Computer Science and Creative Technologies

Coursework or Assessment Specification

Module Details

Module Code	UFCF8S-30-2
Module Title	Advanced Software Development
Module Leader	Zaheer Khan
Module Tutors	Zaheer Khan, Barkha Javed, James Lear
Year	2022-23
Component/Element Number	A Main-sit
Total number of assessments for this module	(1) Part-I: The first covers the analysis and design of a software system. (2) Part-II: The second covers the coding and testing of the design created in the first assignment.
Weighting	100%
Element Description	Group project: **This assignment is to be completed in groups of up to three students.**

Dates

Date issued to students	4 October 2022
Submission Date	(1) **Part-I: November 15, 2022 14:00hrs** Demo session in the same week during the scheduled practical lab sessions (2) **Part-II: January 10, 2023 14:00hrs** Demo session during the same week lab sessions This assessment is NOT eligible for 5 calendar day late submission window
Submission Place	Blackboard
Submission Time	**14:00**
Date to be returned to students	Within 20 working days of the final submission.
Submission Notes	(1) Part-I: Please submit a PDF document containing Use Case, Class and Sequence diagrams. (2) Part-II: Please submit a portfolio as a ZIP file with your code, a PDF report on Agile practices, Test cases used in your program in the form of a table, suitable program running screenshots capturing success/failure testing scenarios, and Individual report. Please note that all the members of a group must be present during the demonstration sessions, and all members must upload both part I and part II individually.

Feedback

Feedback provision will be	On the spot verbal feedback during the demo session and as appropriate written feedback will be uploaded to Blackboard.

Contents

Section 1: Overview of Assessment

This assignment assesses the following module learning outcomes (taken from the module specification):

- *Analyse problems in order to identify software-solution approaches and requirements for computer-based software-intensive systems.*
- *Compare and contrast software development methodologies and choose one suitable for a given application.*
- *Design, implement, test and manage reasonably sized concurrent and distributed software systems, working in a group , considering database and GUI components.*
- *Develop the necessary transferable skills – e.g. communication, delegation, openness, group decision making, flexibility, tolerance - to enable them to work in a group.*
- *Discuss the need for security in the context of system development.*
- *Evaluate emerging software development tools, technologies and methods, e.g. cloud based development, DevOps and AI-based software development tools.*

The assignment is worth 100% of the overall mark for the module.

The assessment is designed to allow you to use your learning and play to your strengths. You can always seek advice and/or interim verbal feedback from your tutors during practical sessions.

Broadly speaking, the assignment requires you to design and implement a moderately realistic object-oriented system. You will produce detailed object models and designs from system requirements; use the modelling concepts provided by UML. You'll have a demo and Questions/Answer session with your tutor(s). You will then map the designs into code and perform unit testing using an automated testing tool. You will demonstrate your system and answer questions about use of your approach and possible alternatives, technologies and tools.

The assignment is described in more detail in section 2.

This is a GROUP assignment and to be completed in groups of up to three students.

Working on this assignment will help you to understand more clearly the concepts, problems, and techniques of object-oriented systems, and how these can be used to design and implement a reasonably large size software systems while working in a team (to a professional standard).

Section 2: Task Specification

All the tasks are based on the following scenario. Please read carefully all information contained within the following passage of text.

Horizon Cinemas Booking System (HCBS)

The Horizon Cinemas (HC) is a successful chain of cinemas in Birmingham, Bristol, Cardiff and London. Each city has at least two cinemas at different locations. HC is looking for an IT solution to manage screening of films and ticket booking system with the objective to make it convenient for customers to book tickets when they visit a cinema booking office.

Each cinema has up to 6 screens and each screen runs between one to four daily shows of a selected film. Film listings, show times and prices may vary from one cinema to another. Seating capacity for each screen varies between 50 to 120. For each screen, seat(s) can be reserved in lower hall or in upper gallery. The following table indicates standard price range for lower hall type tickets for different shows/listings at different times of the day.

Table: Price range for lower hall

	Morning Show (between 8am – noon)	Afternoon Shows (From noon – 1700)	Evening Shows (From 1700 – midnight)
Birmingham	£5	£6	£7
Bristol	£6	£7	£8
Cardiff	£5	£6	£7
London	£10	£11	£12

Typically, lower halls have about 30% of total seats. The ticket price for upper gallery is 20% higher than the lower hall. Upper gallery also offers up to 10 VIP seats at 20% higher price. For example,

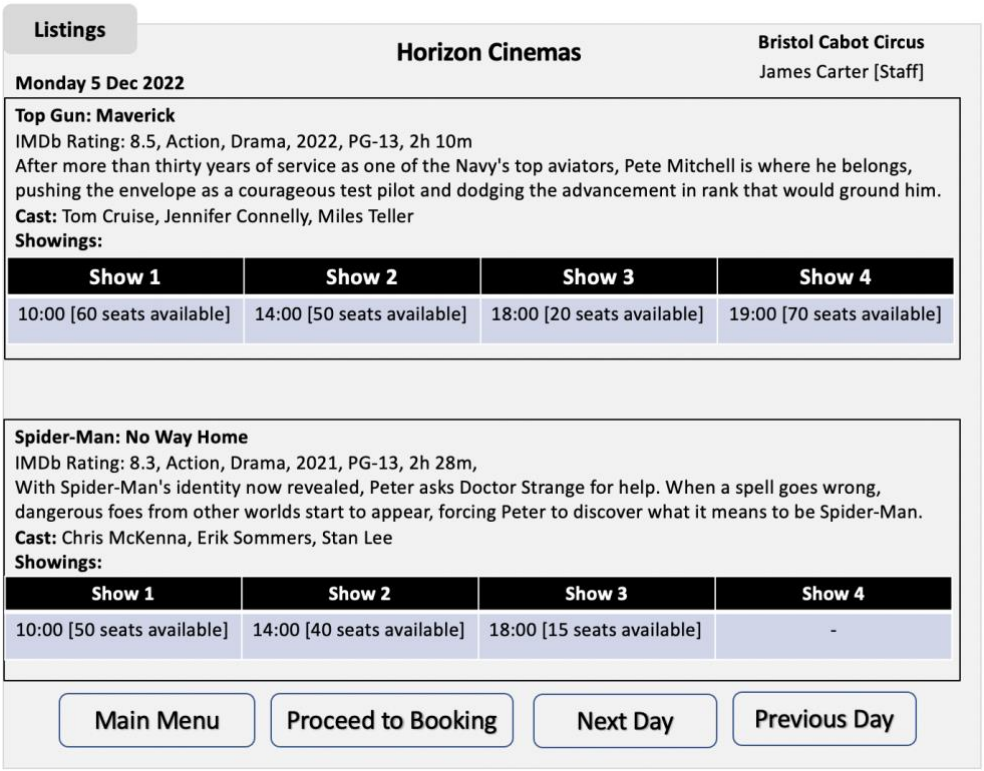
If lower hall price is £10 per ticket, then VIP ticket price would be (£10 + (£10 x 20%)) x 20% = £14.40.

Tickets can also be booked for up to one week in advance. Tickets can be cancelled at least one day prior to film show at 50% cancellation charges. No ticket cancellation allowed on the day of show.

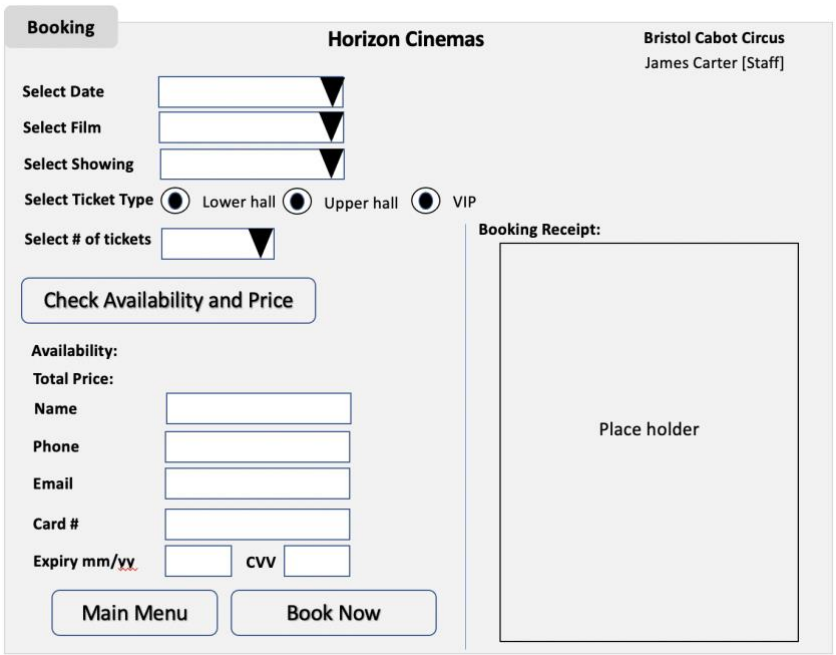
System has three types of users: Booking staff, Admins and Managers.

The system has the following Graphical User Interfaces (GUIs) needs. Please keep it simple and be creative by using labels, text boxes, radio buttons, check boxes, menus, drop-down lists, window tabs, buttons, data tables etc.:

1. **Film Listing:** films with description and actors' details, film genre and age, rating, show times, etc. The following figure depicts an example of how data should look like but be creative and use different GUI widget as you see appropriate:



2. **Booking:** System should allow to fill a booking form that will provide option to check seats availability and total tickets price for a selected date, showing and type of tickets. Booking reference should be unique for each booking. Booking receipt should include Booking reference, Film name, Film date, Showing time, Screen #, Number of tickets, seat numbers, Total booking cost and Booking date. The following figure depicts an example of booking form but be creative and use different GUI widget as you see appropriate:



Only admins can select listings from other HC cinemas and make booking on behalf of customers. Note: You are NOT required to provide a payment mechanism for your system, though you may simulate this.

3. **Cancellation:** There should be an option to cancel booking. There is no cancellation or refund for missed shows.

Managers, Admins and Booking staff can access Film listing, booking and cancellation GUIs/views.

4. **Admin view:** Admin view to manage screening of films. They can add, update and remove listings e.g., add new films, remove film, update show times, attach shows to screen/hall, etc. They can also generate admin reports e.g., number of bookings for each listing, total monthly revenue for each cinema, top revenue generating film, monthly list of staff making number of bookings in sorted order, etc.

Admins and Managers can access Admin GUI/view.

5. **Manager view:** HC also expects to expand their business in existing or new cities in near future and would like their IT solution to allow adding new cinemas. Manager view allows to add new cinemas and listings in existing or new cities. These new cinemas then become available for booking by booking staff.

Only Managers can access Manager GUI/view.

You should also consider other suitable non-functional requirements (e.g., security measures) necessary for the HC booking system. Be creative.

You are the leader of a team of three developers who have been asked to design and implement a system which enables the Managers, Admins and Booking Staff to schedule listings and booking activities and keep track of the cinemas, films, screens and bookings in a simpler and easier way, enforcing the rules described above.

More specifically there are following tasks:

You are asked to produce the following deliverables for the scenario given above.

Part-I (Weightage 40%)
Task 1. Produce a use case diagram to capture the functionality for the system to be built.
Task 2. Produce a class diagram to meet all the requirements captured in the use case diagram.
Task 3. Produce at least three sequence diagrams for some use cases that adds information.

Part-II (Weightage 60%)
Task 4. Write a joint report (300-700 words) outlining the steps that you followed to develop the system in relation to an Agile development episode and justify your choices. You should consider benefits and challenges faced and may consider the following steps:

- **Strategy planning:** This is the initial phase for our system development process where we as developers need to identify system users to avoid building wrong solution and identify role and responsibilities of each team member……

- **Continuous team iterations:** The development phases of our project are more flexible as we continuously iterate between planning and implementation compare to conventional development methodology which is too rigid and strict……

- **Team coordination and communication:** The essence of our project is effective coordination and communication among members either through face-to-face or online conversation…….

- **Simplicity:** Our main objective is to practice simplicity which is important for the system to avoid the structure being too complicated……

- **Work Plan (Not mandatory but if they can produce would be good):** Project Backlog, SPRINT Cycle

Task 5. Develop the system using a suitable Object Oriented Programming language (e.g., Python as we'll use Python programming language for this module) and Database(s) considering all the functionalities described above.

- You should design and implement suitable database(s) and fill it with suitable mock data for testing and demonstration.

- You should be creative and come up with your own User Interface Design for different abovementioned GUI needs.

- You should also implement non-functional requirements (e.g., security measures) to a good standard.

Task 6: Testing is typically a part of the program development - You should use a test strategy to test your system thoroughly. You should identify all the suitable test cases for all the classes implemented. When you test your code, you should make sure that your program does not allow bad data to be stored into your objects. Deliberately feed in your program out of range or wrong data and try to make it fail, see if you can swipe bad values into the member variables. One example test case shown below.

Test Case #	Description	Test dataset/Input	Expected output	Actual output
1	Booking staff is able to get price and book five tickets for a selected listing and date	Selecting film from listing on a specific date and time, ticket type, and quantity	Shows seats availability and total price	As expected [Pass/Fail]

Task 7: Individual Contribution and Reflective Report (300-700 word). Your report should include:

(a) the evidence of your work produced for the above activity with an outline;
(b) reflection on your contribution to the work in relation to skills and knowledge acquired.
(c) provide brief reflection on other suitable emerging technologies and/or methods which could be used for designing and implementation of this system.

Your lab tutor will play role of representative of HC. If you have questions about this assignment, please discuss with your lab tutors. You may also post questions for your lab tutors on the discussion board on Blackboard. You can find the discussion board from the front page of the module Blackboard pages and use the forum under the title "**Group project**".

Section 3: Deliverables

You must use the Blackboard electronic submission system to submit your work. Each student will have to upload complete package individually. Electronically submitted deliverables include:

For Part I: Each student needs to upload the following deliverable.

1. One PDF document that covers Task 1 to Task 3.

For Part II: Each student needs to upload the following deliverables in one .zip file. The naming standard of the zip file is WP1234567.zip where 1234567 is the student Id.

1. For task 4, a reflective group report in PDF format.
2. For task 5, the following deliverables should be compressed in a .zip file.
    a. A software system based on the specifications in section 2, with source code in ZIP file (using e.g., 7Zip). This should contain all the files and folders for the full working system. All program files must have student ID and student name who has written the code. You should also include list of additional Python packages (e.g., using pip list command) needed to run your software. Please do not include python distribution folder in your ZIP file. Provide a text file with instructions on how to start and use your software system e.g., any passwords etc.
    b. relevant Database(s) (e.g. MySQL or MongoDB) dump.
3. For task 6, evidence of testing e.g., test cases, unit tests and outputs. These can be included in a separate PDF document.
4. For task 7, each student will write individual report in PDF format.

Instructions for submission

You must submit your work *before* the stated deadline by electronic submission through Blackboard.

- Multiple submissions can be made to the portal, but only the final one will be accepted.
- **It is your responsibility to submit deliverables in a format stipulated above.** Your marks may be affected if your tutor cannot open or properly view your submission.
- **Do not leave submission to the very last minute**. Always allow time in case of technical issues. Also, save your work frequently to avoid losing your work at the last minute.
- The date and time of your submission is taken from the Blackboard server and is recorded when your submission is complete, not when you click Submit.
- **It is essential that you check that you have submitted the correct file(s),** and that each complete file was received. Submission receipts are accessed from the Coursework tab.
- UWE academic regulations for late submissions will be applied. Please check module handbook.

Note: All submitted work will be checked for Plagiarism using SafeAssign or other suitable software/approaches. Every student undertaking an assignment or other piece of assessed work is required to take, and will be deemed to have taken, full responsibility for all the work submitted by the student. In particular, this includes responsibility for any assessment offence (e.g., Plagiarism, collusion) committed. Assessment offence such as plagiarism or collusion will result in 0 marks.

Section 4: Marks Distribution and Marking Criteria

Part I weightage is 40% and Part II weightage is 60%.
In common with all UWE standard undergraduate assignments, the pass mark for this assignment is 40%. The detailed marking criteria for this assignment is detailed in Appendix-A.

Section 5: Feedback mechanisms

On the spot verbal feedback during the demo session + as appropriate written feedback uploaded to Blackboard. Formative feedback provided in Part-I will be useful to complete Part-II of the assignment.
Please use Blackboard discussion forum to ask questions about this project.

Appendix A. Marking Criteria

Part-I (Weightage 40%)								
	0.0-2.9	3.0-3.9	4.0-4.9	5.0-5.9	6.0-6.9	7.0-8.4	8.5-10.0	Mark & Advice for Improvement
Use case diagram (10%)	Little or no diagram provided or diagram is mostly wrong (overall less than 30% complete).	Partial diagram provided but actors and/or use-cases/relationships are only partially correct (overall less than 40% complete).	Partial diagram provided, actors and/or use-cases/relationships are correct (overall less than 50% complete).	Almost complete diagram provided, actors and/or use-cases/relationships are only partially correct (overall less than 60% complete).	Almost complete diagram provided, actors and/or use-cases/relationships are correct. (overall, less than 70% complete)	Complete diagram provided, actors and/or use-cases/relationships are partially correct (overall less than 85% complete)	Complete diagram provided, actors and/or use-cases/relationships are correct (100% complete)	
Class diagram (10%)	Little or no diagram provided or diagram is mostly wrong (overall less than 30% complete).	Partial diagram provided but Class naming convention, Class relationships, Attributes/methods of the classes and their visibility and type are only partially correct (overall, less than 40% complete).	Partial diagram provided but Class naming convention, Class relationships, Attributes/methods of the classes and their visibility and type are correct (overall, less than 50% complete).	Almost complete diagram provided, Class naming convention, Class relationships, Attributes/methods of the classes and their visibility and type are only partially correct (overall less than 60% complete).	Almost complete diagram provided, Class naming convention, Class relationships, Attributes/methods of the classes and their visibility and type are correct (overall less than 70% complete).	Complete diagram provided, Class naming convention, Class relationships, Attributes/methods of the classes and their visibility and type are partially correct (overall less than 85% complete)	Complete diagram provided, Class naming convention, Class relationships, Attributes/methods of the classes and their visibility and type are correct (100% complete)	
Sequence diagram (at least three	Little or no diagram provided	Partial diagram provided but Lifeline Notation, Activation Bars,	Partial diagram provided, Lifeline Notation, Activation Bars,	Almost complete diagram provided but Lifeline Notation,	Almost complete diagram provided, Lifeline Notation, Activation Bars,	Complete diagram provided but Lifeline Notation, Activation Bars,	Complete diagram provided, Lifeline Notation, Activation Bars,	

diagrams) (10%)	or diagrams are mostly wrong (overall less than 30% complete).	and Message Arrows are only partially correct (overall less than 40% complete).	and Message Arrows are correct (overall less than 50% complete).	Activation Bars, and Message Arrows are only partially correct (overall less than 60% complete).	and Message Arrows are correct (overall less than 70% complete).	and Message Arrows are only partially correct (overall less than 85% complete).	and Message Arrows are correct (100% complete).	
Demo and Individual Q&A (10%)	Absent 0.0 Inadequate (barely able to explain the analysis/design and/or work done) 0.0-3.0 Good (good explanation of analysis/design and/or work done) 3.1-6.0 Very Good (very good explanation of analysis/design and/or work done) 6.1-8.0 Excellent (excellent explanation of analysis/design and/or work done) 8.1-10.							

Part-II (Weightage 60%)

	0.0-2.9	3.0-3.9	4.0-4.9	5.0-5.9	6.0-6.9	7.0-8.4	8.5-10.0	Mark & Advice for Improvement
Agile development (10%)	Little or no description provided (overall less than 30% complete).	Partial description provided but texts/justification are not written reasonably accurately (overall less than 40% complete).	Partial description provided, texts/justification are written reasonably accurately (overall less than 50% complete).	Almost complete description provided but texts/justification are not written reasonably accurately (overall less than 60% complete).	Almost complete description provided, texts/justification are written roughly accurately (overall less than 70% complete).	Complete description provided, texts/justification are written reasonably accurately (overall less than 85% complete).	Complete description provided, texts/justification are written very well and accurately (100% complete).	
	0.0-6.0	6.1-8.0	8.1-10.0	10.1-12.0	12.1-14.0	14.1-17.0	17.1-20.0	Mark & Advice for Improvement
Implementation (20%)	Little or no development	Partial development,	Partial development,	Partial development,	Almost complete development,	Complete development	Complete development	

	(overall less than 30% complete).	compiles and runs, but GUI displays input and output messages only partially correctly (overall less than 40% complete).	compiles and runs, GUI displays input and output messages only partially correctly (overall less than 50% complete).	compiles and runs, but GUI displays input and output messages only partially correctly (overall less than 60% complete).	compiles and runs, GUI displays input and output messages only partially correctly (overall less than 70% complete).	including security measures, compiles and runs, but GUI displays input and output messages only partially correctly. (overall less than 85% complete).	including security measures, compiles and runs, GUI displays input and output messages correctly. One or more Databases are used for persistent storage. (100% complete).	
	0.0-2.9	3.0-3.9	4.0-4.9	5.0-5.9	6.0-6.9	7.0-18.4	8.5-10.0	Mark & Advice for Improvement
Testing (10%)	Little or no test cases (overall less than 30% tested).	Partially tested, (overall less than 40% tested).	Partially tested, (overall less than 50% tested).	Partially tested, (overall less than 60% tested).	Almost tested, (overall less than 70% tested).	Almost tested, (overall less than 85% tested).	Completely tested, (overall less than 100% tested).	
Individual Q&A (10%)	Absent 0.0 Inadequate (barely able to explain the analysis/design and/or work done) 0.0- 3.0 Good (good explanation of coding/testing and/or work done) 3.1- 6.0 Very Good (very good explanation of coding/testing and/or work done) 6.1- 8.0 Excellent (excellent explanation of coding/testing and/or work done) 8.1- 10.0							
Individual report (10%)	No submission 0.0 Inadequate (barely covers individual contribution evidence or reflection) 0.0-3.0 Good (reasonable examples of individual contribution evidence and reflection on learning) 3.1- 6.0 Very Good (good examples of individual contribution evidence and reflection on learning) 6.1-8.0 Excellent (good examples of individual contribution evidence, reflection on learning and emerging technologies) 8.1-10.0							

This Page is Intentionally Left Blank