Full Stack Web Developer Nanodegree Syllabus

Build Complex Web Applications

Before You Start

Thank you for your interest in the Full Stack Web Developer Nanodegree! In order to succeed in this program, we recommend having experience programing in HTML, CSS and Python. If you've never programmed before, or want a refresher, you can prepare for this Nanodegree with Intro to HTML & CSS and Lessons 1-4 of Intro to Computer Science.

Project 0: Launch a Web Server

In this 15-minute code dive you will create a web page and create a web server on your computer to host your web page. The purpose of this Code Dive is to familiarize you with back-end web development and to introduce you to several recurring topics.

- ➔ The client/server software architecture
- ➔ Python's SimpleHTTPServer module to run a local webserver
- ➔ HTTP concepts like port numbers and URL schemes

Project 1: Movie Trailer Website

In this project, you will write server-side code to store a list of your favorite movies, including box art imagery and a movie trailer URL. You will then serve this data as a web page allowing visitors to review their movies and watch the trailers.

Supporting Lesson Content: Programming Foundations with Python

Lesson Title	Learning Outcomes
USE FUNCTIONS	➔ Tour the Python standard library ➔ Use programming library documentation
USE CLASSES: DRAW TURTLES	➔ Use classes and objects to draw graphics
USE CLASSES: SEND TEXT	➔ Use the Twilio web API to send SMS messages

USE CLASSES: PROFANITY EDITOR	➔ Read and write to and from files ➔ Accessing web APIs with the Python urllib library
MAKE CLASSES: MOVIE WEBSITE	➔ Write programs using Object Oriented Programming (OOP) design
MAKE CLASSES: ADVANCED TOPICS	➔ Reuse code with class inheritance ➔ Customize inherited classes with method overriding

Project 2: Build a Portfolio

In this project, you will be provided with a portfolio website design mockup as a PDF-file and will replicate that design in HTML and CSS.

Supporting Lesson Content: Intro to HTML and CSS

Lesson Title	Learning Outcomes
HTML, CSS, AND BOXES	➔ Use Hypertext Markup Language to build websites ➔ Design customized website styles with Cascading Style Sheets ➔ Organize and structure websites with boxes
CSS FRAMEWORKS, RESPONSIVE LAYOUTS	➔ Build sites quickly with CSS Frameworks ➔ Create responsive designs for mobile and desktop screens
BOOTSTRAP AND OTHER FRAMEWORKS	➔ Evaluate and choose the right CSS framework ➔ Build responsive websites using Twitter's Bootstrap framework

Supporting Lesson Content: Responsive Web Design Fundamentals

Lesson Title	Learning Outcomes
WHY RESPONSIVE?	➔ Properly create web pages with mobile-first design ➔ Manage web development by using in-browser development tools ➔ Troubleshoot and debug faulty code
STARTING SMALL	➔ Build HTML elements for any size screen ➔ Use the browser viewport to create consistent user experiences
BUILDING UP	➔ Use media queries and breakpoints to create responsive web page designs ➔ Create flexible HTML elements with Flexbox
COMMON RESPONSIVE PATTERNS	➔ Explore common responsive layout designs
OPTIMIZATIONS	➔ Optimize images, tables and fonts for web sites using CSS.

Supporting Lesson Content: Responsive Images

Lesson Title	Learning Outcomes
GETTING UP AND RUNNING	➔ Use mobile developer tools to write code optimized for mobile devices ➔ Test your web sites on actual mobile devices
UNITS, FORMATS, ENVIRONMENTS	➔ Differentiate between different graphic formats ➔ Generate efficient graphics for your sites ➔ Compress large images to load quickly in mobile browsers
IMAGES WITH MARKUP	➔ Create customized background images and text using CSS ➔ Expand your CSS capabilities by using Unicode text
FULL RESPONSIVENESS	➔ Use source sets to allow devices to choose appropriately sized images ➔ Build accessible image elements using alt tags

Project 3: Multi User Blog

In this project you will be building a multi user blog (along the lines of Medium) where users can sign in and post blog posts as well as 'Like' and 'Comment' on other posts made on the blog. You will be hosting this blog on Google App Engine and you will also be creating an authentication system for users to be able to register and sign in and then create blog posts!

Supporting Lesson Content: Intro to Backend

Lesson Title	Learning Outcomes
FORMS AND INPUTS	➔ Capture user input in web forms and use data to power applications ➔ Create web forms using different input types ➔ Validate user input for use in an application ➔ Protect your application by preventing malicious input
TEMPLATES	➔ Write HTML templates to display web sites using data from a backend server.
DATABASES	➔ Store data for use by a backend server application ➔ Explore the different types of databases ➔ Write database queries using SQL and Python ➔ Harness the power of Google's Datastore for scalable data
USER ACCOUNTS AND SECURITY	➔ Set browser cookies to store data on a client machine ➔ Secure user passwords using cryptography and hash tables

Project 4: Tournament Results

In this project, you will develop a database schema to store the game matches between players. You will then write code to query this data and determine the winners of various games.

Supporting Lesson Content: Intro to Relational Databases

Lesson Title	Learning Outcomes
DATA AND TABLES	➔ Save data in durable storage using database tables ➔ Write queries to access saved data
ELEMENTS OF SQL	➔ Store data using different data types ➔ Use SQL commands to retrieve and organize data ➔ Combine SQL tables to create powerful data queries
PYTHON DB-API	➔ Interact with a database using Python code ➔ Explore PostgreSQL documentation ➔ Protect your database from malicious attacks
DEEPER INTO SQL	➔ Increase the power of databases by using constraints ➔ Normalize data to make databases more efficient ➔ Write reusable views to quickly and efficiently retrieve data
TOURNAMENT DATABASE	➔ Build a tournament organizer that uses a database to store player records, standings and generate player pairings

Project 5: Item Catalog

In this project, you will develop an application that provides a list of items within a variety of categories as well as provide a user registration and authentication system. Registered users will have the ability to post, edit and delete their own items.

Supporting Lesson Content: Full Stack Foundations

Lesson Title	Learning Outcomes
WORKING WITH C.R.U.D.	➔ Model database entries in Python ➔ Write server code to create, read, update and delete database entries interactively.
MAKING A WEB SERVER	➔ Configure a web server to handle requests using HTTP ➔ Allow a web server to read and update data based on HTTP request input
DEVELOPING WITH FRAMEWORKS	➔ Build a functioning web application using the lightweight Flask framework ➔ Respond to HTTP requests with JSON data
ITERATIVE DEVELOPMENT	➔ Plan the design of a complex web application

Supporting Lesson Content: Authentication and Authorization

Lesson Title	Learning Outcomes
AUTHENTICATION VS AUTHORIZATION	➔ Secure your application by verifying users' identities ➔ Control application authorization based on user roles and login state ➔ Use third-party systems to authenticate users
CREATING GOOGLE SIGN-IN	➔ Implement user authentication using Google's OAuth 2.0 tools
LOCAL PERMISSION SYSTEM	➔ Store user data in an application database ➔ Manage user authorization from stored user data
ADDING FACEBOOK AND OTHER PROVIDERS	➔ Implement other authentication providers in a web app

Project 6: Neighborhood Map

In this project, you will develop a single-page application featuring a map of your neighborhood or a neighborhood you would like to visit. You will then add additional functionality to this application, including: map markers to identify popular locations or places you'd like to visit, a search function to easily discover these locations, and a listview to support simple browsing of all locations. You will then research and implement third-party APIs that provide additional information about each of these locations (such as StreetView images, Wikipedia articles, Yelp reviews, etc).

Supporting Lesson Content: Intro to AJAX

Lesson Title	Learning Outcomes
REQUESTS AND APIS	➔ Connect to external web APIs to power asynchronous browser updates
BUILDING THE MOVE PLANNER APP	➔ Use the jQuery Javascript library to build AJAX requests and handle API responses ➔ Handle error responses with AJAX

Supporting Lesson Content: Javascript Design Patterns

Lesson Title	Learning Outcomes
CHANGING EXPECTATIONS	➔ React to changing product specifications and developer expectations ➔ Explore the Model-View-Controller design pattern ➔ Analyze an existing application for MVC structure
REFACTORING WITH SEPARATION OF CONCERNS	➔ Write code with discrete areas of responsibility in an MVC application ➔ Refactor an existing application to make use of modern code design practices
USING AN ORGANIZATIONAL LIBRARY	➔ Build a reactive front end application using an organizational library, knockout.js ➔ Implement knockout models and observable elements in an application
LEARNING A NEW CODEBASE	➔ Use proven strategies to adapt to a new and unfamiliar codebase

Project 7: Linux Server Configuration

In this project, you will take a baseline installation of a Linux distribution on a virtual machine and prepare it to host your web applications, to include installing updates, securing it from a number of attack vectors and installing/configuring web and database servers.

Supporting Lesson Content: Linux Command Line Basics

Lesson Title	Learning Outcomes
GET INTO THE SHELL	➔ Install your own virtual machine on your computer, running a Linux operating system ➔ Practice Linux shell and terminal commands to control your computer without a graphical interface
SHELL COMMANDS	➔ Execute Linux shell commands to interact with your computer's directory structure ➔ Add, move and delete files and folders from your filesystem ➔ Edit files in Ubuntu's built-in text editor - nano
THE LINUX FILESYSTEM	➔ Use more powerful techniques to search and navigate the file structure of a Linux computer.

Supporting Lesson Content: Configuring Linux Web Servers

Lesson Title	Learning Outcomes
INTRO TO LINUX	➔ Explore the historical roots of Linux and some common Linux distributions ➔ Launch the Ubuntu operating system in a virtual machine on your own computer
LINUX SECURITY	➔ Control authorization on a Linux system using super user privileges ➔ Install additional software packages to a Linux system ➔ Manage Linux users and user permissions ➔ Protect a Linux system with a universal firewall
WEB APPLICATION SERVERS	➔ Install an Apache web application server on a Linux system