LEVEL 5

Mobile App Development

Student Guide

Modification History

Version	Date	Revision Description
V1.0	March 2024	For release

Published by: NCC Education Limited, Adamson House, Towers Business Park, Wilmslow Road, Didsbury, Manchester M20 2EZ, UK.

Tel: +44 (0) 161 438 6200 Fax: +44 (0) 161 438 6240 Email: info@nccedu.com
http://www.nccedu.com

CONTENTS

1. Unit Overview and Objectives

This unit will provide students with the key technological principles and methods for delivering and maintaining mobile applications. By the end of the unit, the students should be able to:

- Evaluate requirements for mobile platforms; establish appropriate strategies for development and deployment.
- Interpret a scenario, plan, design and develop a prototype hybrid and native mobile application.

2. Learning Outcomes and Assessment Criteria

Learning Outcomes; **The Learner will:**	**Assessment Criteria;** **The Learner can:**
1. Understand the scope and position of mobile apps within the context of software development	1.1 Explain the key characteristics of mobile apps. 1.2 Discuss the scope and limitations of mobile apps. 1.3 Discuss the issues of developing apps for multiple platforms
2. Demonstrate a critical understanding of the range of current development architectures, platforms, languages and tools	2.1 Explain the key characteristics of native apps 2.2 Explain the key characteristics of cross-platform development frameworks. 2.3 Summaries current trends in development tools 2.4 Discuss the relative merits of native and cross-platform approaches for a well-defined scenario.
3. Plan and design a mobile application with appropriate features	3.1 Explain the SDLC in relation to Mobile Apps. 3.2 Employ appropriate methods to specify and design a well-defined app with limited complexity.
4. Develop application functionality with an appropriate programming language and software development kit (SDK)	4.1 Install and use Android Studio or other appropriate IDE 4.2 Make use of common User Interface elements that can be used to construct the application interface on a mobile platform. 4.3 Write, test and debug native code for a well-defined app.
5. Employ Object Oriented techniques in development of a mobile App	5.1 Explain the key concepts of OO development and their benefits. 5.2 Make appropriate use of inheritance in design/ coding the app. 5.3 Discuss the relative merits of well-known software design patterns used in mobile app development

3. Syllabus

Syllabus			
Topic No	**Title**	**Proportion**	**Content**
1	*Introduction to Mobile App Development*	1/12 2 hours of lectures 2 hours of tutorials 2 hours of laboratory sessions	• What is a Mobile App? • Platforms: smart phone, tablet, wearable devices • Operating systems: E.g.: Android, iOS, Tizen. • Apps in the context of software systems • Challenges for app developers • Developing for multiple platforms • Web apps vs native apps • Quality, Security, • Distribution: Deployment, Market place, users • Trends ***Learning Outcome: 1***
2	*Introduction to Technologies for App Development*	1/12 2 hours of lectures 2 hours of tutorials 2 hours of laboratory sessions	• Technologies: native APIs and SDKs, • HTML5, hybrid Web apps, XML, Ajax, Json. • Hybrid technology: HTML5, CSS, advanced JavaScript, jQuery Mobile. Hybrid technology: HTML5, CSS, advanced JavaScript, jQuery Mobile. • Frameworks (E.g. Ionic, React Native, Framework7, Flutter, PhoneGap). • MVC Design Pattern • Progressive Web Apps ***Learning Outcome: 1,2***
3	App Development Processes	1/12 2 hours of lectures 2 hours of laboratory sessions	• SDLC applied to mobile apps • Selecting technology • Mobile HCI: Usability, IA, standards, UI, interface design, touch design patterns. • Quality Management: testing, debugging, maintenance, updates. • Application types: social, games, data-driven ***Learning Outcome: 2,3***
4	Using an IDE for Mobile App Development	1/12 2 hours of lectures 2 hours of laboratory sessions	• Purpose of IDE • Range of IDEs for Mobile Development • Android Studio overview ○ Downloading/ installing ○ Features ○ Supported platforms ○ Exploring the User Interface ○ Creating a simple App using built-in template ○ Emulator ○ Languages supported ***Learning Outcome: 4***

5	Object Orientated Programming for Mobile Apps (1)	1/12 2 hours of lectures 2 hours of laboratory sessions	• Review of OO concepts and techniques ○ Encapsulation, abstraction, inheritance and polymorphism. • Classes, Objects • Introduction to Kotlin ○ Terminology ○ Data types ○ Variables, values ***Learning Outcomes: 4, 5***
6	Object Orientated Programming for Mobile Apps (2)	1/12 2 hours of lectures 2 hours of laboratory sessions	• Introduction to Kotlin ○ Selection ○ Iteration ○ functions, parameters ○ Classes, inheritance ○ Constructors ○ properties ***Learning Outcome: 4, 5***
7	App Development-Getting Started	1/12 2 hours of lectures 3 hours of laboratory sessions	• Creating a basic App • Views • Adding Images and Text • Constraints ***Learning Outcomes: 4, 5***
8	App Development: Classes and Objects, Database	1/12 2 hours of lectures 3 hours of laboratory sessions	• Buttons and other views • Importing libraries • Introduction to MySQL ***Learning Outcomes: 4,5***
9	App Development: Database	1/12 2 hours of lectures 3 hours of laboratory sessions	• Connecting Kotlin with MySQL • Getting user input • CRUD Example ***Learning Outcomes: 4,5***

10	App Development: Location-Based	1/12 2 hours of lectures 3 hours of laboratory sessions	- Geolocation - Maps and Displayable lists - Camera ***Learning Outcomes: 4***
11	App Development: Navigation	1/12 2 hours of lectures 3 hours of laboratory sessions	- Navigation architecture - Navigating between screens - JETPACK - Search Engine Optimisation (SEO) - Architecture components ***Learning Outcomes: 4,5***
12	Deployment and Testing	1/12 2 hours of lectures 2 hours of tutorials	- Testing - Configuration control - Github - Deployment ***Learning Outcome 1***

4. Related National Occupational Standards

The UK National Occupational Standards describe the skills that professionals are expected to demonstrate in their jobs in order to carry them out effectively. They are developed by employers and this information can be helpful in explaining the practical skills that students have covered in this unit.

Related National Occupational Standards (NOS)
Sector Subject Area: ICT Practitioners **Related NOS:** TECHDUBI1, TECHDUBI3, TECHDUCW1, ESKITU050, ESKITU060, ESKITU061, ESKITU062, ESKITU063

5. Resources

Lecturer Guide: This guide contains notes for lecturers on the organisation of each topic, and suggested use of the resources. It also contains all of the suggested exercises and indicative answers.

PowerPoint Slides: These are presented for each topic for use in the lectures. They contain many examples which can be used to explain the key concepts. Handout versions of the slides are also available; it is recommended that these are distributed to students for revision purposes as it is important that students learn to take their own notes during lectures.

Student Guide: This contains the topic overviews and all of the suggested exercises. Each student will need access to this and should bring it to all of the taught hours for the unit.

5.1 Additional Hardware and Software Requirements

Hardware: A computer running a 64-bit version of Windows (8, 10, or 11), Linux, macOS (10.14 Mojave or later), or ChromeOS.

Internet access for your computer.

Software: Android Studio

6. Pedagogic Approach

Suggested Learning Hours						
Guided Learning Hours				Assessment	Private Study	Total
Lecture	Tutorial	Seminar	Laboratory			
24	6	-	27	40	103	200

The teacher-led time for this unit is comprised of lectures, laboratory sessions and tutorials. The breakdown of the hours is also given at the start of each topic, with 5 hours of contact time per topic.

6.1 Lectures

Lectures are designed to introduce students to each topic; PowerPoint slides are presented for use during these sessions. Students should also be encouraged to be active during this time and to discuss and/or practice the concepts covered. Lecturers should encourage active participation and field questions wherever possible.

6.2 Tutorials

Tutorials provide tasks to involve group work, investigation and independent learning for certain topics. The details of these tasks are provided in this guide and also in the Student Guide. They are also designed to deal with the questions arising from the lectures, laboratory sessions and private study sessions.

6.3 Laboratory Sessions

During these sessions, students are required to work through practical tutorials and various exercises. The details of these are provided in this guide and also in the Student Guide. Some sessions will require more support than others as well as IT resources. More detail is given in this guide.

6.4 Private Study

In addition to the taught portion of the unit, students should undertake private study. Exercises are provided in the Student Guide for students to complete during this time. Teachers will need to set deadlines for the completion of this work. These should ideally be before the tutorial session for each topic, when Private Study Exercises should be reviewed to ensure students are making suitable progress.

7. Assessment

This unit will be assessed by means of an assignment worth 100% of the total mark. This assessment will cover the learning outcomes and assessment criteria given above. Sample assessments are available through the NCC Education Virtual Learning Environment (http://vle.nccedu.com/login/index.php) for your reference.

8. Further Reading List

A selection of sources of further reading around the content of this unit must be available in your Accredited Partner Centre's library. The following list provides suggestions of some suitable sources:

- Forrester, A. et al. (2023) How to build android apps with kotlin : a practical guide to developing, testing, and publishing your first android apps. Second edn. Birmingham, UK: Packt Publishing.
- Griffiths, D. and Griffiths, D. (2019) Head first kotlin : a brain-friendly guide. Sebastopol, CA: O'Reilly (Head first).
- Saumont, P. Y. (2019) The joy of kotlin. Shelter Island, NY: Manning Publications.
- Kousen, K. A. (2020) Kotlin cookbook : a problem-focused approach. First edn. Sebastopol, CA: O'Reilly Media.

Topic 1: Introduction to Mobile App Development

1.1 Learning Objectives

This topic provides an overview of Mobile App Development. The detailed explanation on different type of mobile app and platform used for mobile app are introduced here. This topic covers mobile app operating systems and the challenges faced by mobile app developers. Mobile app quality, security, distribution, deployment, marketplace, users and trends are covered at the end of this topic.

On completion of the topic, students will be able to:

- Explain the key characterises of mobile apps, compare and contrast the different types of mobile apps.
- Distinguish the platforms use for mobile app and explain the different types of mobile app operating systems.
- Discuss the issues of developing apps for multiple platforms.

1.2 Pedagogic Approach

Information and theory of the topic will be presented to the students during lectures. They will then practise the skills during the laboratory sessions. Students are expected to undertake their own private study to understand the theory fully and put the lectures in context.

1.3 Timings

Lectures:	2 hours
Private Study:	8 hours
Tutorial:	2 hours
Laboratory:	2 hours

1.4 Lecture Notes

The following is an outline of the material to be covered during the lecture time and should be read in conjunction with the slides provided.

The structure of this topic is as follows:

- What is a Mobile App?
- Platforms: smart phone, tablet, wearable devices
- Operating systems: E.g.: Android, iOS, Tizen.
- Apps in the context of software systems
- Challenges for app developers
- Developing for multiple platforms
- Web apps vs native apps
- Quality, Security,
- Distribution: Deployment, Marketplace, Users

- Trends

1.5 Tutorial Sessions

The time allocation for this topic is 2 hours.

Exercise 1:

Explain why it is essential to understand Mobile Operating Systems when developing a Mobile App.

Exercise 2:

Research cross-platform app development frameworks such as Flutter, React Native, Kotlin Multiplatform Mobile, Ionic, Swift etc. Based on the features from each framework, suggest a framework for each scenario below and explain why.

Scenario 1: Simple Cross-Platform App with Basic Features

You want to develop a relatively simple app with a limited budget, and your priority is to reach both Android and iOS users with a single codebase.

Scenario 2: Complex Native-Like App for iOS

Your project is iOS-centric, and you need to provide a highly polished, native-like user experience with access to all iOS-specific features and design guidelines.

Scenario 3: Complex Native-Like App for Android

Your project is Android-centric, and you want to create an app that takes full advantage of Android features and Material Design guidelines.

Scenario 4: Cross-Platform App with High Performance and Graphics Requirements

You need a cross-platform app that offers high performance, complex animations, and 2D/3D graphics. The app should be available on both Android and iOS.

1.6 Laboratory Sessions

The time allocation for this topic is 2 hours.

The importance of Wireframing for Mobile Application Development was discussed in the Lecture.

Students will be asked Watch this wireframing tutorial using Figma (1 hour).
https://www.youtube.com/watch?v=BOt3MNB71gI

Then, students will be asked to research for the design of a Restaurant Food Ordering App. Students will be asked to create the wireframes for the Restaurant Food Ordering application pages.

Please talk to students individually to find out how they are progressing with wireframing, give them hints to create usable wireframes, encourage them to brainstorm about the mobile app requirements, encourage critical thinking.

1.7 Private Study

The time allocation for private study in this topic is expected to be 8 hours.

Exercise 1

Based on the Challenges for App Developers listed on slide 34, research and explain each challenge.

Exercise 2:

- Carry on wireframing for lab exercise application pages at home.
- Prepare to show and discuss your created wireframes with your tutor in the next lesson.
- Allocate time for students to present their wireframes in front of the class.

Topic 2: Introduction to Technologies for App Development

2.1 Learning Objectives

This topic introduces the technologies for app development which include native APIs and SDKs. Languages used for mobile app development such as HTML5, CSS 3 Media Queries, XML, AJAX, and JSON are covered in this topic.

On completion of the topic, students will be able to:

- Understand the scope and position of mobile apps within the context of software development.
- Demonstrate a critical understanding of the range of current development architectures, platforms, languages and tools.

2.2 Pedagogic Approach

Information and theory of the topic will be presented to the students during lectures. They will then practise the skills during the tutorial sessions. Students are expected to undertake their own private study to understand the theory fully and put the lectures in context.

2.3 Timings

Lectures:	2 hours
Private Study:	8 hours
Tutorials:	2 hours
Laboratory:	2 hours

2.4 Lecture Notes

The following is an outline of the material to be covered during the lecture time and should be read in conjunction with the slides provided.

The structure of this topic is as follows:

- Technologies: native APIs and SDKs,
- HTML5, hybrid Web apps, XML, Ajax, Json.
- Hybrid technology: HTML5, CSS, advanced JavaScript, jQuery Mobile.
- Frameworks (E.g. Ionic, React Native, Framework7, Flutter, PhoneGap).
- MVC Design Pattern
- Progressive Web Apps

2.5 Tutorial Sessions

The time allowance for tutorials in this topic is 2 hours.

Exercise 1:

CSS selectors are powerful and flexible, allowing user to precisely target and style elements in the HTML documents. Explain CSS selectors listed below and give an example for each.

- Element Selector
- Class Selector
- ID Selector
- Descendant Selector
- Child Selector
- Attribute Selector
- Pseudo-class Selector
- Pseudo-element Selector
- Grouping Selector
- Universal Selector

Exercise 2:

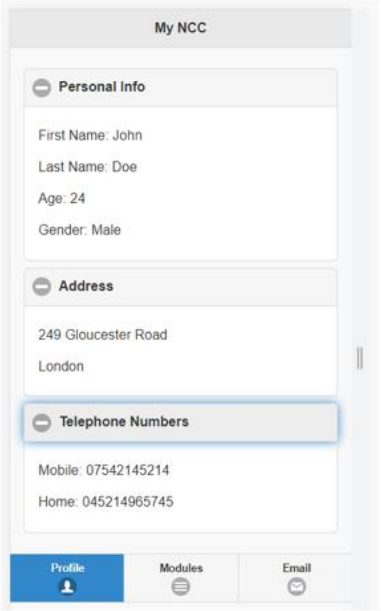
What are the common features and functions you can find in browser DevTools?

2.6 Laboratory Sessions

The time allocation for this topic is 2 hours.

Exercise 1:

Students will be asked to create a Mobile Application (consists of 3 pages). Students will attempt to create the first page and lecturers will help with the coding. This exercise is also part of their private study and students will complete all three pages as part of their homework.


My NCC

Personal Info

First Name: John

Last Name: Doe

Age: 24

Gender: Male

Address

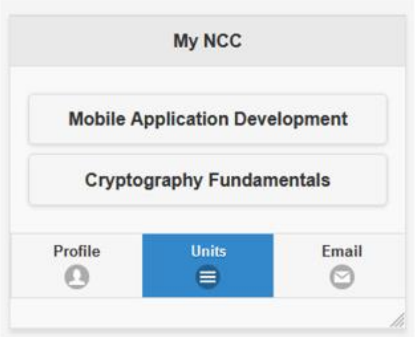
249 Gloucester Road

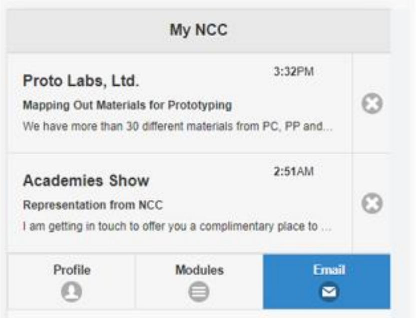
London

Telephone Numbers

Mobile: 07542145214

Home: 045214965745


My NCC


My NCC

Proto Labs, Ltd. 3:32PM

Mapping Out Materials for Prototyping

We have more than 30 different materials from PC, PP and...

Academies Show 2:51AM

Representation from NCC

I am getting in touch to offer you a complimentary place to ...

2.7 Private Study

The time allocation for private study in this topic is expected to be 8 hours.

Exercise 1

What are the advantages of Progress Web Apps (PWA)?

Topic 3: App Development Processes

3.1 Learning Objectives

This topic presents series of processes that guide the creation of a mobile application from conception to deployment and maintenance.

On completion of the topic, students will be able to:

- Demonstrate a critical understanding of the range of current development architectures, platforms, languages and tools.
- Plan and design a mobile application with appropriate features.

3.2 Pedagogic Approach

Information and theory of the topic will be presented to the students during lectures. They will then practise the skills during the tutorial sessions. Students are expected to undertake their own private study to understand the theory fully and put the lectures in context.

3.3 Timings

Lectures:	2 hours
Private Study:	8 hours
Laboratory:	2 hours

3.4 Lecture Notes

The following is an outline of the material to be covered during the lecture time and should be read in conjunction with the slides provided.

The structure of this topic is as follows:

- SDLC applied to mobile apps
- Selecting technology
- Mobile HCI: Usability, IA, standards, UI, interface design, touch design patterns.
- Quality Management: testing, debugging, maintenance, updates.
- Application types: social, games, data-driven

3.5 Laboratory Sessions

The time allocation for this topic is 2 hours.

Exercise 1:

Create a Persona for a Social Media App User

Objective: Develop a detailed persona for a user of a social media mobile application. This exercise will help you gain insights into the user's behaviours, interests, and motivations, which can guide the design and features of a social media app.

Instructions: Select a Social Media Use Case: Choose a specific social media use case for the exercise. For example, the persona could be someone using a platform for professional networking, photo sharing, or content creation.

Define Demographics: Specify key demographic details such as age, gender, occupation, location, and any other relevant demographic factors that align with your chosen use case.

Explore Background and Interests: Dig into the user's background, educational level, and interests. Consider their hobbies, passions, and any challenges they might face in their daily life.

Character Traits: Describe the user's personality traits. Are they extroverted, introverted, tech-savvy, or creative? Understanding these traits will influence the design and engagement strategies of the social media app.

Goals and Motivations: Identify the user's goals and motivations for using a social media app. Are they seeking professional connections, creative inspiration, or a sense of community?

Pain Points and Challenges: Pinpoint any challenges or pain points the user might face in their social media experience. This could include privacy concerns, information overload, or difficulty in finding relevant content.

Preferred Content Consumption: Consider how the user prefers to consume content. Are they more inclined toward visual content, written posts, or interactive features?

Social Behaviour: Explore the user's social behaviour. Do they prefer active participation, lurking, or a mix of both? How frequently do they post, comment, or share content?

Technology Usage: Understand the user's comfort and familiarity with technology. How often do they use mobile devices for social interactions? What other social media platforms are they active on?

Create a Persona:

Synthesize the gathered information to create a detailed persona for the user. Give your persona a name and a brief narrative that encapsulates their background, goals, challenges, and preferences.

Reflection:

Consider how understanding Sarah's persona could inform the design of a social media app tailored to her needs. What features would be most beneficial for her? How could the app address her specific challenges?

Note to Students: Feel free to be creative in crafting your persona. The more detailed and realistic, the better you can understand and address the needs of your potential social media app users.

Exercise 2:

Design Wireframes for Social Media App Features

Objective: Develop mobile wireframes for key features of a social media app. This exercise will help you visualize and plan the user interface for specific functionalities.

Instructions:

Select Social Media Features: Choose two to three specific features commonly found in social media apps. Examples include posting content, commenting, direct messaging, profile creation, or content discovery.

Define User Flow: Outline the user flow for each selected feature. Consider the steps a user would take to complete the action, from opening the app to accomplishing the task.

Identify Key Elements: Break down each feature into key elements such as buttons, text fields, images, and navigation bars. Identify the essential components needed to make the feature user-friendly.

Sketch Wireframes: Begin sketching wireframes for each feature. Use simple shapes and lines to represent the layout and placement of elements on a mobile screen. Focus on the basic structure and flow.

Consider User Interactions: Think about how users will interact with each feature. Include elements like buttons, sliders, or gestures that users might employ to navigate through the app.

Annotation: Add annotations to your wireframes to explain the functionality of each element. Describe what each button does, how users can navigate, and any other relevant details.

Keep it Simple: Prioritize simplicity in your wireframes. Avoid unnecessary details or visual elements that are not crucial to understanding the feature.

Iterate and Refine: Review your initial wireframes and iterate. Consider feedback from peers or make adjustments based on your own analysis. Refine the wireframes to enhance clarity and usability.

Example Features:

Feature: Posting Content

User Flow:

- User opens the app.
- User navigates to the posting section.
- User selects media (photo or video) or enters text.
- User adds optional captions or tags.
- User clicks on "Post" to publish the content.

Key Elements: Media upload button, text field, caption field, post button.

Feature: Commenting on a Post

User Flow:

- User scrolls through the feed.
- User selects a post to view.
- User enters a comment in the comment field.
- User clicks on "Submit" to post the comment.

Key Elements: Comment field, submit button.

Feature: Direct Messaging

User Flow:

- User navigates to the messaging section.
- User selects a contact or starts a new conversation.
- User enters a message in the chat box.
- User clicks on "Send" to deliver the message.

Key Elements: Contact list, chat box, send button.

Submission: Submit your wireframes along with annotations and a brief description of the user flow for each feature.

Note to Students:

This exercise aims to develop your wireframing skills and enhance your understanding of the user interface design process. Focus on clarity, simplicity, and effective communication of functionality in your wireframes.

3.6 Private Study

The time allocation for private study in this topic is expected to be 8 hours.

Exercise 1:

Objective: Gain a comprehensive understanding of Mobile Application HCI principles and apply them through practical exercises.

Private Study Instructions:

Welcome to your independent study session on Mobile App Human-Computer Interaction (HCI)! Below are the topics you'll explore on your own, along with some additional information to guide your learning:

1. Introduction to Mobile App HCI:

- Begin by gaining an overview of HCI principles specifically related to mobile applications.
- Reflect on the importance of user-centered design and usability in mobile app development.
- Take time to explore fundamental HCI concepts independently.

2. Mobile App Design Guidelines:

- Dive into resources discussing design guidelines tailored for mobile applications.
- Explore platform-specific guidelines for both iOS and Android platforms from reliable sources.
- Seek out practical examples of well-designed mobile interfaces for inspiration.

3. Prototyping and Wireframing:

- Explore various prototyping tools like Figma and Sketch through online tutorials and documentation.
- Practice creating a low-fidelity mobile app wireframe independently.
- Seek feedback from online communities or peers to improve your wireframing skills.

4. Usability Testing:

- Research the importance of usability testing in ensuring optimal user experiences.
- Consider conducting your own usability test on a wireframe or existing mobile app.
- Analyze and interpret the results to gain insights into usability issues and potential improvements.

5. Accessibility in Mobile App Design:

- Learn about the importance of designing inclusive interfaces for all users through online resources.

- Seek out practical tips and best practices for incorporating accessibility principles into mobile app design.

- Explore ways to evaluate and enhance accessibility in a sample app independently.

Additional Resources:

- Utilize websites and blogs like Nielsen Norman Group and Smashing Magazine for articles and case studies on usability, user experience, and mobile design.

- Refer to Google Design Guidelines (Material Design) for valuable resources and guidelines specific to designing Android applications.

Topic 4: Using and IDE for Mobile App Development

4.1 Learning Objectives

On completion of the topic, students will be able to:

- Develop application functionality with an appropriate programming language and software development kit (SDK)

4.2 Pedagogic Approach

Information and theory of the topic will be presented to the students during lectures. They will then practise the skills during the tutorial sessions. Students are expected to undertake their own private study to understand the theory fully and put the lectures in context.

4.3 Timings

Lectures:	2 hours
Private Study:	8 hours
Laboratory:	2 hours

4.4 Lecture Notes

The following is an outline of the material to be covered during the lecture time and should be read in conjunction with the slides provided.

The structure of this topic is as follows:

- Purpose of IDE
- Range of IDEs for Mobile Development
- Android Studio overview:
  - Downloading/ installing
  - Features
  - Supported platforms
  - Exploring the User Interface
  - Creating a simple App using built-in template
  - Emulator
  - Languages supported

4.5 Laboratory Sessions

The time allocation for this topic is 2 hours.

Exercise 1: Create your first Android Application and run in an Emulator.

Exercise 2: Create your another Android Application using a different template and run in an Emulator

4.6 Private Study

The time allocation for private study in this topic is expected to be 8 hours.

Exercise 1:

Research and learn event handling for Android mobile app development. This involves a structured approach to cover the basics and practical implementation.

- Introduction to Event Handling
  - Overview of event handling in Android.
  - Understanding UI events: onClick, onTouch, etc.
  - Basic concepts of View and ViewGroup.
  - Android's event propagation mechanism.
- Handling Button Click Events
  - Creating buttons in XML layout.
  - Setting onClick attributes in XML.
  - Implementing onClickListener programmatically.
  - Handling button clicks using Kotlin or Java.
- Touch Events and Gesture Detection
  - Understanding touch events: onTouchEvent.
  - Implementing gesture detection for swipe, pinch, and zoom.
  - Utilizing MotionEvent for touch event details.
  - Practical exercises with touch events.

Tips:

- Refer to official Android documentation for detailed information.
- Explore online tutorials and sample projects for hands-on practice.
- Join Android development communities for discussions and problem-solving.

Topic 5: Object-Oriented Programming for Mobile Apps (1)

5.1 Learning Objectives

The topic aims to provide a solid foundation in object-oriented programming (OOP) principles and the basics of the Kotlin programming language. Learners should be able to apply these concepts to design and implement software solutions using Kotlin, taking advantage of its concise syntax and modern features.

On completion of the topic, students will be able to:

- Gain a strong understanding of object-oriented programming principles
- Be proficient in designing and implementing classes and objects
- Have a solid foundation in using Kotlin for practical programming tasks
- Have essential skills for building robust and maintainable software solutions

5.2 Pedagogic Approach

Information and theory of the topic will be presented to the students during lectures. They will then practise the skills during the tutorial sessions. Students are expected to undertake their own private study to understand the theory fully and put the lectures in context.

5.3 Timings

Lectures:	2 hours
Private Study:	9 hours
Laboratory:	2 hours

5.4 Lecture Notes

The following is an outline of the material to be covered during the lecture time and should be read in conjunction with the slides provided.

The structure of this topic is as follows:

- Introduction to Kotlin
    - Terminology
    - Data types
    - Variables, values
- Classes, Objects
- Review of OO concepts and techniques
    - Encapsulation, abstraction, inheritance and polymorphism

5.5 Laboratory Sessions

The time allocation for this topic is 2 hours.

Exercise 1:

In this lab session, we will be crafting an Android mobile application using Kotlin, comprising one and a half activities (pages).

Throughout the lab, we'll cover the process of incorporating images into a mobile app page and demonstrate the technique of launching one activity (page) from another. Additionally, we will explore designing the user interface using the Activity Design mode, offering insights into creating dynamic pages for a more interactive user experience.

5.6 Private Study

The time allocation for private study in this topic is expected to be 9 hours.

Exercise 1:

Private Study Activity Plan: Create an Android Image Gallery App Using Kotlin.

Objective:

Design and implement an Android application using Kotlin that functions as an image gallery.

Activity Breakdown:

1. Understanding Requirements:

- Define the features and functionalities of the image gallery app.
- Identify user requirements and preferences.

2. Research and Planning:

- Explore existing image gallery apps for inspiration.
- Plan the app's architecture, including navigation and data handling.

3. Setting Up the Project:

- Create a new Android Studio project with Kotlin.
- Set up the project structure and necessary dependencies.

4. Implementing UI Components:

- Design the main layout for the image gallery.
- Implement UI components such as RecyclerView for displaying images.

5. Loading and Displaying Images:

- Write code to load images from the device's storage or an online source.
- Implement functionality to display images in the gallery.

6. Adding Interaction:

- Implement features such as image click events for a detailed view.

7. Testing and Debugging:

- Test the app on different devices and screen sizes.
- Debug and address any issues that arise.

8. Documentation and Reflection:

- Document the code for future reference.
- Reflect on the development process, noting any challenges and lessons learned.

Additional Tips:

- Take short breaks to maintain focus.
- Utilise online resources and documentation for problem-solving.
- Experiment with Kotlin's features to enhance code efficiency.
- Stay organized by keeping code, resources, and documentation well-structured.

This 9-hour study activity is designed to guide you through the process of creating a functional Android image gallery app using Kotlin. Adjust the time allocation based on your progress and familiarity with the tools and concepts.

Topic 6: Object-Oriented Programming for Mobile Apps (2)

6.1 Learning Objectives

This topic explores different types of loops in Kotlin (for, while, do-while). Students will learn how to use loops for repetitive tasks and iterations and gain OOP concepts like classes and inheritance. At the end of the topic, students will understand the principles of encapsulation, abstraction, and polymorphism.

On completion of the topic, students will be able to:

- Understand and apply conditional statements (if, else if, else) in Kotlin.
- Develop the ability to make decisions in programs based on different conditions.
- Master the use of loops in Kotlin, including for, while, and do-while loops.
- Apply loops effectively for repetitive tasks and iterations in programs.
- Comprehend the concept of functions in Kotlin, including their declaration, definition, and usage.
- Grasp the fundamentals of Object-Oriented Programming (OOP) with a focus on classes and inheritance.

6.2 Pedagogic Approach

Information and theory of the topic will be presented to the students during lectures. They will then practise the skills during the tutorial sessions. Students are expected to undertake their own private study to understand the theory fully and put the lectures in context.

6.3 Timings

Lectures:	2 hours
Private Study:	9 hours
Laboratory:	2 hours

6.4 Lecture Notes

The following is an outline of the material to be covered during the lecture time. Please also refer to the slides.

The structure of this topic is as follows:

Kotlin Programming with:

- Selection
- Iteration
- Functions and parameters
- Classes, inheritance
- Constructors
- Properties

6.5 Laboratory Sessions

The time allowance for Laboratory in this topic is 2 hours.

Exercise 1:

In this lab session, we will be orchestrating the development of an Android mobile application utilising Kotlin, featuring a trio of activities (pages). The outcomes of these activities will be visually presented in a PowerPoint slide.

Throughout this lab, we'll delve into incorporating dynamic elements into our mobile app pages. Specifically, we will explore the dynamic addition of images and checkboxes, demonstrating the use of if statements, loops, and ArrayLists to enhance the app's functionality.

Furthermore, this labwork will guide you through the process of creating dynamic pages, showcasing a versatile approach to designing interactive user interfaces.

6.6 Private Study

The time allocation for private study in this topic is expected to be 9 hours.

Exercise 1: **Creating a Todo App Without a Database Using Kotlin.**

Objective: Design and implement a simple Todo application for Android using Kotlin, focusing on functionality without incorporating a database.

Duration: 9 hours

Activity Breakdown:

1. Understanding Requirements:

- Define the key features and functionalities of the Todo app.
- Identify user requirements for task management.

2. Research and Planning:

Explore different approaches for managing tasks without a database. This implies that the application will not retain data, and upon restarting, no previous data will be available. All data will be temporarily stored in memory during the application's runtime.

Plan the app's user interface, and task management logic.

3. Setting Up the Project:

- Create a new Android Studio project with Kotlin.
- Set up the necessary project structure, layouts, and dependencies.

4. Implementing Task List UI:

- Design the main layout for the Todo app, including the task list view.
- Implement UI components such as RecyclerView to display tasks.

5. Adding Task Creation:

- Implement functionality to add new tasks to the task list.
- Allow users to input task details.

6. Implementing Task Deletion:

- Add functionality to delete tasks from the list.
- Implement a responsive user interface for task deletion.

7. Task Status Management:

- Allow users to mark tasks as completed or pending.
- Implement a visual cue for task status.

8. Testing and Documentation:

- Test the app on different devices, ensuring smooth functionality.
- Document the code and any relevant information for future reference.

Additional Tips:

- Take short breaks to maintain focus.
- Use Kotlin's features effectively to write clean and concise code.
- Test the app thoroughly to catch and address any potential issues.
- Consider user experience in the design to make the app intuitive.

This 9-hour study activity is designed to guide you through the process of creating a functional Android Todo app without the use of a database. Adjust the time allocation based on your familiarity with the tools and concepts.

Topic 7: App Development – Getting Started!

7.1 Learning Objectives

This topic introduces

On completion of the topic, students will be able to:

- create a basic mobile application using Kotlin, including setting up the project structure, defining the main activity, and incorporating necessary resources.
- Students will learn how to work with different types of views such as buttons, text fields, and labels in their Kotlin app, enabling them to design interactive user interfaces.
- Students will gain proficiency in adding images and text to their Kotlin app interface, allowing them to enhance the visual appeal and provide informative content within the application.
- Students will acquire the skills to utilize constraint layout to design responsive and adaptive user interfaces, ensuring consistent appearance across various screen sizes and orientations.

7.2 Pedagogic Approach

Information will be transmitted to the students during the lectures. They will then practise the skills during the laboratory sessions and extend their understanding during private study time. The tutorial will then provide an opportunity to review the key ideas and obtain further guidance and support.

7.3 Timings

Lectures:	2 hours
Laboratory Sessions:	3 hours
Private Study:	9 hours

7.4 Lecture Notes

The following is an outline of the material to be covered during the lecture time. Please also refer to the slides.

The structure of this topic is as follows:

- Creating a basic App
- Views
- Adding Images and Text
- Constraints

7.5 Laboratory Sessions

The laboratory time allocation for this topic is 3 hours.

Exercise 1:

Create the Android Kotlin app for creating a user profile with a profile picture selection and name entry.

7.6 Private Study

The time allocation for private study in this topic is expected to be 9 hours.

Exercise 1:

Local User Profile Storage and Retrieval

Prerequisites:

Basic knowledge of Android development.

Android Studio installed on your computer.

Activity Outline:

Part 1: Setting Up the Project

Create a New Project: Open Android Studio and create a new Android project.

Choose an Empty Activity template.

Design the User Interface: Design a layout with an EditText for the user's name, an ImageView for the profile picture, and buttons for saving and retrieving data.

Part 2: Save User Profile Data Locally

Implement Data Model: Create a data model class (e.g., UserProfile) to represent user profile data with fields for name and image URI.

Implement Saving Functionality: Implement code to save user profile data locally using SharedPreferences or Room Database.

Allow users to input their name, select a profile picture from the gallery, and save the data.

Test Saving Functionality: Run the app on an emulator or device to test the saving functionality.

Verify that the user profile data is saved locally.

Enhance UI: Add visual indicators or messages to inform the user when data is successfully saved.

Part 3: Retrieve and Display User Profile Data (3 hours)

Implement Retrieval Functionality: Implement code to retrieve user profile data from local storage.

Display the retrieved data in the UI.

Test Retrieval Functionality: Run the app and ensure that the saved user profile data is retrieved and displayed correctly.

Handle cases where no data is available or if the data format is incorrect.

Enhance UI: Improve the UI to provide a better user experience when displaying retrieved data.

Part 4: Finalise and Refine

Code Review and Refactoring: Review your code for any improvements or optimizations.

Refactor the code if necessary to follow best practices.

Documentation: Add comments to explain complex sections of your code.

Document any dependencies or external libraries used.

Final Testing: Perform thorough testing to ensure the app functions as expected.

Address any bugs or issues discovered during testing.

Additional Challenges (Optional):

Implement Image Compression: Improve the app by implementing image compression techniques to reduce the size of stored images.

Explore Other Local Storage Options: Experiment with different local storage options such as SQLite databases or external storage.

Implement Dark Mode: Enhance the app's UI by implementing a dark mode feature.

This activity provides a comprehensive hands-on experience in Android development, covering data storage, retrieval, and user interface design. Participants will gain practical experience in working with local storage, user interface elements, and data model classes in an Android app using Kotlin.

Topic 8: App Development: Classes and Object, Databases

8.1 Learning Objectives

This topic provides an overview of the importance of data persistence in mobile apps and the role of databases.

On completion of the topic, students will be able to:

- Explore the Room Persistence Library as a higher-level abstraction over SQLite, providing enhanced features for database interactions.
- Develop the ability to define and manage database entities, relationships, and migrations using Room.

8.2 Pedagogic Approach

Information will be transmitted to the students during the lecture. They will then practise the skills during the laboratory sessions and extend their understanding during private study time. The tutorial will then provide an opportunity to review the key ideas and obtain further guidance and support.

8.3 Timings

Lecture:	2 hours
Laboratory Sessions:	3 hours
Private Study:	9 hours

8.4 Lecture Notes

The following is an outline of the material to be covered during the lecture time. Please also refer to the slides.

The structure of this topic is as follows:

- Buttons and other views
- Importing libraries
- Introduction to MySQL

8.5 Laboratory Sessions

The laboratory time allocation for this topic is 3 hours.

Exercise 1:

Create the Android Kotlin app for displaying NCC student list using RecylerView.

8.6 Private Study

The time allocation for private study in this topic is expected to be 9 hours.

Exercise 1: Comprehensive Exploration of Entity Relationship Diagrams (ERD): Research Theoretical Foundations and Hands-On Application.

1. Introduction to ERD
Theoretical Understanding:

- Definition and purpose of ERD.
- Basic components: entities, attributes, relationships.
- Importance in database design.

Practical Activity:

- Draw a simple ERD on paper or using a digital tool for a familiar scenario (e.g., library management system).

2. Entity and Attribute Identification
Theoretical Understanding:

- Guidelines for identifying entities and attributes.
- Different types of attributes (simple, composite, derived).
- Keys: primary and foreign keys.

Practical Activity:

- Identify entities and attributes for a real-world scenario (e.g., online shopping system).

3. Relationship Modelling
Theoretical Understanding:

- Types of relationships: one-to-one, one-to-many, many-to-many.
- Cardinality and participation constraints.
- Associative entities.

Practical Activity:

- Model relationships for a university database (e.g., students, courses, professors).

4. Advanced Concepts

- Theoretical Understanding:
- Subtypes and supertypes.
- Aggregation.
- Generalisation and specialisation.

Practical Activity:

- Enhance the previous university database with advanced concepts.

5. Tools and Notations
Theoretical Understanding:

- Overview of ERD tools (e.g., Lucidchart, draw.io).
- Crow's Foot, Chen, or UML notations.

Practical Activity:

- Choose a tool and create an ERD for a business process.

6. Case Study and Problem Solving (1 hour)

Theoretical Understanding:

- Case study analysis.
- Problem-solving strategies in ERD.

Practical Activity:

- Solve a complex problem scenario using ERD.

7. Best Practices and Optimisation (1 hour)

Theoretical Understanding:

- Normalization and denormalization.
- Best practices in ERD design.

Practical Activity:

- Optimise an existing ERD for better performance.

Topic 9: App Development: Database

9.1 Learning Objectives

This topic provides an overview

On completion of the topic, students will be able to:

- Understand the fundamentals of database connectivity.
- Learn how to establish a connection between Kotlin (programming language) and MySQL (relational database management system).

9.2 Pedagogic Approach

Information will be transmitted to the students during the lectures. They will then practise the skills during the laboratory sessions and extend their understanding during private study time. The tutorial will then provide an opportunity to review the key ideas and obtain further guidance and support.

9.3 Timings

Lectures:	2 hours
Laboratory Sessions:	3 hours
Private Study:	9 hours

9.4 Lecture Notes

The following is an outline of the material to be covered during the lecture time. Please also refer to the slides.

The structure of this topic is as follows:

- Connecting Kotlin with MySQL
- Getting user input
- CRUD Example

9.5 Laboratory Sessions

The time allocation for this topic is 3 hours.

The purpose of the lab work "Implementation of Android Kotlin app for NCC user login and registration" is to guide students through the process of creating an Android app using the Kotlin programming language. The specific focus is on implementing features for user login and registration within the context of an NCC application.

The tasks outlined in the lab PowerPoint slide include steps for creating UI components (such as text fields and buttons) for user input, designing the user interface for registration and login screens, and implementing the underlying logic to handle user registration and login functionalities. Additionally, the students are instructed to integrate networking libraries (e.g., Volley) to communicate with a PHP Services for user authentication.

Exercise 1: User Registration

Exercise 2: User Login

9.6 Private Study

The time allocation for private study in this topic is expected to be 9 hours.

Review and Enhance User Registration

Review Previous Implementation:

- Go through the existing code for user registration and login.
- Understand the database schema and PHP scripts.

Enhance User Registration Form:

- Add more advanced form fields (e.g., date of birth, profile picture).
- Implement client-side validation.

Advanced User Profile Management

Implement User Profile Image Upload:

- Allow users to upload a profile picture.
- Handle image upload on the server using PHP.

Enhance User Profile Display:

- Implement a detailed user profile display with all user information.
- Consider implementing image caching for better performance.

Implementing Additional Features

Implement Password Recovery:

- Allow users to reset their passwords via email.
- Implement the necessary PHP scripts.

Security Considerations:

- Focus on security aspects, including input validation, secure password handling, and protecting against common security threats.

Topic 10: App Development Location-Based

10.1 Learning Objectives

This topic provides an overview

On completion of the topic, students will be able to:

- Explain the fundamentals of Android's location services and how they provide access to device location.
- Integrate the Google Maps SDK into an Android application.
- Integrate the device's camera to capture images or videos within the application.
- Implement lists to display location-based information, such as nearby places or user-generated content

10.2 Pedagogic Approach

Information will be transmitted to the students during the lectures. They will then practise the skills during the laboratory sessions and extend their understanding during private study time. The tutorial will then provide an opportunity to review the key ideas and obtain further guidance and support.

10.3 Timings

Lectures:	2 hours
Laboratory Sessions:	3 hours
Private Study:	9 hours

10.4 Lecture Notes

The following is an outline of the material to be covered during the lecture time. Please also refer to the slides.

The structure of this topic is as follows:

- Displayable lists
- Geolocation
- Camera

10.5 Laboratory Sessions

The laboratory time allocation for this topic is 3 hours.

The purpose of the lab work "Implementing Google Maps in an Android Kotlin app" is to guide students through the process of integrating an Android app with Google Map using the Kotlin programming language. The tasks outlined in the lab PowerPoint slide.

Exercise 1: Implementing Google Maps in an Android Kotlin app

10.6 Private Study

The time allocation for private study in this topic is expected to be 9 hours.

Private Study: Extending Google Maps with Markers and RecyclerView Integration.

Objective:

Extend the existing Google Maps integration in your Android Kotlin app to include markers at specific locations and integrate a RecyclerView to display additional information.

Prerequisites:

- Basic understanding of Android app development with Kotlin.
- Completed the initial Google Maps integration in your app.

Study Plan:

1. Overview of Google Maps Markers:

- Understand the concept of markers in Google Maps.
- Explore the various customization options for markers.

2. Adding Markers Programmatically:

- Learn how to add markers to specific locations on the map.
- Customize markers with different icons and titles.

3. Handling Marker Clicks:

- Implement click listeners for markers.
- Respond to marker clicks with actions like displaying information or navigating to another screen.

4. Introduction to RecyclerView:

- Understand the purpose and benefits of RecyclerView.
- Learn how RecyclerView efficiently handles large datasets.

5. Creating RecyclerView Layout:

- Set up a layout for the RecyclerView item that will display additional information about the marked locations.

6. Populating RecyclerView with Data:

- Create a data structure to hold information about the marked locations.
- Implement an adapter to bind data to the RecyclerView.

7. Integrating RecyclerView with Google Maps:

- Display RecyclerView alongside the Google Map in the layout.
- Synchronise the selected marker data with the corresponding item in the RecyclerView.

8. Enhancing User Experience:

- Implement smooth transitions between selecting a marker on the map and scrolling to the corresponding item in the RecyclerView.
- Explore additional features like clustering for a better visual representation.

9. Testing and Debugging:

- Test the extended functionality on various devices and screen sizes.
- Debug any issues related to marker placement, RecyclerView data, or interactions.

10. Advanced Features (Optional):

- Explore advanced features like info windows for detailed information.
- Implement animations or transitions to enhance the user experience.

11. Documentation and Code Cleanup:

- Document the changes made to the codebase.
- Clean up the code, ensuring readability and adherence to best practices.

12. Q&A and Troubleshooting Session:

Resources:

- Google Maps Android API Documentation
- Android RecyclerView Guide

Topic 11: App Development Navigation

11.1 Learning Objectives

This topic begins with the introduction of

On completion of the topic, students will be able to:

- Understand the principles and benefits of the Navigation Architecture Component.
- Apply the various techniques for navigating between different screens in an Android app.
- Use the Android Jetpack library, its components, and explain their role in modern Android app development.
- Recognise the importance of SEO in the context of web development.
- Explore and implement Android Architecture Components, including LiveData, ViewModel, and Room.

11.2 Pedagogic Approach

Information will be transmitted to the students during the lectures. They will then practise the skills during the laboratory sessions and extend their understanding during private study time. The tutorial will then provide an opportunity to review the key ideas and obtain further guidance and support.

11.3 Timings

Lectures:	2 hours
Laboratory Sessions:	3 hours
Private Study:	9 hours

11.4 Lecture Notes

The following is an outline of the material to be covered during the lecture time. Please also refer to the slides.

The structure of this topic is as follows:

- Navigation architecture
- Navigating between screens
- JETPACK
- Search Engine Optimisation (SEO)
- Architecture components

11.5 Laboratory Sessions

The laboratory time allocation for this topic is 3 hours.

The purpose of the lab work "Implementation of an Android Kotlin app using Jetpack Navigation with three pages" is to guide students through the process of implementing an Android app with Jetpack Navigation. The tasks outlined in the lab PowerPoint slide.

Exercise 1: Implementation of an Android Kotlin app using Jetpack Navigation with three pages

11.6 Private Study

The time allocation for private study in this topic is expected to be 9 hours.

Exercise 1: Enhance the existing Android Kotlin app (labwork: *Implementation of an Android Kotlin app using Jetpack Navigation with three pages*) with additional features and improve the user experience.

Implementing Fragment4 (New Feature)

Create Fragment4:

- Create a new fragment (e.g., Fragment4) with a unique layout.
- Design the UI to showcase a new feature or information.

Add to NavGraph:

- Update the nav_graph.xml file to include the new fragment.
- Connect Fragment3 to Fragment4 with a navigation action.

Update ActionBar:

- Modify the ActionBar title to reflect the current destination.

Adding RecyclerView to Fragment1 (Enhancement)

Enhance Fragment1:

- Integrate a RecyclerView into Fragment1 to display a list of items.
- Design a custom layout for RecyclerView items.

Populate RecyclerView:

- Load sample data or fetch data from a mock API.
- Implement an adapter to populate data into the RecyclerView.

Implementing Camera Integration (New Feature)

Create Fragment5:

- Design a new fragment (e.g., Fragment5) for camera-related features.
- Include UI elements for capturing photos or videos.

Integrate Camera API:

- Implement the logic to capture photos or videos using the Camera API.
- Handle runtime permissions for camera access.

Additional Tips:

Testing:

- Allocate some time for testing each new feature or enhancement.
- Ensure proper error handling and user feedback.

Documentation:

- Update code comments and documentation to reflect the new changes.
- Include details about the purpose and usage of each fragment.

Deployment:

If time allows, explore the process of deploying the app to a test environment or distributing it to testers.

Topic 12: Deployment and Testing

12.1 Learning Objectives

This topic provides an overview of

On completion of the topic, students will be able to:

- Understand the importance of testing frameworks like JUnit and Espresso.
- Explain how to create effective test cases and mocking dependencies.
- Understand the principles of configuration control,
- Manage changes to the Android Kotlin codebase efficiently.
- Describe the use of GitHub for version control in Android Kotlin projects.
- Explain the deployment process for Android Kotlin applications.

12.2 Pedagogic Approach

Information and theory of the topic will be presented to the students during lectures. They will then practise the skills during the tutorial sessions. Students are expected to undertake their own private study to understand the theory fully and put the lectures in context.

12.3 Timings

Lectures:	2 hours
Tutorials:	2 hours
Private Study:	8 hours

12.4 Lecture Notes

The following is an outline of the material to be covered during the lecture time and should be read in conjunction with the slides provided.

The structure of this topic is as follows:

- Testing
- Configuration control
- Github
- Deployment

12.5 Tutorial Notes

The time allowance for tutorials in this topic is 2 hours.

Assignment Workshop:

Workshop Objectives:

To guide students through the process of completing the assignment within the deadline.

Workshop Materials:

- Laptops with Android Studio Installed
- Assignment Guidelines and Requirements
- Sample Code Snippets
- Checklists for Assignment Submission

Evaluation:

- Informal Assessment: Continuous monitoring of student progress and engagement.
- Q&A Session: Opportunity for students to seek clarification and guidance.
- Peer Collaboration: Encourage students to collaborate and troubleshoot together.

Notes to Instructors:

Encourage Collaboration: Foster a collaborative learning environment where students can work together and share insights.

12.6 Private Study

The time allocation for private study in this topic is expected to be 8 hours.

The students' private study focus for this week is on working on the assignment.