Full Stack Web Development Course Outline

Prepared by Chris Brickhouse, Instructor 10/17/2017

Detailed below is a course outline with references and resources as well as diagrams and visual tools. If you have any questions contact cbrickhouse@gmail.com. Each student will be assigned a Student ID that they will use to access the various servers I will be setting up for them.

I thought we'd have a little fun with this, most kids love music so we will be making a simple music blog that also "sells" albums. We will be building an application in which a user can view real album reviews, add them to their shopping cart and check out. We will demonstrate creating login processes, front end and back end work as well as creating database tables, administering web servers and other helpful skills. By the end of the class, the student will have created a fully functioning professional web application that will look good across all devices - mobile, tablet, computer.

Nov 29th - Class 1: Intro To Web Design - HTML & CSS

1. Introduce students to each other
    a. Discuss previous tech usage and experience
    b. Familiarity with web technologies
2. Intro to browser
    a. View source
    b. F12 for tools
        i. Inspect elements
        ii. Console
        iii. Network - view requests
        iv. View cookies and app vars
        v. View DOM and globals
3. Explain flow of data from user to browser to server, and back to the user via the browser.
    a. Post
    b. Get
    c. Success vs errors
        i. 200 for success
        ii. 404 not found
        iii. 500 error
4. Brief overview of layers to a web application, including graphical representation
5. Intro to Notepad++
6. Build a simple page using Notepad++ and Browser.
    a. Console - inspect element
    b. console.log for later use
7. HTML
    a. Tag structure - <tag></tag>
        i. Self closing tags </>
    b. The Following tags will be used:
        i. Body Structure tags
            1. Html, head, body, title

        2. Styles
        3. Scripts
        4. Link tag (not a tag, internal links)
        5. Meta tags
    ii. Formatting
        1. Em (italic)
        2. Bold
        3. Blockquote
    iii. Structure
        1. Div
        2. Span
    iv. Tables
        1. Table, thead, th, tbody, tr, td.
    v. Forms
        1. Input, select, button, checkbox, radio.
    vi. Fonts
        1. Font awesome
        2. Google fonts
    vii. Importance of commenting code
        1. Later, show how to comment in each language
8. CSS
    a. Different ways to implement
        i. Element inline via "style=..."
        ii. File inline via "<style ...>" section
        iii. Separate CSS file
        iv. Pros and Cons of each method.
    b. A mention of Browser differences/incompatibility
    c. Syntax: similar to javascript
    d. CSS Selectors
        i. Tag
        ii. Class
        iii. ID
        iv. Multiple selectors + drill down, Combining Selectors such as ".my-class span"
        v. Common Naming Standards
            1. CSS Class - Lowercase separated by dashes
            2. Id's - Lowercase, underscores
        vi. Overriding styles, cascading.
    e. Colors and formatting
        i. Explain hexadecimals
        ii. Explain different units
            1. Px
            2. Em
            3. pt
    f. Center align page
    g. Margins, padding
        i. Apply top, right, bottom, left in a single line
        ii. Apply top and bottom, left and right in single line with 2 commands
    h. Backgrounds

i. Styling tables
j. Rounded corners
k. Positioning - absolute, relative, fixed
l. Display - block, inline block, none, grid, table
m. Floating divs
    i. Two column layout
    ii. Three column layout
    iii. Clear both
n. Media queries
    i. Customize styles for different screen sizes
o. Build:
    i. Horizontal navigation
    ii. Contact us form

Dec 6th - Class 2: Intro To Web Programming - Javascript

1. Different ways to implement
    a. Script src=
    b. Inline
    c. Events in tags
2. A mention of Browser differences/incompatibility
3. Syntax - curly braces, parens
4. The concept of "this".
5. Explanation of the Document Object Model (DOM)
6. Change div content
7. Hide/show content
8. String manipulation
9. Alerts
    a. Regular alert
    b. Confirm
    c. prompt
10. Loops and constructs
    a. If then else, else if.
    b. Loops
        i. For
        ii. While
        iii. Do while
    c. Switch case, break
11. Conditionals
    a. truthy
    b. coercion (== vs. ===)
    c. common mistake: assignment vs. comparison (= vs ==)
    d. Ternary Operator (this == that) ? this : that;
    e. Boolean evaluation - if (item) = if (item == true), (!item) = if (item != true)
    f. .indexOf to find occurance of string
12. Variables
    a. Scope
        i. Global

        ii. Local
    b. Object types - int, string, concat + conversions
    c. parseFloat and parseInt to add numbers
    d. Arrays
        i. Looping through arrays
        ii. Objects inside arrays
        iii. .push to add item
        iv. .split to split string into array
            1. Split comma list and loop
            2. Split space list
    e. Javascript objects inside arrays
13. Functions
    a. Parameters
    b. Optional params
    c. Return types
    d. Objects
        i. Properties
        ii. Methods
    e. Method chaining
14. Timer and interval (setTimeout/setInterval) for polling ajax
15. String manipulation
    a. .substring
    b. .indexOf
    c. Use substring for left and right, mid
16. Prototyping
    a. Add left, right, mid to string
    b. Add dateadd to date
17. Events (in tags)
    a. Onclick
    b. Onmouseover
    c. Onchange
    d. Attach event handlers in script
18. AJAX
    a. Cross browser ajax call
    b. Use text and .json files.
    c. Handle errors
19. JSON
    a. What is JSON (JavaScript Object Notation)
        i. Syntax
        ii. Validation (JSONLint)
        iii. Build:
            1. Simple json file with just one record
            2. Multiple records
20. Regex as a concept
21. Build:
    a. 99 bottles of beer to show programming constructs
    b. Form validation for contact us form previously built

Dec 13th - NO CLASS

Dec 20th - Class 3: Intro to Bootstrap

1. Build on what we learned with CSS
2. Installing bootstrap with CDN
3. Grid system
    a. Explain sizing for different screen sizes
4. Center align page user container
5. Fluid pages
6. Intro to containers, rows and columns
7. Build 2 + 3 column layout. Switch previously built CSS layout to Bootstrap.
8. Tables
9. Components
    a. Navs
        i. Mobile nav
    b. Tabs
    c. Buttons / button groups
        i. Add click events to buttons
    d. Collapse to replace hide/show
    e. Forms
    f. Modal
    g. cards
10. States
    a. Alert
    b. Info
    c. Default
    d. Warning
11. Customizing for mobile, sizing items down and rearranging for smaller screens
12. Build:
    a. - layout for admin page.
        i. Classes
        ii. Students
        iii. Orders

Jan 3rd - Class 4:- Jquery + Jquery UI

1. What is Jquery/UI
2. Installing Jquery with CDN
3. Introducing selectors and how to use with Jquery
    a. By ID
    b. By class
    c. By tag
    d. Explain differences between these and the advantages of each
4. Concept of $(this)
5. .val(), .text()
6. .prop() / .attr()

7. Document.ready - fires when document is ready to be manipulated
8. $.each - how to iterate through tags
9. Serialize form for sending. Name fields the same as model.
10. Data attributes for reusable functionality
11. e.preventDefault to stop form from submitting. Also return false on click.
12. Return false to stop function.
13. Change backgrounds
14. Events
    a. Click
    b. Change
    c. Mouseover
    d. keyup
15. .append()
    a. Add row to table
    b. Add li to list
    c. Add div to parent div
16. Exercise:
    a. Sort divs based on data attributes
17. .:first, :last - show how to get first and last and insert before and after those with .before and .after.
18. .remove()
19. $.load
    a. Load local text file
    b. Load local html
    c. Execute function upon completion
    d. Use loading gif
20. .length - use this to determine if object exists
21. .on - assign event handler to object when it is created
22. $.ajax
    a. Post
    b. Get
    c. How to return data, events
        i. Success
        ii. Error
    d. Build:
    e. Simple ajax call to .txt file, then .json.
23. Extending JQuery
24. Jquery UI
    a. Draggable
    b. Droppable
    c. Sortable
    d. Dialog
    e. Connected lists

Jan 10th - Class 5: Database - SQL

1. What is SQL (structured query language)
2. What is T-SQL (transact sql, for scripting)

3. Using SQL Management Studio
    a. SQL Express installed
    b. Connect to sql express
4. Create sql databases for all students
5. Data types
        i. Int (also increment)
            1. IDs
            2. Primary keys
            3. Foreign keys
        ii. bit
        iii. Varchar (max), Nvarchar
        iv. Decimal
        v. Float
        vi. filestream/blob
6. Build tables
    a. Instructors, admins.
    b. Students
    c. Classes
    d. Student Classes
    e. Alter table to add new column programmatically
    f. Advanced
        i. Add trigger to table
        ii. Indexes so speed up
        iii. Full text indexing
7. Data Calls
    a. Select
        i. Inner join
        ii. Outer join
        iii. Subquery
        iv. With (nolock) to avoid locks
    b. Insert
        i. Insert into select from for mass insert and data massaging
    c. Update
        i. Joined update
    d. Delete
8. Stored procs
    a. Make proc to do add and update at the same time
9. Loops
10. Cte
11. Cursors
12. Functions
13. Views
14. Exercise: lost decimals

Jan 17th - Class 6: Back End Programming, Servers, Publishing

1. Intro to FTP and publishing
2. Set up ftp site

3. Assign ten sites
4. Set up websites in IIS
    a. Application pools
        i. 32 bit support
        ii. Needs to be .net 4+
    b. Restarting sites
    c. Exploring options
    d. Custom errors
    e. Windows auth
5. Intro to Visual Studio
6. Have students sign up for VS Online if they haven't already
7. Create project in VS Online
8. Intro to .NET Core 2.0
    a. Setting up connection strings in the appsettings.json file
    b. Set up entity framework in the Startup.cs
    c. Set up bundling and minification
9. Nuget
    a. Show users how to import using the package manager console
    b. Show users how to use the graphical package manager to uninstall and reinstall
10. Basic operations
    a. New project
    b. Add to source control
    c. Save project
    d. Open project
11. Debugging and Testing
    a. Build
    b. Debug
    c. Breakpoints
    d. Stepping through code
    e. Watches
12. Add custom css and javascript to project
13. Add existing items
14. Exploring the folder structures of a web app

Jan 24th - Class 7: Back End Programming Part 1

15. Object oriented programming
    a. Using to import framework and other classes
        i. Using system.io to loop through directory
    b. What is a variable
        i. Data types
            1. Int, bool, string, etc
            2. Nullable types (?)
        ii. Lists of objects - string, int, other
    c. Logic operators and commands
        i. if/then/else/elseif
            1. Explain curly braces
            2. Explain parens and param grouping

3. Explain short circuiting logic - if first matches, don't run second
4. Explain and/or for logic

ii. Tertiary - short way of doing if logic to set variables, check var existence

iii. Loops

1. Foreach
2. For i = whatever
3. Do while
4. while

d. What is a namespace

e. What is a class

i. Creating object classes

ii. How to instantiate class (new object)

iii. How to overload instantiation to load with different data

iv. How to load data into object manually (new object{prop=prop})

f. What are properties

g. What are methods

i. Return types

1. How to call a function from another function and return data
2. Int, bool, string, etc

ii. Class access modifiers

iii. Static methods are shared

iv. Inline functions

h. Polymorphism

i. Inheritance

ii. Overrides

iii. Overloads

1. Same function name, which function is used is controlled by matching params

i. Base class has virtual methods that can be overridden or overloaded

i. Use shape analogy to display how inheritance and overriding works

j. Objects available via web server

i. Request.form

1. Iterate form fields
2. Also send as function param

ii. Request.querystring

1. Also can send as function param

iii. Request.cookie

1. Response.cookie to set

iv. Httpcontext to get current info

16. Explain the layout of an MVC Project, basics

a. App_start - filters and script setup

i. Bundleconfig

1. Setup script packages
2. Bundle different packages together for different uses, protect admin js
3. Setup CSS packages
4. BundleTable.EnableOptimizations = True

ii. Filterconfig

1. Add methods to import various filters
2. Add attributes to methods, like authorizeattribute

iii. Routeconfig
1. Set up routing for various items
2. Explain default routing - /controller/view/id/

b. App_Code - for any utilities needed
i. If it won't compile, change action to compile in properties. Defaults to content
c. Bin - add third party dlls to this
d. Content - all css content goes here
e. Images - all assets, images
f. Scripts - all javascript
g. Views - explain shared views, explain views and routing fully later,
h. Global.asax - events that are fired when application and page loads
i. Web.config - configuration file, has data sources, set up forms authentication

Jan 31st - Class 8: Back End Programming Part 2

1. MVC
a. What is MVC/VM (model/view/controller/viewmodel)
i. Models
ii. Model matches the data structure. Properties match fields.
iii. No logic in models.
iv. Use entity framework to create lists of objects, dbsets (cover more later)
b. Viewmodels
i. Viewmodels combine models to populate views
ii. Use viewmodels for almost all complex operations to provide data to the view.
iii. Use lazy loading properties so they will only be executed when needed.
iv. Viewmodels aren't officially part of the MVC framework but they come in handy when dealing with serving models to views.
c. Views
i. This is the design code. Actual logic should be limited, as most of the logic will be done with the ViewModel.
ii. Loops through properties of viewmodels to populate any needed data.
d. Controllers
i. Controllers serve as a central web service to serve JSON and HTML as well as execute save functions
ii. Used in routing to views.
iii. Used to load models and viewmodels and pass that data to the view.
1. Result types
a. JSONResult - serves lists of objects, single objects, etc
b. ContentResult - serves HTML chunks and strings
c. ViewResult - to populate view
d. PartialViewResult - partial views don't use layouts. This can also be used by Jquery .load.
2. Explain passing a model to controller
3. Explain matching javascript models to backend to pass model
iv. AuthorizeAttribute - lock down actions to secure data
v. AllowAnonymous to make public, good for home controller
vi. Use controller for autosuggest, list of objects

Feb 7th - Class 9: Entity Framework

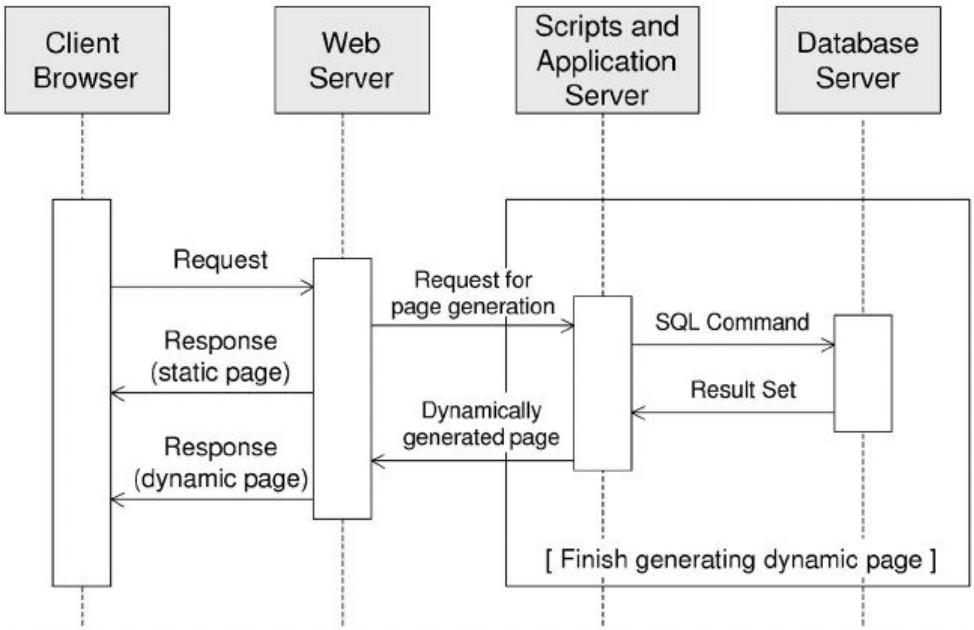
e. Set up data source in web.config
f. Set up models class, base name = data source name
g. Dbsets of all models
h. Key attribute in model
i. Table name attribute to point to table in db
j. Use view models to populate models from entity framework
    i. Single object, .find by id
    ii. List of objects from query
k. LINQ
    i. Writing queries to populate lists of models
    ii. Lambda as well as linq syntax
    iii. Joins
l. Updating data
    i. Pass model from front end and set equal to object from dbcontext
    ii. Set individual properties
    iii. Specify updated fields
    iv. Write function to add as well as update
        1. Entitystate added or modified
    v. Do.savechanges

Feb 14th - Class 10: Bringing It All Together

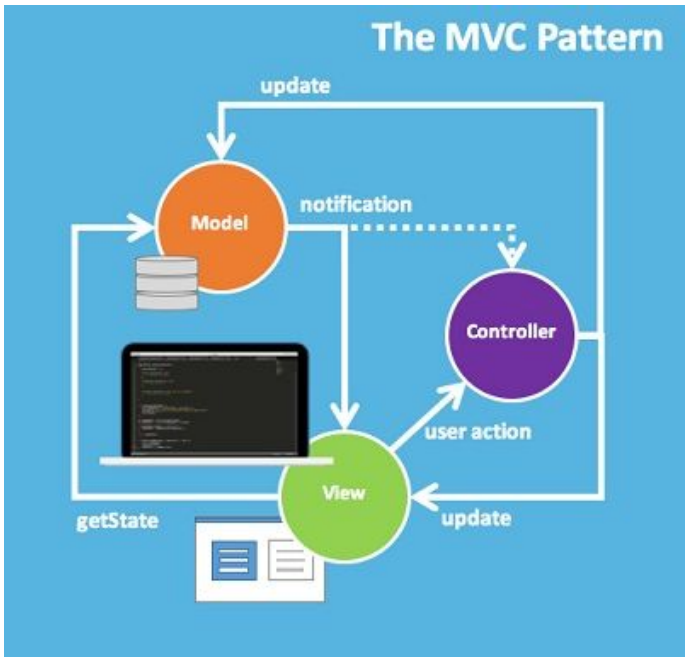
1. Gather all previously built pieces of code to finalize application
2. Copy earlier login form for admin login, forms auth
3. Logout function to clear cookies
4. Create admin section
    a. Classes
    b. Students
    c. Schedules
5. Finalize front end
    a. Create listing of classes
    b. Create class information page
    c. Allow user to add item to cart
    d. User checks out
    e. Checkout will be a single page application
    f. Course will be added to user's schedule, email sent.
    g. User can add multiple courses to cart
    h. Data Validation using javascript
6. Testing back end and front end, fixing bugs
7. Finalizing app
8. Student receives certificate of completion!

Diagrams / Visual Tools

General Web App Flow



MVC Flow



Resources / Continued Learning

All Questions

- **http://stackoverflow.com**

General

- https://scotch.io/ - Programming Tutorials

HTML/CSS

- http://htmldog.com/guides/html/beginner/
- https://www.codecademy.com/en/tracks/htmlcss
- http://csslint.net/ - CSS validation

Javascript

- https://www.codecademy.com/learn/introduction-to-javascript
- https://developer.mozilla.org/en-US/docs/Learn/Getting_started_with_the_web/JavaScript_basics
- https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Objects/Object-oriented_JS

JSON/Ajax

- http://www.json.org/
- https://jsonlint.com/ JSON Validation

Jquery/ Jquery UI

- https://jquery.com/
- https://jqueryui.com/

- https://learn.jquery.com

Bootstrap

- http://getbootstrap.com/
- http://fontawesome.io/
- https://mdbootstrap.com/bootstrap-tutorial/

C#/ASP.NET

- https://www.asp.net/mvc/overview
- http://www.learncs.org/
- https://www.codeproject.com/Articles/207797/Learn-MVC-Model-View-Controller-step-by-step-in
- https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/classes-and-structs/polymorphism

SQL

- https://mva.microsoft.com/product-training/sql-server
- https://msdn.microsoft.com/en-us/library/jj592907(v=vs.113).aspx

Entity Framework

- https://msdn.microsoft.com/en-us/library/aa937723(v=vs.113).aspx
- https://www.codeproject.com/Articles/739164/Entity-Framework-Tutorial-for-Beginners