Module 40 Emerging Technologies

Module title	Emerging Technologies
Module NFQ level (only if an NFQ level can be demonstrated)	8
Module number/reference	BSCH-ET
Parent programme(s)	Bachelor of Science (Honours) in Computing Science
Stage of parent programme	Award stage
Semester (semester1/semester2 if applicable)	Semester 2
Module credit units (FET/HET/ECTS)	ECTS
Module credit number of units	5
List the teaching and learning modes	Direct, Blended
Entry requirements (statement of knowledge, skill and competence)	Learners must have achieved programme entry requirements.
Pre-requisite module titles	Modules from previous stages
Co-requisite module titles	None
Is this a capstone module? (Yes or No)	No
Specification of the qualifications (academic, pedagogical and professional/occupational) and experience required of staff (staff includes workplace personnel who are responsible for learners such as apprentices, trainees and learners in clinical placements)	Qualified to as least a Bachelor of Science (Honours) level in Computer Science or equivalent and with a Certificate in Training and Education (30 ECTS at level 9 on the NFQ) or equivalent.
Maximum number of learners per centre (or instance of the module)	60
Duration of the module	One Academic Semester, 12 weeks teaching
Average (over the duration of the module) of the contact hours per week	3
Module-specific physical resources and support required per centre (or instance of the module)	One class room with capacity for 60 learners along with one computer lab with capacity for 25 learners for each group of 25 learners

Analysis of required learning effort		
	Minimum ratio teacher / learner	**Hours**
Effort while in contact with staff		
Classroom and demonstrations	1:60	18
Monitoring and small-group teaching	1:25	18
Other (specify)		
Independent Learning		
Directed e-learning		
Independent Learning		57
Other hours (worksheets and assignments)		32
Work-based learning – learning effort		
Total Effort		125

Allocation of marks (within the module)					
	Continuous assessment	**Supervised project**	**Proctored practical examination**	**Proctored written examination**	**Total**
Percentage contribution	50%	50%			100%

Module aims and objectives

The module introduces the learner to the fundamentals behind virtualization, Docker technology. They are introduced to the core concept behind how containers work and concept of continuous development and delivery with Jenkins. Learners are given practical experience of deploying containers and building clusters to serve a complete solution of a service or services.

Minimum intended module learning outcomes

On successful completion of this module, the learner will be able to:

1. Use tools and methods appropriate for large-scale development projects
2. Implement the concept and usage of Continuous Integration and Deployment (Jenkins)
3. Apply Docker technology in building containers
4. Install single and multiple dockers
5. Monitor a VM instances and Docker containers
6. Introduce Open flow, SDN, NFV

Rationale for inclusion of the module in the programme and its contribution to the overall MIPLOs

Docker technology is becoming one of the major deployment tools for middle ware due to concise size and ease of building and delivery in Agile and rapid development today, such knowledge of how Dockers work and how it is build and deployed is a key skill in any development practice in the 21st century for computer science.

Appendix 1 of the programme document maps MIPLOs to the modules through which they are delivered.

Information provided to learners about the module
Learners receive a programme handbook to include module descriptor, module learning outcomes (MIMLO), class plan, assignment briefs, assessment strategy, and reading materials.

Module content, organisation and structure

- Description of virtual machines and Dockers
- Basics of Virtual Machines
- Basics of Containers
- How CPU is Virtualized
- How Storage is Virtualized
- How Network is Virtualized
- Nested Virtualization
- Hardware Features Assisting Virtualization
- Deploying Virtual Machines
- Orchestrating Containers
- Datacenters and Virtualization
- Continuous delivery development and it is components, Jenkins in details
- Introduction to Open flow, SDN and NFV structure and implementation with Docker

Module teaching and learning (including formative assessment) strategy
The module is taught as a combination of lectures and lab sessions. The lecture sessions assist the learner in exploring the theoretical underpinnings of Docker. The practical lab sessions give learners the opportunity to implement virtual instances of Docker.

Assessment is divided into two. There are two take home assignments that build the learners skills with Docker building and deployment. Finally, there is an end of semester project that deals with problems with a cluster of containers.

Timetabling, learner effort and credit

The module is timetabled as one 1.5-hour lectures and one 1.5-hour lab per week. The number of 5 ECTS credits assigned to this module is our assessment of the amount of learner effort required. Continuous assessment spreads the learner effort to focus on small steps before integrating all steps into a realizing a robot exhibiting AI behaviours.

There are 36 contact hours made up of 12 lectures delivered over 12 weeks with classes taking place in a classroom. There are also 12 lab sessions delivered over 12 weeks taking place in a fully equipped computer lab. The learner will need 57 hours of independent effort to further develop the skills and knowledge gained through the contact hours. An additional 32 hours are set aside for learners to work on worksheets and assignments that must be completed for the module.

The team believes that 125 hours of learner effort are required by learners to achieve the MIMLOs and justify the award of 5 ECTS credits at this stage of the programme.

Work-based learning and practice-placement

There is no work based learning or practice placement involved in the module.

E-learning

The college VLE is used to disseminate notes, advice, and online resources to support the learners. The learners are also given access to Lynda.com as a resource for reference.

Module physical resource requirements

Requirements are for a classroom for 60 learners equipped with a projector, and a 25 seater computer lab for practical sessions with access to AWS is needed to run clusters of Dockers and deployment, virtual images can be also used in labs on virtual box. (this may change should more suitable technologies become available).

Reading lists and other information resources

Recommended Text

Goasguen, S., 2015, Docker Cookbook: Solutions and Examples for Building Distributed Applications, Sebastopol: O'Reilly Media

Secondary Reading:

Mouat, A. (2016) *Using Docker: Developing and Deploying Software with Containers.* Sebastopol: O'Reilly Media

Specifications for module staffing requirements

For each instance of the module, one lecturer qualified to at least Bachelor of Science (Honours) in Computer Science or equivalent, and with a Certificate in Training and Education (30 ECTS at level 9 on the NFQ) or equivalent.. Industry experience would be a benefit but is not a requirement.

Learners also benefit from the support of the programme director, programme administrator, learner representative and the Student Union and Counselling Service.

Module Assessment Strategy

The assignments constitute the overall grade achieved, and are based on each individual learner’s work. The continuous assessments provide for ongoing feedback to the learner and relates to the module curriculum.

No.	Description	MIMLOs	Weighting
1	Assignment 1 : Dockers construction and deployment	1,2,3	25%
2	Assignment 2 : Building your cluster on Google Cloud Platform	4-6	25%
3	Major Docker based project	1-9	50%

All repeat work is capped at 40%.

Sample assessment materials

Note: All assignment briefs are subject to change in order to maintain current content.

Assignment 1

You are required to build two containers using Docker. These containers can be built starting from an official base image. Official Images has a list of different official base images. You will create a client/server system with Docker on a Linux host system.

- Server :
    - This container runs a server application which will create a file of size 1KB with random text data in "/serverdata" and then transfer the file to the client along with the checksum.
    - The server application itself can be built using any language you are comfortable with. But, the container should include all the packages that are required to run your application.

- Client:
    - The client container runs an application that connects to the server, recieves the file that the server sends and saves it in "/clientdata".
    - Verify that the file is received properly at the client side by verifying the checksum.
    - The client application again can be written in any language that you are comfortable with, but the container should include all the necessary packages.

You need to create a user-defined network in Docker and run both these containers on the network created. The containers should run these applications by default (i.e, on run command). You are required to clearly state in a README file that what should be done to get into the container shell instead of running the applications by default (i.e, on run command).

Assignment 2

Your group is required to build a cluster on Google Cloud Platform and build a simple web service that require few containers. To create the cluster on Google Compute Engine, you should take the following 4 steps:

- Setup Google Compute Engine
- Install Docker
- Create your cluster using dockerized ElastiCluster
- Test your cluster with ClusterJob

You are required to work in groups and present various ideas that can be implement on this setup.

Assessment 3: Group Project

This group project is the continuation of the work completed by your group in assignment 2. You will take the base system developed in the assignment 2 and will extend it. You are required to discuss and agree the details of the extensions of the base system and the new functionalities and features. You are also required to discuss the approach that you may adopt and the technologies that may decide to sue. Your proposal should be based on the following general guidelines:

- Use Jenkins continuous and delivery Docker and connect to GitHub repository where code is changing.
- Use NoSQL database repository container for collecting analytics and data mining.

Your proposal must consist of the following 3 things:

- A detailed description of main objective of your project.
- A list of functionalities / features with a brief description of each.
- A rough timeline/Gantt chart of the project.

This is to be submitted by the group before an agreed deadline. These proposals will then be reviewed by the lecturer and assessed (complexity, feasibility, functionality). Once the project has been accepted, you can start work on it immediately and are required to submit and present it on an agreed deadline.