CKCS145 Full Stack Developer - Sample Course Outline

N.B. If this stand-alone course is delivered virtually, it will have a low resource threshold.

Courses at Ryerson consist of 12 3-hour teaching modules plus a final exam (so 13 weeks). Course developers may use this outline template to complete 8 to 10 content topics around which to build the course. As a guideline, each topic should have one or more Learning Objectives. These objectives should have supporting Teaching Strategies and Assessment(s) which trace back to the overall objectives of the module.

In order to ensure compliance with Ryerson University's Course Management Policy, all Course Outlines must follow the layout in this template. The Senate Course Management Policy specifies that any changes made to the Course Outline after course start-up must be discussed with and communicated to the students in your class. Please do so via the Announcements area of your course in Brightspace by D2L. In addition, the Outline should include:

- any course prerequisites and exclusions (remember, these are the same Outlines that prospective students see);
- any instructions or limitations on student use of e-mail to contact faculty (e.g.: "Students must use the Messages tool, NOT e-mail, to contact their instructor")
- an indication of when the first assessment will be returned to students (so that they can make an informed decision about remaining in the course or withdrawing from it, prior to deadline).

The following competencies framework has been used for this project: ESCO – European Skills/Competencies, Qualifications and Occupations. These ESCO skills/competencies are mapped to this project. For the Full Stack Developer Course and Assessment, the competency mapping is as follows:

Working with Computers:

- S5.0 - working with computers
- S5.1 - programming computer systems
- S5.5 - accessing and analysing digital data
- S5.6 - using digital tools for collaboration, content creation and problem solving

Information Skills:

- S2.7 Analyzing and Evaluating Data

AGENDA

The following section documents course details as required by the Senate Course Management Policy. Please complete the details for all sections.

Course Description

Excel in Full Stack developers' tools Git, Github and the Unix Shell. Distribute via Linux server to a virtual machine in order to host and to install updates to your web applications. Combine HTTP and key networking skills with DNS, NAT, IPv6, latency and bandwidth. Problem-solve with tcpdump. Build with JavaScript, SQL and Python multi-user web applications on Google. Leverage external web APIs to perform asynchronous browser updates and to employ the jQuery JavaScript library to handle API responses. Note: This course conducts virtual labs in AZURE.

Course Focus and Scope

Full stack web development covers all the layers of web development including: DevOps, front end development, backend development, and databases. The core approach that will be followed is called the “12 Factor App”. Core topics will include JavaScript, SQL, Python, Continuous Integration (CI)/Continuous Delivery (CD), RESTful architecture and HTTP, exposing and consuming APIs, domain modeling and database design, web frameworks, and modern JavaScript user interfaces.

Course Learning Outcomes

Upon the successful completion of this course, students should be able to:

- Utilize Python application programming interfaces (APIs)
- Create RESTful services supported by Python, Flask, and SQL Alchemy
- Create a rich modern UI with a responsive web design that supports a variety of devices.
- Apply the techniques necessary for basic web development
- Build and deploy a fully functional web application

Teaching Methods

The teaching and learning strategies in this course include:

- core textbook and web-based readings
- videos
- weekly labs
- hands-on practical experience in the virtual computer lab
- online discussions

Course Schedule

Module 1: Introduction to Full Stack Development

Topic(s):

- Choosing a code editor
- Linux shell basics
    - man pages and --help
- Using JavaScript and Python virtual environments
- Creating a “Hello World” web application
- Running a local web server for application development/testing

Learning Objectives:

By the end of the module, students should be able to:

- Create and run a JavaScript/Python/Flask web application
- Run a web server locally on a Linux machine

Readings & Resources:

- Grinberg, M. (2018). *Flask Web Development: Developing Web Applications with Python* (2nd ed.). O'Reilly Media, Inc.
    - Chapter 1: Installation
    - Chapter 18: Additional Resources
- Reitz, K., & Real Python. (2022). The Hitchhiker’s Guide to Python: Zen of Python.
- Thoughtbot, Inc. Mastering the Shell.

Graded Assessments:

- Lab 1: Creating a web application with /Python/Flask (5% of the final grade)

Module 2: Version Control Using Git

Topic(s):

- Committing to version control with Git
- Basic Git commands
    - Committing
    - Branching
    - Reading diffs
- Git configuration
- Pushing / Pulling committed code to GitHub
- GitHub pull requests / pull request review

Learning Objectives:

By the end of the module, students should be able to:

- Develop a working knowledge of Git and GitHub
- Commit a web application to version control (local and remote)

- Write good Git commit messages

Readings & Resources:

- Github Git cheat sheet
- 5 Useful tips for a better commit message
- Pope, T. (2008). A note about Git commit messages. tbaggery.
- Thoughtbot, Inc. Mastering Git

Graded Assessments:

- Lab 2: Opening your first pull request and code review (5% of the final grade)

Module 3: Web Development

Topic(s):

- The “DevOps movement” and its importance for modern web development
    - Continuous integration
    - Continuous delivery
- Managing multiple environments for your application
- Dev/Prod (Developement/Productivity) parity
- Deploying a Python/Flask web application to Heroku

Learning Objectives:

By the end of the module, students should be able to:

- Create, commit, and deploy a web application to the cloud

Readings & Resources:

- Grinberg, M. (2018). *Flask Web Development: Developing Web Applications with Python* (2nd ed.). O'Reilly Media, Inc.
    - Chapter 17: Deployment
- Wiggins, A. (2017). Dev/prod parity. The Twelve-Factor App.
- Humble, J. (2017). Patterns. Continuous Delivery.

Graded Assessments:

- Lab 3: Deploying the application (5% of the final grade)

Module 4: HTTP Requests and Endpoints

Topic(s):

- HTTP
    - GET, POST, PUT/PATCH, DELTE, HEAD
    - Response codes: 1xx, 2xx, 3xx, 4xx, 5xx
- RESTful architectures and CRUD
- Web application URLs and routing
- The request-response cycle
- RESTful APIs

    - Exposing an API endpoint

Learning Objectives:

By the end of the module, students should be able to:

- Create a web form that makes an HTTP POST request to the application
- Respond to an HTTP POST with success/failure messaging
- Utilize the 6 core HTTP verbs (GET, POST, PUT/PATCH, DELETE, HEAD)
- Apply HTTP verbs to map to web applications (CREATE, RETRIEVE, UPDATE, DELETE)

Readings & Resources:

- Grinberg, M. (2018). *Flask Web Development: Developing Web Applications with Python* (2nd ed.). O'Reilly Media, Inc.
    - Chapter 2: Basic Application Structure
    - Chapter 14: Application Programming Interfaces
- Fielding, R. T. (2000). Chapter 5 Representational State Transfer (REST).
- Video: RailsConf 2017: In relentless pursuit of REST by Derek Prior [36:18]

Graded Assessments:

- Lab 4: Building an application skeleton with Python/Flask/SQL and API endpoints (15% of the final grade)

Module 5: The MVC (Model-View-Controller) Paradigm and Web User Interfaces

Topic(s):

- Rendering static files
- Templates (Jinja)
- Web forms (HTML / CSS)
- Using a CSS framework (Bootstrap)

Learning Objectives:

By the end of the module, students should be able to:

- Render both dynamic and static web pages with Flask and Jinja
- Create forms that make HTTP POST requests and handle success/failure feedback
- Leverage a CSS framework for rapid development

Readings & Resources:

- Grinberg, M. (2018). *Flask Web Development: Developing Web Applications with Python* (2nd ed.). O'Reilly Media, Inc.
    - Chapter 3: Templates
    - Chapter 4: Web Forms
- Bootstrap Documentation

Graded Assessments:

- Lab 5: Creating static mockups with HTML/CSS (10% of the final grade)

Module 6: Database Usage

Topic(s):

- Domain modelling
- Postgresql basics
- Database design basics
- Working with an ORM (SQLAlchemy)
- Database migrations

Learning Objectives:

By the end of the module, students should be able to:

- Store and retrieve data from a database
- Gain a cursory understanding of the 3NF (usage of normalization principles to reduce data duplication and data anomalies)
- Describe the major associations (belongs to/has one/has many)
- Write SQL queries (with/without code)

Readings & Resources:

- Grinberg, M. (2018). *Flask Web Development: Developing Web Applications with Python* (2nd ed.). O'Reilly Media, Inc.
    - Chapter 5: Databases
    - Chapter 12: Followers
- PostgreSQL. (2022). About PostgreSQL.

Graded Assessments:

- Lab 6: Integrating static mockups with dynamic data (10% of the final grade)

Module 7: JavaScript: Part 1

Topic(s):

- User Interfaces with JavaScript and React.js
- React components
- Development environments for JavaScript web applications

Learning Objectives:

By the end of the module, students should be able to:

- Explain the difference between a server rendered and a client rendered application
- Examine use cases for a Single Page Application (SPA)
- Be familiar with JavaScript, React.js, and the related ecosystem
- Create a front end application with JavaScript (React.js)
- Build React components

- Run a JavaScript web application on a local Linux server

Readings & Resources:

- React Documentation (2022)
  - Create a new react app
  - Thinking In React
- Facebook. (2022). Create react app - Getting started.
- Github. (2022). Create react app.

Graded Assessments:

- Lab 7: Mocking up a rich UI with JavaScript (10% of the final grade)

Module 8: JavaScript: Part 2

Topic(s):

- JavaScript frameworks
- Deployment of JavaScript web application
- Usage of API endpoints

Learning Objectives:

By the end of the module, students should be able to:

- Analyze the front end and the back end interact over HTTP
- Describe distributed applications
- Use jQuery for making HTTP requests
- Consume a Flask-based API endpoint from a JavaScript web application

Readings & Resources:

- React Documentation (2022)
  - Advanced Guides
  - AJAX and APIs
- jQuery

Graded Assessments:

- Lab 8: Integrating the rich UI with JavaScript and RESTful APIs (10% of the final grade)

Module 9: Web Services and Data Consumption

Topic(s):

- Consuming a third-party API with Python
  - HTTP GET requests to a publicly available API
  - Parsing JSON or XML
- Web harvesting with Python
  - Extracting data from public web pages
  - Parsing HTML or XML

- Opening Data Canada

Learning Objectives:

By the end of the module, students should be able to:

- Get data from an API using the Requests package
- Describe the difference between consuming a first-party API and a third-party API
- Analyze web harvesting or web data extraction and open data consumption from different formats

Readings & Resources:

- Ronquillo, A. (2022). Python's Requests library (Guide). Real Python.
- Video: Python Requests tutorial - How to call a weather API [11:28]
- Reitz, K., & Real Python. (2022). The Hitchhiker's guide to Python: HTML Scraping.

Graded Assessments:

- Lab 9: Pulling in and storing real world data (10% of the final grade)

Module 10: Debugging and Testing Applications

Topic(s):

- TDD (Test Driven Development)
- Red-Green-Refactor Cycle
- Setting up CircleCI for continuous integration
- The Test Pyramid

Learning Objectives:

By the end of the module, students should be able to:

- Apply the basic steps to test a web application
- Explain why testing web applications is important
- Set up continuous integration with CircleCI and continuous delivery with CircleCI and Heroku

Readings & Resources:

- Grinberg, M. (2018). *Flask Web Development: Developing Web Applications with Python* (2nd ed.). O'Reilly Media, Inc.
    - Chapter 15: Testing
- Facebook. (2022). Jest - Testing react apps.
- Thoughtbot, Inc. (2022). Fundamentals of TDD.
- Fowler, M. (2018). The Practical Test Pyramid.

Graded Assessments:

- Lab 10: Basic automated testing (5% of the final grade)

Module 11: Security

Topic(s):

- Domain names and DNS (domain name system)
    - Setting up a free domain with Namecheap
    - Configuring the domain with Heroku
- SSL (HTTPS)
    - Setting up a free SSL with let's encrypt
    - Configuring the SSL with Heroku
- NAT (Network address translation) / IPv4 and IPv6
- Problem-solve with tcpdump

Learning Objectives:

By the end of the module, students should be able to:

- Set up and configure a domain name to point to the web application
- Set up and configure SSL for that domain name
- Develop basic knowledge of NAT and IPv6

Readings & Resources:

- Heroku Dev Center (2022)
    - Custom domain names for apps
    - Heroku SSL
- Wikipedia. (2022). Network address translation.
- Wikipedia. (2022). IPv6.
- Tcpdump. (2022). Man page of tcpdump.

Graded Assessments:

- Lab 11: Adding a domain name & SSL certificate (5% of the final grade)

Module 12: Performance Optimization

Topic(s):

- Performance
    - Backend: databases / code quality
    - Front end: latency / bandwidth / page speed
        - Using Google's page speed insights
    - Debugging with Chrome console / developer tools
- Optimization

Learning Objectives:

By the end of the module, students should be able to:

- Identify pros and cons of premature optimization and decide when to optimize and when not to optimize
- Find the highest leverage places to optimize
- Use the network tab of the Chrome console

- Apply extreme programming principles such as You Ain't Gonna Need It (YAGNI)

Readings & Resources:

- Grinberg, M. (2018). *Flask Web Development: Developing Web Applications with Python* (2nd ed.). O'Reilly Media, Inc.
    - Chapter 16: Performance
- Video: Mike Müller - Faster Python programs - Measure, don't guess - PyCon 2019 [3:18:00] (Note: Students are not required to watch the entire video; only watch clips pertaining to the project undertaken)
- Video: Ned Batchelder - Big-O: How code slows as data grows - PyCon 2018 [28:50]
- Video: Matt Davis - Python performance investigation by example - PyCon 2018 [30:35]

Graded Assessments:

- Lab 12: Optimizing your application (5% of the final grade)

Module 13: Project Presentations & Course Wrap-up

Graded Assessments:

- Lightning-Talk Presentations (5% of the final grade)

Course Text and Materials

Required Textbook(s)
Grinberg, M. (2018). *Flask web development: Developing web applications with Python* (2nd ed.). O'Reilly Media, Inc.
ISBN: 9781491991725

Recommended Readings or Resources

EVALUATION

Marking Scheme

Assessment	Course Weight %	Week Assigned	Week Due
Lab 1: Creating a web application with Python/Flask/SQLAlchemy	5	1	1
Lab 2: Opening your first pull request and code review	5	2	2
Lab 3: Deploying the application	5	3	3
Lab 4: Building an application skeleton with Python/Flask/SQL and API endpoints	15	4	4
Lab 5: Creating static mockups with HTML/CSS	10	5	5
Lab 6: Integrating static mockups with dynamic data	10	6	6
Lab 7: Mocking up a rich UI with JavaScript	10	7	7
Lab 8: Integrating the rich UI with JavaScript and RESTful APIs	10	8	8
Lab 9: Pulling in and storing real world data	10	9	9
Lab 10: Basic automated testing	5	10	10
Lab 11: Adding a domain name and SSL certificate	5	11	11
Lab 12: Optimizing your application	5	12	12
Lightning-Talk Presentations	5	13	13
Total	**100%**		

Assignment Descriptions

Labs

Throughout the course, students will build a complete web application. The full development of the application will be broken into 12 labs to be completed in class. Students will be able to select one of three predefined application ideas (as defined by the Instructor) or propose their own application idea (approval at the discretion of the Instructor).

Lightning-Talk Presentations

Lightning-Talk Presentations will occur in the final class. Students will have the opportunity to present several features of their web application in the presence of their instructor, peers, and employers. These presentations are approximately 10 to 15 minutes followed by a Q & A period.

Late Assignments

Most assignments build on top of the work of previous weeks, so missed assignments will have to be made up before the next class. Falling behind on the progress of assignments will affect the results of your project.

Participation Details

The online discussion board is an excellent way to enhance your learning and practice critical thinking. Discussing content in an online environment allows you to reflect before contributing and take time to consider other student postings. By providing opportunities for networking and community building, the discussion board can reduce the feeling of isolation that sometimes occurs in online courses.

From Ryerson University Policy

Ryerson University is a learning, teaching, and work community of students, faculty and staff, committed to providing a civil and safe environment which is respectful of the rights, responsibilities, well-being and dignity of all of its members.

Etiquette Guidelines

- Treat online forums as academic, public-speaking places. Post comments in the same way you would speak in a traditional classroom—politely and respectfully. Forums are a place for discussion and debate about the content you are studying. They are a way of getting to know the abilities and strengths of your peers and instructor(s) and an opportunity to share your views and ideas.
- Respect diversity. There will be multiple perspectives and experiences shared relating to course content and subject matter practice. You may disagree with someone's perspective or have a different one, but positioning any perspective as "right" or "wrong" should be avoided.
- Read and respond to peer postings. If someone comments on your thread or asks a question, monitor and reply.
- Keep criticism constructive and positive. Reference course readings and content to make suggestions or recommendations.
- Participate frequently. You may be assessed for attendance and participation via weekly forums.
- Be concise. You, your instructor, and your peers have many posts to read each week. Unless your instructor states otherwise, keep your initial postings and responses brief and meaningful (one to two short paragraphs) and include references and links.

Issues Awareness

- Discussion forums can sometimes move off topic; avoid tangents and assist with redirection to keep postings contextual.
- The instructor is the course expert and will address any incorrect information in forums with guidance and support as needed.
- Inappropriate forum behaviour should be reported to the instructor immediately. Allow the instructor time to respond and take action. Do not engage an inappropriate peer directly.
- Your instructor may provide a separate course Q & A forum. This is the ideal place to post general questions about assignments and schedules and to seek clarification on forum issues. Your peers may have similar questions, so it will benefit everyone to ask publicly. Personal issues should be communicated with your instructor outside of this forum.
- You may also have a course "coffee shop" where you can socialize with peers about non-course topics. The Etiquette Guidelines above apply to this social area and your instructor will check in to ensure that all students are using the forum appropriately.
- Your instructor may opt to form smaller groups out of the larger class to reduce the number of posts each student must read or to enable group assignments.

Missed Term Work or Examinations and Course Repeats

Missed Term Work or Examinations

Students are expected to complete all assignments, tests, and exams within the time frames and by the dates indicated in this outline. Exemption or deferral of an assignment, term test, or final examination is only permitted for a medical or personal emergency or religious observance (the request must be received within the first two weeks of the course). The instructor must be notified by email **prior to the due date or test/exam date** or as soon as possible after the date, and the appropriate documentation must be submitted. For absence on medical or religious-observance grounds, official forms may be downloaded from the Ryerson website or picked up from The Chang School at Heaslip House, 297 Victoria Street, Main Floor.

Course Repeats

Senate GPA Policy prevents students from taking a course more than three times. For the complete GPA Policy, see Policy 46 on the Ryerson Senate Policies website.

Plagiarism

The Ryerson Student Code of Academic Conduct defines *plagiarism* and the sanctions against students who plagiarize. All Chang School students are strongly encouraged to go to the academic integrity website and complete the tutorial on plagiarism.

Use of the Plagiarism-Detection Service
The work submitted by students in this course will be submitted to Turnitin. Students who do not want their work submitted to this plagiarism-detection service must, by the end of the second module, consult with the instructor to make alternate arrangements.

OTHER COURSE INFORMATION

Departmental Policies and Course Practices

N/A

Specific Details on IT Requirements

Course work must be completed on a computer running a VirtualBox virtual machine. The virtual machine will be available for download in the first class.

Ryerson Student Email

All students in full- and part-time graduate and undergraduate degree programs and all continuing education students are required to activate and maintain their Ryerson online identity at ryerson.ca/accounts in order to regularly access Ryerson's email (either Rmail or Gmail), RAMSS, the my.ryerson.ca portal and learning system, and other systems by which they will receive official university communications.

Student Support

If you are experiencing technical or administrative issues with your course, help is available from Student Support for Distance Courses via email at distance@ryerson.ca or by phone from Monday to Friday, 9 a.m.–5 p.m., at (416) 979-5315.