# Research Project Assessment

**This assessment is for the final research project only.** Use the assessment.md file for all mini-projects.

## Instructions

Before submitting your research project draft for grading, confirm:

1. The manuscript .qmd for your project is in the root directory and knits to .pdf without error.
2. The knitted .pdf of your draft is in the root directory, with the same filename as the .qmd.
3. This research-assessment.md file is in the root directory of your project repo.
4. Dr. Dowling and your section TA are collaborators on your GitHub repo with permissions to pull/push.

To complete this assessment:

1. Complete the basic information section in full.
2. Confirm all links are correct and accessible
3. Check off all objectives you are attempting to demonstrate
   1. To earn 30 points you must demonstrate each objective. However, you do not need to attempt all objectives with each draft if your goal is to build the project over time.
   2. If the objective is demonstrated somewhere other than the .qmd, add a note in the grader comments section for where to find it (e.g., “see data-cleaning.R lines 20-30”).
4. Optionally, complete the reflection section, which may earn engagement points.

## Basic information

Name:

CNetID:

Section:

Research project title:

Submission date:

Submission number (1-4):

Project GitHub repository URL:

Filename of manuscript .qmd:

Filename of knitted .pdf:

## Overall requirements

Overall requirements for the research project are as follows:

1. The project must be a research project. It must provide background on a research topic, ask at least one research question, use data to attempt to answer that question, report the results of the data analysis, and interpret the results in the context of the research question.
2. The project must be contained in a github repository that follows git best practices and includes all necessary files to run the project from start to finish, including:
   1. The .qmd file for the manuscript
   2. All data files used in the project
   3. All scripts used in the project
   4. A README.md file & .gitignore file
3. The project must be reproducible – a reader should be able to clone the repo and run the .qmd from start to finish without error. The .qmd file should include:
   1. A YAML header with all fields necessary for an APA manuscript
   2. Setup source chunks that load libraries, read in data, set chunk options, set seed, etc.
   3. Minimally, an IMRD structure (Introduction, Methods, Results, Discussion), though it may be more complex
   4. Integration of markdown and code chunks throughout, following best practices for using code chunks
   5. Figures and tables rendered in code chunks
   6. Inline R code & references to render data-dependent text
   7. At least 1 descriptive analysis and 1 hypothesis test, either in code chunks or sourced scripts
   8. Frequent and informative code comments throughout
4. The .qmd file should knit/render to an APA7 formatted manuscript with one click and no errors. The knitted manuscript should include:
   1. A title page with title, author, and institutional affiliation
   2. An abstract (this may be minimal, but should exist)
   3. Narrative text comprising a complete research report
   4. APA7 references, both in-text citations and a References page
   5. Publication-ready figures (2+) and tables (1+)
   6. Results of all analyses presented in-text (and where appropriate, in tables), with no raw R output; where possible, all text should be data-dependent and rendered with inline R code
   7. Quarto generated references to all figures and tables
   8. Statistical analyses and figures interpreted in narrative text
5. The .qmd should render a .pdf identical to the .pdf you submit for grading

## Assessment

The final project must demonstrate each of 30 the course learning objectives, each worth 1 point.

Below each learning objective is a list of general expectations for meeting that objective. You should aim to meet all expectations to earn a point for meeting the objective, but these are not rigid requirements. For example, writing a complex and creative function that uses multiple arguments and returns a complex output could meet the “parse and define functions and arguments” objective, even if it is only used in one context.

Refer to the website for general tips on meeting these objectives and an FAQ.

### GitHub and R Studio

1. Create and maintain a repo with sensible organization and naming conventions
   1. All folder and file names are informative
   2. Uses relative paths correctly
   3. Does not have duplicate/redundant elements
      * ☒ Objective attempt
      * ☐ Objective met
      * Grader comments:
2. Maintain an informative and up-to-date README.md
   1. Includes description of repo purpose, data use, research questions, etc.
   2. Outlines the repo structure with file tree or similar
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:
3. integrate a GitHub repo with an R studio project, including .gitignore file
   1. All scripts run and all notebooks render if the repo is cloned to another location
   2. .gitingore comprehensively excludes unnecessary, private, and very large files and is be commeted appropriately
      * ☒ Objective attempt
      * ☐ Objective met
      * Grader comments:
4. effectively use version control
   1. Used frequent, informative commit messages
   2. Relies on document revisions rather than manually created new versions
      * ☒ Objective attempt
      * ☐ Objective met
      * Grader comments:

### R programming

1. Find, install, require, and load R packages
   1. No errors occur when running scripts in a new environment
      1. If packages other than the “class packages” listed on the resources page are used, code to install/require them is included *and commented out*
      2. When a reader opts-in to installing packages by uncommenting the code, it runs without errors
   2. Uses more than one function to install/load/require packages (including those used in commented code)
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:
2. Use arithmetic, comparison, and logical operators
   1. Uses all three types of operators
   2. Uses multiple operators in data transformation pipelines and/or inline R code
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:
3. Parse and define functions and arguments
   1. Defines at least one function with at least one argument in code chunks or sourced scripts
   2. User-defined function(s) run(s) without error and produces expected output in at least 2 contexts
   3. Functions are well-documented with comments
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:
4. Parse and write conditional statements and/or loops
   1. Uses conditional in multiple contexts, including dplyr pipelines
   2. Uses multiple types of conditional/loop functions (e.g., if\_else(), case\_when(), for, while)
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:
5. Use readr functions to read in and write out data
   1. Reads in data from at least one source in code chunk or sourced script
   2. Writes out intermediate and/or final datasets in code chunks or sourced scripts
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:
6. Use dplyr and tidyr functions to transform data
   1. Uses at least 3 unique dplyr functions
   2. Uses at least 1 tidyr function in a data transformation pipeline
   3. Combines dplyr and tidyr functions in a data transformation pipeline
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:
7. Use stringr functions to work with string variables
   1. Uses ate least 2 unique stringr functions
   2. Uses stringr functions in a data transformation pipeline
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:
8. Use forcats functions to work with factor variables
   1. Uses ate least 2 unique forcats functions or one function in 2 unique contexts (with different purposes)
   2. Uses forcats functions in a data transformation pipeline
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:

### Data visualization with ggplot2

1. Produce 1- and 2-variable plots with geom\_\* layers
   1. Creates at least 2 figures with different geom\_\* layers (e.g., a scatter plot and a bar plot)
   2. At least one plot is multi-variable
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:
2. Use dynamic aesthetics to group data
   1. Uses at least 2 unique data-mapped aes() arguments (e.g., color, shape, size) to group data in a plot in one or multiple plots
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:
3. Use facets to create parallel plots
   1. Uses facets in at least 2 ways (e.g., using both facet\_wrap() and facet\_grid()), modifying the number of rows and columns, using free vs fixed scales, etc.)
   2. Combines facets with other dynamic grouping aesthetics
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:
4. Create publication-quality plots using theme and labs layers
   1. Plots have informative titles, axis labels, and legends
   2. Fonts are stylized professionally and legibly (e.g., adjusted size/angle/justification)
   3. Variables display in plain English (e.g., “Age (years)” not “child\_age\_yrs”
   4. Uses at least 1 static aesthetic (e.g., color, shape, size) that improves visual clarity without mapping to data
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:

### Data analysis

1. Perform simple descriptive analyses with multiple data types
   1. Calculates summary/descriptive statistics for at least 1 numeric variable (e.g., mean, standard deviation)
   2. Calculates summary/descriptive statistics for at least 1 non-numeric variable (e.g., frequencies, proportions)
   3. Presents results in narrative text, table, or plot
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:
2. Perform simple hypothesis testing analyses for multiple data types
   1. Performs at least 1 hypothesis test for numeric data (e.g., t-tests, linear regression)
   2. Performs at least 1 hypothesis test for factor data (e.g., chi-square, ANOVA)
   3. Presents results in narrative text, table, or plot
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:
3. Present and interpret statistics in manuscript narrative
   1. Presents and interprets results of analyses in narrative text, like the results section of a journal article, including all information appropriate for a given analysis (e.g., effect size, p-value, confidence interval – dependent on analysis type and results)
   2. Discriminates between statistically signficiant and non-signficant statistics, where applicable
   3. Discriminates between informative and non-informative statistics and presents only the former in narrative text
   4. Uses dynamic inline R code to render data-dependent text
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:

### BibTeX

1. Render APA7 in-text citations with BibTeX syntax for multiple citation forms
   1. Cites at least 3 sources in-text
   2. Uses at least 2 citation forms (e.g., (Author, Year), Author (Year), etc.)
   3. May use cite\_r() to cite R and R packages
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:
2. Render an APA7 references page from a .bib file
   1. Includes all sources cited in-text
   2. Formats references in APA7 style
   3. Presents accurate, complete, and error-free references
   4. May include R and R package citations with cite\_r()
   5. May include references not cited in-text
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:

### Notebooks and code chunks

1. Create and effectively use code chunks following best practices
   1. Uses informative names/labels
   2. Includes frequent, informative comments
   3. Follows the “1-chunk-1-thing” rule
   4. Chunks are distributed throughout the manuscript, sensibly placed near the text they support
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:
2. Use code chunks to set up a quarto document
   1. Sources at least 1 .R script and/or reads in necessary data
   2. Loads packages in at least 1 code chunk
   3. Sets preferences/options in at least 1 code chunk
   4. Organizes setup chunks sensibly
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:
3. Render publication-quality tables, figures, and images from code chunks
   1. Produces at least 1 table or image with a caption
   2. Produces at least 1 figure/plot with a markdown caption
   3. Captions are informative, complete, and render correctly
   4. All tables and figures are referenced in the narrative text (e.g., Figure 1)
   5. References render without error and link to the correct table/figure in pdf/html output
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:
4. Execute descriptive analyses and/or hypothesis testing in code chunks
   1. At least 1 code chunk executes a descriptive analysis (e.g., summary(), table())
   2. At least 1 code chunk executes a hypothesis test (e.g., t.test(), chisq.test())
   3. Results are presented in narrative text, table, or plot
   4. Results are not displayed as raw R output
   5. Chunks are organized sensibly and appear near the text they support
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:

### R Markdown and Quarto

1. Create and maintain a quarto document YAML header
   1. Includes all necessary metadata, output options, and formatting options necessary to render an APA styled document (or other specified style if appropriate for the project)
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:
2. Use quarto R Markdown to compose an academic manuscript
   1. Uses at least 2 unique text styles (e.g., bold, italics, code)
   2. Uses at least 2 unique header levels
   3. Includes at least 1 list
   4. Includes at least 1 footnote
   * NOTE: This is going to be one of the most flexible objectives to demonstrate. You need to demonstrate a range of markdown skills and use them to make a readable, informative manuscript. Hitting the four points above should do that, but you can use your judgment about what kind of markdown features will best serve your project. No matter what, you should use markdown to follow APA7 guidelines.
     + ☐ Objective attempt
     + ☐ Objective met
     + Grader comments:
3. Use inline R variables to replace static text
   1. Replaces static text with inline R references in at least 3 unique numeric contexts
   2. Replaces static text with inline R references in at least 1 character context
   3. Ideally, uses inline R references for *all* data-dependent text
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:
4. Run inline R functions to render dynamic data-dependent text
   1. Uses inline R functions to render at least 3 unique data-dependent text outputs (e.g., performs rounding, calculates means, subtracts one list length from another, etc. – inline rather than in a code chunk)
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments:
5. Use knitr and quarto to produce an APA7 formatted 1-click PDF manuscript
   1. Produces a PDF output that is formatted in APA7 style
   2. PDF includes all necessary elements (e.g., title page, abstract, body, references)
   3. PDF renders without error and includes all text, tables, and figures
   4. No additional steps are needed (e.g., finding data, determining necessary packages to install and load, running unsourced scripts)
      * ☐ Objective attempt
      * ☐ Objective met
      * Grader comments: