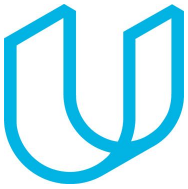
Full Stack Web Developer Nanodegree Syllabus



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 programming languages like Python and Javascript. If you've never written code before, we recommend starting with the Introduction to Programming Nanodegree Program, which will prepare you for this and other career-focused Nanodegree programs.

Prerequisites:

- ➔ You will need to be able to communicate fluently and professionally in written and spoken English.
- ➔ To enroll, you should also have experience in the following courses or skills:
  - ◆ Programming with Python or another object-oriented programming language.
  - ◆ Programming with JavaScript Data Structures including Lists, Arrays, Dictionaries. Free Course
  - ◆ Git/GitHub. Free Course
  - ◆ Introduction to HTML. Free Course

Educational Objectives:
Students will learn about building out the infrastructure that powers and supports the many web, desktop, mobile and integrated applications in the world.

Length of Program*: 160 Hours
Textbooks required: None
Instructional Tools Available: Classroom Video lectures, Study Hub with Mentors, Knowledge.

**The length is an estimation of total hours the average student may take to complete all required coursework, including lecture and project time. If you spend about 10 hours per week working through the program, you should finish in 16 weeks, so approximately 4 months. Actual hours may vary.*

PART 1

Welcome to the Program!

Welcome to the Full Stack Web Developer Nanodegree program. This is your first step on your journey to become a Full Stack Developer. Learn what this program is all about as well as how to find support along your learning journey.

Estimated time: 35 minutes

PART 2

Developer's Tools

Brush up your knowledge of essential developers' tools such as the Unix shell, Git, and Github; then apply your skills to investigate HTTP, the Web's fundamental protocol and basic networking skills like DNS, NAT, IPv6, Bandwidth, Latency and how to use tcpdump to explore the packages in the network.

Estimated time: 36 hours 45 minutes

Lesson Title	Learning Outcomes
Shell Workshop	➔ The Unix shell is a powerful tool for developers of all sorts. Get a quick introduction to the basics of using it on your computer
What is Version Control?	➔ Version control is an incredibly important part of a professional programmer's life. In this lesson, you'll learn about the benefits of version control and install the version control tool Git!
Create a Git Repo	➔ Now that you've learned the benefits of Version Control and gotten Git installed, it's time you learn how to create a repository.
Review a Repo's History	➔ Knowing how to review an existing Git repository history of commits is extremely important. You'll learn how to do just that in this lesson.
Add Commits to a Repo	➔ A repository is nothing without commits. In this lesson, you'll learn how to make commits, write descriptive commit messages, and verify the changes you're about to save to the repository.
Tagging, Branching, and Merging	➔ Being able to work on your project in isolation from other changes will multiply your productivity. You'll learn how to do this isolated development with Git's branches.

Undoing Changes	➔ Help! Disaster has struck! You don't have to worry, though, because your project is tracked in version control! You'll learn how to undo and modify changes that have been saved to the repository.
Working with Remotes	➔ You'll learn how to create remote repositories on GitHub and how to get and send changes to the remote repository.
Working on Another Developer's Repository	➔ In this lesson, you'll learn how to fork another developer's project. Collaborating with other developers can be a tricky process, so you'll learn how to contribute to a public project.
Staying In Sync With a Remote Repository	➔ You'll learn how to send suggested changes to another developer by using pull requests. You'll also learn how to use the powerful `git rebase` command to squash commits together.
Requests & Responses	➔ In this lesson, you will examine HTTP requests and responses by experimenting directly with a web server, interacting with it by hand.
The Web from Python	➔ In this lesson, you will write HTTP servers and clients in Python.
HTTP in the Real World	➔ In this lesson, you will examine a number of practical HTTP features that go beyond basic requests and responses.
From Ping to HTML	➔ Start exploring the network using low-level command-line tools such as ping and netcat.
Names and Addresses	➔ Investigate the Domain Name System (DNS), which translates hostnames into IP addresses.
Addressing and Networks	➔ Investigate the structure and use of Internet addresses, including network blocks, interfaces, network address translation (NAT), and IPv6.
Protocol Layers	➔ Use tcpdump to examine packets that make up the requests and responses for three protocols: ping, DNS, and HTTP.
Big Networks	➔ Learn more about bandwidth, latency, filtering, and other properties that matter when users are accessing your application over the Internet.

PART 3

Databases with SQL and Python

Master SQL databases and build multi-user web applications using the Flask framework, SQLAlchemy, and authentication providers such as Google and Facebook.

- Project: Logs Analysis

Estimated time: 26 days

Lesson Title	Learning Outcomes
Data and Tables	➔ Learn the principles behind relational data organization: tables, queries, aggregations, keys, and joins.
Elements of SQL	➔ Use the select statement to retrieve data from tables ➔ Use the insert statement to add data to tables ➔ Combine SQL tables using joins and aggregations to create powerful queries
Python DB-API	➔ Interact with a database from Python code ➔ Connect a Python web application to an SQL database ➔ Discover and fix security problems with database-backed apps
Deeper into SQL	➔ Create tables using normalized forms ➔ Use keys to express relationships between tables ➔ Write reusable views to quickly and efficiently retrieve data
Project: Logs Analysis	➔ In this project, you'll practice your SQL skills by building a reporting tool that summarizes data from a large database.

PART 4

Servers, Authorization, and CRUD

Learn the CRUD pattern (Create, Read, Update, Delete) and how it relates to RESTful architectures and to the operations of a database-backed web service. Learn the difference between authentication and authorization and some best practices in developing a login system.

- Project: Improve Your LinkedIn Profile
- Project: Optimize Your GitHub Profile
- Project: Project: Item Catalog

Estimated time: 50 days

Lesson Title	Learning Outcomes
Working with CRUD	➔ Learn the CRUD pattern (Create, Read, Update, Delete) and how it relates to RESTful architectures and to the operations of a database-backed web service.
Making a Web Server	➔ Build a web service in Python that uses your database to implement CRUD operations.
Developing with Frameworks	➔ Apply the Flask framework in Python to build web services more quickly and reliably.
Iterative Development	➔ Learn the basics of agile and iterative development while building a restaurant menu application.
Authentication vs Authorization	➔ Learn the difference between authentication and authorization and some best practices in developing a login system.
Creating Google Sign-in	➔ Investigate OAuth and build third-party sign-in into your web applications using Google's authentication services.
Local Permission System	➔ Use a local permission system to protect pages based on each user, not just whether that user is logged in.
Adding Facebook and Other Providers	➔ Discover additional OAuth providers and add Facebook authentication in your application, giving more choices for third-party auth.
What's and Why's of APIs	➔ Discover the foundation of Web APIs, the common systems for passing and updating information to web applications.
Accessing Published APIs	➔ Practice gathering and reading information from web APIs that already live on the internet.
Creating Your Own APIs	➔ Begin creating your own API endpoints and learn to serialize your data to package it up for transfer across the web using HTTP.
Securing Your API	➔ Control access to your APIs by limiting who can access the resources behind them and ensuring that only authorized users can read and modify data.
Writing Developer-Friendly APIs	➔ Customize your APIs to be usable and approachable by the developers who will consume them.

Project: 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.

PART 5

Deploying to Linux Servers

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 and configuring web and database servers.

- Project: Project: Linux Server Configuration

Estimated time: 28 days

Lesson Title	Learning Outcomes
Intro to Linux	➔ Gain an understanding of the Linux operating system and how it differs from other operating systems you may have experienced in the past. ➔ Launch the Ubuntu operating system in a virtual machine on your own computer
Linux Security	➔ Dive deep into Linux Security to ensure your service remains stable and free from attackers.
Web Application Servers	➔ Install all of the required software to turn your Linux server into a full-fledged web application server and host your very own application! ➔ Install an Apache web application server on a Linux system.
Project: 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 and configuring web and database servers.

Extracurricular material

PART 1

Web Accessibility

Explore the diversity of different users experience with websites and applications. Learn about using screen readers practically and recognize the challenge of building web experiences for all users.

Lesson Title	Learning Outcomes
Accessibility Overview	➔ Explore the diversity of different users experience with web sites and applications. Learn about using screen readers practically and recognize the challenge of building web experiences for all users.
Focus	➔ Manage focus - the location on a page that receives input from the keyboard. Discover how some users navigate website entirely with the keyboard, and how to optimize their experience.
Semantics Basics	➔ Dive into the differences between visual UI and semantically designed accessible UI. Add semantic elements to HTML to create a user interface that works for everyone.
Navigating Content	➔ Implement effective semantic navigation using headings, link text and landmarks.
ARIA	➔ Sometimes an HTML element may not have a role or value assigned semantically. In this lesson, you'll use ARIA attributes to provide context for screen readers.
Style	➔ Incorporate CSS styling into your accessible web design and use accessible color schemes to improve accessibility.

PART 2

JavaScript Design Patterns

React to changing product specifications and developer expectations, explore the Model-View-Controller design pattern, and analyze an existing application for MVC structure.

Lesson Title	Learning Outcomes
Changing Expectations	➔ Learn why well-structured code is vitally important to a web app's structure, especially as the app gets larger. Explore how you can use MV* organizational framework to create cleaner projects.
Refactoring with Separation of Concerns	➔ Begin refactoring your Cat Clicker code and learn the best ways to improve its structure.

PART 3

Intro to AJAX

Connect to external web APIs to power asynchronous browser updates and use the jQuery JavaScript library to build AJAX requests and handle API responses.

Lesson Title	Learning Outcomes
Requests and APIs	➔ Learn how to request data from third party APIs using jQuery's AJAX functions. Examine AJAX queries in live applications and investigate APIs you can use in your own!
Building the Move Planner App	➔ Follow along as Cameron uses The New York Times API to build a moving planner app. Learn how to handle errors and how to debug your AJAX methods.