Week 5: Technical Writing with Quarto

DATA 510: Data Science Capstone

Lucas P. Cordova, Ph.D.

2026-06-08

Week 5: why Quarto for capstone technical writing, logistics from M1 proposal through M4/M5 write-ups, installation via VS Code and the command line, hands-on Quarto essentials for R users, and a teaser on portfolio sites. Includes a copy-paste write-up skeleton for deliverables/M4-writeup-draft/.

Table of contents

1 Learning Objectives

1.1 Today's Objectives

1.2 What You Will Leave With

You will be able to:

1. **Explain** why Quarto fits capstone deliverables (HTML, PDF, reproducible code).
2. **Map** your M1 proposal content into a long-form write-up workflow inside the DS3 `deliverables/` tree.
3. **Install and verify** Quarto with VS Code and the command line.
4. **Author** a basic `.qmd` with YAML, R chunks, figures, and citations; understand render commands for HTML and PDF (hands-on in the Git session).
5. **Describe** how the final write-up becomes a portfolio centerpiece (details in a follow-up lecture).

1.3 Course Connection

Week 2 covered *what* good capstone research prose argues. **Today** covers *how* you publish it. Your write-up rough draft (M4, week 12) and final write-up (M5) must read like **online publication**, not a pasted notebook dump.

Block 1 is 6:00 to 7:30 (90 min). Block 2 is Git plus skeleton render (05-2 deck), then data engineering consultations. Clone, render PDF, commit, and Canvas upload happen in Block 2, not Block 1.

2 Part 1: Why Quarto

2.1 Why Quarto

2.2 You Already Know Most of This

In the MSDS program you have written **R Markdown** (`.Rmd`):

- YAML header at the top
- Prose in Markdown
- Executable `{r}` chunks
- Knitr weaves code output into the document

Quarto (`.qmd`) is the modern publishing engine built on that same habit. If you can read an `.Rmd`, you are one short migration away from a capstone write-up toolchain.

2.3 One Source, Many Outputs

Rather than maintaining four separate documents, you should maintain **one** Quarto source and render the formats you need for me, peer POs, and employers.

2.4 Executable Documents (Jupyter-Like, R-Native)

Like a **Jupyter notebook**, a Quarto document can **run code at render time** and embed outputs:

- Figures and tables update when data changes
- Readers (and future you) see how results were produced
- Unlike copy-pasting console output into Word, the narrative and computation stay linked

Unlike Jupyter alone, Quarto is built for **long-form technical writing** with sections, cross-references, bibliographies, and multi-format export. That is exactly what a capstone write-up needs.

2.5 Why This Matters for DATA 510

Your write-up rough draft must be:

- A complete report (question, methods, findings, ethics, limitations)
- Written for **online publication**
- Stable after M4 (methods and findings should not keep pivoting)

Quarto gives you **HTML for the web** and **PDF for Canvas** from the same source. Your peer POs can open HTML without guessing which notebook cell mattered.

2.6 This Course Site Is Quarto

Everything at courses.lpcordova.phd/data510 is built with Quarto:

- Website pages from `.qmd`
- These slides from RevealJS output
- Shared themes in `_quarto.yml`

If the toolchain works for an entire course site, it works for your **one** capstone report.

2.7 Quarto Is Multilingual (One Line)

Quarto also runs **Python**, **Julia**, and other engines. Tonight we focus on **R** because that is what this cohort uses daily. The document structure is the same if you add a `{python}` chunk later.

3 Part 2: From Proposal to Write-Up

3.1 Proposal to Write-Up

3.2 The Milestone Document Line

The **proposal** is a planning document. The **write-up** is the permanent record of what you actually did and found.

3.3 Reuse Your Proposal (Do Not Restart)

Your M1 PDF already has prose you can **migrate**:

Proposal section	Write-up section
Problem framing and RQs	Introduction
Impact and stakeholders	Introduction / Impact
Data sources and engineering	Data and data engineering
Ethics and risks	Ethics and responsible use
Planned analysis	Methods and evaluation
(new after you run code)	Results and visualizations
Timeline risks	Discussion and limitations

Add a short **"Changes since M1"** paragraph if your methods or data shifted. Honesty beats silent rewrites.

3.4 Where Files Live in Your DS3 Repo

```
deliverables/
  M1-proposal/           # you submitted PDF here
  M2-data-summary/
  M3-poster-draft/
  M4-writeup-draft/      # writeup.qmd lives here (start now)
```

```
  M5-final/
notebooks/                # exploration and heavy analysis
src/                      # scripts imported or summarized in the write-up
```

Rule of thumb: notebooks hold the messy work; the write-up **summarizes and cites** the final pipeline. Peer POs should not need to run twelve notebooks to grade your story.

3.5 When to Start Quarto (Answer: This Week)

When	What to do in `writeup.qmd`
Week 5 (now)	Skeleton + paste proposal sections; stub Results
Week 7 (M2)	Data section matches stable panel; add QC figures
Week 10 (M3)	Pull key figures from poster into Results
Week 12 (M4)	Full draft; methods and findings **freeze**
Week 14 (M5)	Polish prose, citations, render final HTML + PDF

Starting early means you are not writing twelve sections from a blank page in week 12. Fight me if you want, but the students who start now thank themselves in July.

3.6 Required Write-Up Sections (M4 Rubric)

Your draft must include the following sections:

1. Title, team, affiliations
2. Abstract
3. Introduction (problem and RQs)
4. Impact and stakeholders
5. Related work / background
6. Data and data engineering
7. Ethics and responsible use
8. Methods and evaluation
9. Results and visualizations
10. Discussion, limitations, future work
11. Conclusions and recommendations
12. Reproducibility and references

The skeleton file in this lecture folder mirrors these headings.

3.7 Weekly Habit: Render Before Studio

Before each Iteration Review:

1. `quarto render deliverables/M4-writeup-draft/writeup.qmd`
2. Link the HTML (or PDF) in your `README.md` summary
3. Peer POs read the **rendered** artifact, not your git diff fantasies

4 Part 3: Install and Verify

4.1 Install Quarto

4.2 Two Paths (Use Both)

Path	Best for
Command line (`quarto`)	Reproducible renders, README instructions, CI-style habits
VS Code	Daily authoring, Quarto preview, R chunk run controls

I prefer you **author in VS Code** (or Cursor, same extensions) and **verify in Terminal** before milestone submissions. Same file, two ways to catch path bugs.

4.3 Step 1: Install Quarto CLI (macOS)

1. Download the **macOS** installer from Quarto: Download
2. Open the `.pkg` and complete installation
3. Open Terminal:

```
quarto --version
quarto check
```

`quarto check` confirms R, knitr, and optional PDF tools. Fix anything it flags before week 12 panic.

If you are on Windows or Linux for this course, use the matching installer from the same Quarto installation page. Tonight's lab machines are macOS-first.

4.4 Step 2: PDF Support (TinyTeX)

Canvas milestones often want **PDF**. On macOS:

```
quarto install tinytex
```

Then re-run `quarto check`. First PDF render downloads LaTeX packages; it is slow once, fast after.

4.5 Step 3: VS Code Setup

You likely already use **VS Code** (or Cursor) for Python and git. Add Quarto on top:

1. **Install extensions** (Extensions panel, or Command Palette `Extensions: Install Extensions`):
   - **Quarto** (Publisher: Quarto)
   - **R** (Publisher: REditorSupport) if not already installed
2. **Open your DS3 repo as a folder:** File > Open Folder > select repo root (not a single file floating nowhere)
3. **Create or open** a `.qmd` file (e.g. `deliverables/M4-writeup-draft/writeup.qmd`)
4. **Render from VS Code:**
   - Command Palette: `Quarto: Render` (or click **Render** in the editor toolbar when the Quarto extension recognizes the file)
   - Or integrated terminal: `quarto render path/to/writeup.qmd`

Preview: Command Palette `Quarto: Preview` (live HTML reload), same engine as `quarto preview` in Terminal.

Docs: Quarto VS Code extension

4.6 Step 4: R in VS Code (Quick Check)

For `{r}` chunks to execute at render time you need **R on PATH** and packages **knitr** / **rmarkdown**:

```
install.packages(c("knitr", "rmarkdown"))
```

In VS Code, the R extension can run individual lines or chunks for debugging; **milestone submission** should still come from a full `quarto render` so output matches what graders see.

4.7 Live Demo Checklist

On your machine, right now:

- ☐ `quarto --version` prints 1.x
- ☐ `quarto check` passes R/knitr
- ☐ VS Code Quarto and R extensions installed
- ☐ Repo opened as a **folder**; sample `.qmd` renders to HTML from Terminal or `Quarto: Render`

If your neighbor fails, compare `quarto check` output. Nine times out of ten it is a missing package, missing extension, or wrong working directory.

4.8 Git Hygiene

The DS3 template `.gitignore` already ignores `.quarto/` cache.

Commit: `.qmd` source, rendered PDF/HTML for milestones, figures referenced in the doc.

Do not commit: `.quarto/` scratch, huge raw data, secrets.

5 Part 4: How to Use Quarto

5.1 Quarto Essentials

5.2 Document Anatomy

Every capstone write-up follows this shape (one `.qmd` file):

```
---
title: "Your Project Title"
format:
  html: default
  pdf: default
---

# Introduction
Prose here.

# R chunk here (summary(cars))
```

Three layers: **YAML metadata**, **Markdown prose**, **executable chunks**.

5.3 YAML You Will Actually Use

Key	Purpose
`title`, `subtitle`, `author`	Cover metadata
`format: html` / `pdf`	Output targets (list both for capstone)
`number-sections: true`	Numbered headings in PDF
`toc: true`	Table of contents in HTML/PDF
`bibliography: refs.bib`	Citation database
`execute:`	Global chunk defaults (`echo`, `warning`, `message`)

R Markdown users: `.Rmd` used `output: html_document`. Quarto uses `format: html`. That is the main YAML migration pain point.

5.4 R Code Chunks

Chunk options (capstone essentials):

- `echo: true` shows code (methods transparency)
- `echo: false` hides boilerplate imports
- `fig-cap` + `label:` enables `@fig-scatter` cross-refs
- `cache: true` only for expensive stable chunks (not while debugging)

5.5 Cross-References

In prose: use cross-ref syntax with your figure label (example: `@fig-scatter`).

Tables get `#| label: tbl-summary` and a caption option similarly.

Cross-refs are how you write like a report, not like a folder of images named `final_v3_REALLY_final.png`.

5.6 Citations

1. Create `refs.bib` in the same folder as `writeup.qmd`
2. Add to YAML: `bibliography: refs.bib`
3. Cite inline: `@liu2016` or `@liu2016 [p. 352]`
4. Quarto generates the References section (add `# References` or use `reference-section-title`)

5.7 Include Fragments (Teams)

For parallel editing, split sections:

Each teammate owns a file; one person integrates renders. Optional for teams of three; solo students can ignore this.

5.8 Render Commands

From your repo root (paths adjusted):

```
# One-shot render (all formats in YAML)
quarto render deliverables/M4-writeup-draft/writeup.qmd

# Live HTML preview with reload
quarto preview deliverables/M4-writeup-draft/writeup.qmd
```

VS Code: Command Palette `Quarto: Render` on the open `.qmd` = `quarto render` on that file. Use the integrated terminal when you want the exact command for your README.

5.9 Working Directory Pitfalls

If chunks cannot find `data/processed/panel.csv`:

- **File > Open Folder** on your repo root in VS Code, then render from the integrated terminal at that root
- Or use `here::here("data/processed/panel.csv")`
- Document the render command and working directory in `README.md`

Wrong `wd` is the number one "it works on my laptop" capstone bug.

5.10 Skeleton File

Copy from this lecture:

examples/writeup-skeleton.qmd

Into your repo:

`deliverables/M4-writeup-draft/writeup.qmd`

Paste proposal prose into the marked sections. Leave Results as stubs until M2/M3.

5.11 Hands-On: Block 2 (Git Session)

We **do not** render the skeleton in Block 1. Many of you have not cloned your DS3 repo yet, and rendering belongs with **Git commit and push**.

In Block 2 I will walk through:

1. Clone (or open) your repo
2. Copy the skeleton into `deliverables/M4-writeup-draft/`
3. Render a **PDF** from the skeleton
4. Commit, push, and upload the PDF to **Canvas**

See the Git session slides for step-by-step commands.

5.12 Quick Quiz: Commit Rendered Output?

Should you commit the rendered PDF/HTML for milestone submissions?

```
A. No, only `.qmd` source ever
B. Yes for graded milestones so that Lucas and peer POs open artifacts without rendering
C. Only commit PDF if you use Word
D. Only commit on the final week
```

. . .

B. DS3 convention: source **plus** rendered milestone artifacts in `deliverables/`. I click PDF; you still maintain `.qmd` as source of truth.

6 Part 5: Portfolio Teaser

6.1 Portfolio and Publication

6.2 Why a Public Write-Up

The syllabus expects a **portfolio-quality** artifact: employers should see your question, methods, and findings without joining our Canvas course.

Your final write-up becomes a **permanent public record**, linked from your project website and eventually your professional portfolio.

6.3 Write-Up as Portfolio Centerpiece

One project page tells the story; the Quarto write-up is the depth layer.

6.4 Quarto Website Projects (Preview)

Later you will create a **Quarto website** (separate project):

```
quarto create-project my-portfolio --type website
```

That gives you `index.qmd`, `_quarto.yml`, and a `projects/` folder for case studies. Your capstone write-up becomes **one project entry** with summary, hero figure, and links.

6.5 GitHub Pages (One Sentence)

Render the site to `_site/` or `docs/`, push to GitHub, enable **Pages** on that folder. Your write-up HTML becomes a public URL. Full walkthrough is the **next** technical writing / portfolio session.

6.6 What to Do This Week

1. **Tonight (Block 2):** render skeleton PDF, commit to repo, submit PDF on Canvas (Git session)
2. Paste proposal sections into Introduction / Data / Ethics / Methods stubs when you have time
3. Link the rendered HTML or repo path in your next Iteration Review
4. Keep coding in `notebooks/`; summarize in the write-up as results mature

7 Wrap-Up

7.1 Before Block 2

7.2 Tonight's Checklist

- ☐ `quarto check` passes on your machine
- ☐ VS Code Quarto extension installed
- ☐ You can explain **one source, many outputs** in one sentence
- ☐ You know where M4 write-up lives (`deliverables/M4-writeup-draft/`)
- ☐ You know Block 2 covers **Git + skeleton PDF render + Canvas upload**

Block 2: Git session first, then data engineering consultations. Bring your GitHub repo URL (or be ready to create one from the DS3 template).

7.3 Coming Next

Follow-up lecture: **building your portfolio site** around this write-up (Quarto website, GitHub Pages, project landing page). Today you learned the engine; next time we ship the showcase.

8 References

8.1 References

1. Quarto Project. (2024). *Quarto Guide.* https://quarto.org/docs/guide/
2. Xie, Y., Allaire, J. J., & Grolemund, G. (2018). *R Markdown: The Definitive Guide.* https://bookdown.org/yihui/rmarkdown/
3. Cordova, L. P. DATA 510 Write-Up Rough Draft. https://courses.lpcordova.phd/data510/project/write-up-rough-draft.html
4. Cordova, L. P. DATA 510 Project Website and Dissemination. https://courses.lpcordova.phd/data510/project/project-website.html
5. Quarto Project. (2024). *Visual Studio Code.* https://quarto.org/docs/tools/vscode.html