QBN1 — QBN1 TASK 3: PROGRAM DEPLOYMENT

DEPLOYMENT — D602
PRFA — QBN1

Preparation Task Overview Submissions Evaluation Report

COMPETENCIES

4162.1.3 : Implements a Function

The learner implements a function to call and receive information between multiple systems for deployment.

4162.1.4 : Deploys a Data Based Product

The learner deploys a data product based on project requirements.

INTRODUCTION

The data analyst's project is not complete until they have guided it through the process of bringing a data product from development to deployment.

In this task, you will write an API in either Python or R in GitLab. You will then create a Dockerfile that packages your API code and runs a web server to allow HTTP requests to your API. You will then explain how you wrote your code and the challenges you overcame. Finally, you will provide a video demonstrating the live API running from a deployed Docker container.

SCENARIO

Your leadership at the airline is pleased with the machine learning pipeline you generated and has passed it to other business units. To make route planning easier, the operations team has asked you to deploy the model in a format that is easy to integrate with other software tools they are already using. Specifically, they have asked you to create an API to provide departure delay predictions given a departure airport, an arrival airport served from that departure airport, a departure time, and an arrival time. This API should be deployed via a web server so that other software tools can send HTTP requests to the API and receive responses in JSON format. Both the model API and the web server should be deployed together using a Docker image, which is the company standard for deploying models for testing purposes. Also standard is implementation of unit testing of code to ensure it runs properly before including it in any Docker images.

Your code should be stored in the GitLab repository created for you, and you should provide multiple updates of your code within the repository so that users in other business units can track the changes you make to the code over time. Your GitLab repository has automation features implemented that will check your code upon submission, create a Docker image and store it in the GitLab Docker container registry, and then deploy that image to a running container. The live API may be accessed using the URL generated when the GitLab automation runs.

Help

REQUIREMENTS

Your submission must represent your original work and understanding of the course material. Most performance assessment submissions are automatically scanned through the WGU similarity checker. Students are strongly encouraged to wait for the similarity report to generate after uploading their work and then review it to ensure Academic Authenticity guidelines are met before submitting the file for evaluation. See Understanding Similarity Reports for more information.

Grammarly Note:
Professional Communication will be automatically assessed through Grammarly for Education in most performance assessments before a student submits work for evaluation. Students are strongly encouraged to review the Grammarly for Education feedback prior to submitting work for evaluation, as the overall submission will not pass without this aspect passing. See Use Grammarly for Education Effectively for more information.

Microsoft Files Note:
Write your paper in Microsoft Word (.doc or .docx) unless another Microsoft product, or pdf, is specified in the task directions. Tasks may not be submitted as cloud links, such as links to Google Docs, Google Slides, OneDrive, etc. All supporting documentation, such as screenshots and proof of experience, should be collected in a pdf file and submitted separately from the main file. For more information, please see Computer System and Technology Requirements.

You must use the rubric to direct the creation of your submission because it provides detailed criteria that will be used to evaluate your work. Each requirement below may be evaluated by more than one rubric aspect. The rubric aspect titles may contain hyperlinks to relevant portions of the course.

A. Create your subgroup and project in GitLab using the provided web link and the "GitLab How-To" web link by doing the following:

- Clone the project to the IDE.
- Commit with a message and push when you complete each requirement listed in parts B and D.

  Note: You may commit and push whenever you want to back up your changes, even if a requirement is not yet complete.

- Submit a copy of the GitLab repository URL in the "Comments to Evaluator" section when you submit this assessment.
- Submit a copy of the repository branch history retrieved from your repository, which must include the commit messages and dates.

B. Write an API with the code templates provided in either the FastAPI package in Python or the plumber package in R that accepts the following HTTP endpoints. Submit *at least* **two** versions of your code to the GitLab repository demonstrating a progression of work on your code.
   1. “/” should return a JSON message indicating that the API is functional.
   2. “/predict/delays” should accept a GET request specifying the arrival airport, the local departure time, and the local arrival time. It should return a JSON response indicating the average departure delay in minutes.

C. Write *at least* **three** unit tests for your API code, using the pytest package in Python or the testthat package in R, that test features of endpoints given *both* correctly formatted and incorrectly formatted requests. Submit *at least* **two** versions of your code to the GitLab repository demonstrating a progression of work on your code.

D. Write a Dockerfile referencing the requirements.txt file as appropriate that packages your API code and runs a web server to allow HTTP requests to your API. Submit *at least* **two** versions of your Dockerfile to the GitLab repository demonstrating a progression of work on your code.

E. Provide an explanation of how you wrote your code, including any challenges you encountered and how you addressed those challenges.

F. Provide a video demonstrating the live API running from a deployed Docker container. You must issue *at least* 1 well-formatted request and 1 ill-formatted request and demonstrate that the API responds appropriately.

G. Acknowledge sources, using in-text citations and references, for content that is quoted, paraphrased, or summarized.

H. Demonstrate professional communication in the content and presentation of your submission.

File Restrictions

File name may contain only letters, numbers, spaces, and these symbols: ! - _ . * ' ( )
File size limit: 200 MB
File types allowed: doc, docx, rtf, xls, xlsx, ppt, pptx, odt, pdf, csv, txt, qt, mov, mpg, avi, mp3, wav, mp4, wma, flv, asf, mpeg, wmv, m4v, svg, tif, tiff, jpeg, jpg, gif, png, zip, rar, tar, 7z

RUBRIC

A:GITLAB REPOSITORY

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
A GitLab repository is not provided.	The subgroup and project are created in GitLab, but 1 or more of the given actions are not completed, or they are completed incorrectly.	The subgroup and project are created in GitLab correctly, and all of the given actions are completed correctly.

B1:API ENDPOINT "/"

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
An API code is not provided.	In the API code provided, "/" does not return a JSON message.	In the API code provided, "/", returns a JSON message indicating that the API is functional.

B2:API ENDPOINT "/PREDICT/DELAYS"

NOT EVIDENT	APPROACHING COMPETENCE	COMPETENT
An API code is not provided.	In the API code provided, "/predict/delays" does not provide arrival airport, local departure time, local ar-	In the API code provided, "/predict/delays" provides the arrival airport, the local departure time, the local arrival