Republic of Rwanda
Ministry of Education

RTB | RWANDA TVET BOARD

SWDMA501

MOBILE APPLICATION DEVELOPMENT

Develop Mobile Application using Flutter

Competence

RQF Level: 5

Learning Hours 100

Credits: 10

Sector: ICT AND MULTIMEDIA

Trade: SOFTWARE DEVELOPMENT

Module Type: SPECIFIC

Curriculum: ICTSWD5001 -TVET CERTIFICATE V IN SOFTWARE DEVELOPMENT

Copyright: © Rwanda TVET Board, 2024

Issue Date: February 2024 2024-25

<table>
<tr><td>Purpose statement</td><td colspan="6">This specific module describes the knowledge, skills and attitude required to Develop Mobile Application using Flutter. This Module is intended to prepare learners pursuing TVET Level 5 in Software development. Upon completion of this module, the learner will be able to Apply Basics of Dart, Implement UI designs, Integrate backend functionality and Publish Application.</td></tr>
<tr><td>Learning assumed to be in place</td><td colspan="6">N/A</td></tr>
<tr><td rowspan="6">Delivery modality</td><td colspan="2">Training delivery</td><td>100%</td><td colspan="2">Assessment</td><td>Total 100%</td></tr>
<tr><td colspan="2">Theoretical content</td><td>30%</td><td rowspan="4">Formative assessment</td><td>30%</td><td rowspan="4">50%</td></tr>
<tr><td colspan="2">Practical work:</td><td rowspan="3">70%</td><td rowspan="3">70%</td></tr>
<tr><td>Group project and presentation</td><td>20%</td></tr>
<tr><td>Individual project /Work</td><td>50%</td></tr>
<tr><td colspan="3"></td><td colspan="2">Summative Assessment</td><td>50%</td></tr>
</table>

Elements of Competence and Performance Criteria

Elements of competence	Performance criteria
1. Apply Basics of Dart	1.1 Development environment is correctly prepared according to the operating system.
	1.2 Dart concepts are appropriately applied according to the dart standards.
	1.3 Object-oriented principles are properly implemented in accordance with the dart syntax.
	1.4 Dart Libraries and packages are appropriately used based the application requirements
	2.1 Flutter environment is properly prepared based on system requirements.

2. Implement UI designs	2.2 Flutter's widget system is correctly utilized based on responsiveness principles for mobile application
	2.3 State management is efficiently implemented based on app functionality.
	2.4 Flutter's rich set of pre-designed widgets are effectively used based on industry standards and design guidelines.
3. Integrate backend functionality	3.1 External services with flutter packages are effectively integrated based on industry standards and best practices.
	3.2 Storage management is correctly implemented based on data integrity and security standards.
	3.3 Modular microapps are properly implemented based on application requirements.
	3.4 Error handling mechanisms are effectively performed in accordance with industry standards.
	3.5 Tests are efficiently performed based on the reliability of the mobile application requirements.
	3.6 Codebase issues are correctly debugged based on relevant techniques.
4. Publish Application	4.1 Installable files for mobile platforms are correctly generated based on the manufacturer's guide.
	4.2 Submissions to relevant app stores of the app generated builds are correctly published following their respective submission guidelines.
	4.3 Post-deployment issues are correctly addressed based on quality standards and user expectations.

KNOWLEDGE, SKILLS AND ATTITUDE		
Knowledge	**Skills**	**Attitudes**
• Description of Dart Concepts • Description of Flutter Concepts • Description of Android App Concepts • Description of iOS App Concepts	• Performing Cross platform app development • Performing Android app publication • Performing iOS app publication • Performing Microapps development • Performing Widget based UI designing • Applying Dart programming	• Creativity • Flexibility • Innovative • Having Patience • Being a Team Player

Course content	
Learning outcomes	**At the end of the module the learner will be able to:** 1. Apply Basics of Dart 2. Implement UI designs 3. Integrate backend functionality 4. Publish Application
Learning outcome 1: Apply Basics of Dart	**Learning hours: 20**
Indicative content	
• **Preparation of development environment** ✓ Introduction to Dart	

- Dart features and characteristics
- Dart frameworks
- Use cases.

✓ Description of key terms
- Data types
- Variables
- Control flow structures.
- Functions
- Native Apps
- Cross Platform

✓ Installation of key tools (Windows and Apple)
- Dart SDK
- Integrate dart with the Code Editor (Visual Studio Code)
- IDE for specific operating system
- Testing dart environment.

- **Applying the dart concept.**

✓ Declaration of variables
- Data types
- Naming convention

✓ Implement control flow structures.
- Conditional statements
- Sequence switch statements.
- Iterating statements

✓ Using functions
- Using built-in functions

- Declaring functions
- Parameters and return types.
- Calling functions

- **Applying Object Oriented Programming (OOP)**
  - ✓ Classes and Objects
  - ✓ Inheritance
  - ✓ Polymorphism
  - ✓ Encapsulation
  - ✓ Abstraction
- **Using dart libraries and packages**
  - ✓ Importing and using libraries
  - ✓ Exploring built-in Dart libraries
  - ✓ Managing dependencies with pub (Dart's package manager)
  - ✓ Using external packages for enhanced functionality

Resources required for the learning outcome	
Equipment	• Computer
Materials	• Third party libraries, Internet
Tools	• VS Code, Dart SDK
Facilitation techniques or Learning activity	• Demonstration, • Simulation • Group work • Practical exercise • Individualized • Group discussion
Formative assessment methods /(CAT)	• Written assessment • Presentation • Practical assessment

Learning outcome 2: Implement UI designs	Learning hours: 30

Indicative content

- **Preparation of flutter environment**
  - ✓ Introduction to Flutter framework
    - Definition
    - Purpose
    - Features
  - ✓ Widgets
    - Definition
    - Types
    - Widget Lifecycle
  - ✓ State management
    - Definition
    - Packages
    - Libraries
    - Methods
  - ✓ Environment Set Up
    - Installation of Flutter SDK
    - Installation of IDE (Android Studio, XCode)
    - Configuration of development environment
  - ✓ Creating a new Flutter project
- **Applying flutter's widget system**
  - ✓ Stateful and stateless
  - ✓ Widget Tree and Hierarchy
    - Parent-child relationships in Flutter
    - Widget composition
    - Row and Column

        - Container
        - Expanded
        - Stack
    - ✓ Core widgets
        - Text and Styling
        - Image and Asset
        - Interactive (Buttons, Gesture).
        - Layout Widgets
- **Implementation of state management.**
    - ✓ Using packages.
        - GetX package
        - Provider package
    - ✓ Using pattern
        - Redux pattern
        - Business Logic Component (BLoC) pattern
    - ✓ Using setState() method
    - ✓ Using the riverpod solution.
    - ✓ Using navigation and routing.
        - Navigator
        - Route
        - BottomNavigationBar
        - TabBar and TabBarView
- **Using pre-designed widgets**
    - ✓ Material Design Widgets
    - ✓ Cupertino Widgets
    - ✓ Flutter Icons
    - ✓ Third-Party Packages

Resources required for the indicative content	
Equipment	• Computer (Windows, MacOS), Mobile Phone (Android, iPhone)
Materials	• Material Design components, Flutter Icons
Tools	• Internet, VS Code, Android Studio, Xcode, Emulators, Flutter SDK
Facilitation techniques or Learning activity	• Demonstration, • Simulation • Group work • Practical exercise • Individualized • Group discussion
Formative assessment methods /(CAT)	• Written assessment • Presentation • Practical assessment

Learning outcome 3: Integrate backend functionality.	**Learning hours: 30**
Indicative content	

- **Integration of external services**
  - ✓ Description of HTTP Requests
    - GET
    - POST
    - PUT
    - DELETE
    - UPDATE

- PATCH

✓ Adding dependencies

✓ Adding calls

✓ Handle Responses

✓ Parsing JSON Data

✓ Perform Authentication and Authorization

✓ Push notifications(firebase)

✓ Implement Security measures

- Secure Data Storage
- Secure Network Communication
- Input Validation and Output Encoding

- **Implement storage management.**

✓ Data integrity

✓ Security standards.

✓ Use Local Data Storage

- Working with Shared Preferences
- Working with SQLite
- Working with File Storage

- **Implementation of microapps**

✓ Description of modular microapp

- Definition
- Structure

✓ Applying modular microapps concept

- Project structure

- Dependency injection
- Shared components

✓ Perform microapp Build configuration

- Configure each module pubspec file
- Android gradle
- iOS build settings

- **Perform Error Handling**

✓ Description

- Definition of error handling
- Error classes

✓ Exception Management

- try-catch block
- OnError
- CatchError

✓ Rethrowing exceptions

- Finally block
- on clause

- **Perform testing**

✓ Description

- Definition
- Importance
- Testing Levels

✓ Implement Types of testing

- Unit Tests
- Widget Tests
- Integration Tests
- Functional Tests
- UI Tests
- Performance Tests
- Regression Tests
- Cross-Platform Testing
- Security testing
- End-to-End (E2E) Tests
- Mocking and Stubbing
- Code Coverage Tests

✓ Test device Responsiveness.

- Select testing tools
- Test using Emulator and Simulator
- Test using Physical Device
- Perform manual Testing
- Perform automated testing

✓ Test Reliability

- Unit and Widget Testing
- Integration and End-to-End Testing
- Edge Case and Stress Testing

- Performance Testing and User Acceptance Testing (UAT)

- **Debug Codebase issues.**
  - ✓ Description of key terms
    - Codebase
    - Debug
    - Logging
    - Flutter DevTools
    - Isolation
    - Assertion
    - Breakpoints
    - Print statement
  - ✓ Applying debugging methods
  - ✓ Code Reviews
  - ✓ Prepare documentation

Resources required for the indicative content	
Equipment	• Computer (Windows, MacOS), Mobile Phone (Android, iPhone)
Materials	• Material Design components, Flutter Icons
Tools	• Internet, VS Code, Android Studio, Xcode, Emulators, Flutter SDK, DevTools
Facilitation techniques or Learning activity	• Demonstration, • Simulation • Group work • Practical exercise

	• Individualized • Group discussion
Formative assessment methods /(CAT)	• Written assessment • Oral assessment • Practical assessment

Learning outcome 4: Publish Application	**Learning hours: 20**
Indicative content	

- **Generation of installable files**
  - ✓ Description
    - Types of builds
    - Installable file (Android & iOS)
  - ✓ Compilation Models
    - Just-in-Time (JIT) Compilation
    - Ahead-of-Time (AOT) Compilation
    - Hot Reload and Hot Restart mechanisms
  - ✓ Perform builds generation
    - iOS (IPA)
    - Android (APK)
- **Submission of application files**
  - ✓ Prepare Store Assets
    - App Icon
    - App Screenshots
    - App promotional materials/previews

- ✓ Create Developer account registration
  - Google Developer Console
  - Apple Developer Account
- ✓ Generate App Release Builds(.ipa, .aab)
- ✓ Configure app setting
  - App ID(Android & iOS)
  - App Description
  - Set up app listing
  - Distribution
- ✓ upload App bundles (Android & iOS)

- **Address post deployment issues**
  - ✓ Monitor crash reports (UXCam,Sentry)
  - ✓ Performance degradation
  - ✓ Applying of App Store Optimization (ASO)
  - ✓ Compatibility problems
    - Based on Operating System type
    - Based on Operating System version (API Level, iOS version)
  - ✓ Perform Hot fixing

Resources required for the indicative content	
Equipment	• Computer
Materials	• Google Developer Console Account, Apple Developer Account
Tools	• Internet, Flutter Inspector, DevTools, Browser
Facilitation techniques	• Demonstration, • Simulation • Group work

	• Practical exercise • Individualized • Group discussion
Formative assessment methods /(CAT)	• Written assessment • Presentation • Performance assessment

Integrated/Summative assessment (For specific module)

KLEIN is a growing online retail start-up located in Rwamagana, they don't have a robust presence in the digital marketplace by offering a seamless shopping experience to its customers. They deal in a wide range of products including clothes, shoes, bags, household equipment, jewellery and cosmetics. They are hiring a mobile app developer to create a comprehensive e-commerce cross-platform mobile application for them. The mobile app should allow users to browse a wide range of products, easily add items to their cart, securely make purchases, and track their orders in real-time. As a Flutter developer, you are hired to design and implement a feature-rich app that meets KLEIN's requirements and exceeds their expectations.

Tasks:

1. Develop screens for browsing products with search, filtering and multi-language capabilities.

2. Implement user authentication features.

3. Implement a shopping cart that allows adding, updating, removing items and performing checkout.

4. Implement real-time order tracking and history views

5. Conduct testing to ensure functionality, performance, and compatibility.

6. Provide all possible reports

Information:

1. API has been provided.
2. Login must be a module of the whole system

Resources	
Tools	Flutter SDK, Dart SDK, Visual Studio Code, XCode, Appium, Flutter Driver, Widget Inspector, Windows OS, MacOS
Equipment	Computer, Android Mobile Phones, iOS Mobile Phone
Materials/ Consumables	Papers, Pens, internet

Assessable outcomes	Assessment criteria (Based on performance criteria)	Indicator	Observation		Marks allocation
			Yes	No	
1. Apply Basics of Dart (20%)	1.1 Development environment is correctly prepared	Ind.1 Dart SDK is installed.			1
		Ind.2 Code Editor is installed.			1
		Ind.3 IDE for a specific OS is configured.			2
		Ind.4 Dart Environment is tested			2
	1.2 Dart concepts are appropriately applied	Ind.1 Variables are applied			2
		Ind.2 Control flow structures are implemented			2
		Ind.3 Functions are Used			2
	1.3 Dart Libraries and packages are appropriately used	Ind.1 Imported Libraries are used			2
		Ind.2 Built-in libraries are used			2
		Ind3. Dependencies are managed			2

		Ind.4 External packages are applied			2
2. Implement UI designs (30%)	2.1 Flutter environment is properly prepared	Ind.1 Flutter SDK is installed			2
		Ind.2 IDE is configured			2
		Ind.3 New Flutter project is created			2
	2.2 Flutter's widget system is correctly utilized	Ind.1 Stateful and stateless widgets are used			2
		Ind.2 Widget Tree and Hierarchy are respected			2
		Ind.3 Core widgets are applied			2
	2.3 State management is efficiently implemented	Ind.1 User logins are managed			3
		Ind.2 Products can be browsed			3
		Ind.3 Products can be added to cart			3
		Ind.4 Products can be filtered			3
	2.4 Flutter's rich set of pre-designed widgets are effectively used	Ind.1 Material Design Widgets are used			2
		Ind.2 Flutter Icons are used			2
		Ind.3 Multi languages are applied			2
		Ind.1 JSON Data are used			2

3.Integrate backend functionality (30%)	3.1 External services with flutter packages are effectively integrated	Ind.2 Authentication is implemented			2
		Ind.3 Push notifications are used			2
	3.2 Storage management is correctly implemented	Ind.1 Login sessions are stored locally.			1
		Ind.2 Stored data are encrypted.			2
		Ind.3 Data integrity is respected.			1
		Ind.4 Errors are handled			1
	3.3 Modular microapps are properly implemented	Ind.1 Project structure is respected			2
		Ind.2 All pubspec files are configured			2
		Ind.3 Android gradles and IOS build settings are configured.			2
	3.4 Tests are efficiently performed	Ind.1 App is running on Emulator			2

		Ind.2 App is running on a physical device.			2
		Ind.3 Widget components are tested			2
		Ind.4 Performance is tested			1
	3.5 Codebase issues are correctly debugged	Ind.1 App properly debugged			2
		Ind.2 Codes are Reviewed			2
		Ind.3 Debugging report is generated.			2
4. Publish Application (20%)	4.1 Installable files for mobile platforms are correctly generated	Ind.1 Hot Reload and Hot Restart mechanisms are used.			4
		Ind.2 iOS (IPA) Build is generated			3
		Ind.3 Android (APK) is generated			2
	4.2 Submissions to relevant app stores of the app generated builds are correctly published	Ind.1 App Icons are designed			2
		Ind.2 App Screenshots are designed			3
		Ind.3 App Release Builds are generated			3

	4.3. Address post deployment issues	Ind 1: Crashes are Monitored			2
		Ind 2: Compatibility problems are fixed			2
Total marks					100
Percentage Weightage					100%
Minimum Passing line % (Aggregate): 70%					

References:

1. Miola, A. (2020). Flutter Complete Reference: Create Beautiful, Fast and Native Apps for Any Device. KDP: Amazon LLC.
2. Napoli, M. L. (2019). Beginning Flutter: A Hands On Guide to App Development 1st Edition. Indianapolis: John Wiley & Sons.
3. Rose, R. (2022). Flutter and Dart Cookbook 1st Edition. Calfornia: O'Reilly Media.
4. Thornton, E. (2020). Flutter For Beginners: A Genius Guide to Flutter App Development. KDP: Amazon Digital Services LLC.
5. Tutorials Point. (2019). Flutter. Retrieved from Tutorials Point (I) Pvt. Ltd:

Glossary

IDE: Integrated Development Environment

APK: Android Package Kit

iOS: iPhone Operating System

IPA: iOS package App Store

SDK: Software Development Kit

OS: Operating System

.aab: Android App Bundle

E2E : End-to-End

Author's Note Page

Authoring institution

Copies available from:

Authoring institution's Adrress