Unit 25: Full Stack Development

Delivery guidance

Approaching the unit

Computer systems are routinely made up of a number of front-end and back-end systems. Understanding the whole deployment stack, is a highly sight after skill. Either as a manager that must oversee specialists in a particular area, or as developer working in one area, understanding how it forms part of a whole integrated system helps improve the quality of the solutions you produce.

The focus of this unit is the design and development of a software solution that utilises a wide range of technologies. The unit brings together use of front-end and back-end processes, and infrastructure to solve problems. It is expected that the learners will already have a fundamental understanding of programming concepts before starting the unit. Learners would be expected to have completed Unit 4 Programming before beginning this unit. It may be beneficial to learners to have also completed Unit 13 Software Testing before studying this unit, but it is not vital that they do. This unit is best scheduled later in a delivery plan for example in the second year of a two-year programme.

The emphasis of the unit is the development of highly sought-after problem-solving skills. In Learning Aim, A learners will explore the concept of “the stack” and how full stack solutions are used to solve problems and meet user and organisational In Learning Aims B and C learners will develop effective programming skills to design and create a solution that makes use of an identified stack to implement a solution that utilises front-end and back-end processes.

For learning Aims B and C a suitable integrated development environment (IDE) and suitable back-end resources including databases and server technology. This unit does not require use of any specific stack and centres should explore which platforms would be best suited to the needs of the centre and the learners. Due to the restrictions on local networks, learners may find it difficult to run server processes on the local resources. The use of cloud-based resources is recommended as this will reduce the strain on centre infrastructure and provide learners with flexibility and exposure to industry standard technologies. Some cloud providers, such as Google Cloud Platform and Amazon Web Services, offer a range of learning materials and even provide free tier access to systems for learners. It is highly recommended that you explore these platforms and others, to identify which system would be best suited to your delivery.

This delivery guide does not cover everything that needs to be delivered for completion of this unit but gives examples of delivery methods. You should refer to the specification for full details of all the content that needs to be covered.

Delivering the Learning aims

Learning aim A you could start by looking at some well-known computing platforms such as Facebook, Amazon and Uber. Start by looking in broad terms at the different types of processes and technologies involved and identifying where front-end and back-end processes work and interact with each other.

Explore the common approaches to full stack development, identifying some of the key technologies involved and common stacks that are employed by professional developers. As learner familiarity with the stack grows, explore the application of the stack in more details. You should explore the specific technologies used, how they are utilised in modern software development lifecycles and wider implications for stakeholder of utilising full stack technologies.

Delivery of this learning aim would benefit from the use of high quality, detailed case studies as well as talks and presentations from professionals in industry. You should provide learners with opportunities to explore existing uses of full stack technologies in a range of contexts.

Learning aim B you could start with considering the concept of 'client needs'. It would be helpful to draw on knowledge from previous units where learners will have considered functional and non-functional requirements and revisit how to break a problem down in to its component parts (decomposition) and identify ways in which these problems can be solved. Initially learners should consider 'big-picture' issues, and it may be helpful for learners to explore a range of different scenarios and industries to gain an appreciation of how these may impact on the needs of a client.

Provide examples of documentation form different stages of the planning and design stages so learners can identify what should be produced, and what content is required for different stakeholders. Learners should be able to investigate a problem within a range of contexts and produce design documents that solve identified problems. Where learners have programming experience, it is likely that their previous experience previously will be more heavily skewed to front-end development. You may need to spend extra time introducing back-end concepts.

Depending on which units learners have studied before this unit, and the time that has passed since, you may have to revisit design documentation such as flow charts ad pseudocode in addition to the introduction of documents learners may not have had experience of previously such as data designs and UML diagrams. You should start with smaller, less complex problems and slowly build the scale and complexity of what learners would be required to design. It may also be helpful to initially scaffold tasks by providing partially completed designs that learners can improve.

Learning aim C focuses on the development of software development skills to produce complete solutions that utilises a full stack implementation including front-end and backend processes. It is important that learners appreciate that the development/programming skills and testing and refinement skills are interlinked. The need to test code should not be an afterthought but a key part of the coding practice, to ensure solutions are of the highest quality.

Provide opportunities for learners to develop practical programming skills to solve both front-end and back-end computing problems. You may need to initially start with isolated tasks to build skills, as well as possibly initially introducing the concepts of front-end and back-end separately, but you should look to integrate these as soon as possible. Start with simple problems and slowly build the scale and complexity.

Learning aims B and C are closely linked. While some parts will be taught separately, such as how and why a particular design document is used, or specific programming skills, learners cannot effectively design a solution until they fully understand how to develop a solution. Being aware that there are strong links between the practical skills and the development of a solution will greatly help the learners; in order to be able to design an effective solution they must have a strong understanding of what the practical possibilities are.

Assessment model

Learning aim	Key content areas	Recommended assessment approach
A Explore tools and technologies for full stack development	**A1** Full stack development **A2** Implications of full stack development	A written report exploring how and why full stack development is utilised, the benefits it can bring, and any associated drawbacks and wider concerns.
B Design a full stack solution for an identified client	**B1** Software project proposal **B2** Software design documentation	A portfolio of evidence detailing the development and testing of a software solution, which may include: • design documentation for the software solution • completed software solution (including initial and refined version) • copy of the software solution's source code • test documentation and user feedback • analysis of feedback throughout design and development, and evidence of refinement of designs and solution • evaluation of the development, testing and refinement process.
C Develop a full stack solution for an identified client	**C1** Creating software solutions **C2** Developing high quality software solutions **C3** Testing and reviewing software solutions	

Assessment guidance

This unit is internally assessed. There is a maximum number of two summative assignments for this unit. Tutors should refer to the assessment guidance in the specification for specific detail, particularly in relation to the requirements for Pass, Merit and Distinction grades.

The first assignment requires the learner evaluate how and why full stack development is used to solve different problems. They will explore the fundamental concepts of full stack development and the tools it provides to solve problems and deploy computing solutions. They should consider the use of full-stack development in at least three different contexts, supporting their evaluation with examples taken form the contexts. They should also consider the impacts that the use of the identified technologies would have on individuals and organisations.

Learners could produce the evidence for this in different ways in including a written formal report or a presentation to the group. If a presentation is used then assessors could use video recording combined with an observation sheet to cite which assessment criteria the learner has met, with appropriate commentary supporting the reason for awarding a particular grade. A blog or some form of audio or visual evidence would also be acceptable and would allow learners to develop their creativity, provided the information is communicated in a clear and detailed manner using appropriate language.

The second assignment requires the learner to design and develop a computing solution using a full stack deployment. Learners will respond to a scenario which provides a set of general requirements which they will use develop a detailed proposal and a full set of design documentation. The proposal and documentation should demonstrate how they will meet the identified needs and will inform the development and testing of their proposed full stack solution. Learners will be required to provide evidence of the development and testing of the solution. Finally, learners will be required to provide a supported evaluation of the extent to which their solution will meet the needs of the client.

Getting started

This gives you a starting place for one way of delivering the unit, based around the recommended assessment approach in the specification.

Introduction
Depending on which units the learners have studied before this, you may wish to briefly look at the concepts of computational thinking and how they can use this to help solve problems, particularly how a larger project can be broken down and solved. Explain to learners that the unit focuses on providing a solid understanding of the key knowledge, skills and behaviours required of a software developer. They should understand that a solution is made up of many interrelated tools and technologies and even a solution such as the one they will develop could in turn be components of a much larger system. The unit should provide learners with skills and competencies to successfully enter the workplace in a junior development role or allow them to progress to the study of computer science at a higher level. Where possible incorporate practical activities to develop hands on skills that learners can apply to a range of contexts.
Learning aim A: Explore tools and technologies for full stack development
• Learning aim A should give learners an understating of current tools and technologies, available to developers, and how they are used as part a 'stack'. • You could start by looking at some well-known computing platforms such as Facebook, Amazon and Uber. Start by looking in broad terms at the different types of processes and technologies involved and identifying where front-end and back-end processes work and interact with each other. Through guided discussions, get learner to think about why these processes are divided in to different parts of the stack and how this aids users and developers. • Provide learners with opportunities to interact with full-stack deployed solutions, in which they can explore the way in which they work. You may have to do some initial preparation and build some examples for learners to explore. However, these can be quite simple in terms of their functionality, and may consist of only 1 or 2 functions, but they should demonstrate how front-end and back-end processes pass data between them and how they are each reliant on the other. Providing these 'bare-bones' examples is also

a good way to de-mystify the process and can make the subject matter less daunting for learners. This is a also a good way to introduce the learner to the practical aspects and move beyond the theoretical. • Where possible provide links to practical tasks. For example, looking at the different tools provides opportunities to start to explore the practical skills that will be used in Learning Aims B and C. • Some of the content can often feel quite theoretical in nature, particularly when considering impact of the technologies. In these cases, still try to provide opportunities for learners to engage in a more practical way. For example, where time and resources allow, providing learners with access to systems that they can interact with will greatly understand how these solutions would impact users.
Learning aim B: Design a full stack solution for an identified client
• Start by recapping skills and knowledge developed in Unit 4, reminding learners of the skills needed to decompose a problem scenario and develop correct designs. • Give learners plenty of opportunities to decompose problems and create designs for solutions using appropriate design documentation. • Spend time exploring the concepts of front-end and back-end design, considering how the requirements for data are related to expected outcomes for the user. • Spend some time working on the concept of abstraction and how software, wherever possible, hides the complexities of the data and processes in the back-end. • After some tutor-led input into conventions for designing software solutions, you could provide learners with opportunities to apply these skills to given scenarios. Initially it can be helpful to separate aspects of the design process, for example, you could provide the learners with already decomposed problems ,and a defined sets of requirements, from which they can design just the algorithms. • Once they are comfortable with the individual stages then provide them with opportunities to practice the whole process. • The quality and clarity of design documents is very important to ensuring a final solution is appropriate. Provide learners with examples of good and not so good design documents that they can analyse. You could also provide learners with a completed system that they have to 'reverse engineer'. In these types of activities, the learner must use a completed system to

produce what they think the design documents might have looked like. Reverse engineering tasks can help learners more clearly see the link between design and final product. • It can also be helpful to link this learning aim with Learning Aim C. Once learners have sufficient practical skills, you could provide opportunities for them to create systems from prewritten design documentation. Varying the quality of the documents you present them allows learners to see issues that can arise if design documentation is not of sufficient quality.
Learning aim C: Develop a full stack solution for an identified client
• Much of the learning for this learning aim should be in the form of practical tasks. Provide opportunities for learners to spend time developing coding skills for back-end and front-end solutions. • You could start by building on the learners' previous experience of coding, which is likely to be limited to simple front-end coding and scripting. While building the complexity of the problems they solve in the front end, introduce back-end concepts such as pulling data from relatively simple file use cases (such a .csv or JSON) and slowly build up to the use of more complex data structures and back-end programming such as SQL. • Getting the learner to build on exiting code bases can help develop understating more quickly. Providing learners with solutions that have limited functionality, to which they should add additional features, will help develop understand of larger scale systems. • As learners grow in confidence you can provide less scaffolding and get learner to produce the more and more program code themselves. Time will need to be spent introducing use of more advanced libraries and frameworks, to add greater functionality to the learners' solutions. This can be achieved through a combination of tutor-led practical demonstrations, self-directed online learning and practical challenges • Provide opportunities for learners to develop solutions for a range of scenarios and different users. • Learners should be encouraged to move beyond simplistic and basic solutions. They should be remined that this unit should prepare them for becoming a computing professional and they should strive for high quality solutions. Provide the learners with opportunities to consider how ensure given opportunities to consider how they can ensure that their software is efficient, readable, maintainable and robust.

- Following tutor led input, you could provide opportunities for leaners to work collaboratively, e.g. testing and debugging each other's code, parried code reviews, paired programming. You could then get learners to identify improvements that could be made both in terms of the solution itself and the processes the learner followed.
- Provide learners with examples of good and not so good code, exploring the factors that lead to high quality software solutions.
- You should emphasise the concept of testing as an integral part of the development process and not as a separate isolated task.
- You could introduce specific types and methods of testing by providing learners with example projects/solutions in various stages of completion and with various levels of functionality. Learners could test the solutions against a defined set of requirements and document the results. For solutions that lack functionality, the learners could use their testing to identify issues and then perform the corrective actions which can then be recorded.
- Explore how the definition of requirements, and performance indicators etc at design stage should influence the testing and review process, to ensure that the solution is being tested both for functionality and the extent to which it meets the needs of the users.
- As learners' familiarity with the testing and debugging solutions increases, provide more complicated problems and less obvious issues.

Details of links to other BTEC units and qualifications, and to other relevant units/qualifications

This unit links to:

- Unit 4: Programming.
- Unit 13: Software Testing.

Videos

https://www.youtube.com/watch?v=cGwSm8xDSwI Backend Development – PHP and My SQL tutorial

https://www.youtube.com/c/Freecodecamp

A collection of tutorials and range of programming languages – Front end and back end

Websites

https://www.w3schools.com/

Programming tutorials for wide range of languages

https://www.codecademy.com/

Online programming courses in a range of languages

https://cloud.google.com/docs/open-tutorials

Google cloud platform tutorials and documentation

https://aws.amazon.com/education/awseducate/

Amazon Web Services – AWS educate

Platform for learners to gain free training materials and cloud-based resources

Pearson is not responsible for the content of any external internet sites. It is essential for tutors to preview each website before using it in class so as to ensure that the URL is still accurate, relevant and appropriate. We suggest that tutors bookmark useful websites and consider enabling students to access them through the school/college intranet.