| | |
|---|---|
| **Unit Title** | Advanced Secure Programming (blended) |
| **FHEQ Level** | Level 6 |
| **Unit Code** | CYS20304 |
| **Credit Value** | 30 |
| **Unit Type** | Subject |

| **Learning Hours** | | | |
|---|---|---|---|
| **Staff – Student Contact Hours** | | **Independent Study Hours** | |
| Classes | 45 | Independent study | 200 |
| Supervised access to Ravensbourne resources | 15 | Preparation for assessment | 40 |
| **Total** | | 300 | |

## Unit Description

This unit develops the students' ability to design, implement and maintain moderately complex, realistically-sized programs. It builds upon the basic programming techniques introduced in first year and extends to implementation of more complex real-world programs. Examples of such systems include simulations, visualisation tools, drawing packages, database systems etc. Such systems may be implemented in the context of non-traditional computing environments such as mobile "apps".

Students will also explore vulnerabilities and issues with IoT based devices and potential security threats they may cause to the integrity and security of the system.

Secure programming is the process of effective system analysis, design and development to ensure the newly developed program is safe from cyber-attacks. This Unit will introduce you to various techniques that can be applied for secure software development and to ensure the system is not exposed to cyber-attacks.

A well specified program which is designed and well tested is less prone to cyber-attacks. Students will preferably use Object-Oriented programming languages such as Python or C++. Students will be exposed to advanced programming features which make a program efficient, effective and secure. The aim of this unit is to provide a test bed to students to develop skills in secure programming and apply them on a small project for practice.

You will develop advanced programming skills with a selected programming language.

The Five Principles underpin the Mindsets and Skillsets Manifesto and are the foundation upon which all course curriculum frameworks and unit specifications are based. The relevant Principles as stated below have been mapped against the Learning Outcomes

relevant to each course unit and at each level (see Programme Specifications for full description of the Five Principles):

1. Cultivate / Where the individual thrives.
2. Collaborate / Where disciplines evolve.
3. Integrate / Where education engages industry.
4. Advocate / Where purpose meets practice.
5. Originate / creativity meets technology.

## Unit Indicative Content

### Secure Software Development

### Industry-wide Knowledge

- Introduction to secure programming
- Effective System Analysis and Design
- Software Vulnerabilities and Prevention
- Coding Practices
- Web and Mobile Security
- Client side and server-side vulnerabilities
- Secure Software Lifecycle
- Programming Environment and UML
- Object Oriented Programming
- MVC and PHP Secure Programming
- Web Data Integrity
- Use of PHP and ASP .Net Core 2.0
- C, C++ or C#

### CyBOK knowledge areas

- Cryptography
- Software Security
- Secure Software Lifecycle

## Unit Aims

1. Apply client-side and server-side security technique
2. Design a moderately secure software system, using a suitable object-oriented design for a multi-platform environment (desktop and cloud)
3. Implement, test, and package the software for client deployment using a modern

object-oriented development tool.

4. Evaluate the effectiveness of security of the newly developed program

| Unit Learning Outcomes |
|---|
| LO 2 Concept/Ideation<br><br>Critically appraise and evaluate appropriate research materials to generate workable concepts or strategic project themes that inform and underpin project development. Related Principle: ORIGINATE |
| LO 4 (Pre) Production<br>Demonstrate systematic working knowledge, production skills, selection, application and understanding of a selection of processes, materials and methods that inform creative and academic practice.<br>Related Principle: COLLABORATE |
| LO 6 Critical and creative mindsets Evaluate a range of critical approaches in order to form an independent position<br>Related Principle: ORIGINATE |

**Learning and Teaching Methods**

This unit will be delivered using a combination of:

- Lectures / Seminars
- Online activities
- Self-directed independent study
- Peer learning, group discussion, guest speakers

| Assessment methods and tasks | |
|---|---|
| **Assessment tasks** | **Weighting (%)** *(one grade or multi-grade unit)* |
| A portfolio of practical outcomes which might include tests, experiments, research and development material and research log | 30% |
| Artefact Demonstration (12-15 minutes) | 70% |

**Indicative Assessment Criteria**

Analyse and propose a software project, taking into account client-side and server-side security, and determine which techniques might be more suitable for development approach. (LO2)

Develop the proposed solution and critically evaluate the security implications, issues and potential solutions to newly designed software system (LO6)

Apply and evaluate testing and implementation techniques for a secure software (LO4)

Analyse the outcomes of the project and propose solutions for various software vulnerabilities (LO2)

Apply best coding practices during development of your software (LO4)

Analyse the importance of cryptography and discuss potential techniques to apply on your artefact (LO2)

## Essential Reading list

The Cyber Security Body of Knowledge. (2019). 1st ed. The National Cyber Security Centre.

Oliveira, Jason de, and Michel Bruchet. Learning ASP.NET Core 2.0: Build Modern Web Apps with ASP.NET Core 2.0, MVC, and EF Core 2. Packt Publishing, 2017.

**Recommended Reading List**

Long, Fred. Java Coding Guidelines: 75 Recommendations for Reliable and Secure Programs. Addison-Wesley, 2014.

Troelsen, Andrew W., and Philip Japikse. Pro C 7 with .NET and .NET Core. Apress, 2017.