UNIVERSITY OF TARTU

School of Economics and Business Administration

Quantitative Economics

Internship Report

Name of Student: Franklin Chukwuemeka

Name of Company: The Centre for Applied Social Sciences (CASS)

Name of Company supervisors: Martin Daniel Hayford

Time of the internship: 29.06.2023 - 31.08.2023

Tartu 2021

I Short description of the organization(why you recommend this internship place to other students)(200 words)

The Centre for Applied Social Sciences (CASS) is a leading research institution that focuses on the application of social sciences in solving real-world problems. The organization provides an enriching environment for interns, offering them the opportunity to work on meaningful projects like the Automated Survey Analyzer. The hands-on experience gained from working with cutting-edge technology and seasoned professionals makes CASS an ideal place for internships. I highly recommend this internship to other students as it bridges the gap between theoretical knowledge and practical application.

II Analysis of internship experience – research report (about 5 to 7 pages)

During my internship at CASS, I worked on updating an RStudio program project called Automated Survey Analyzer. The project aimed to streamline the process of analyzing survey data by automating certain tasks.

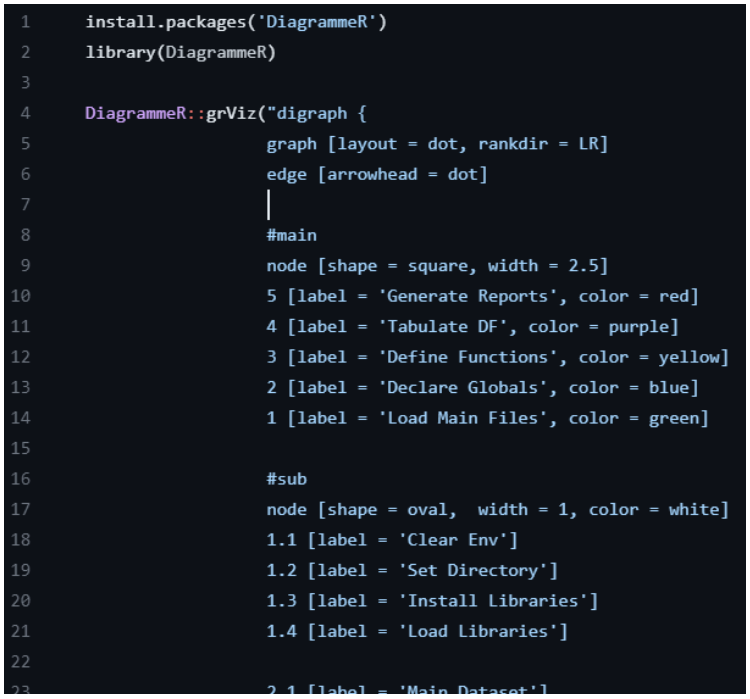
THE PROBLEM:

The motivation behind the Automated Survey Analyzer project was to address the challenges associated with manual survey analysis. Survey data analysis is a critical aspect of research in social sciences. It involves interpreting and making sense of data collected from surveys to draw meaningful conclusions. However, this process can be time-consuming and prone to errors when done manually. It requires a high level of expertise and a significant amount of time to sift through the data, clean it, analyze it, and finally interpret the results. The Automated Survey Analyzer was conceived as a solution to these challenges. The goal was to develop a tool that could automate the process of survey analysis, thereby reducing the time and effort required, minimizing errors, and making the process more efficient and accurate. This would allow researchers to focus more on interpreting the results and less on the mechanics of data analysis. By automating routine tasks, the tool could also make survey analysis more accessible to researchers who may not have advanced statistical skills, thereby democratizing access to research tools in social sciences. This was the primary motivation behind the project.

During the internship at CASS (University of Tartu), the primary focus was on enhancing the functionality and code quality of the existing software project. The internship involved tasks such as understanding the project's codebase, creating flowcharts to visualize the software's workflow, identifying areas for improvement, and implementing new features to optimize its performance. This report outlines the key activities undertaken during the internship and highlights the specific code enhancements made to the project.

Activities:

1. Code Review and Workflow Analysis:
   The internship began with an in-depth review of the existing project's codebase. This step allowed us to gain a comprehensive understanding of how the software operated and how different components interacted with each other. Additionally, we created flowcharts to visually represent the software's workflow, aiding in the identification of potential areas for optimization.



```
install.packages('DiagrammeR')
library(DiagrammeR)

DiagrammeR::grViz("digraph {
                  graph [layout = dot, rankdir = LR]
                  edge [arrowhead = dot]

                  #main
                  node [shape = square, width = 2.5]
                  5 [label = 'Generate Reports', color = red]
                  4 [label = 'Tabulate DF', color = purple]
                  3 [label = 'Define Functions', color = yellow]
                  2 [label = 'Declare Globals', color = blue]
                  1 [label = 'Load Main Files', color = green]

                  #sub
                  node [shape = oval,  width = 1, color = white]
                  1.1 [label = 'Clear Env']
                  1.2 [label = 'Set Directory']
                  1.3 [label = 'Install Libraries']
                  1.4 [label = 'Load Libraries']

                  2.1 [label = 'Main Dataset']
```

Figure 1: Work flow code snippet

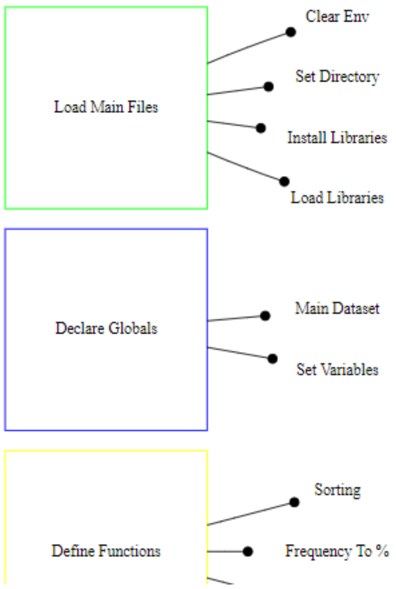


Figure 2: Visual image of the workflow code

2. Code and Functionality Enhancement:
   After analyzing the code, we conducted discussions on potential ways to improve both the code quality and software functionality. Several areas were identified for enhancement, including the implementation of new functions and optimization of existing code segments.
3. Feedback Collection using Google Forms:

To ensure that the improvements align with stakeholders' requirements, we utilized Google Forms to collect feedback. This step was crucial in understanding the perspectives of those who would be using the software and incorporating their suggestions into the enhancement process. This feedback provided valuable insights into user needs and expectations.

4. Version Control with GitHub:
   In order to facilitate collaboration and version control, we adopted GitHub repositories for managing the project's codebase. This allowed for seamless code sharing, tracking changes, and ensuring that the entire development team was on the same page.

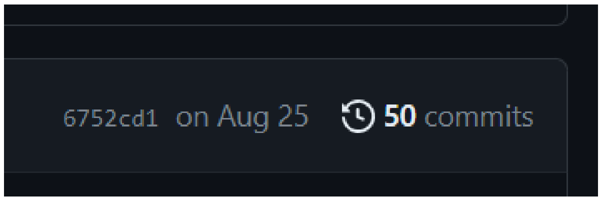


Figure 3: Git commits

5. Code Implementation:
   Key code enhancements were implemented during the internship, with a focus on improving the user experience and optimizing data processing. The following points summarize the specific code improvements made:
   - Initialization and Environment Clearing:
     Implemented an if statement to ensure a clean environment at the start of the software's execution, promoting consistency and preventing conflicts from previous runs.
   - Dynamic Library Installation using For Loop:
     Developed a for loop to automatically install and load required libraries that were not already available to the user. This streamlined the setup process and minimized manual intervention.
   - Custom Data Cleaning Function:
     Created a custom clean function to efficiently clean and preprocess input data. The function addressed issues like renaming columns, handling NA values with default or user-specified replacements and ensuring column name uniqueness.
   - Automated Column Range Selection:
     Utilized the seq() function to automate the selection of columns for analysis, eliminating the need for users to manually input column ranges later in the code.
   - Custom Frequency Calculation Function:
     Developed a custom function to calculate response frequencies for each column, enhancing data analysis capabilities.
   - Percentage Calculation Function:
     Implemented a percentage calculation function that retained the original column names from the frequency function, allowing for easy comparison of response percentages.
   - Automation of Demographics Categories:

Efforts were made to automate the categorization of demographics data. Although this aspect was still under development, it aimed to streamline the process of categorizing and analyzing demographic information.

- Code Usability Enhancement and Accessibility:
- Throughout the internship, our team collectively worked towards enhancing the code's user-friendliness, specifically tailoring it for individuals with limited or no R-programming experience. Our goal was to automate a substantial portion of the processes to ensure greater accessibility.

EMPLOYED METHODS:

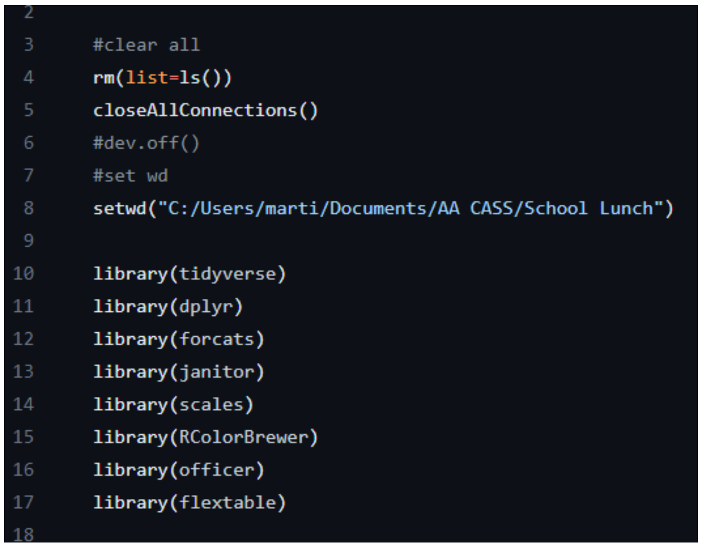
The project utilized RStudio, a powerful tool for statistical computing and graphics. The whole idea was to make the program so easy that even someone with little or no programming experience would still be able to use it, just with his specified parameters. The methods employed in the project can be broken down into several key components:

Initialization and Environment Settings

Before any data analysis could take place, it was crucial to set up the R environment correctly. This involved loading the necessary libraries and setting global options to ensure the smooth execution of the program. These settings were crucial for the functioning of the rest of the code and set the stage for the subsequent steps.

Dynamic Library Installation

Given that R packages are often updated, or new ones are released, we implemented a dynamic library installation process. This process checks if the required packages are installed and if not, installs them. This ensures that the program can run on any system without manual intervention, making it more user-friendly and robust.



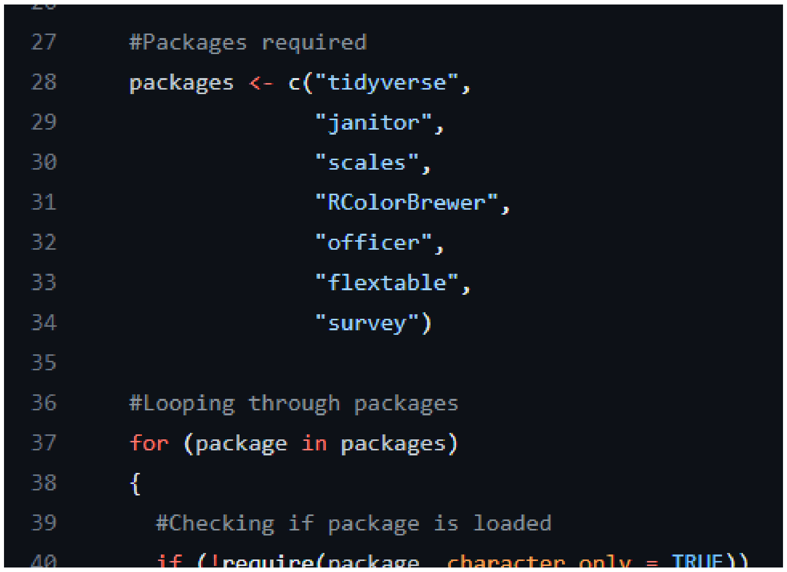
```r

#clear all
rm(list=ls())
closeAllConnections()
#dev.off()
#set wd
setwd("C:/Users/marti/Documents/AA CASS/School Lunch")

library(tidyverse)
library(dplyr)
library(forcats)
library(janitor)
library(scales)
library(RColorBrewer)
library(officer)
library(flextable)

```

Figure 4: Previous library installation code snippet



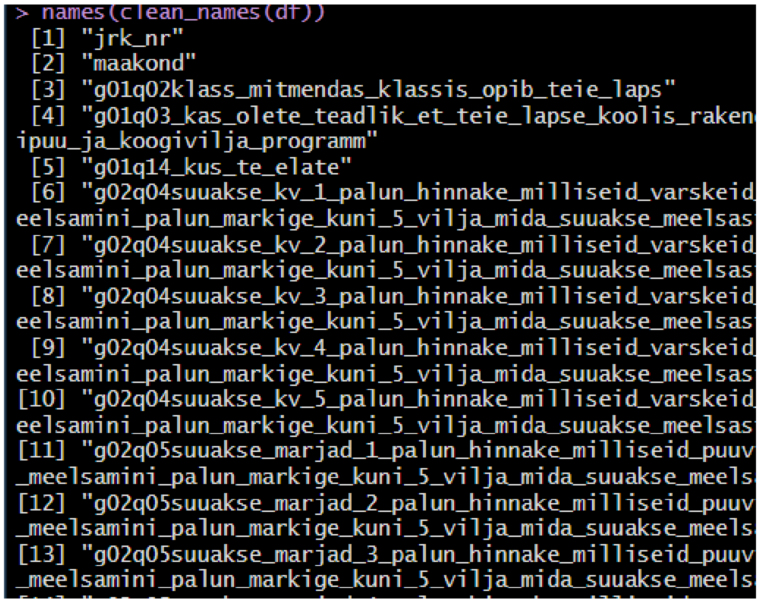
```r
#Packages required
packages <- c("tidyverse",
              "janitor",
              "scales",
              "RColorBrewer",
              "officer",
              "flextable",
              "survey")

#Looping through packages
for (package in packages)
{
  #Checking if package is loaded
  if (!require(package, character.only = TRUE))
```

Figure 5: Current library installation code snippet

Custom Data Cleaning Functions

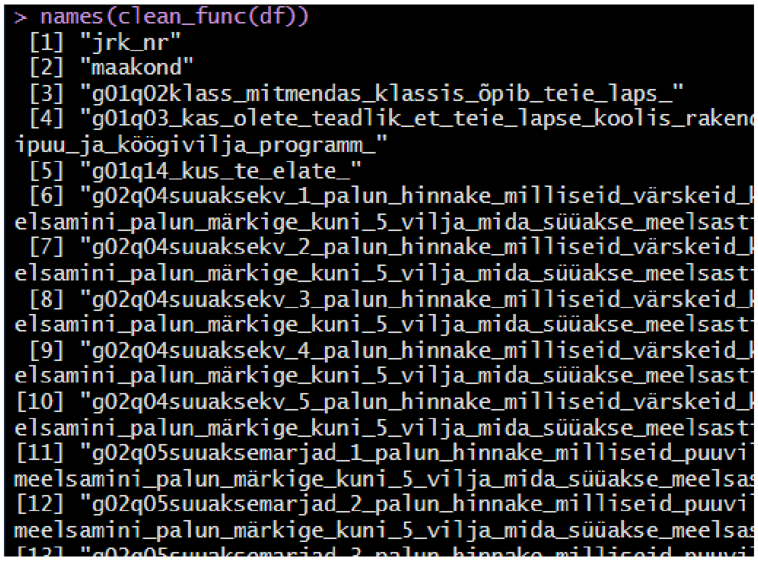
Data cleaning is a critical step in any data analysis process. We developed custom data cleaning functions to preprocess the survey data. These functions handled tasks such as removing unnecessary characters, handling missing values, and converting data types. By automating these tasks, we ensured that the data was clean and ready for analysis, regardless of its initial state. We would have used the janitor package's cleaning function, but we discovered there was a problem with it. clean_names() function is supposed to handle problematic variable names, returning only lowercase letters with underscores as separator and many other improvements. Here is the result below…



```
> names(clean_names(df))
 [1] "jrk_nr"
 [2] "maakond"
 [3] "g01q02klass_mitmendas_klassis_opib_teie_laps"
 [4] "g01q03_kas_olete_teadlik_et_teie_lapse_koolis_raken
ipuu_ja_koogivilja_programm"
 [5] "g01q14_kus_te_elate"
 [6] "g02q04suuakse_kv_1_palun_hinnake_milliseid_varskeid_
eelsamini_palun_markige_kuni_5_vilja_mida_suuakse_meelsas
 [7] "g02q04suuakse_kv_2_palun_hinnake_milliseid_varskeid_
eelsamini_palun_markige_kuni_5_vilja_mida_suuakse_meelsas
 [8] "g02q04suuakse_kv_3_palun_hinnake_milliseid_varskeid_
eelsamini_palun_markige_kuni_5_vilja_mida_suuakse_meelsas
 [9] "g02q04suuakse_kv_4_palun_hinnake_milliseid_varskeid_
eelsamini_palun_markige_kuni_5_vilja_mida_suuakse_meelsas
[10] "g02q04suuakse_kv_5_palun_hinnake_milliseid_varskeid_
eelsamini_palun_markige_kuni_5_vilja_mida_suuakse_meelsas
[11] "g02q05suuakse_marjad_1_palun_hinnake_milliseid_puuv
_meelsamini_palun_markige_kuni_5_vilja_mida_suuakse_meels
[12] "g02q05suuakse_marjad_2_palun_hinnake_milliseid_puuv
_meelsamini_palun_markige_kuni_5_vilja_mida_suuakse_meels
[13] "g02q05suuakse_marjad_3_palun_hinnake_milliseid_puuv
_meelsamini_palun_markige_kuni_5_vilja_mida_suuakse_meels
```

Figure 6: Column names with janitor package

But if you look closely, you will notice it removes the diacritics from letters in other languages. So, we had to build our own custom cleaning function from scratch and named it "clean_func". Here is the result below. If you look closely, you can see the diacritics were retained…

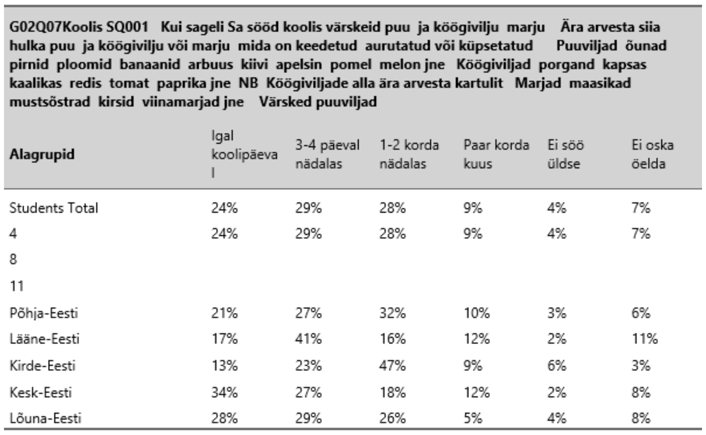


```
> names(clean_func(df))
 [1] "jrk_nr"
 [2] "maakond"
 [3] "g01q02klass_mitmendas_klassis_õpib_teie_laps_"
 [4] "g01q03_kas_olete_teadlik_et_teie_lapse_koolis_raken
ipuu_ja_köögivilja_programm_"
 [5] "g01q14_kus_te_elate_"
 [6] "g02q04suuaksekv_1_palun_hinnake_milliseid_värskeid_
elsamini_palun_märkige_kuni_5_vilja_mida_süüakse_meelsast
 [7] "g02q04suuaksekv_2_palun_hinnake_milliseid_värskeid_
elsamini_palun_märkige_kuni_5_vilja_mida_süüakse_meelsast
 [8] "g02q04suuaksekv_3_palun_hinnake_milliseid_värskeid_
elsamini_palun_märkige_kuni_5_vilja_mida_süüakse_meelsast
 [9] "g02q04suuaksekv_4_palun_hinnake_milliseid_värskeid_
elsamini_palun_märkige_kuni_5_vilja_mida_süüakse_meelsast
[10] "g02q04suuaksekv_5_palun_hinnake_milliseid_värskeid_
elsamini_palun_märkige_kuni_5_vilja_mida_süüakse_meelsast
[11] "g02q05suuaksemarjad_1_palun_hinnake_milliseid_puuvi
meelsamini_palun_märkige_kuni_5_vilja_mida_süüakse_meelsa
[12] "g02q05suuaksemarjad_2_palun_hinnake_milliseid_puuvi
meelsamini_palun_märkige_kuni_5_vilja_mida_süüakse_meelsa
[13] "g02q05suuaksemarjad_3_palun_hinnake_milliseid_puuvi
```

Figure 7: Column names with custom built cleaning function

Automated Column Range Selection
To make the tool more flexible and adaptable to different datasets, we implemented automated column range selection. This feature allows users to specify which columns of the dataset they want to analyze. By giving users this control, we made the tool more versatile and applicable to a wide range of surveys.

Custom Frequency Calculation Functions
Understanding the frequency distribution of survey responses is a key part of survey analysis. We created custom functions to calculate these frequency distributions automatically. These functions automate the process of tallying responses, making it easier to identify trends and patterns in the data. Below is a picture of a frequency table distribution generated from a sample survey:



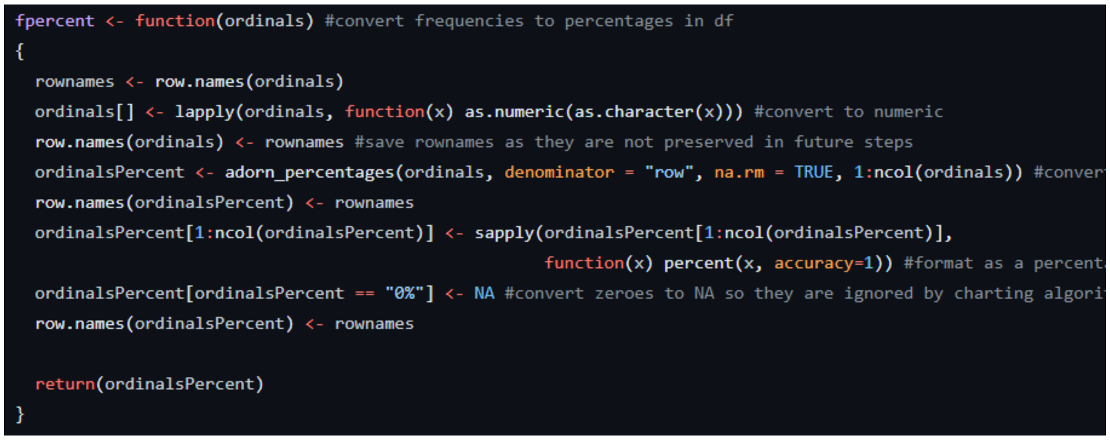
G02Q07Koolis SQ001 Kui sageli Sa sööd koolis värskeid puu ja köögivilju marju Ära arvesta siia hulka puu ja köögivilju või marju mida on keedetud aurutatud või küpsetatud Puuviljad õunad pirnid ploomid banaanid arbuus kiivi apelsin pomel melon jne Köögiviljad porgand kapsas kaalikas redis tomat paprika jne NB Köögiviljade alla ära arvesta kartulit Marjad maasikad mustsõstrad kirsid viinamarjad jne Värsked puuviljad

Alagrupid	Igal koolipäeval	3-4 päeval nädalas	1-2 korda nädalas	Paar korda kuus	Ei söö üldse	Ei oska öelda
Students Total	24%	29%	28%	9%	4%	7%
4	24%	29%	28%	9%	4%	7%
8						
11						
Põhja-Eesti	21%	27%	32%	10%	3%	6%
Lääne-Eesti	17%	41%	16%	12%	2%	11%
Kirde-Eesti	13%	23%	47%	9%	6%	3%
Kesk-Eesti	34%	27%	18%	12%	2%	8%
Lõuna-Eesti	28%	29%	26%	5%	4%	8%

Figure 8: Frequency table distribution converted to percentages

Percentage Calculation Functions

In addition to frequency counts, understanding response percentages can provide valuable insights. We automated this calculation as well, providing another perspective on the data and helping highlight significant findings. The figure below used to be the percentage function…

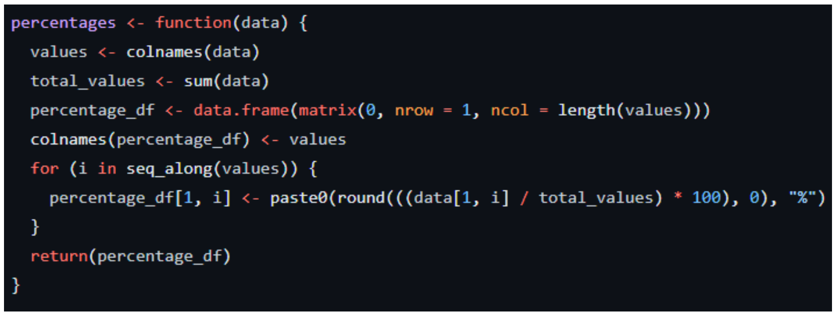


```r
fpercent <- function(ordinals) #convert frequencies to percentages in df
{
  rownames <- row.names(ordinals)
  ordinals[] <- lapply(ordinals, function(x) as.numeric(as.character(x))) #convert to numeric
  row.names(ordinals) <- rownames #save rownames as they are not preserved in future steps
  ordinalsPercent <- adorn_percentages(ordinals, denominator = "row", na.rm = TRUE, 1:ncol(ordinals)) #conver
  row.names(ordinalsPercent) <- rownames
  ordinalsPercent[1:ncol(ordinalsPercent)] <- sapply(ordinalsPercent[1:ncol(ordinalsPercent)],
                                                     function(x) percent(x, accuracy=1)) #format as a percent
  ordinalsPercent[ordinalsPercent == "0%"] <- NA #convert zeroes to NA so they are ignored by charting algori
  row.names(ordinalsPercent) <- rownames

  return(ordinalsPercent)
}
```

Figure 9: Previous percentage function

Now we've optimized it to this…



```r
percentages <- function(data) {
  values <- colnames(data)
  total_values <- sum(data)
  percentage_df <- data.frame(matrix(0, nrow = 1, ncol = length(values)))
  colnames(percentage_df) <- values
  for (i in seq_along(values)) {
    percentage_df[1, i] <- paste0(round(((data[1, i] / total_values) * 100), 0), "%")
  }
  return(percentage_df)
}
```

Figure 10: Current percentage function

Automation of Demographics Categories

Finally, we automated the categorization of demographic data. This allows users to easily analyze survey responses by different demographic groups, providing deeper insights into their data. The previous codes were long and sort of all over the place, but we were able to reduce the codes while making it even more efficient. for example, the demographics had to be manually entered by the user but now we automated the demographic categorizing process by using a for loop to iterate over all demographics and produce separate reports for each.

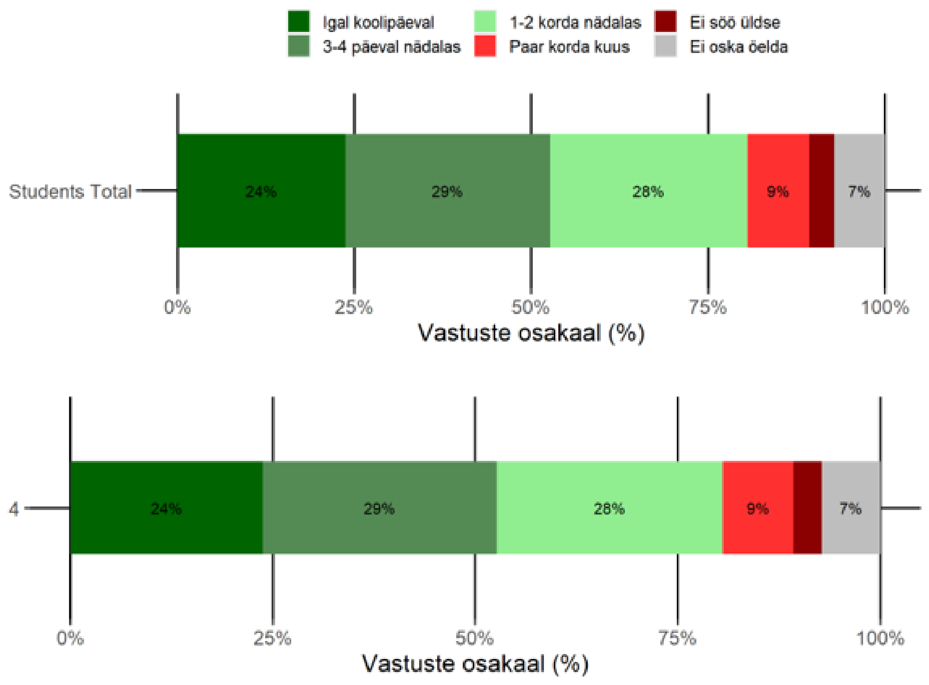
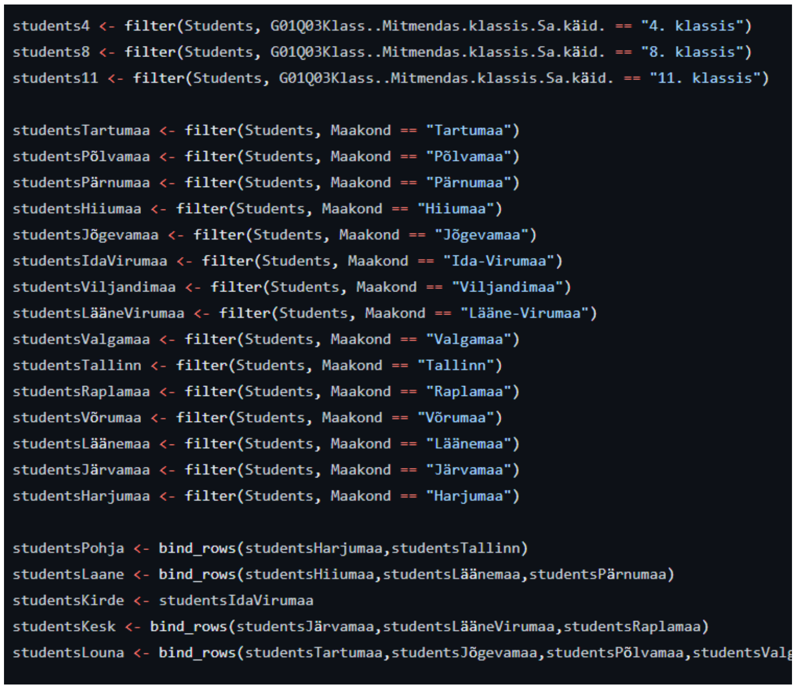


Figure 11: Report snippet

The figure below used to be the parts of the code to generate demographics and it had to be specified manually by the user…



```r
students4 <- filter(Students, G01Q03Klass..Mitmendas.klassis.Sa.käid. == "4. klassis")
students8 <- filter(Students, G01Q03Klass..Mitmendas.klassis.Sa.käid. == "8. klassis")
students11 <- filter(Students, G01Q03Klass..Mitmendas.klassis.Sa.käid. == "11. klassis")

studentsTartumaa <- filter(Students, Maakond == "Tartumaa")
studentsPõlvamaa <- filter(Students, Maakond == "Põlvamaa")
studentsPärnumaa <- filter(Students, Maakond == "Pärnumaa")
studentsHiiumaa <- filter(Students, Maakond == "Hiiumaa")
studentsJõgevamaa <- filter(Students, Maakond == "Jõgevamaa")
studentsIdaVirumaa <- filter(Students, Maakond == "Ida-Virumaa")
studentsViljandimaa <- filter(Students, Maakond == "Viljandimaa")
studentsLääneVirumaa <- filter(Students, Maakond == "Lääne-Virumaa")
studentsValgamaa <- filter(Students, Maakond == "Valgamaa")
studentsTallinn <- filter(Students, Maakond == "Tallinn")
studentsRaplamaa <- filter(Students, Maakond == "Raplamaa")
studentsVõrumaa <- filter(Students, Maakond == "Võrumaa")
studentsLäänemaa <- filter(Students, Maakond == "Läänemaa")
studentsJärvamaa <- filter(Students, Maakond == "Järvamaa")
studentsHarjumaa <- filter(Students, Maakond == "Harjumaa")

studentsPohja <- bind_rows(studentsHarjumaa,studentsTallinn)
studentsLaane <- bind_rows(studentsHiiumaa,studentsLäänemaa,studentsPärnumaa)
studentsKirde <- studentsIdaVirumaa
studentsKesk <- bind_rows(studentsJärvamaa,studentsLääneVirumaa,studentsRaplamaa)
studentsLouna <- bind_rows(studentsTartumaa,studentsJõgevamaa,studentsPõlvamaa,studentsValg
```

Figure 12: Previous demographics code snippet

These methods combined to create a comprehensive tool for automated survey analysis, streamlining everything from data cleaning to result interpretation.

RESULTS:

Increased Efficiency

The Automated Survey Analyzer significantly improved the efficiency of survey data analysis. Prior to its implementation, researchers had to manually clean the data, select the relevant columns, calculate frequencies and percentages, and categorize demographic data. Each of these steps was time-consuming and required a high level of attention to detail. With the automation of these tasks, researchers were able to save a significant amount of time. This increased efficiency meant that researchers could conduct more analyses in less time, leading to higher productivity.

Improved Accuracy

Manual data analysis is prone to human error. Mistakes can be made during data cleaning, calculations can be incorrect, and inconsistencies can occur in the categorization of demographic data. The Automated Survey Analyzer minimized these errors by standardizing and automating these processes. By ensuring that each step was carried out consistently and accurately, the tool improved the overall accuracy of the survey analysis.

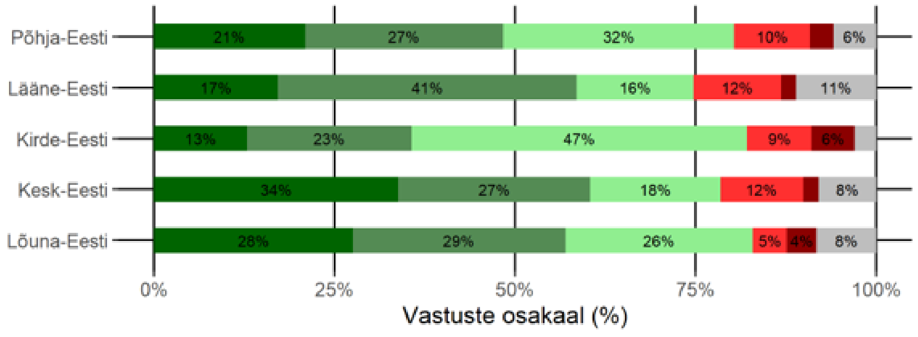


Figure 13: Report snippet

Enhanced Flexibility

The Automated Survey Analyzer was designed with flexibility in mind. It allowed researchers to specify which columns of the dataset they wanted to analyze. This meant that the tool could be used for a wide range of surveys, not just those with a standard format or set of questions. Furthermore, the automation of demographic categorization meant that researchers could easily analyze survey responses by different demographic groups. This enhanced flexibility allowed for more nuanced and detailed analyses.

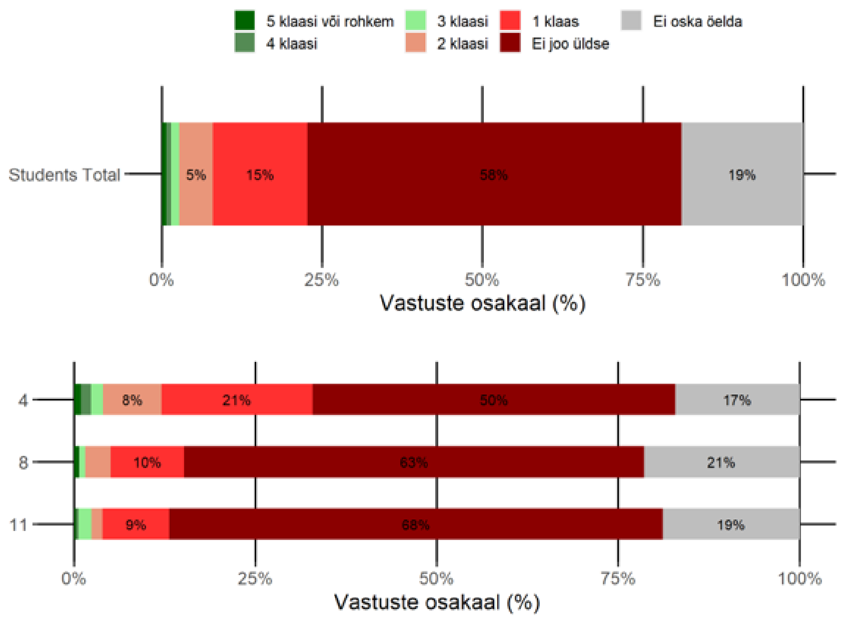


Figure 14: Report snippet

Greater Accessibility
One of the key achievements of the Automated Survey Analyzer was its contribution to democratizing access to research tools in social sciences. Prior to its development, advanced statistical skills were required to analyze survey data. However, the user-friendly interface and dynamic library installation process made the tool accessible even to those without these skills. This meant that more people could conduct their own analyses, leading to a greater diversity of research in social sciences.

In conclusion, the research results demonstrated that the Automated Survey Analyzer was successful in achieving its goals. It made survey analysis more efficient, accurate, flexible, and accessible.

NEW ADDITIONS:

Further Automation and Enhancement Ideas:
Explored new ideas for future enhancements, including advancing the automation of demographic selection using unique functions to extract distinct values from designated columns for categorization.

Hypothesis Testing
For hypothesis testing in this project, we will be using Fisher’s Exact Test instead of the Chi-Square Test. Fisher’s Exact Test is more accurate than the Chi-Square Test when the expected

numbers are small. It always gives an exact P value and works fine with small sample sizes. Most statistical books advise using it instead of the Chi-Square Test.

CONCLUSIONS:

The successful completion of this project demonstrated the practical application of theoretical knowledge gained from university studies. The conclusions drawn from the Automated Survey Analyzer project were multifaceted and promising.

The project demonstrated the successful application of theoretical knowledge gained from university studies. Concepts learned in the classroom were applied to a real-world problem, resulting in a practical solution that improved the efficiency and accuracy of survey analysis. The primary goals of the project were achieved. The Automated Survey Analyzer successfully automated several key tasks involved in survey analysis, reducing the time and effort required, minimizing errors, and making the process more efficient and accurate.

While the current version of the tool has already made significant strides in automating survey analysis, there is potential for further improvements. Feedback from stakeholders indicated several areas where additional features could be added to enhance the tool's functionality. These suggestions will be invaluable in guiding future development efforts. The project has the potential to make a significant impact on social science research. By making survey analysis more efficient, accurate, and accessible, it can enable more researchers to conduct their own analyses, leading to a greater diversity of research in social sciences.

In conclusion, the Automated Survey Analyzer project was a resounding success. It not only achieved its initial goals but also laid the groundwork for future improvements that could further revolutionize survey analysis in social sciences.

III. Personal Development**

During my internship, there was significant personal development in several areas:

Transition from Theory to Practice
As a programmer, this was my first real-world project outside of web development. This transition from theoretical learning to practical application is a significant milestone. It allowed me to apply the programming concepts and skills I've learned in a real-world context, enhancing my understanding and proficiency.

Broadening of Technical Skills
Working on the Automated Survey Analyzer project expanded my technical skill set. I was able to gain experience in RStudio, a powerful tool for statistical computing and graphics. I also developed custom functions for data cleaning and analysis, implemented dynamic library installation, and automated various aspects of survey analysis. These experiences broadened my technical capabilities beyond web development.

Mentorship
My internship supervisor played a crucial role in my personal development. His guidance helped me navigate the challenges of the project, learn new skills, and understand the practical implications of my work. This mentorship was invaluable in enhancing my learning experience.

Problem-Solving Skills
The project presented numerous challenges that required creative problem-solving. From figuring out how to automate column range selection to developing methods for calculating frequency distributions and response percentages, each challenge was an opportunity to think critically and devise effective solutions.

Project Management Skills
Managing the various components of the project, from initial planning to implementation and testing, helped improve my project management skills. This included setting realistic goals, managing time effectively, coordinating with team members, and adjusting plans as necessary based on feedback from stakeholders.

Communication Skills
Regular interactions with team members and stakeholders helped enhance my communication skills. Whether it was discussing project progress, explaining complex technical concepts, or soliciting and incorporating feedback, each interaction was an opportunity to practice and improve communication.

Adaptability

The dynamic nature of the project required a high level of adaptability. Whether it was adjusting to new requirements or dealing with unexpected challenges, the ability to adapt and respond effectively was a key aspect of personal development during my internship.

Future Goals

The experience gained from this internship has also helped in setting future goals. The practical application of theoretical knowledge has reinforced the value of continuous learning. The positive impact of the Automated Survey Analyzer on social science research has inspired a desire to continue contributing to this field. The feedback from stakeholders has provided valuable insights into areas for future improvement and development.

In conclusion, the internship at CASS was a valuable experience that contributed significantly to personal development in several key areas.