WDD 330 Syllabus

Description

This course is designed to give students the skills required to create web applications using HTML, CSS , and JavaScript. It is intended to help the student learn to do this without the aid of third-party frameworks or libraries. Because of this, the course focuses on how to solve larger, ill-structured business problems by designing and creating web applications.

Learning Model Architecture

Prepare:

Each week you will work directly with your team. You help your team become successful if you come prepared for these team meetings. This preparation includes thinking about the technologies you are exploring, completing the assignments given during the previous team meeting, and doing the necessary research to plan the next steps for your team. Just as in life, if you do more than fulfill the minimum requirements assigned to you, you will be more successful in your team and in the class. Magnify your professional calling.

Teach One Another:

Team meetings and a collaborative environment have been established to enable each of you to draw from the strengths of others so that your weaknesses can become strengths. This requires effort on the part of both the knower and the learner so that 'both [may be] edified' (see Doctrine and Covenants 50:22).

Ponder/Prove

Pondering is integral to success in both this course and life. You should be pondering and reflecting not only on what you are learning but how you are learning it. Self-understanding is vital to professional life. It is also fundamental to our eternal exaltation.

Proving is also integral to success. You should prove what you think you know through experimentation. Just finding an example of some principle

on the web or from your team is insufficient for successful learning. You must be playing with and generating your own examples to understand how technologies work and what they do. This way you become a blessing to yourself and your current and future team members.

Standards

Be responsible for your own education. It is important that you prepare yourself each week to contribute actively in learning with your fellow students. Be respectful of each other’s time and be prompt to any meetings that may be scheduled.

NODE and NPM

Node is a JavaScript runtime that has gained a lot of popularity in the last few years. There are many helpful tools and utilities that can be run using it. NPM is the Node Package Manager. It gets installed automatically with Node and is used to add and remove utilities.

MANAGING THE MODERN FRONTEND WORKFLOW

Development workflow has become quite complicated for Web development. Let's take a medium-sized project for example. It could have dozens of JavaScript files, several CSS files, 3rd party libraries, and who knows how many icons, fonts, images, etc. involved. It might be using a CSS preprocessor like SASS or it could be transpiling the JavaScript to make sure that new features will work in older browsers. It's no wonder that developers have produced tools to help manage it. This activity will introduce a simple implementation of some of these tools. The tools fall into three categories:

Package managers: These keep track of all of the external dependencies for our app. This includes development tools and libraries we might be using. It not only knows which packages to download, but it tracks versions as well. We are using **npm** for our package manager.

Bundlers: Bundlers handle the compiling, transpiling, concatenating, minifying, and moving around of assets in our project. We are

using **Snowpack** as our bundler. Other common bundlers are Parcel and Webpack.

Task managers: These keep track of what needs to be done and when. There will generally be scripts defined in the task manager for each phase of development. Our project is fairly simple so we are just using **npm** again for task manager. Other common managers are **Grunt** or **Gulp**.

We will be using an NPM/Node-based workflow for this course this semester. It is important that you understand how these tools work together.

PREPARATION

Do not treat these activities as a task to check off in as little time as possible. Instead, always treat them as opportunities to learn.

Some activities and discussions will be designed to move beyond the readings...give examples and use cases, talk about potential problems, etc. Because of this, it is extremely important that you ask for clarification on parts of the readings that didn't make sense. Your instructor is happy to talk about them...but will assume everything is good unless you ask.

Tips

- Read for understanding, not completion.
- Don't skip the code examples and exercises! Make sure to spend time reviewing these.
- Take, good notes of questions that arise in your mind as you read.
- If there's a chapter in the book that's not assigned, it is not because those topics are not important...but often because you are expected to already understand those concepts from a previous course. You may want to read some of those unassigned chapters if the topics look unfamiliar to you.
- There is not a specific list of exercises given because not everyone needs the same thing.

QUESTION

What should I do if none of my fellow mate is responding to my email or message sending to them ?