Android Full Stack Development

System Requirements

- Min 8 GB Ram or More
- 2nd generation Intel Core or newer, or AMD CPU with support for a Windows Hypervisor

Android Topics

1. UI/UX Concepts
   1.1. Linear Layout
   1.2. Relative Layout
   1.3. Constraint Layout
   1.4. Material Design
   1.5. Drawables Vector Images
   1.6. XML Uses
2. API Integration
   2.1. Retrofit
3. DataBase
   3.1. Room DataBase
4. Permission
5. RecycleView
6. Google Map Integration
7. FireBase
8. Session Management
9. Android Architecture Components
   9.1. MVVM
   9.2. Navigation Component
10. Android Studio Emulator
11. Testing Application
    11.1. Unit Testing
12. Application Deployment to PlayStore
13. Git
14. Project Coverage on Individual Topics
    14.1. User Mangement Application

Core java Topics

1. Control statements: Java provides various control statements such as if-else, switch-case, while loop, for loop, etc. These statements help in controlling the flow of a program.

2. Data types: Java supports various data types such as primitive data types (int, double, boolean, etc.) and reference data types (class, array, interface, etc.). Understanding data types is essential for writing Java programs.

3. Classes and objects: Java is an object-oriented programming language, and classes and objects are at the heart of it. Understanding the concept of classes and objects is crucial for creating Java programs.

4. Inheritance: Inheritance is one of the key features of object-oriented programming, and Java supports it. Inheritance enables code reuse, and it helps in creating a hierarchical structure of classes.

5. Polymorphism: Polymorphism is another critical feature of object-oriented programming, and Java supports it. Polymorphism enables objects to take multiple forms, and it helps in creating flexible and extensible code.

6. Exception handling: Exception handling is an essential aspect of writing robust and error-free code. Java provides various mechanisms for handling exceptions, such as try-catch, throws, and finally blocks.

7. Input/output: Input/output operations are fundamental to any programming language, and Java provides various classes and methods for performing input/output operations.

8. Collections: Collections are used for storing and manipulating groups of objects. Java provides various collection classes, such as ArrayList, HashSet, LinkedList, etc.

9. Multithreading: Multithreading is the ability of a program to execute multiple threads simultaneously. Java supports multithreading, and it provides various classes and methods for creating and managing threads.

Kotlin Concepts

1. Variables and data types: Kotlin supports variables that can be declared using the var and val keywords. Variables can hold different data types such as Int, Double, Boolean, etc. Kotlin also provides nullable types to handle null values.

2. Control flow: Kotlin provides various control flow statements such as if-else, when, while, and for loop. The when statement is a powerful replacement for the switch statement in Java.

3. Functions: Functions in Kotlin are similar to functions in other programming languages. They can have parameters and return values. Kotlin also supports default parameter values, named arguments, and lambda expressions.

4. Classes and objects: Kotlin is an object-oriented programming language, and it provides support for classes and objects. In Kotlin, classes are declared using the class keyword, and objects are created using the object keyword.

5. Inheritance: Kotlin supports inheritance, and classes can inherit properties and methods from a parent class using the : symbol. Kotlin also supports overriding methods and properties of a parent class.

6. Interfaces: Interfaces are similar to Java interfaces, and they can define abstract methods and properties that must be implemented by a class. Kotlin also supports default methods and properties in interfaces.

7. Null safety: Kotlin provides null safety features to handle null values, which can cause NullPointerExceptions. Kotlin uses the ? symbol to denote nullable types and the !! operator for forcing a non-null value.

8. Extension functions: Extension functions allow adding new methods to existing classes without modifying the class itself. Kotlin extension functions can be used to provide additional functionality to third-party libraries.

9. Collections: Kotlin provides various collection classes such as List, Set, Map, and Array. Kotlin also supports functional programming concepts such as filter, map, and reduce.

10. Coroutines: Coroutines are a concurrency concept in Kotlin that allow writing asynchronous code in a more readable and concise manner. Coroutines can be used to perform long-running operations without blocking the main thread.