase Notifications : Push and local notifications, customization, deep linking.
   b. Firebase Analytics : Track user behavior, log events, analyze metrics.
   c. Firebase Crashlytics : Error reporting, custom error handling, crash analytics.
   d. Best Practices : Privacy, security, and improving app performance.
9. **App Deployment and CI/CD Pipelines**
   a. Introduction to CI/CD : Overview, GitHub Actions setup, basic pipeline.
   b. Automating Builds & Tests : Build commands, automated testing, dependency management.
   c. Advanced GitHub Actions : Multi-platform builds, matrix builds.
   d. Deploying Apps : Prepare for release, automate deployment, Firebase App Distribution.
   e. Best Practices : CI/CD best practices, branching, notifications, monitoring.

Cost Structure

The following outlines the payment options available for the additional topics:

Payment Option	**Total Cost (Excluding GST)**
Upfront Payment (₹3000 × 9)	₹27,000
PAP/ISA Payment (₹4800 × 9)	₹43,200

Features Comparison

Compare the features of different course options:

Features	MEAN and MERN		
	7 Months Course	**1 Year Course**	**Addon: Advanced Topics**
Duration	29 Weeks	52 Weeks	11 Weeks
Repeat Count	12 Weeks	24 Weeks	3 Weeks
Number of Main Projects	2	3	0
Expert Mentorship	❌	✅	❌
Money Back Guarantee	❌	✅	❌
Fitness Activities	❌	✅	❌
Termination	✅	❌	✅
Group Project	❌	✅	❌

Java Spring Boot Full Stack

Introduction

This document outlines the additional topics to be incorporated into the current 7 months course. These topics are designed to enhance the learning experience and provide students with essential skills relevant to today's technological landscape.

Overview of Additional Topics

Total Duration: 9 Weeks

Detailed Breakdown of Each Topics

1. **OOAD - (1 Week)**
   a. Topics: SDLC, programming paradigms, OOP advantages, core concepts (Abstraction, Encapsulation, Polymorphism, Inheritance), design patterns (Creational, Structural, Behavioral), UML diagrams.
   b. Hands-on: Create UML diagrams, map them to Python code.
   c. Project: Analyze, design, and implement using UML and Python.
   d. Testing Basics: Overview of testing OOP.

2. **AI Integration & Testing - (1 Week)**
   a. AI Tools: GitHub Copilot, TabNine, CodeWhisperer for auto-completion and code generation.
   b. Testing: Introduction to TDD (Red, Green, Refactor), unit, integration, functional, and E2E testing.
   c. Backend Testing: JUnit 5, AssertJ, Mockito, integration testing.
   d. Frontend Testing: Thymeleaf templates, React unit testing (if applicable).
   e. API Testing: Selenium setup, writing E2E tests, testing full application flows.

3. **Microservices and Mini Project - (5 Week)**
   a. Spring Boot Microservices : Microservices architecture, Eureka, Zuul, Spring Cloud Config, Resilience4j, Zipkin, Feign.
   b. Docker : Containers, networking, volumes, building and deploying microservices.
   c. Kubernetes : Container orchestration, deployments, services, volumes.

d. CI/CD Pipelines : Automation, code quality, deployment, monitoring, rollbacks.
e. Micro Frontends :Single SPA, Module Federation, integrating micro frontends with microservices.
f. Build and deploy Spring Boot microservices using Docker and Kubernetes.
g. Implement CI/CD pipelines for automated deployment and monitoring.
h. Understand and apply micro frontend architecture using Single SPA and Module Federation.
i. Gain hands-on experience with distributed systems, containerization, and orchestration

4. **Advanced Database Operations with Spring Boot + Messaging and Event-Driven Architecture - (1 Week)**
    a. Database Migrations:
        i. Liquibase/Flyway for schema management.
        ii. Custom queries with @Query.
        iii. Pagination and sorting in Spring Data JPA.
    b. Caching with Redis:
        i. Configuring Redis for caching.
        ii. Using caching annotations in Spring Boot.
    c. Messaging and Event-Driven Architecture:
        i. Introduction to RabbitMQ/Kafka.
        ii. Publishing and consuming messages in Spring Boot.

5. **System Design and DB Design - (2 Weeks)**
    a. Topics: High-level architecture, workflow diagrams, scalability, caching, security planning, CI/CD strategy.

Cost Structure

The following outlines the payment options available for the additional topics:

Payment Option	**Total Cost (Excluding GST)**
Upfront Payment (₹3000 × 10)	₹30,000
PAP/ISA Payment (₹4800 × 10)	₹48,000

Features Comparison

Compare the features of different course options:

Features	**MEAN and MERN**		
	7 Months Course	**1 Year Course**	**Addon: Advanced Topics**
Duration	29 Weeks	52 Weeks	11 Weeks
Repeat Count	12 Weeks	24 Weeks	3 Weeks
Number of Main Projects	2	3	0
Expert Mentorship	❌	✅	❌
Money Back Guarantee	❌	✅	❌
Fitness Activities	❌	✅	❌
Termination	✅	❌	✅
Group Project	❌	✅	❌

Game Development using Unity

Introduction

This document outlines the additional topics to be incorporated into the current 7 months course. These topics are designed to enhance the learning experience and provide students with essential skills relevant to today's technological landscape.

Overview of Additional Topics

Total Duration: 19 Weeks

Detailed Breakdown of Each Topic

1. **3D Game Development Essentials - (3 Weeks)**
    a. Unity 3D interface, 3D object manipulation, Blender basics, materials, textures, lighting, camera controls, player movement.
    b. Unity physics (Rigidbody, colliders), collision handling, triggers, physics-based player controls.
    c. Animator basics, importing characters, Mixamo animations, CharacterController, blend trees.
    d. Introduction to URP and HDRP, setting up projects, working with Render Pipeline Assets.
    e. URP/HDRP lighting, high-quality assets, PBR materials, VFX Graph, terrain sculpting.
    f. Racing game basics, car physics (Wheel Collider), track design, checkpoints, player controls.
    g. ame polishing (visuals, audio), leaderboards, optimization (Profiler, LOD, culling), exporting.
    h. Projects :
        i. Simple 3D Environment
        ii. 3D Ball Rolling Game
        iii. Animated 3D Character
        iv. Realistic 3D Environment
        v. Racing Game Prototype
2. **Game Cutscenes & Storytelling Techniques - (2 Week)**
    a. Timeline Basics
        i. Creating and managing Timeline assets
        ii. Understanding tracks (Animation, Activation, Audio)
    b. Animating with Timeline
        i. Attaching assets and adding keyframes
        ii. Playback controls and animation refinement
    c. Cinemachine Integration
        i. Setting up Cinemachine for smooth camera transitions
        ii. Using Dolly Track, Depth of Field, and Lens Effects

d. Audio and Effects
   i. Adding background music, dialogue, and effects
   ii. Syncing sound with animations and cutscenes
e. Cutscene Creation & Integration
   i. Planning and choreographing cinematic sequences
   ii. Triggering cutscenes within gameplay
   iii. Seamless transitions between cutscenes and gameplay
f. Projects:
   i. Basic Animation - Animate a GameObject using Timeline.
   ii. Cinemachine Camera Setup - Create smooth camera transitions.
   iii. Short Cutscene - Develop a cinematic scene with animation, audio, and camera work.
   iv. Advanced Cutscene - Create a complex cutscene with multiple tracks and scripted events.

3. **AR Foundation Basics - (3 Weeks)**

a. Introduction to AR Development with Unity
   i. Basics of AR, AR Foundation, and platform-specific considerations (iOS/Android).
b. Setting Up AR Foundation
   i. Installing AR Foundation, configuring settings, and setting up AR Session.
c. Basic AR Interactions
   i. Raycasting, touch inputs for object placement, and real-world surface detection.
d. Placing 3D Objects in AR
   i. Creating and positioning 3D objects in real-world spaces.
e. Simple AR App & Game
   i. Building an app with 3D object placement, and developing an AR game (e.g., food cooking).
f. Touch Gestures & Object Interactions
   i. Implementing drag-and-drop and touch gestures for AR manipulation.
g. Optimizing AR Apps
   i. Performance improvements, LOD, object pooling, and memory management.
h. Projects:
   i. Basic AR Object Placement
   ii. Simple AR Game Prototype
   iii. Advanced AR Game Development
   iv. Portfolio Compilation

4. **VR and Specialized Game Projects - (4 Weeks)**

a. Introduction to VR : Setup Unity, integrate Oculus, and explore XR Interaction Toolkit.
b. Setting Up VR Project : Create Unity project, configure XR Plug-in, set Android platform, and

add XR Rig.

c. Basic VR Controls : Enable head tracking, integrate Oculus controllers, and test object interaction.
d. Interactive VR Scene : Build a simple VR room, add colliders, and create interactive elements.
e. Deploying & Testing on Oculus: Set up Oculus Developer Tools and deploy the project to Oculus Quest.
f. Optimization for VR : Optimize lighting, reduce details, and ensure smooth performance.
g. Advanced VR Interactions : Implement grabbing, throwing, and teleportation in VR.
h. Simple VR Game : Design basic gameplay, implement grabbing/throwing, teleporting, and scoring.
i. Designing VR Experience : Define theme, create layout, and list interactive elements.
j. Environment & Interactions : Build environment, add colliders, and set up puzzles.
k. Sound & UI : Add sound effects, create World Space Canvas for UI elements.
l. Shader Effects : Set up URP/HDRP, create custom shaders, and use post-processing.
m. Final Optimization : Refine prototype, optimize performance using Unity Profiler.
n. Projects :
    i. Basic VR Scene: Create a simple VR scene with interactive objects.
    ii. VR Game Prototype: Develop a VR game with grabbing, throwing, and scoring.
    iii. Escape Room VR: Build a VR escape room with puzzles and interactive elements.
    iv. Shader & Post-Processing: Design custom shaders and apply post-processing effects.

5. **Open World Game Development - (1 Week)**

a. Defining Theme and Objective: Set the game's theme (e.g., exploration, survival) and objectives (e.g., resource collection, adventure).
b. Terrain Creation: Use Unity's Terrain Tool to sculpt large landscapes, apply textures (grass, sand), and add water bodies (rivers, lakes).
c. Foliage and Props: Populate terrain with trees, bushes, and environmental props. Add dynamic elements like swaying grass and trees using wind zones.
d. Lighting: Configure lighting settings, add directional light for sun, enable dynamic shadows, and implement a day/night cycle.
e. Core Gameplay Mechanics: Implement player movement, interaction with resources, and basic gameplay loops (e.g., resource collection to unlock new areas).
f. Optimization: Use occlusion culling, LOD Groups, async loading, and optimize textures/models to improve performance.
g. Building Prototype: Integrate terrain, foliage, lighting, and gameplay mechanics into a unified prototype.

6. **AI and NPCs in Games (1 Week)**

   a. Introduction to AI: Define AI's role for NPCs (e.g., patrolling, chasing, interacting) and outline needed behaviors (idle, patrol, aggressive).
   b. AI Navigation: Enable NavMesh and mark walkable areas, assigning NavMesh Agent for NPC movement.
   c. Basic AI Behaviors: Create a state machine for NPCs with idle, patrol, and chase states based on triggers (e.g., distance to player).
   d. Advanced AI Behaviors: Use FSM and Behavior Trees for decision-making, with additional actions like fleeing or attacking.
   e. NPC Interactivity: Implement player-NPC interactions (e.g., dialogue, quests) using custom scripts and animations.
   f. Debugging and Testing AI: Visualize detection ranges and paths, test edge cases, and fine-tune parameters.
   g. Integrating AI: Enhance the game by adding NPCs to dynamic areas and test their integration within the gameplay loop.
   h. Optimizing Performance: Use culling, coroutines, and efficient animation/pathfinding for smoother NPC performance.

7. **Interactive and Immersive Features (5 Weeks)**

   a. Game Systems:
      i. Add inventory, crafting, and quest systems with reusable design.
      ii. Implement inventory UI, crafting recipes, and quest tracking.
   b. Hyper-Casual Game:
      i. Simple mechanics (e.g., jumping, dodging), minimalistic style.
      ii. Focus on replayability and monetization through ads and in-app purchases.
   c. VR & AR Integration:
      i. Set up VR and AR modes using Unity's XR tools.
      ii. Implement seamless switching between VR and AR with interactive features.
   d. Cinematic Games:
      i. Develop a narrative-driven game with interactive gameplay using Unity's Timeline and Cinemachine for cutscenes.
   e. Multiplayer Setup:
      i. Use Mirror or Photon for networking, create matchmaking, and handle race logic.
      ii. Integrate leaderboards, social features, and optimize performance.
   f. Hybrid Platformer:
      i. Create a hybrid game with 2D and 3D elements.
      ii. Implement 2D platformer mechanics, and design transitions between 2D and 3D modes.

Cost Structure

The following outlines the payment options available for the additional topics:

Payment Option	Total Cost (Excluding GST)
Upfront Payment (₹3000 × 19)	₹57,000
PAP/ISA Payment (₹4800 × 19)	₹91,200

Features Comparison

Compare the features of different course options:

Features	MEAN and MERN		
	7 Months Course	1 Year Course	Addon: Advanced Topics
Duration	32 Weeks	52 Weeks	19 Weeks
Repeat Count	12 Weeks	24 Weeks	6 Weeks
Number of Main Projects	2	3	0
Expert Mentorship	❌	✅	❌
Money Back Guarantee	❌	✅	❌
Fitness Activities	❌	✅	❌
Termination	✅	❌	✅
Group Project	❌	✅	❌