DATA ANALYSIS CAPSTONE PROJECT GUIDELINES | AUGUST 2024 COHORT*

TABLE OF CONTENTS

Project Overview

Project Deadline: Submission by 11:59 AM (Noon) EST on the last Friday of Module 4.

Your projects are your number one way to show off your skills and land a job as a developer. This document will provide an outline of the minimum requirements to pass Code:You. We encourage you to go well beyond what is detailed here, show your progress to your mentors regularly, and strive for something you want to use to impress employers.

Please read this document carefully and be sure to ask any questions you may have early on.

**Requirements and Guidelines are subject to change at any time.*

Expectations by Module:

Module 3: Develop your project plan during this phase. Regularly consult with mentors to refine your ideas and implementation strategies. Turn in the project plan assignment.

Module 4 (Capstone): The "Turn In Form" for your project will have been distributed. Submit your project before the due date. You can continue working up until the deadline. Ensure your project is thoroughly tested and reviewed by a mentor.

Project Requirements

To pass your project, ensure all the following criteria are met. Maintain regular communication with your mentors to stay aligned with these guidelines.

Capstone Presentation and Interview

1. Capstone Project Overview:

- Be prepared to explain your capstone project.
- Discuss the problem you aimed to solve and how your project addresses it.

2. Code Functionality and Use:

- Demonstrate how your code supports your project's functionality.
- Highlight a specific section of your code that you're particularly proud of or found challenging to implement.

3. Future Plans and Reflection:

- Reflect on your experience with Code:You.
- Share a key skill or lesson you've learned and how it will influence your future projects or career.

GitHub Repository and Commits

Source control is a cornerstone of development, enabling version tracking, collaboration, and project management. GitHub provides developers with tools to track changes, collaborate, and present their work professionally.

Requirements:

- Upload your project to a GitHub repository with at least **10 distinct commits**.

- Use Git commands consistently to demonstrate ongoing progress, not just a final upload.

README File

A README file serves as a roadmap for your project, providing essential details to ensure clarity and accessibility for collaborators and reviewers.

Requirements:

- Write a detailed README file explaining your project in at least one paragraph.
- Identify and describe **three or more features** you integrated from the provided list.
- Include any special instructions for reviewers to run your project.

Visual Appeal

In development, the user experience (UX) and visual design are just as critical as technical functionality. Your project should be engaging, visually cohesive, and aligned with industry standards.

Requirements:

- Design your project to be **visually appealing**, following current industry trends.
- Ensure text is correctly spelled, legible, and consistent across all components.
- Use a clear and consistent design for headings, buttons, forms, and interactive elements.
- Research other applications and websites for inspiration. Emulate styles and functionalities that enhance usability and appeal.
- Choose a **color palette** and **font stack** to create a polished and professional design.

By meeting these standards, your project will demonstrate both technical proficiency and design sensibility, making it attractive to mentors, reviewers, and potential employers.

Data Analysis Project Requirements

Data Analysis Implementation:

Develop a data analysis program utilizing pandas, matplotlib, and/or numpy. Execute an analysis project on 2 or more data pieces and produce comprehensive data visualizations via Tableau, Jupyter, Plotly, Matplotlib, or similar tools. Ensure data ingestion, analysis, and display are well-documented and reproducible. Data cleaning steps should be clearly outlined.

Review and Polishing:

Include a section in your project that explains your data interpretation. This highlights your written communication skills, emphasizing clarity and coherence. Ensure that someone unfamiliar with your project can comprehend its content and purpose.

Feature Selection:

Choose at least one item from each category in the Features List provided.
Consider including more than 5 features in case of issues with other selections.

Data Analysis Capstone Features List

Choose **ONE** item from each table below to meet the project requirements.

1. **Loading data.**

FEATURE	DIFFICULTY	RESOURCES
Read TWO data files (JSON, CSV, Excel, etc.).	Easy	
Read in TWO text data sources (in any format). For example, email chains or different pages from a book.	Intermediate	
Read TWO data sets in with an API (or use two different APIs that have data you can combine to answer new questions).	Intermediate	A Cool List of Public APIs
Scrape TWO pieces of data from anywhere on the internet and utilize it in your project.	Intermediate	RealPython article on webscraping
Set up a local database and read data in with SQLite or SQLAlchemy	Hard	SQLite docs

2. **Clean and operate on the data while combining them.**

FEATURE	DIFFICULTY	RESOURCES
Clean your data and perform a pandas merge with your two data sets, then calculate some new values based on the new data set.	Intermediate	Merge and join
Clean your data and perform a SQL join with your data sets using either plain sql or the pandasql Python library.	Intermediate	SQL Join resource pandasql docs
If you're using text data, get some information from your separate documents and summarize them in a DataFrame. This isn't *literally* a join but accomplishes a similar idea. For example, getting the most frequent word distributions from both documents and then summarizing them in a table.	Intermediate	Natural Language Processing

3. **Visualize / Present your data.**

FEATURE	DIFFICULTY	RESOURCES
Make 3 matplotlib or seaborn (or another plotting library) visualizations to display your data.	Easy	Matplotlib basics Seaborn basics
Make a Tableau dashboard to display your data	Intermediate	Tableau Basics
Make at least 1 Pandas pivot table and 1 matplotlib/seaborn plot. Pivot tables are a way to summarize your data and present it easily in a way that isn't just a graph. They can be useful when combined with graphs.	Intermediate	Pandas pivot tables
Make a visualization with Bokeh. You can create interactive online	Intermediate / Hard	Bokeh Gallery

visualizations with this, but it is more involved than the other plotting libraries! Very cool though.		

4. **Best practices: Enhance your project to a higher tier that will impress employers and help other programmers understand your project. Choose 2 from this table.**

FEATURE	DIFFICULTY	RESOURCES
Utilize a virtual environment and include instructions in your README on how the user should set one up	Intermediate	conda virtual environments (note: you can do this with a few different modules, so either venv or virtualenv are fine as well)
Write 3 unit tests and include instructions on how the user can run them. This will mostly only apply if you're building custom functions and classes.	Intermediate	Pytest docs
Build a custom data dictionary and include it either in your README or as a separate document. This will only apply if your data set does not already *have* a data dictionary or if you're building a custom data set. For an example, see the resources to the right.	Easy	Data Dictionary example (look under the 6th row on this Louisville Metro Data data set)
Any other "best practices" your mentor can think of: this is open to interpretation, but if your mentor has a particular idea for a best practice about your specific project, that will meet the requirement for this table.	n/a	Discuss w/ mentor

5. **Interpretation of your data.**

REQUIREMENT	DIFFICULTY	RESOURCES
Annotate your code with markdown cells in Jupyter Notebook, write clear code comments, and have a well-written README.md. Tidy up your notebook, and make sure you don’t have any empty cells or incomplete cells that don’t do anything. Make sure it’s all functional before your final github commit.	Intermediate	A Guide to Good Comments A Guide to Markdown
Annotate your .py files with well-written comments and a clear README.md (only applicable if you’re not using a jupyter notebook).	Intermediate	A Guide to Good Comments A Guide to Markdown

Review Process:

Our project reviewers consist of mentors, training team and other staff. We will evaluate the project based on a general rubric for capstone projects. Projects must meet all the requirements to pass the session. Having a passing project is only part of the requirements to graduate a session or class.

Projects are evaluated through the following steps:

- The project reviewer clones your project using Git.
- They follow your readme instructions to set up/run your project.
- They will test your app for the requirements in this document and also look through the code
- Reviewers will try only the most minimal troubleshooting steps if your project does not run “out of the box”

Testing Your Project:

Sharing and testing your capstone project offers invaluable benefits for both personal and professional growth. By sharing your project with peers, mentors, or industry professionals, you gain valuable feedback and insights that can help refine and improve your work. Embracing the principles of sharing and testing not only enhances the quality of your project but also cultivates essential skills in communication, collaboration, and problem-solving, paving the way for success in your future endeavors.

Successful testing includes:

- Identification of potential issues or bugs
- Verification of functionality and performance
- Ensuring compatibility across different devices and platforms
- Evaluation of user experience and interface usability
- Confirmation of security measures and data integrity
- Validation of adherence to project requirements and specifications
- Gathering feedback for improvement and refinement

The reviewers **MUST** be able to run your project for it to pass. For your project to pass review, ensure that reviewers can run it smoothly. Test thoroughly and provide clear instructions for setup. Maintain modularity and portability, avoiding database dependencies like SQLServer. Opt for portable solutions like SQLite to enhance accessibility and ease of use.

Mentor Engagement:

Regularly consult with mentors about your project. Share your ideas early and ensure you're on track to a successful project.

Additional Resources

For additional support in your coding project, consider tapping into online communities like Stack Overflow for problem-solving, using version control systems such as Git and GitHub for collaboration, and consulting comprehensive documentation like MDN for guidance. Exploring relevant open-source projects can also offer inspiration and resources. Integrating these tools and resources will streamline your project's development process and enhance its quality.

- **The Ultimate Design Principles Guide For Developers -** https://bootcamp.uxdesign.cc/the-ultimate-design-principles-guide-for-developers-d4aa58937283
- **Adobe Colors** - https://color.adobe.com/create/color-wheel
- **Coolors** - https://coolors.co/
- **An Interactive Guide to Flexbox** - https://www.joshwcomeau.com/css/interactive-guide-to-flexbox/
- **An Interactive Guide to CSS Grid** - https://www.joshwcomeau.com/css/interactive-guide-to-grid/
- **Regex Cheat Sheet** - https://www.keycdn.com/support/regex-cheat-sheet
- **SOLID: The First 5 Principles of Object Oriented Design** - https://www.digitalocean.com/community/conceptual-articles/s-o-l-i-d-the-first-five-principles-of-object-oriented-design
- **RapidAPI** - https://rapidapi.com/hub

Free APIs

This list is just the beginning; there are countless more APIs available, with new ones emerging regularly. Consider it a starting point to explore the vast world of APIs. Don't hesitate to reach out to your mentors for suggestions on other APIs they may be aware of.

OpenWeatherMap API: Access weather data for any location worldwide. https://openweathermap.org/api

NASA AP: Explore space-related data including images, videos, and information about planets, satellites, and more. https://api.nasa.gov/

GitHub API: Access GitHub repositories, user data, and much more. https://developer.github.com/v3/

The New York Times API: Access articles, headlines, and other content from The New York Times. https://developer.nytimes.com/apis

Google Maps API: Integrate maps, routes, and location data into your applications. https://developers.google.com/maps

OpenStreetMap API: Similar to Google Maps but based on open data.
https://wiki.openstreetmap.org/wiki/API

Twitter API: Access tweets, user data, trends, and more from Twitter.
https://developer.twitter.com/en

Reddit API: Access posts, comments, and other data from Reddit.
https://www.reddit.com/dev/api/

CoinGecko API: Access cryptocurrency data, including market data, historical data, and more.
https://coingecko.com/en/api

Unsplash API: Access a vast collection of high-quality, free-to-use images.
https://unsplash.com/developers

The Movie Database (TMDb) API: Access movie and TV show data, including details about cast, crew, ratings, and more. https://www.themoviedb.org/documentation/api
COVID-19 Data API: Access data related to the COVID-19 pandemic, including case counts, vaccination rates, and more. https://covid19api.com/

PokeAPI: Access data about Pokémon, including Pokémon species, moves, abilities, and more.
https://pokeapi.co/

REST Countries API: Access information about countries, including details like population, currencies, languages, and more. https://restcountries.com/

Open Food Facts API: Access data about food products, including ingredients, nutritional information, and more. https://world.openfoodfacts.org/data

Demo Day:

After the completion of the pathways, Code:You will host a Demo Day for graduates of the program to show off their projects to the community. Employers, mentors, and other members of the tech community will be invited to see the projects and meet you - the developer of that project!

Presentation slots for the event will be invite-only, as we will not have enough time for every Code:You student to participate (sorry, there are hundreds of projects!). To incentivize you to do as well as you can on your project, invites to present at Demo Day will be based on the most impressive projects to employers. What determines that selection is ultimately subjective - how do you compare the visual presentation of one project to the technical skill level of another? But we will strive to be as fair as we can during the process.

Policy for using AI/Machine Learning Prompts for writing code:

AI should not replace essential human elements of writing and reviewing code. Students are expected to use AI technologies ethically, respecting copyright laws, privacy norms, and the intellectual property rights of others. AI resources should be used responsibly and not abused for the ease of use in creating original content. AI should not be used to engage in plagiarism, cheating, or any form of dishonesty. AI work must not be submitted in the place of student work.

AI tools should be used as an aid rather than a substitute for human expertise. Students are accountable for the code generated with AI assistance, and it must adhere to existing coding standards and practices. All AI-generated code should be scrutinized to ensure that it aligns with project requirements, just like any manually written code.

Failing to adhere to these rules may result in not passing the class.

Documentation

Proper documentation is vital for any well-maintained coding project, but it’s imperative when introducing AI-generated code into your work.

Documenting the AI-generated code helps make the project more transparent. It allows staff and mentors to understand how AI models are utilized within the codebase.

If AI tools are used to write code, it must be documented in the Read-Me and in comments in the code. This documentation should cover the purpose of the code and the manner of usage.

Best Practice: Encapsulate AI-Generated Code

Encapsulate AI-generated code into defined modules or functions to improve readability and usability immediately. Code encapsulation means wrapping a piece of code, typically a set of statements or functions, within a higher-level structure such as a function, class, or module. The encapsulated code is then treated as a single unit with a well-defined interface and can be invoked or interacted with as a cohesive entity.

Encapsulation promotes modular design by breaking down complex systems into smaller, manageable units. Each encapsulated unit can be developed, tested, and maintained independently, making the codebase more modular and easier to understand.

Encapsulated units can also be reused in different parts of a program or other projects.

Sample Project Outline: Analysis and Visualization of Weather Patterns and CO2 Emissions

I. Introduction

- Objective: Develop a visually appealing and technically sound project analyzing the relationship between weather patterns and CO2 emissions.
- Tools and Technologies: Pandas, Matplotlib, Seaborn, SQLite, Tableau, Jupyter Notebook.
- Goals:
    - Adhere to standard interaction conventions.
    - Ensure consistency and readability across the project.
    - Emulate industry trends in design.
    - Provide comprehensive data analysis and visualization.
    - Document the project thoroughly for clarity and reproducibility.

II. Visual Appeal and Design Consistency

1. Industry Trends Analysis
    - Research and analyze current design trends in data dashboards focusing on environmental data.
    - Identify compelling styles and functionalities to emulate.
2. Design Framework
    - Color Palette: Select a harmonious color palette that reflects environmental themes (e.g., greens, blues).
    - Font Stack: Choose consistent fonts for headings, buttons, and body text.
    - Layout and Navigation: Ensure predictable and standard interaction elements (menus, forms, buttons).
    - Text Accuracy: Verify all text for correct spelling and legibility.

III. Data Acquisition and Cleaning

1. Loading Data
    - Read Two Data Files: Load historical weather data (CSV) and CO2 emissions data (Excel).
    - API Data Integration: Fetch current weather data from a weather API and recent CO2 emission data from an environmental API.

- Web Scraping: Scrape additional weather data (e.g., temperature trends) and CO2 data (e.g., emission sources) from relevant websites.

2. Data Cleaning and Merging
    - Clean datasets using Pandas (handling missing values, correcting data types).
    - Merge datasets on common attributes (e.g., date, location) and calculate new values such as average temperature and CO2 emission rates.
    - For text data: Perform NLP tasks to extract summaries from weather reports and emission studies and combine them into a DataFrame.

IV. Data Analysis and Visualization

1. Visualizations
    - Create at least three visualizations using Matplotlib or Seaborn, such as:
        - Line plots for temperature trends over time.
        - Bar charts comparing CO2 emissions by year.
        - Scatter plots showing the correlation between temperature and CO2 levels.
    - Develop a Tableau dashboard to present key insights interactively.
2. Advanced Visualization
    - Explore creating interactive visualizations using Bokeh, such as interactive maps showing temperature and CO2 emission hotspots.

V. Best Practices Implementation

1. Virtual Environment
    - Set up a virtual environment and provide setup instructions in the README.
2. Unit Testing
    - Write three unit tests for custom functions or classes used in data cleaning and analysis.
    - Include instructions on how to run the tests.
3. Data Dictionary
    - Develop a custom data dictionary to explain all variables and data points, included in the documentation.

VI. Data Interpretation and Documentation

1. Code Annotation

    - Use markdown cells in Jupyter Notebook to annotate the code, explaining each step of the analysis.
    - Ensure clear and concise comments within the code.
2. Project Documentation
    - Write a comprehensive README.md file covering:
        - Project overview and objectives.
        - Data sources and cleaning steps.
        - Analysis and visualization methods.
        - Instructions for setting up the environment and running the project.
        - Summary of findings and interpretations, highlighting any observed trends or correlations.

VII. Review and Polishing

1. Internal Review
    - Perform a thorough review to ensure the project meets all requirements.
    - Verify the accuracy and readability of visual components.
2. External Feedback
    - Present the project to peers or mentors for feedback.
    - Incorporate feedback to enhance clarity, functionality, and visual appeal.

VIII. Final Submission

- Ensure the project is fully functional and well-documented.
- Prepare the project for final presentation or submission to potential employers.

This outline provides a structured approach to developing a data analysis project focused on weather patterns and CO2 emissions. It includes steps for design consistency, data handling, visualization, best practices, and thorough documentation, making the project both attractive and engaging to employers.