Middlesex University London

Science and Technology

Computer Science

Full Stack Development

CST3144

Module Leader: Dr Luca Piras

Semester 1, Academic Year 2024-25

12 weeks

Online location of handbook

This handbook can also be accessed via My Learning at: link.

Other formats available

This handbook is available in a large print format. If you would like a large print copy or have other requirements for the handbook, please contact the Disability Support Service disability@mdx.ac.uk

Disclaimer

The material in this handbook is as accurate as possible at the date of production. You will be notified of any minor changes promptly. If there are any major changes to the module you will be consulted prior to the changes being confirmed. Please check the version number on the front page of this handbook to ensure that you are using the most accurate information.

Other documents

Your module handbook should be read and used alongside your programme handbook and the information available to all students on My Learning and MyMDX including the Academic Regulations. Your programme handbook can be found on the My Learning programme page for your programme.

Table of Contents

1 Welcome

This module aims to develop a deep understanding of the latest full stack programming techniques, frameworks, and methodologies used by industry to develop the next generation software, which can be deployed on a wide range of devices and systems. The students will investigate, develop, and deploy the latest programming language standards, which are fundamental to app development, and currently being widely employed in industry. Modern programming frameworks will be introduced to provide the essential software architecture for large-scale software development, and the ability to target a wide range of platforms. The module will cover the three most important components of a complete architecture for a software application: Front-End, Back-End, and System Administration. This handbook will guide you on important aspects, including how to approach to the module, how it is organised, what the assessment entails, and related rules.

Dr Luca Piras

2 The module teaching team

Please see below details of the teaching team for this module.

Module Leader: Dr Luca Piras		
	Room number:	*TG11*
	Email:	l.piras@mdx.ac.uk
	Telephone number:	+44 (0)20 8411 2700
	Office hours:	Please email for an appointment
Dubai Lecturer: **Dr Chinnu George**		
	Room number:	Dubai Knowledge Park, Blocks 4, 16, 17 & 19
	Email:	c.george@mdx.ac.ae
	Telephone number:	+971 (0)4 367 8100
	Office hours:	Please email for an appointment
Mauritius Lecturer: **Dr Zia Lallmahomed**		
Photo	Room number:	Coastal Road, Uniciti
	Email:	z.lallmahomed@mdx.ac.mu
	Telephone number:	[230] 403 6400
	Office hours:	Please email for an appointment

3 Communication with the teaching team

Students may contact staff via e-mail and by making an appointment to see them (for instance, online with a Zoom call).

Staff will contact students by e-mail, the My Learning module page and via lectures and seminars. The team may send urgent group and/or individual messages about the module to you by email, so it is important that you read your University email regularly.

In the first instance problems should be dealt with by talking to a member of the module team. You can give feedback on this module to the Module Leader, your Student Voice Leader, to your Personal Tutor, and through the end of module evaluation survey.

4 Module overview

Aims

This module aims to develop a deep understanding of the latest full stack programming techniques, frameworks, and methodologies used by industry to develop the next generation software, which can be deployed on a wide range of devices and systems. The students will investigate, develop, and deploy the latest programming language standards, which are fundamental to app development, and currently being widely employed in industry. Modern programming frameworks will be introduced to provide the essential software architecture for large-scale software development, and the ability to target a wide range of platforms. The module will cover the three most important components of a complete architecture for a software application: Front-End, Back-End, and System Administration.

Learning outcomes

Knowledge

LO1	Justify and deploy the latest programming standards, technologies, and strategies required for advanced software development.
LO2	Evaluate the main characteristics, strengths, and weaknesses of the latest software architecture frameworks.
LO3	Critically evaluate the methodologies of developing platform-independent software, and the strength and weakness of existing solutions.

Skills

LO4	Create a Front-End solution by employing best practices for advanced software development.
LO5	Deploy and manage essential technologies for dependency management.
LO6	Design and develop a Back-End solution with advanced technologies including software hosting and data storage.

Syllabus

Topics that will be covered throughout the module include:

- Programming Front-End solutions.
- Advanced interactivity for Front-End solutions.
- Version Control for software systems.
- Refactoring of software systems.
- Advanced Design and Development of software applications.
- Full Stack Development and Dependency Management.
- Advanced Back-End technologies and Administration.
- Programming Platform-Independent software applications.
- Communication between Front-End and Back-End technologies.
- Deployment of Software Applications.
- App Hosting and Data Storage.

Learning and teaching strategy

Weekly contact hours:

- Workshop: 2 hrs (activities including for example Theory, Practice, Tutorials, CW Demo, CW Support, CW Q&A, etc., depending on the stage of the Module)
- Laboratory: 3 hrs
- Drop-in online session: 1 hr

Short key concept videos will be made available.

The focus of teaching will be a mixture of activities, lab-based practical work, and discussions, using a Project-Based Learning (PBL) approach. Skills and experience are built up progressively through weekly lab sessions in order that the students can implement a full stack software application. Various tools for software development will be introduced throughout the module as they become necessary. The student will develop their individual application through in-class discussions, peer-to-peer collaboration with other students, and self-study.

Delivery method:

☒ On-campus/Blended

☐ Distance Education

Assessment scheme

(a) Formative assessment scheme
Students will receive formative feedback on summative assessments tasks. Students will be expected to work towards a complete software application, mastering related architecture and technologies. The students will be formatively assessed by using a Project-Based Learning (PBL) approach, which will support students in the labs.

(b) Summative assessment scheme
The coursework gives students the opportunity to put into practice the theories, frameworks, and libraries covered in the module. The coursework will focus on the Full-Stack aspects of a complete software application development.

Task: Full Stack Software Application Development (Individual) This application development coursework will allow students to apply knowledge and skills in the latest programming standard to design and implement a full-stack software application architecture.

The deliverable of this coursework will consist of software code and students are expected to run a short demonstration of the application.

Weighting	Specification e.g. word count / duration / no. of pages	LO mapped to	Anonymously marked	Ethics approval required
100%	Demonstration (maximum 10 minutes). Zip file of the project (code of the resulting software) must be no more than 10MB. Deadline: week 11.	1- 6	☒ *No* ☐ *Yes*	☒ *No* ☐ *Yes – individual student* ☐ *Yes – group approval* ☐ *Yes – whole module*

In order to pass the module, the student will be required to achieve either:

☒ an overall aggregate of grade 16;

☐ an overall aggregate of grade 16 with a minimum of grade 16 in each assessment component

☐ an overall aggregate of grade 16 with a minimum of grade 17 in each assessment component

☐ an overall aggregate of grade 16 with a minimum of grade of 18 in each assessment component

Seen examination	**........%**
Unseen examination	........%
Coursework (no examination)	100%

Type	**Hours**
Independent study	228
Scheduled Teaching	72

Research Ethics

Statement 3:

- **The teaching, learning, assessment and research activities undertaken in this module have been considered and are <u>not likely to require ethical approval.</u>**
- However, please seek advice if undertaking the module entails carrying out any research activities involving **human participants, human data, animals/animal products, precious artefacts, materials or data systems.** If you submit work that includes data gathered from or about people, this may be treated as academic misconduct and could lead to fail grade being awarded.
- Further information about ethics can be found at the following link: http://mdx.mrooms.net/enrol/index.php?id=12277 (Log in required)

5 Learning resources

This module has a variety of learning resources available for you to use to support your learning. These include recorded lecture, lecture slides, feedback, and key reading materials. These can be accessed online via the module page. Please visit the module page regularly to make use of these.

6 Expectations of studying this module

The module team is here to help and support you achieve your goals. One of the key elements to successfully completing this module is engaging with all of the learning opportunities we offer as well and working with your peers to support one another.

This module is designed as a combination of contact sessions, directed study and independent study. This means you must participate in all the allocated sessions and you must complete all set prework and activities outside them. Students are expected to take an active part in all learning sessions whether these are online or on campus, for example lectures, lab sessions, practical classes, etc.

To make the most of this module please complete the following every week.

- Complete any potential prework indicated in preparation for learning sessions. This may be watching videos, reading through set material or chapters and completing activities. Please make notes of points you need to clarify and discuss these in learning sessions with module tutors.
- Read through the notes making a note of any points you need to discuss with your tutor.
- Complete the set activities before the next session, making a note of any points you need to discuss with your tutor.
- Go to the module My Learning page regularly and use all the resources, for example indicated reading, links, extra material, etc. Make a note of anything you wish to discuss with your tutor.
- Complete potential further reading indicated.

The module team is committed to support you and your fellow students whilst you undertake this module. In order for you to get the most out of sessions you need to come prepared and ready to contribute. Please ensure that any work set by the team has been completed before activities indicated. After each class please review what has been covered and make a note of anything you would like clarification on.

6.1 Attendance and Engagement

Engaging with online and on-campus in-person learning and activities is integral to your success. Middlesex University supports you to achieve your full potential through a number of strategies, all of which provide a supportive learning environment online, remotely, face-to-face, or blended.
Further information on attendance and engaging with your programme will be available at your Induction and general information is available at the weblink below.

https://mymdx.mdx.ac.uk/campusm/home#pgitem/419149/t

6.2 Professional behaviour and online conduct

The programme of study you are undertaking is underpinned by developing professional behaviour and attitude. You are expected to behave in a professional, supportive manner to your peers and teachers. You must come to sessions prepared and ready to contribute where appropriate. Please remember that your University ID should be carried with you always whilst on campus and you must be able to identify yourself if asked to do so. Please conduct your email communication with fellow students, tutors and all relevant staff in a formal and courteous manner.

6.3 Academic Integrity and Misconduct

Academic Integrity is a set of principles and values to show that you work in a professional, honest and ethical way. You should be aware of the University's academic integrity and misconduct policies and procedures. Taking unfair advantage over other students in assessment is considered a serious offence by the University. Action will be taken against any student who contravenes the regulations through negligence, foolishness or deliberate intent. Academic misconduct takes several forms, in particular:

- **Plagiarism** – using extensive unacknowledged quotations from, or direct copying of, another person's work and presenting it for assessment as if it were your own effort. This includes the use of third party essay writing services.
- **Collusion** – working together with other students (without the tutor's permission), and presenting similar or identical work for assessment.
- **Infringement of Exam Room Rules** – Communication with another candidate, taking notes to your table in the exam room and/or referring to notes during the examination.
- **Self-Plagiarism** – including any material which is identical or substantially similar to material that has already been submitted by you for another assessment in the University or elsewhere.
- **Unauthorised use of Artificial Intelligence** – Appropriate use of Artificial Intelligence (AI) for this Module is described in section 7.1.

Links to the relevant University regulations and additional support resources can be found here:

Student Success Essentials Course includes useful information about how to approach your assessments and complete them with honesty. The course also describes what plagiarism (cheating) is and how to avoid it so you don't face any disciplinary action. For successfully completing this course, you will be awarded a certificate that will verify the knowledge you have gained. Certificates can be shared and promoted via LinkedIn and other digital channels. You will have to log into to MyMDX and then MyLearning to access the course. https://mdx.mrooms.net/course/view.php?id=17199

Full details on academic integrity and misconduct and the support available can be found at https://mymdx.mdx.ac.uk/campusm/home#pgitem/419149/t

The Academic Integrity and Misconduct policy is available in our Public Policy Statements (under Academic Quality) at: Our policies | Middlesex University London (mdx.ac.uk)

Referencing & Plagiarism: Suspected of plagiarism?:

http://libguides.mdx.ac.uk/c.php?g=322119&p=2155601
Referencing and avoiding plagiarism:

https://mymdx.mdx.ac.uk/campusm/home#pgitem/419258

The Middlesex University Students' Union (MDXSU) Advice Service offers free and independent support in making an appeal, complaint or responding to any allegations of academic or non-academic misconduct.
https://www.mdxsu.com/advice

6.4 Extenuating circumstances:

There may be difficult circumstances in your life that affect your ability to meet an assessment deadline or affect your performance in an assessment. These are known as extenuating circumstances or 'ECs'. Extenuating circumstances are exceptional, seriously adverse and outside of your control. For further information search 'Extenuating circumstances' in MyMDX.

7 Assessment

Formative Feedback: Formative assessments help show that you are learning and understanding the material covered in this module and allow us to monitor your progress towards achieving the learning outcomes. Although formative assessments do not directly contribute to the overall module mark they do provide an important opportunity to receive feedback on your learning.

In this module, students will receive formative feedback on summative assessments tasks. Students will be expected to work towards a complete software application, mastering related architecture and technologies. The students will be formatively assessed by using a Project-Based Learning (PBL) approach, which will support students in the labs.

Summative assessment: Summative assessment is used to check the level of learning for the module. It is summative because it is based on accumulated learning during the course. It is the summative assessment that determines the grade that you are awarded for the module.

There is 1 assessment component in this module:

- Coursework 1
    - ***Task: Full Stack Software Application Development*** (Individual) This application development coursework will allow students to apply

knowledge and skills in the latest programming standard to design and implement a full-stack software application architecture.

- The deliverable of this coursework will consist of software code and students are expected to run a short demonstration of the application.

The table below specifies the associated deadlines:

Summative assessment	Weighting	Deadline	Feedback
Coursework 1	100%	2nd December at 2pm	Within 15 working days since student's demonstration

In order to pass this module, you need to pass the assessment with a minimum grade of 40% or equivalent.

Before you submit your work for final grading, please ensure that you have accurately referenced the work. It is your responsibility to check the spelling and grammar, as all written assessments will assess technical proficiency in the English. This means accurate and effective spelling, punctuation and grammar. Details of how it will be assessed will be provided in the marking criteria for each assessment and the University overall approach can be found within the Grade Criteria Guide in the University Regulations https://www.mdx.ac.uk/about-us/policies (scroll to university regulations)

Reasonable adjustments will be made for those students who have a declared disability/specific learning condition which would affect performance in this area.

Reassessment for this module normally takes place in the following way:

Normally in April with demonstration of improvement of the coursework, and ability to clearly describe and demonstrate it.

Further information is available in MyMDX

7.1 CourseWork 1 (CW1): Full Stack Software App Development (Individual) (100%)

Deadline: 2nd December at 2pm.

Type of CW: Individual Work.

Demonstration: the demonstration will be recorded by the Lecturer for internal and external moderation purposes.

IMPORTANT. A submission could receive zero marks if it fails any of the following requirements:

- The student must **book an available slot for the demo** of a CW**, by the CW submission deadline,** via the related "Individual Demonstration Booking System" on the Module Page, **AND** the work must be **demonstrated within the demonstration weeks scheduled,** during the **demo slot booked** (**IMPORTANT**: see more details and guidance on demo booking at **section 7.2.4 in the handbook**, and for guidance on other important related aspects at **section 7.2 in the handbook**).
- The work must be **demonstrated** and **explained satisfactorily during demonstration**, i.e., student can explain what the code does.

Appropriate use of AI
AI is a powerful tool that can increase the productivity of programmers. In this Module you are welcome to use AI tools like GitHub Copilot, ChatGPT and CodeWhisperer to help you to write code for your coursework.

AI tools can help you to program and write more efficiently, but they are not going to complete the coursework for you, they frequently make mistakes, and they are unlikely to follow the technology requirements in the coursework descriptions. If you use these tools to generate the code for you *without* understanding the technologies or coursework requirements, without being able to describe your code properly during demonstration, then you will get a low mark for this Module.

We strongly recommend that you spend time understanding how technologies we cover work and write code with these technologies without the help of AI. Then you will be in a much better position to use AI tools to *help* you to generate drafts of pieces of code for your project. Your understanding of the technologies will enable you to change this code to match the project requirements, integrate it, debug it, and fix bugs.

Task. For this coursework, you need to create the Front-End of a fictitious web app, which allows students and their parents to buy after school classes and activities. Then, you need to build the Back-End for the app. MongoDB will be used for storing the data, and data exchange between the app and the database will be done through REST API implemented using Express.js. The Back-End will run on a Node.js server.

Submission

Instructions for the text area to compile when uploading your submission:

- please, indicate the same information requested below for the README file

Instructions for the zip file, it must contain the following elements, and must be no more than 10MB:

- your code in **2 folders** respectively related to:
    - your **Vue.js App**.
    - your **Express.js App** (**do not include the 'node_modules' folder**, otherwise, the zip will be too big to submit).
- a README file with the following links:
    - [**Vue.js App**] the link to your GitHub Repository.
    - [**Vue.js App**] the link to your GitHub Pages from where the app can directly run.
    - [**Express.js App**] the link to your GitHub Repository.
    - [**AWS (or render.com) Express.js App**] the link to the AWS (or render.com) route that returns all the lessons.
- the **'lesson'** and the **'order' collections** exported from your MongoDB Atlas. See here for how to export collection in MongoDB Compass (https://docs.mongodb.com/compass/current/import-export#export-data-from-a-collection).
- the requests you created in Postman. See here for how to export requests in Postman (https://learning.postman.com/docs/getting-started/importing-and-exporting-data/#exporting-collections).

Overall Requirements and Instructions

- You can use an external CSS library such as Bootstrap. Make sure the library file or online link is included in the submission.

A submission could receive zero marks if it fails any of the following requirements:

- **[Front-End]**
    - the App must be implemented by using Vue.js framework. Other frameworks or related technologies (e.g., React, AngularJS, Svelte, Apache, XAMPP) are NOT allowed.
    - Any JavaScript library is NOT allowed if it duplicates or replaces features provided by Vue.js framework. Check with the tutor if not sure.
- **[Back-End and Connection details for the Front-End]**
    - the Back-End server must use "Node.js"; others such as Apache or Xampp are not allowed.
    - in relation to hosting the Back-End server, it is not allowed to use AWS S3, nor AWS EC2, nor other cloud-based hosting solutions (e.g., Heroku). The only allowed ones are AWS or render.com, with the technologies covered in the related Lecture.
    - the REST API must be developed with "Express.js".
    - the Front-End data access must be achieved with "promise" using "fetch" function; "XMLHttpRequest" or library such as axios:js are not allowed.
    - the data must be stored in "MongoDB Atlas" and retrieved via your Express.js App; local MongoDB or any other databases are not allowed.

- connection to MongoDB (in your Express.js App) must use the native Node.js driver only; libraries like Mongoose are not allowed.

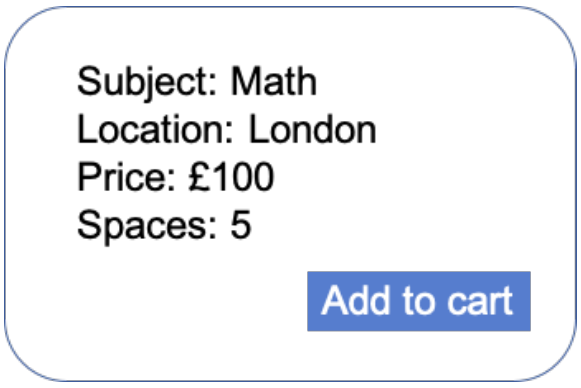

Figure 1. A Lesson element of the App

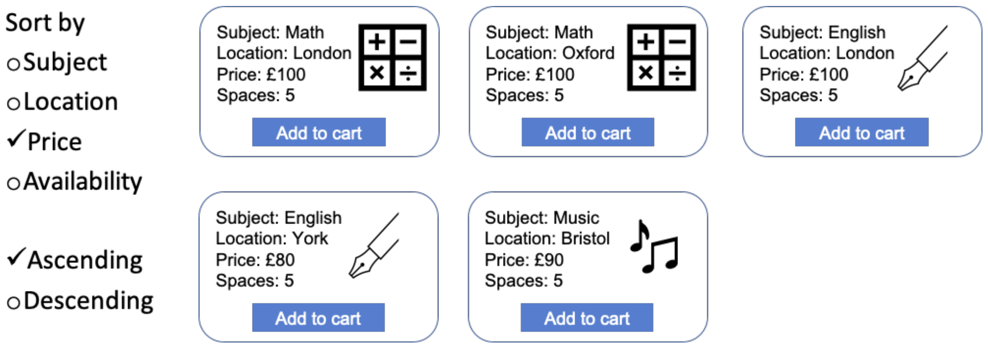

Figure 2. Sort functionality and List of Lessons

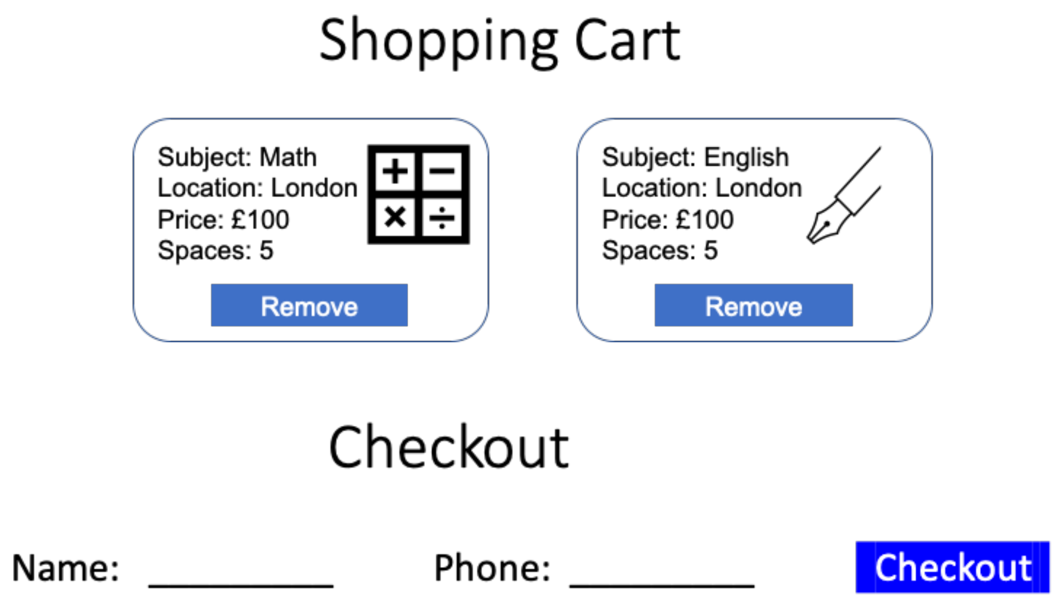

Figure 3. Shopping Cart and Checkout

Marking criteria

- **General Requirements (20%):**

  A. [GitHub Repositories] the code of the Vue.js App must be hosted in a GitHub repository, with at least 10 commits, and the code of the Express.js App must be hosted in another GitHub repository, with at least 10 commits (6% if 2 separated repositories are used, 3% if only 1 repository is used).
  B. [GitHub Pages] the Vue.js App must be hosted and demonstrated on/via GitHub Pages and connected (via Fetch) to your AWS (or render.com) Express.js App (7% if requirement is fully covered, 3% if app is running locally).
  C. [AWS (or render.com)] the Node/Express server must be hosted on Amazon AWS (https://aws.amazon.com/) or on Render (https://render.com) (7% if requirement is fully covered, 3% if the server is run locally, 0% if the server is hosted in another cloud-based solution).

- **[Front-End] “Display List of Lessons” functionality (7%):**

  A. there should be at least 10 lessons, and each lesson should have 5 spaces (or availability) (1%).
  B. each lesson should have at least (5%): Subject (1%), Location (1%), Price (1%), Spaces (or availability: this indicates how many spaces are left) (1%), a Font Awesome icon (or an Image) (1%).
  C. v-for must be used for the display of the lesson list (1%).

- **[Front-End] Sort functionality (10%):**

A. the user can choose to sort the lessons by one of the following attributes (8%): subject (2%), location (2%), price (2%), or spaces (i.e. availability) (2%).
B. there must be an option to sort in ascending or descending order (order dependent on the sorting attribute selected), which should work for each of the attributes (2%).

- **[Front-End] "Add to Cart" functionality (5%):**

  A. each lesson must have an "Add to Cart" button (1%).
  B. the button is always visible, and only enabled when space is larger than 0 (1%).
  C. clicking the button once (related interactions implemented by using v-on) will add one space to the shopping cart, reducing the remaining space by one (2%).
  D. once there is no more space, i.e. space = 0, the "Add to cart" button should be disabled but still visible, i.e. clicking it will not further reduce "space" nor add lessons to the cart (1%).

- **[Front-End] "Shopping Cart" functionality (5%):**

  A. the shopping cart button should only be enabled after at least one lesson is added to cart (1%).
  B. clicking the shopping cart button should show the cart page, and clicking the button again goes back to the lesson page (1%).
  C. the shopping cart, in the cart page, should show all the lessons added (1%).
  D. in the shopping cart page, the user should be able to remove lessons from the shopping cart; the removed lesson is added back to the lesson list (in the lesson page) (2%).

- **[Front-End] Checkout functionality (6%):**

  A. the checkout is part of the shopping cart page (not part of the lessons page) (1%).
  B. the "checkout" button is always visible and only enabled (clickable) after valid "Name" and "Phone" are provided (2%).
  C. the "Name" must be letters only and the "Phone" must be numbers only; the check must be done using JavaScript (suggestion: regular expressions) (2%).
  D. clicking the "checkout" button should display a message confirming the order has been submitted (1%).

- **[Front-End (and Back-End)] Search Functionality (10%):**

  - [Intro] This is the challenge component of this coursework, and it is not expected that everyone can complete it. The solution is not fully covered in the lecture or lab, so you need to research it.

- [Feature Description] The goal is to add a full-text search feature,
- The user can search for a lesson without specifying which attribute to search on.
- For example, searching for "a" should return all the lessons with "a" in its "title" or "location" or "price" or "availability".

- Solutions provided are marked as follows.

[Base Marks (provided depending on which following approach is chosen)]

A. [Approach 1] (2%) "Implemented only in the Front-End", you can implement this feature using Vue.js and/or an existing JavaScript library (could not be a Vue.js library).

   OR

- [Approach 2] (7%) "Implemented functionally in the Back-End and graphically in the Front-End", the difference with "Approach 1" above is that in this case the search needs to be performed in the Back-End (Express + MongoDB), not directly and exclusively in the Front-End, but the Front-End will receive (e.g., via REST API) results from the Back-End and manage the graphical aspects for showing the search results. You cannot use any existing library to implement this functionality. Otherwise, you will not receive any mark for this part. The points for this approach are divided as follows.
  a. "Fetch and Visual Aspects" (3%), in the front end, a "fetch" request should be created to send the search information to the Back-End, and related filtered results obtained should be shown in the Front-End.
  b. "Express API" (4%), an Express.js route should be created to handle the search request, and to return the search results from the MongoDB. The student should implement this as a GET route ("/search"), and should be able to test it also without using the Front-End.

[Further Mark]

B. "search as you type" (3%), there is also this mark if the search supports "search as you type", i.e. the search starts when the user types the first letter (displaying all the lessons containing that letter), and the result list is dynamically filtered as more search letters are entered (similar to Google Search).

- **[Back-End] MongoDB should have (8%):**

   A. a collection for lesson information (minimal fields: topic, price, location, and space) (4%).
   B. a collection for order information (minimal fields: name, phone number, lesson IDs, and number of space) (4%). Suggestion: the element lessonIDs can contain 1 or more lesson IDs, depending on how many different kinds of lessons you have in your order. Other solutions could be accepted: in fact, how you design this is not the primary aspect

of this module, therefore it is up to you to find a reasonable solution that works satisfactorily.

- **[Back-End] Middleware Functions implemented in the Express.js Server should include (8%):**

  A. a “logger” middleware that outputs all requests to the server console; the student needs to be able to inspect and explain the log (4%).
  B. a static file middleware that returns lesson images, or an error message if the image file does not exist; the student needs to be able to test and demonstrate these aspects (4%).

- **[Back-End] REST API implemented in the Express.js Server should include and be tested as indicated in the following (the student must be able to test all the routes properly and explain them) (12%):**

  A. one GET route /lessons, which returns all the lessons as a Json, demonstrated with a Postman request prepared in advance (3%). Example:
     [
     {‘ topic’: ‘math’, ‘location’: ‘Hendon’, ‘price’: 100, ‘space’: 5},
     {‘ topic’: ‘math’, ‘location’: ‘Colindale’, ‘price’: 80, ‘space’: 2},
     {‘ topic’: ‘math’, ‘location’: ‘Brent Cross’, ‘price’: 90, ‘space’: 6},
     {‘ topic’: ‘math’, ‘location’: ‘Golders Green’, ‘price’: 95, ‘space’: 7},
     ]
  B. one POST route, which saves a new order to the “order” collection, demonstrated with a Postman request prepared in advance (4%).
  C. one PUT route, which can update any attribute in a lesson of the “lesson” collection (suggestion/requirement: this will be used, after an order is submitted, for updating the available spaces: updating to any number the available spaces, and not just increasing or decreasing), demonstrated with a Postman request prepared in advance (5% if the requirement is fully covered, 2% if the solution is just increase or decrease).

- **[Front-End] Fetch Functions implemented in the Front-End should include (9%):**

  A. one fetch that retrieves all the lessons with GET (3%).
  B. one fetch that saves a new order with POST after it is submitted (3%).
  C. one fetch that updates the available lesson space with PUT after an order is submitted (3%).

7.2 Extension, Late Penalty, Incentive System and Demonstration Process

7.2.1 Extension

All extensions must be applied through the *Extenuating Circumstances service* (see Section “Extenuating circumstances:” for more details). **Please do not contact the module leader for extension.**

7.2.2 Late Penalty

The late penalty rule of the University is included in the rules of the extenuating circumstances. For more details read the Section “Extenuating circumstances:”, which indicates you how to reach the information on late penalty rule available in MyMDX. **Please do not contact the module leader for late penalty.**

7.2.3 An Incentive System for Motivating Students to Have their Demo in The First Weeks, and for Properly Balancing Demonstrations

Demonstrations of the assessment are scheduled in different weeks sessions. In order to distribute our Demonstrations, and to reward Students for completing on time:

1. **[First Week]** if a student books and gives the Demo in the first scheduled week, the student is rewarded with 4 points.
2. **[Second Week]** if a student books and gives the Demo in the second scheduled week, the student is rewarded with 2 points.
3. **[Third Week]** for the 3rd week there are 0 reward points.
4. **[Exceeding Reward Points]** if the sum composed of “grade achieved + plus Rewards Points” exceeds the maximum possible grade, any excess points are lost. For example, if a student has 4 rewards points and scores 98 on the assessment, the total grade becomes 100, using only 2 Reward Points, and the other 2 Reward Points exceeding are forfeited.

7.2.4 Demonstration Process and Important Requirements/Instructions

The following applies to the demonstration process of all coursework parts.

A submission could receive zero marks if it fails any of the following requirements:

- The student must **book an available slot for the demo** of a CW**, by the CW submission deadline,** via the related "Individual Demonstration Booking System" on the Module Page.
- The work must be **demonstrated within the demonstration weeks scheduled,** during the **demo slot booked**.

Demonstration Process: Important Instructions (obtaining a grade, submission, demonstration and booking).

Requirements for obtaining a grade:

- to **submit** your work by the CW deadline **AND**
- to **book** your Demo by the CW deadline **AND**
- to **attend and give** your booked Demo.

Avoiding Issues related to Booking:

- when the booking system is created, there are **enough slots for the number of students** of the Module (everything is scheduled and **planned according to the number of students, time, and available rooms**).
- **if you leave your booking at the last minute**, there could not be the slot you prefer, or slots could have been all booked.
- accordingly, it is **your responsibility** and **part of a valid submission to book your demo**. Thus, if you try to book at the last minute and there are no more available slots, it is your fault and unfortunately you get 0 for that part; however, if you get an **official extension/deferral from UniHelp**, and the Lecturer is notified from UniHelp, your Demo can be scheduled at the **next possible opportunity** (in this case, **after getting an official extension/deferral from UniHelp, contact the Lecturer** for asking a new scheduling of your demo).

Avoiding Issues related to not attending your Booked Demo:

- it is **your responsibility** and **part of a valid submission to attend your booked demo** and give your demo satisfactorily; if you **cannot attend your demo**, and you have **not important reasons**, it is your fault and unfortunately you get 0 for that part.
- However, if you cannot attend the demo (you have booked) **due to important reasons**:

    1. check if there are **still available slots**, and, in that case, feel free to **book another one autonomously**.
    2. if there are **no other available slots**, and if **you cannot really attend your booked demo due to important reasons**, if you get an **official extension/deferral from UniHelp**, and the Lecturer is notified from UniHelp, your Demo can be scheduled at the **next possible opportunity** (in this case, **after getting an official extension/deferral from UniHelp, contact the Lecturer** for asking a new scheduling for your demo).

7.3 Feedback on your assignments

You will be provided with feedback on all assessments that is helpful and informative, consistent with aiding the learning and development process. The nature of the feedback shall be determined at programme level but may take a variety of forms including: written comments; individual and group tutorial feedback; peer feedback; or other forms of effective and efficient feedback. If you have submitted a formative or draft assessment, you will receive feedback but no grade. The comments should inform you about how well you have done or tell you about the areas for improvement. All assignments should be submitted online unless specified in assessment briefs.

Oral feedback will be provided during demonstration, and written feedback will be provided in UniHub in electronic form.

Feedback on summative assessments will normally be provided within 15 WORKING DAYS of the demonstration date booked and attended.

7.4 How is your assignment mark agreed?

External Examiners (external academic experts) review what we deliver at a programme level. The University reviews a sample of your work to quality assure the grades and feedback you received from the person who marked your work. Our External Examiners will sample a selection of modules from a programme, with more focus of outcomes between modules within a programme.

The following diagram provides an overview of the marking process for your module assessment. Further information on the role of external examiners can be found at. https://www.mdx.ac.uk/about-us/policies/academic-quality/handbook (section 4)

1. You submit your assignment
2. The first marker grades the work and provides feedback; this could be completed anonymously depending on the assessment type.
3. A moderator or second marker reviews a sample of the work to quality assure the grades and feedback, to ensure they are accurate. A final mark for the work is agreed between the first marker and the moderator or second marker.
4. A sample of work, from a selection of modules across the programme, is sent to the External Examiner to check that the grading and feedback is at the right level and in line with external subject benchmarks (this applies to levels 5, 6 & 7only)
5. Your final grades are submitted to the Programme assessment board.

7.4.1 Results Confirmation

First Semester: Provisional Grades: At the end of your first semester, you can see your module grades in the Grades and Progress tile within MyMDX. These grades are provisional and not yet confirmed.

Second Semester: Final Grades and Progression: After your second semester, the Programme Assessment Board will confirm your grades. Then, your final module results, progression status, or finalist classification will be posted in the Grades and Progress tile within MyMDX.

Need Help or More Info?

- **University Guide**: Find detailed information in the University Guide in the Grades and Progress tile within MyMDX.
- **Support Team**: Ask your Progression and Support Team Officer for advice.
- **Regulations**: Check the University regulations for more details.

Anonymous Marking Assessment Policy

We have worked with the Middlesex University Students' Union (MDXSU) to create an anonymous marking policy, in response to student feedback. Anonymous marking ensures that your identity (your name, student number and other personal/identifiable information) is not made available to academics when they are marking your work. This means that you can have confidence that your assessments will be marked fairly and consistently. However, there are some forms of assessment for which anonymity cannot be guaranteed and these are recognised in the policy. The Anonymous Marking Assessment Policy is available at:

https://www.mdx.ac.uk/__data/assets/pdf_file/0037/563599/anonymous-marking-assessment-policy.pdf

8 Indicative Learning Planner

Week	Workshops and Labs	Assessment
1	Programming Front-End Solutions	
2	Advanced Interactivity for Front-End Solutions	
3	Version Control of Software Systems and Refactoring of Software Systems	
4	Advanced Design and Development of Software Applications	
5	Full Stack Development and Dependency Management	
6	Advanced Back-End Technologies and Administration	
7	Programming Platform-Independent Software Applications	
8	Communications between Front-End and Back-End Technologies	
9	Deployment of Software Applications	
10	App Hosting and Data Storage; Review and Coursework Time	Individual Work Lab Demo
11	Review and Coursework Time	Individual Work Lab Demo
12	Review and Coursework Time	Individual Work Lab Demo

9 University 20-point Scale

20-point scale	General scale
1	80% - 100%
2	76% - 79%
3	73% - 75%
4	70% - 72%
5	67% - 69%
6	65% - 66%
7	62% - 64%
8	60% - 61%
9	57% - 59%
10	55% - 56%
11	52% - 54%
12	50% - 51%
13	47% - 49%
14	45% - 46%
15	42% - 44%
16	40% - 41%
17	35% - 39%
18	30% - 34%
19	0% - 29%
20	Non-participation