Department of Computer Science
Exchange Studies - BSc School of Computer Science

Kalen Jeffrey Wallin
Social Security Number: 781000-2997

Course		*Date*	*ECTS*	*Grade*	*Rank*
T-213-VEFF	Web-Programming	11.04.2022	6	8,5	17-36/187
T-637-GEDE	Game Engine Architecture	22.04.2022	6	8,5	14-20/37
		Total ECTS:	**12,00**		
		Average:		**8,50**	

ECTS required for Exchange Studies - BSc School of Computer Science: 180

Reykjavik 4.8.2022

Confirmed by

NEMENDASKRÁ
REGISTRY

Certificate of Attendance

Department of Computer Science
Exchange Studies - BSc School of Computer Science

Kalen Jeffrey Wallin
Soc.sec: 781000-2997

Course: T-213-VEFF	Web-Programming	
Semester: 20221	**ECTS:** 6	**Grade:** 8,5

Description

This course will introduce the basics of web applications and design patterns. The focus will be on the protocols and standards that are used in all server-side web frameworks, and OOP concepts used when implementing design patterns. Students will also learn how to improve security in websites, how to handle errors gracefully, how to write and consume web services, as well as learn about the standards and best practises in their implementation. Furthermore, students will learn how to connect to a database and other types of data storage in application code. Finally, the future of web applications and web standards will be discussed.

Course objectives:

REYKJAVIK UNIVERSITY | Menntavegi 1 | 102 Reykjavik | Sími +354 5996200 | ru@ru.is | www.ru.is

Certificate of Attendance

- This course will introduce the basics of web development. The course will cover basic network terminology to understand how web applications function. Furthermore, essentials in frontend and backend development will be covered. In addition, important topics in web development, such as web security, testing and debugging, and social role of web applications will be touched upon.

Knowledge and comprehension

- define and contrast client-side and server-side web applications
- summarise the content of HTTP requests and responses
- list different HTTP methods and explain their purpose
- explain the features of the different HTTP methods
- define and explain key language concepts of HTML, CSS, and JavaScript
- define accessibility for web applications and give examples for accessible/not accessible code
- predict the behaviour and look of a web application based on its source code
- predict the behaviour of asynchronous JavaScript code
- discuss web application testing and contrast different testing techniques
- summarise the different constraints of REST in relation to RESTful APIs
- discuss the correctness of HTTP response status codes for different endpoints of a RESTful API
- list and explain the most important web security threats according to the OWASP TOP 10

???????**Application and analysis**

- develop basic client-side web applications using HTML, CSS, and JavaScript
- make use of AJAX to enrich web applications with asynchronous behaviour
- debug and test basic client-side web applications
- analyse web application source code for errors
- choose the correct HTTP request method for different REST endpoints
- build a RESTful backend application
- make use of a database to persist data in a backend application
- analyse an existing RESTful API and point out shortcomings
- deploy a server-side application to an online cloud provider
- test and debug server-side applications
- develop tests for common web security threats
- inspect web application source code for potential security threats

???????**Synthesis and evaluation**

- propose improvements to web application source code
- improve existing web application source code
- assess existing code for errors and security vulnerabilities
- compare different testing techniques for web applications
- design a RESTful API according to given requirements
- convert a backend API so that it conforms to the REST style
- debate the importance of testing and debugging for web application development
- debate the societal role of web applications and the ethical impact this has on web development

Assessment methods:

Course: T-637-GEDE	Game Engine Architecture	
Semester: 20221	**ECTS:** 6	**Grade:** 8,5

Description

Certificate of Attendance

The course covers the theory and practice of game engine software development, bringing together topics that range from large-scale software architectures and modern game programming paradigms to the design and implementation of subsystems for memory management, interface devices, resource management, rendering, collision, physics and animation. Through practical lab exercises and group projects, the students will get technical hands-on experience in C++ game development, including the use and development of supporting tool pipelines.

Course objectives:

- Be able to explain game engines and their role in game development.
- Be able to sketch the typical components of a run-time game architecture.
- Be able to explain programming paradigms and data structures that are commonly used in game development
- Be able to understand what goes on in the rendering pipeline.
- Be able to explain engine sub-systems that deal with start-up/shut-down, memory management, engine configuration, file system, game resources, game loop, rendering loop and interface devices
- Be able to explain game engines and their role in game development.
- Be able to use and extend a C++ graphics engine to develop tech demos.
- Be able to use industry standard C++ development and version control tools.
- Be able to apply 3D math, covering points, vectors and matrices, for solving game world problems.Be able to import resources from Digital Content Creation tools.
- Be able to read input from game interface devices.
- Be able to program a basic vertex and fragment shader.Be able to use a particle system to create visual effects.
- Be able to use a physics library for realistic object behavior.
- Be able to analyze and compare existing game engines with respect to game development goals and system requirements.
- Be able to research, design, implement and present a tech demo of a low-level engine feature.
- Be able to design new game engines or engine sub-systems, based on established practices and an insight into various architectural decisions (pros and cons).

Assessment methods:

Calculated average grade: 8,50

Total 12 credits of 180

REYKJAVIK UNIVERSITY | Menntavegi 1 | 102 Reykjavik | Sími +354 5996200 | ru@ru.is | www.ru.is

Certificate of Attendance

REYKJAVIK UNIVERSITY | Menntavegi 1 | 102 Reykjavik | Sími +354 5996200 | ru@ru.is | www.ru.is