University of Northampton	**Faculty Of Art, Science and Technology** **Field of Computing**
Module Level:	Level 5
Module Code + Name:	CSY2091 | Mobile Application Development
Credit Value:	20
Module Leader:	Emmanuel Ofori-Attah | Emmanuel.ofori-attah@northampton.ac.uk
Assessment Code + Type:	PJ1
Assessment Deliverable(s) as stated in the Module Specification:	This module is designed to give an understanding of the technologies (hardware/software) and how these are utilised in a modern networks. This understanding is then use to develop the necessary skills to design and implement (programming) software to be deployed within modern networks.
Weighting (%):	**70%**
Submission Date:	18/05/2025 23:59
Feedback and Grades:	See NILE for feedback under Assessment and Submission

Learning Outcomes aligned to this assessment

Aim: This module is designed to give an understanding of the technologies (hardware/software) and how these are utilised in a modern networks. This understanding is then use to develop the necessary skills to design and implement (programming) software to be deployed within modern networks.

Subject-Specific Knowledge, Understanding & Application

1. identify and respond to inter-relationships between components of modern mobile computing and application development, and explain / interpret them in relation to a complex situation
2. analyse the principles of mobile device programming to enable the design and implementation of applications which have some complexity
3. Evaluate and appraise the use of emerging computing technologies in fixed and mobile contexts.

Changemaker & Employability Skills

4. Identify, select and justify approaches to use from a range of both predefined and self-generated, creative solutions, using clearly defined / accepted problem solving strategies
5. Identify and solve straightforward problems including some abstract problems related to mobile application development and/or modern networks

NOTE: Students who cannot demonstrate understanding of their work, will not pass.

The module tutor may invite you for an online or Physical viva-voce. Poor demo/viva could negatively influence other sections in the marking criteria and result in an overall fail grade.

Assessment Overview

This assignment requires you to complete all assignment tasks to achieve the learning outcomes.

Assessment Tasks

Welcome to Grace Tasty Bites, a busy restaurant that wants to streamline its operations. Right now, the restaurant tracks staff shifts, schedules, and food orders manually, which leads to confusion and inefficiencies. To solve this, you need to develop a Restaurant Management App that allows:

- Admins to manage staff, assign shifts, calculate earnings, and generate invoices.
- Employees to check their shifts, accept or decline them, and view their earnings.
- Customers to browse the menu, order food, and receive a digital receipt.

Kotlin and SQLite Database

Basic System Requirements

Admin Panel (For Restaurant Management)

- Staff & Role Management
  - Add, update, delete, and view staff.
  - Assign roles (Chef, Waiter, Cashier, Manager).
- Shift Management
  - Assign shifts to employees.
    - Employees can accept or decline shifts.
    - View a list of employees working on a particular day.
    - Track the number of shifts each employee completes.

- Payroll & Invoice Generation
  - Calculate total shifts worked by each employee.
  - Generate an invoice at the end of the month showing:
  - Total shifts worked
  - Pay per shift
  - Total earnings
  - Employees should be able to download their monthly earnings report (File Storage).

- Menu Management
  - Add, update, delete, and view menu items.
  - Employee Panel (For Staff Members)

- Employee Panel (For Staff Members)
  - Shift Management
    - View assigned shifts.
    - Accept or decline shifts.
    - See their schedule for the week.
    - View total shifts completed.
    - Download monthly earnings invoice.

Customer Panel (For Ordering & Receipt Generation)

- Menu & Ordering
    - Browse the restaurant’s menu.
    - Add food items to the cart and place an order.
- Receipt Generation
    - Generate and download a receipt (PDF or text file) after ordering.

Note: To get a higher grade, you will need to add enhancements that are deemed relevant to the assignment. Please note that any additional enhancement has to be deemed relevant in order to receive full marks.

*See deliverables for details.

DELIVERABLES

All the below system requirements (A, B and C below) **MUST** be delivered to achieve a passing grade for this assignment.

A) Technical Report

The report should, ideally, consist of the following sections (in order):

1. Username and password for all relevant accounts
2. A list of all the features implemented in a tabular format. For example:

Feature	Implemented (Partial/Full)	Any comments
Add a post	Full	No error validation.
View all posts	Full	

3. Explanation of the main sections/fragments of the code. Provide information that would be useful for another developer (not an end user!) who may want to extend/maintain your system.
4. Screenshots of the system showing all key features
5. Evidence of Testing:
   . Blackbox Testing: Test **logs** providing information of all the tests carried out (including any failed tests for functionality not implemented)
      . Please provide screenshots and tables as part of the testing.
   . List of any bugs and/or weaknesses in your system (if you do not think there are any, then say so). Bugs that are declared in this list will lose you fewer marks than ones that you do not declare.
   See Appendix 2 for a suggested outline of the technical report.

B) Source Code

The source code must be well documented with necessary comments. Consistent and clear indentation of the code is also important. Source code needs to be submitted in two forms:

(i) As a single ZIP archive (.zip file consisting of all the files).

(ii) A commented full listing in a separate Word document named "Full Source Code Listing".

C) Video Demonstration

In addition to the report, you **must** submit a video demo (URL) of your assignment. The demo should be about 10 minutes long (maximum:15 minutes) and should cover all your work in a logical way. You should explain the main phases of design and implementation covering the main fragments of code. **Your video must have clear audio and a visible face to ensure the marker can identify you. If your video lacks both a face and audio, it will not be marked, as ownership of the work cannot be verified. The video should include a walkthrough of the software, showcasing its key features. Additionally, the module tutor may request an online viva-voce, and a poor demonstration or viva performance could negatively impact other areas of the marking criteria.**

It is crucial that your video remains accessible at the time of marking. Videos that are not publicly available during marking will result in a failed grade. All video links must be submitted through the designated video link submission system—submissions via email will not be accepted.

Submission Procedure:

To do this, go to the NILE site for this module and use the link labelled “Assessment and Submission” there you will see the relevant.

- E-Submission of documents through Turnitin on NILE as TWO separate WORD documents.

1. Document 1 = **Report**
2. Document 2 = **FullSourceCodeListing**
3. E-Submission of a single ZIP archive that contains all the source code files (kt)), data files and ReadMe file. The archive must be named with your student ID, e.g. *12345678.zip* where 12345678 is your student ID. To do this, go to the NILE site for this module and use the link labelled ‘Submit your work’. Clicking on the link (*SourceCodeEsubmission*), will take you into the submission form, where you can upload your ZIP archive using the ‘Attach File’ button (*Browse for Local File*). Finally, click the *Submit* button.
4. When submitting your video demonstration, use of Kaltura (https://video.northampton.ac.uk/) is recommended. You must ensure that the video link is accessible to the marker (do not set it to private access).

- Failure to follow the above submission guidelines may result in a capped or fail grade.

Appendix 2 - Technical Report Outline

1. Introduction
   - Give a brief overview of the project scope, aims and objectives.
   - An explanation of the importance/justification of the proposed application.
   - Background Research into existing Applications
2. System Specification
   - A comprehensive list of all features implemented, categorised as basic requirements and additional enhancements.
   - For each feature, specify whether it's fully or partially implemented, along with any relevant comments.
3. Interface Design
   - Provide a visual representation of the application's key features through selected screenshots.
   - Highlight any innovative design elements or user interface design principles deployed.
4. Build/Implementation
   - Provide an explanation of the main sections and fragments of code, highlighting the code you are the most satisfied with.
   - Provide insights useful for other developers who may extend or maintain the system, focusing on architectural choices and design patterns employed.
5. Testing
   - Provide evidence of testing efforts, including black box testing logs.
   - Description of test cases, including both successful and failed scenarios.
   - Identification and discussion of any bugs or weaknesses in the system, demonstrating awareness of potential areas for improvement.
6. Conclusion
   - Provide a summary of key deliverables and achievements.
   - Reflection on the additional features implemented beyond the basic requirements.
   - Discussion on the potential for further enhancements or future iterations of the application, including suggestions for improving functionality, usability, and scalability.
   - Reflection on lessons learned and insights gained during the development process.
7. References
   - Provide a list of any resources, frameworks, or libraries used in the development of the application

Assessment Submission

→ The deadline is 11.59pm (British time) on the due date *provided on* NILE

→ Submit your work on NILE, under: Assessment and Submission

→ The completion and submission of your assignment is your responsibility

→ Only work submitted through NILE will be marked

→ Submission should be in the appropriate format e.g. *word document, zip file, video files*

→ You must grant relevant access to videos and links

Work correctly uploaded to Turnitin will get a receipt for proof of submission. Submission not through Turnitin, will have a green banner at the top of the screen for successful submission.

Academic Integrity

- → The UON's Policy on Academic Integrity and Misconduct must be strictly implemented
- → Submitting this assignment means that this is entirely your own individual work
    - → All work for this submission must be your (your group's) own
    - → All work for this submission must be based on module content
- → All work for this assignment must be developed originally & solely for this submission
- → You may discuss work with other students, but any code written should be your own
- → All sources must be referenced and clearly cited
- → You must submit all items of the assessment according to the submission procedure stated in this document
- → Failing to meet the university's guidance on AI and Plagiarism will impact your grade
- → Failure to follow the submission procedure may impact your grade
- → A high similarity report from TurnItIn my impact your grade, subject to investigation

***Note**: See University Of Northampton Guidance on Plagiarism*

Grading:

Your grade is dependent on achievement of the specified learning outcomes for this assessment. The rubric is used as a standard benchmark, so assignments are marked equally. The grading rubric is based on the criteria you are assessed on. See the marking criteria rubric *provided on NILE.*

- → Marks are given for assignment requirements only
- → Late submissions, within 7 days of deadline, maximum grade is a pass, (40)
    - → This does not include resits, which are already capped at pass (40)
- → Late submissions, more than 7 days after the deadline, are fails

- → Grades above 39% are a pass mark
- → Passed assessments cannot be retaken to improve the grade
- → Students with grades less than pass (40) get 1 opportunity to resit the assessment
- → Standard resit grades are capped at a pass (40)

Extensions and Mitigating Circumstances

Students who experience extreme unprecedented circumstances that impact their study can appeal for extensions or mitigating circumstances to extend the deadline of their assessments.

- → A student who attends an examination or submits an assessment declares themselves 'fit to sit' and cannot afterwards submit a claim for Mitigating Circumstances.

Marking Criteria

See below

Sample Marking Criteria

	A	B	C	D	F	G
Design (10%)	Excellent design of program and user interface.	Good quality design of program and user interface.	Satisfactory design of program and user interface.	Adequate design of program and user interface.	Faulty design of class diagrams. Very little discussion of the overall design.	No submission or no submission of merit
Functionality (40%)	All basic system requirements are met and criteria for (B) and many significant additional features.	All criteria for (C) and some significant additional features.	Most basic system requirements are met.	Most basic system requirements are met however works partially as designed.	Most basic system requirements are not met.	No submission or no submission of merit
Technical Report (20%)	Excellent description of main sections of the code. All sections covered in the right order. Any assumptions and bugs/weaknesses are clearly stated. Evidence of both white box and black box testing with extensive code coverage.	Good description of main sections of the code. All sections covered in the right order. Evidence of both white box and black box testing with good code coverage.	Satisfactory description of main sections of the code. Most sections are covered.Evidence of either white box or black box testing with satisfactory code coverage.	Poor description of main sections of the code. Very Little covered.Little evidence of either white box or black box testing with satisfactory code coverage.	No description of code sections.No evidence of any white box or black box testing.	No submission or no submission of merit. No submission or no submission of merit
Code quality and Efficiency (20%)	Code is very well structured to enable reusability and debugging. Excellent work on error handling.	Code is well structured to enable reusability and debugging. Good work on error handling.	Some thought has been given on how the code is structured. Some work on error handling.	Little thought has been given on how the code is structured. Some work on error handling.	Hardly any thought on how the code is structured.	No submission or no submission of merit
Demonstration (10%)	Covers all implemented features in sufficient detail. Any known bugs are highlighted. Validation is tested (e.g., entering invalid values)	Covers all implemented features in sufficient detail	Satisfactory demo but needs more details.	A poor virtual demonstration	A very poor virtual demonstration	No submission or no submission of merit