Full Stack Web Developer Syllabus

Contact Info

While going through the program, if you have questions about anything, you can reach us at support@udacity.com. For help from Udacity Mentors and your peers visit the Udacity Classroom.

Nanodegree Program Info

The goal of the Full Stack Web Developer Nanodegree program is to equip learners with the unique skills they need to build database-backed APIs and web applications. A graduate of this program will be able to: Design and build a database for a software application Create and deploy a database-backed web API (Application Programming Interface) Secure and manage user authentication and access control for an application backend Deploy a Flask-based web application to the cloud using Docker and Kubernetes

Prerequisite Skills

A well-prepared learner is able to:

- Write and test software with Python or another object-oriented programming language
- Query a SQL database using SELECT
- Write to a SQL database using INSERT
- Fetch and display data from an API using AJAX or Fetch
- Organize data using JSON (JavaScript Object Notation)

Required Software

- Python 3.6
- pip
- GIT 2.23 or latest
- Code Editor
- PostgreSQL, Psycopg2
- Kubernetes, Kubectl
- Flask-sqlAlchemy
- AWS CLI
- EKSCTL
- Docker
- Heroku Account
- Gunicorn Webserver

- jose node
- ionic

Version: 1.0.0

Length of Program: 107 Days*

** This is a self-paced program and the length is an estimation of total hours the average student may take to complete all required coursework, including lecture and project time. Actual hours may vary.*

Part 1: Welcome to the Program!

Part 2: SQL and Data Modeling for the Web

In this part, you'll be building out the data models and database for an artist/venue booking application. The fictitious startup Fy-yur is building a website that facilitates bookings between artists who can play at venues, and venues who want to book artists.

Project: Fyyur: Artist Booking Site

In this project, you'll demonstrate your new SQL and data modeling skills by creating a site to help coordinate bookings between artists and venues.

Supporting Lessons

Lesson	Summary
Course Introduction	Understand what this course will cover and the learning objectives achieved.
Interacting with Databases	Interact with relational databases. Review SQL and the Client-Server Model. Use Postgres, understand DBAPIs, and use psycopg2.
SQLAlchemy Basics	Learn to use SQLAlchemy and SQLAlchemy ORM to work with a relational database in Python.
SQLAlchemy ORM in Depth	Get to know Model.query and the SQLAlchemy Object Lifecycle to master how to query for and change records in a database.
Build a CRUD App with SQLAlchemy - Part 1	Build out the ability to read and create todo items in our To-do app, handling changes from the database to the views.
Migrations	Handle changes to your database schema over time using a version control system involving migration files.
Build a CRUD App with SQLAlchemy ORM - Part 2	Finish developing our To-Do app with update and delete functionality. Model relationships with To-Do lists.

Part 3: API Development and Documentation

In this part, you will use the skills you've developed to build a Trivia API. The goal of this project is to use APIs to control and manage a web application using existing data models. You'll be given a set of data models and the application front end. Your task will be to implement the API in Flask to make the Trivia game functional.

Project: Trivia API

In this project, you'll use your new API and API documentation skills to build a trivia API.

Supporting Lessons

Lesson	Summary
Introduction to APIs	In this lesson, you'll learn what APIs are, how they're implemented, and why they're important. We'll also dive into the concepts of Internet Protocols and REST.
HTTP and Flask Basics	In this lesson, you'll learn about HTTP—including methods, requests, responses and status codes— as well as how to set up a Flask app, and implement and test endpoints using Curl.
Endpoints and Payloads	In this lesson, you'll learn about how Flask is extensible in itself to handle different kinds of methods, more complex endpoints, and to return formatted data to the client.
API Testing	This lesson teaches students the benefits and purposes of testing an API, Test-Driven Development for APIs and implementing the tests using unittest.
API Documentation	To round out the course, students will consider and write API documentation and project documentation so their projects can be hosted, shared, and used by other developers with clarity.

Project: Improve Your LinkedIn Profile

Find your next job or connect with industry peers on LinkedIn. Ensure your profile attracts relevant leads that will grow your professional network.

Part 4: Identity and Access Management

In this part, you will build the backend for a coffee shop application. You'll add user accounts and authentication to your application and use role-based access management strategies to control different types of user behavior in the app.

Project: Coffee Shop Full Stack

In this project, you'll demonstrate your new authentication and authorization skills by creating a full-stack application for a coffee shop menu.

Supporting Lessons

Lesson	Summary
Foundation	Set the groundwork for understanding information security and refresh your understanding of the technologies used in future lessons.
Identity and Authentication	Explore frequently used methods of identifying who is making requests on web systems. Implement modern software patterns to accomplish this goal across the stack.
Passwords	Understand and overcome common pitfalls of the ubiquitous password authentication design pattern.
Access and Authorization	Limit access to specific resources or actions by restricting requests only to authorized request to particular users and groups of users. Implement role-based access controls (RBAC) across the stack.
Thinking Adversarially	Stay one step ahead of attackers by implementing a secure development process and knowing how to keep informed on the cutting edge of security research.

Part 5: Server Deployment, Containerization and Testing

In this part, you will create a container for your Flask web app using Docker and deploy the container to a Kubernetes cluster using Amazon EKS. By the end of the project, you will have deployed your application live to the world, where it should be accessible by IP address.

Project: Deploy Your Flask App to Kubernetes Using EKS

Supporting Lessons

Lesson	Summary
Introduction	Welcome to the Server Deployment, Containerization, and Testing course!
Containers	An introduction to containers and Docker
Deployment	Deploy into Kubernetes using continuous delivery

Project: Optimize Your GitHub Profile

Other professionals are collaborating on GitHub and growing their network. Submit your profile to ensure your profile is on par with leaders in your field.

Part 6: Capstone Project

You will now combine all of the new skills you've learned and developed in this course to construct a database-backed web API with user access control. You will choose what app to build and then you'll design and build out all of the API endpoints needed for the application and properly secure them for use in any front end application (web or mobile).

Project: Capstone Project

Supporting Lessons

Lesson	Summary
Capstone Prep and Deployment	In this lesson you'll be introduced to the Capstone project and Heroku, a new tool you'll use to deploy your API.

Udacity

Generated Sat Feb 27 12:45:02 PST 2021