**Advanced Computer Science II**

Performance Task

# **Create — School Store Layout**

## **Overview**

Good web design doesn't happen by accident. Now that you've gained a strong foundation in programming, this course takes a deeper look at front-end design, expanding on what is covered in Foundations. You'll learn how to design websites that look great on any device and you'll be equipped to deeply understand and create your own responsive design framework!

Students will focus on remembering the basics learned during AP CS II for website design and development. They will work as a class to design and build a responsive and complete layout for the Eagles’ Nest. Later Performance Tasks will see the students adding interactivity and a working database onto the framework they construct here.

## **Assessment**

You will be provided with 15 hours of class time to complete and submit the following:

* A video of your program running
* Written responses about your program and development process
* Program Code

Your teacher will share submission guidelines that include suggestions for creating video and PDF files.

## **General Requirements**

This performance task requires you to develop a program based on user feedback from a client. It is strongly recommended that a portion of the program involve some form of collaboration with another student in your class. Your program development process must involve a significant portion of work completed independently that requires a significant level of planning, designing, and program development.

You are required to:

* Iteratively design, implement, and test your program.
* Independently create at least one significant part of your program.
* Create a video that displays the running of your program and demonstrates its functionality.
* Write responses to questions about your program.
* Include your entire program code.

## **Program Requirements**

Your program must demonstrate a variety of capabilities and implement several different language features that, when combined, produce a result that cannot easily be accomplished without computing tools and techniques.

Your final draft must demonstrate:

* Explain how to test software for bugs and find solutions to problems they can anticipate.
* Use debugging strategies to correct errors in code.
* Document their own or their team’s computational processes when creating a program in a way that allows others to follow and understand.
* Collaborate with others to collect feedback on a digital project they or their team made, identify areas for improvement, and implement changes.
* Apply the principles of UI (user interface) design to create a digital project that balances aesthetic design with practical application.
* Define layout and design theories (Design Principles, Color Theory, Elements of Design, and Typography)
* Create a user-friendly project that meets provincial and/or other known accessibility standards and accounts for a wide range of human diversity.

## **Submission Requirements**

### 1. **Video**

Submit one video in .mp4, .wmv, .avi, or .mov format that demonstrates the running of at least one significant feature of your program. Your video must not exceed 1 minute in length and must not exceed 30MB in size.

### 2**. Written Responses**

Submit one PDF document in which you respond directly to each prompt. Clearly label your responses 2a – 2e in order. Your response to all prompts combined must not exceed 750 words, exclusive of the Program Code.

## **Program Purpose and Development**

1. Provide a written response or audio narration in your video that:

Identifies the programming language.

* Identifies the purpose of your program.
* Explains what the video illustrates.

(Approximately 150 words)

1. How have you used this program/website to meet a customer's requirements? How did you learn to get your requirements? Did the customer have a say and how the site was designed and how it functions? What feedback did you receive from the customer? How did you use feedback? (Approximately 200 words)
2. Did you use github to manage the project? How many branches exist for the project? How did you coordinate using github? How did you track different versions of your website? Does the github include a technical document? What components exist in your technical document? (Approximately 200 words)
3. What kind of data Structures did you use in the background of your website? How did you use different modules/libraries in your website? How did you break down your code into smaller parts? How did you reuse code in your website without copy and pasting it? How did you adapt existing structure for your website without re-inventing the wheel? (Approximately 200 words)
4. Capture and paste your entire program code into the PDF.

* Include comments or citations for program code that has been written by someone else.

## Tasks

### [**Activity 1 - Introduction to JavaScript**](https://docs.google.com/presentation/d/1bsP3iC7UZJodeGJcpu9i71OWXtCwEcag4FMD4g0SIp0/edit?usp=sharing)

**Description**

Welcome to the JavaScript course! Start here! This series digs into the things you need to write larger and larger applications with JavaScript. This is where it gets real!

Time To Complete: 10 - 12 Hours

### [**Activity 2 - JavaScript in the Real World**](https://docs.google.com/presentation/d/1YvRPKXIiYE4vhLADD0c_ETqb-cabmOOKN6nW5IK18Oc/edit?usp=sharing)

**Description**

Let's look at a few more practical applications of JavaScript and learn about a few useful tools that are widely used in the industry.

Time To Complete: 2 - 3 Hours

### [**Activity 3 - Asynchronous JavaScript and APIs**](https://docs.google.com/presentation/d/17YFeeGyUifUOpuEs1ZTXN3xafJIj22XOIdgOmjNuxs0/edit?usp=sharing)

**Description**

This section explores asynchronous JavaScript and event loops, and how it's useful in fetching data from web servers using APIs.

Time To Complete: 4 - 5 Hours

### [**Activity 4 - Testing JavaScript**](https://docs.google.com/presentation/d/1IIpDhvM-rdkPjX6AZe7exgxE8HBm0i-kkZS-kWWaPSo/edit?usp=sharing)

**Description**

Test driven development is an important skill in today's dev world. This section digs into the details of writing automated JavaScript tests.

Time To Complete: 4 - 5 Hours

### [**Activity 5 - React JS**](https://docs.google.com/presentation/d/1ov_f9n_0eRbG9-YtQ9Julc7QE7cv_NaegIzqpua62WM/edit?usp=sharing)

**Description**

In this section you will learn the basics of the most popular frontend framework, React JS. Students will develop a prototype shopping page in a group.

Time To Complete: 7-8 Hours

### [**Activity 6 - JavaScript and the Backend**](https://docs.google.com/presentation/d/1FMa4osnch-ihNaMbby9FdXG312Z1KBRfZ89r2ohPWVU/edit?usp=sharing)

**Description**

A real web app needs a back end in order to persist its data and do sensitive operations. Here you'll learn how to use ajax to send data requests to your back end or how to outsource your backend to a Backend-as-a-Service company like Firebase.

Students will have the opportunity to test and modify their shopping cart app. Based on alpha test and generic feedback.

Time To Complete: 1-2 Hours

### **Activity 7 - Share**

**Description**

Students will share their shopping cart/school store app with the marketing students and get feedback from them on the functionality and design of the overall app. They will record/document their feedback.

Time To Complete: 1-2 Hours

### [**Activity 8 - Reflect**](#_vdvycs9tc3jl)

**Description**

Students will reflect on the fundamentals of programming and how it reflects all they learned throughout this project.

Time To Complete: 1 Hour