Coding Boot Camp © 2019. All Rights Reserved.

Instructions:

Evaluate the homework against the outlined criteria in the below rubric, assigning a rating to each criterion. Add points earned across all criteria and convert the total points to a letter grade, assigning a “+” or “-” letter grade designation at your discretion.

A (+/-)	90+	C (+/-)	40-64	F (+/-)	<15
B (+/-)	65-89	D (+/-)	15-39		

Notes:

The deployed assignment utilizes the **SQLAlchemy** library to retrieve data from a database which is used to generate charts and an API. The source code should also be deployed to **Github** or **Gitlab.**

Rubric for Surfs Up:

	Mastery 20 points	**Approaching Mastery 15 points**	**Progressing 10 points**	**Emerging 5-0 points**	**Incomplete**
Precipitation Analysis	The submission does all of the following: ✓ Gets the correct results for the last year of data (note that the last day in the dataset is 8/23/2017) ✓ Creates a pandas dataframe using the `date` and `precipitation` columns ✓ Sorts the dataframe by date ✓ Makes a plot using pandas with `date` as the x and `precipitation` as the y variables	The submission does 3 of the following: ✓ Gets the correct results for the last year of data (note that the last day in the dataset is 8/23/2017) ✓ Creates a pandas dataframe using the `date` and `precipitation` columns ✓ Sorts the dataframe by date ✓ Makes a plot using pandas with `date` as the x and `precipitation` as the y variables	The submission does 2 of the following: ✓ Gets the correct results for the last year of data (note that the last day in the dataset is 8/23/2017) ✓ Creates a pandas dataframe using the `date` and `precipitation` columns ✓ Sorts the dataframe by date ✓ Makes a plot using pandas with `date` as the x and `precipitation` as the y variables	The submission does 0-1 of the following: ✓ Gets the correct results for the last year of data (note that the last day in the dataset is 8/23/2017) ✓ Creates a pandas dataframe using the `date` and `precipitation` columns ✓ Sorts the dataframe by date ✓ Makes a plot using pandas with `date` as the x and `precipitation` as the y variables	No submission was received -OR- Submission was empty or blank -OR- Submission contains evidence of academic dishonesty
Station Analysis	The submission does all of the following: ✓ Correctly outputs the number of stations in the dataset (9) ✓ Correctly finds the most active station by using `count` (USC00519281) ✓ Gets the min, max, and average temperatures for the most active station (USC00519281)	The submission does 3 of the following: ✓ Correctly outputs the number of stations in the dataset (9) ✓ Correctly finds the most active station by using `count` (USC00519281) ✓ Gets the min, max, and average temperatures for the most active station (USC00519281) ✓ Correctly plots a histogram for	The submission does 2 of the following: ✓ Correctly outputs the number of stations in the dataset (9) ✓ Correctly finds the most active station by using `count` (USC00519281) ✓ Gets the min, max, and average temperatures for the most active station (USC00519281) ✓ Correctly plots a histogram for	The submission does 0-1 of the following: ✓ Correctly outputs the number of stations in the dataset (9) ✓ Correctly finds the most active station by using `count` (USC00519281) ✓ Gets the min, max, and average temperatures for the most active station (USC00519281) ✓ Correctly plots a histogram for the	

	✓ Correctly plots a histogram for the last year of data using `tobs` as the column to count.	the last year of data using `tobs` as the column to count.	the last year of data using `tobs` as the column to count.	last year of data using `tobs` as the column to count.	
API SQLite Connection & Landing Page	The Flask Application does all of the following: ✓ Correctly generates the engine to the correct sqlite file ✓ Uses `automap_base()` and reflects the database schema ✓ Correctly saves references to the tables in the sqlite file (measurement and station) ✓ Correctly creates and binds the session between the python app and database	The Flask Application does 3 of the following: ✓ Correctly generates the engine to the correct sqlite file ✓ Uses `automap_base()` and reflects the database schema ✓ Correctly saves references to the tables in the sqlite file (measurement and station) ✓ Correctly creates and binds the session between the python app and database	The Flask Application does 2 of the following: ✓ Correctly generates the engine to the correct sqlite file ✓ Uses `automap_base()` and reflects the database schema ✓ Correctly saves references to the tables in the sqlite file (measurement and station) ✓ Correctly creates and binds the session between the python app and database	The Flask Application does 0-1 of the following: ✓ Correctly generates the engine to the correct sqlite file ✓ Uses `automap_base()` and reflects the database schema ✓ Correctly saves references to the tables in the sqlite file (measurement and station) ✓ Correctly creates and binds the session between the python app and database -OR- ✓ Flask app does not start	
API Static Routes	The static routes do all of the following: **Precipitation route** ✓ Returns the jsonified precipitation data for the last year in the database ✓ Returns json with the date as the key and the value as the precipitation **Stations route** ✓ Returns jsonified data of all of the stations in the database **Tobs route** ✓ Returns jsonified data for the most active station (USC00519281) for the last year of data	The static routes do 3 of the following: **Precipitation route** ✓ Returns the jsonified precipitation data for the last year in the database ✓ Returns json with the date as the key and the value as the precipitation **Stations route** ✓ Returns jsonified data of all of the stations in the database **Tobs route** ✓ Returns jsonified data for the most active station (USC00519281) for the last year of data	The static routes do 2 of the following: **Precipitation route** ✓ Returns the jsonified precipitation data for the last year in the database ✓ Returns json with the date as the key and the value as the precipitation **Stations route** ✓ Returns jsonified data of all of the stations in the database **Tobs route** ✓ Returns jsonified data for the most active station (USC00519281) for the last year of data	The static routes do 0-1 of the following: **Precipitation route** ✓ Returns the jsonified precipitation data for the last year in the database ✓ Returns json with the date as the key and the value as the precipitation **Stations route** ✓ Returns jsonified data of all of the stations in the database **Tobs route** ✓ Returns jsonified data for the most active station (USC00519281) for the last year of data -OR- ✓ Flask app does not start	
API Dynamic Route	The dynamic route does all of the following: **Start route** ✓ Route accepts the start date as a parameter from the URL ✓ Returns the min, max, and average temperatures calculated from the given start date to the end of the dataset	The dynamic route does 3 of the following: **Start route** ✓ Route accepts the start date as a parameter from the URL ✓ Returns the min, max, and average temperatures calculated from the given start date to the end of the dataset	The dynamic route does 2 of the following: **Start route** ✓ Route accepts the start date as a parameter from the URL ✓ Returns the min, max, and average temperatures calculated from the given start date to the end of the dataset	The dynamic route does 0-1 of the following: **Start route** ✓ Route accepts the start date as a parameter from the URL ✓ Returns the min, max, and average temperatures calculated from the given start date to the end of the dataset	

	Start/end route ✓ Route accepts the start and end dates as parameters from the URL ✓ Returns the min, max, and average temperatures calculated from the given start date to the given end date	**Start/end route** ✓ Route accepts the start and end dates as parameters from the URL ✓ Returns the min, max, and average temperatures calculated from the given start date to the given end date	**Start/end route** ✓ Route accepts the start and end dates as parameters from the URL ✓ Returns the min, max, and average temperatures calculated from the given start date to the given end date	**Start/end route** ✓ Route accepts the start and end dates as parameters from the URL ✓ Returns the min, max, and average temperatures calculated from the given start date to the given end date -OR- ✓ Flask app does not start	

Rubric for Surfs Up - Bonus Analyses:

	Mastery **20 points**	**Progressing** **10 points**	**Emerging** **0 points**	**Incomplete**
Optional Analyses	The submission does all of the following: **Trip Temperature Analysis** ✓ Uses the calc_temps function to get the min, max, and average temperatures for a date range of their choosing ✓ Uses the calculated temperatures to generate a bar chart with an error bar. **Daily Rainfall Average** ✓ Calculates the min, max, and average temperatures for each day of their trip and appends them to a list. ✓ Creates a dataframe from the list and generates a stacked line chart plotting the min, max, and average temps for each day of their trip	The submission successfully does only 1 of the optional analyses: **Trip Temperature Analysis** ✓ Uses the calc_temps function to get the min, max, and average temperatures for a date range of their choosing ✓ Uses the calculated temperatures to generate a bar chart with an error bar. -OR- **Daily Rainfall Average** ✓ Calculates the min, max, and average temperatures for each day of their trip and appends them to a list. ✓ Creates a dataframe from the list and generates a stacked line chart plotting the min, max, and average temps for each day of their trip	The submission attempts one or both of the following, but fails: **Trip Temperature Analysis** ✓ Uses the calc_temps function to get the min, max, and average temperatures for a date range of their choosing ✓ Uses the calculated temperatures to generate a bar chart with an error bar. **Daily Rainfall Average** ✓ Calculates the min, max, and average temperatures for each day of their trip and appends them to a list. ✓ Creates a dataframe from the list and generates a stacked line chart plotting the min, max, and average temps for each day of their trip	No submission was received -OR- Submission was empty or blank -OR- Submission contains evidence of academic dishonesty