Project Update – 08/07/2021

# What you did and created?

This last week:

* Had a call with Hannah:
  + Discussed Database design, especially the key point not to repeat data.
  + They shared the books and sources they have been using to develop their database.
  + I explained my intended approach to handling all my data in multiple data frames. Generally discussed the benefits of storing data in R and CSVs vs in a SQLDB
  + Discussed a potential ID generation issues Hannah was having with their database.
  + Discussed RShiny and what Hannah had found so far using it (coding, maintaining, outputs)
* Started reviewing the books Hannah had shared in particular: *Database design for mere mortals*.
* Arranged a meeting with Amelie for next Monday to compare notes on projects and in particular data scraping.
* Gateway to Research Data Collection
  + Implemented error handling, logging, and progress monitoring functionality in the code to better understand and handle fail data collection attempts.
  + Ran Gateway to Research data collection script successfully generating data for almost 30000 projects with only 6 failed projects.
* Grants on the Web
  + Continued reviewing the different panel page types, reviewing the different layouts and class tags used.

Before this week:

* Developed, tested, and ran Theme’s data collection.
* Started to develop and test various methods of collecting data from panels.
  + Reviewed different panel page layouts and used the rvest Selector Gadget tool to identify common table names and page layouts.
* Started to develop and did several test-runs of collecting data from the Gateway to Research API.
  + Explored JSON document layout and navigating the output from the fromJSON() function, parsing it into the relevant tables.

# What decision have you made and how they were made?

* Decided to implement error catching, progress monitoring, and logging of the data scraping functionality. This is because the script would typically take 3-4hrs to run and an error halfway through would interrupt and stop the progress. These meant that:
  + Errors were handled in the code and problematic links were saved for later investigation.
  + The progress of the program could be monitored with a progress bar that showed the precent completion,
* Added a 5-minute wait after an error occurred on a connection request. This was because smaller wait times appeared to lead to more errors occurring.
* Implemented a wait time after each successful json request that was the same as the response time. This was to prevent requests being sent over too quickly and overloading the website while also keeping a reasonable response time.
* Reviewed use of R and CSVs to store data over using SQL. Decided to continue using R and CSVs:
  + Unfamiliar with setting up and running a SQL database so would require extra time and efforts.
  + The benefits provided by SQL (stricter architecture, data types) would be less useful for my project where I am creating and entering the data and not an end user (like in Hannah’s project)

# Relevant References

|  |  |  |
| --- | --- | --- |
| **Name** | **Link** | **Notes** |
| Advanced R | http://adv-r.had.co.nz/Exceptions-Debugging.html#condition-handling | Used to develop error handling functionality |
| Stack Overflow: Advanced TryCatch statements | https://stackoverflow.com/questions/12193779/how-to-write-trycatch-in-r | Used to develop complex error handling functionality |
| GtR-1-API-v3.3 | <https://gtr.ukri.org/resources/GtR-1-API-v3.3.pdf> | Used to develop API access expressions |
| Progress Package Readme | https://github.com/r-lib/progress#readme | Used vinettes/tutorial to develop progress bar functionality |
| Rvest | https://rvest.tidyverse.org/ | Used to develop scaping from HTML documents functionality |
| JSONLITE | <https://cran.r-project.org/web/packages/jsonlite/vignettes/json-aaquickstart.html> | Used to request and convert JSON files into dataframes |

# To do items for coming week

* Create a database schema and review the data needed in each table.
* Complete Panel data collection (code development, testing, and running)
* Prepare person data for gender prediction.
* Start reviewing and cleaning the data.