User Manual

Table of Contents

User Manual

1. Introduction

1.1 Purpose of the Application

The application is designed to provide users with generic solvers for quadratic spline interpolation and polynomial regression, and a simplex implementation for the diet optimization problem. It consists of an RShiny application and a shinyApp web application.

1.2 Key Features

- Application Implementations
  - RShiny application
  - shinyApp Web Application
- Generic Solvers
  - Quadratic Spline Interpolation
  - Polynomial Regression
- Simplex Implementation
  - Diet Problem Solver

1.3 Dependencies

- RShiny Application

  Ensure that the following dependencies are installed on your system before using the application:
  - R (version 4.3.0 or later) and RStudio (version v2023.04 or later)
  - R Packages: shiny, DataTables (DT), shinyWidget, and shinydashboard

- shinyApp Web Application

  There are no known dependencies required to access the shinyApp web application.

User Manual

2. Getting Started

2.1 System Requirements

Before using the RShiny application and the shinyApp web application, ensure that your system meets the following minimum requirements:

- Software Requirements:
  - *Operating System:* Windows 10, macOS 10.13, Ubuntu 18.04, or a compatible operating system
  - *Web Browser:* Google Chrome (recommended), Mozilla Firefox, Safari, or Microsoft Edge (latest versions)
- Internet Connection (shinyApp web application):
  - *Stable Internet Connection:* A stable internet connection is required to access and use the web application. The application may involve data retrieval or real-time processing that depend on having access to an active internet connection.
- R Environment (RShiny Application):
  - *R Version:* R 4.3.0 or later
  - *RStudio Version:* RStudio v2023.04 or later
  - *Packages:*
    - Shiny (version 1.7.5 or later),
    - DataTables (version 0.30 or later)
    - shinyWidgets (version 0.8.0 or later)
    - shinydashboard (version 0.7.2 or later)

User Manual

2.2 Installation

- RShiny Application

  After ensuring that the minimum system requirements are met, to install the application, follow these steps:

  1. From the .zip file, extract all the files to a folder of your choice.
  2. Ensure that the files are complete by checking if the following .R files and .csv file are present in the folder:
     a. init.R
     b. run.R
     c. server.R
     d. ui.R
     e. nutritional_values.csv
  3. Proceed with accessing the application.

- shinyApp Web Application

  There are no required installation steps for the shinyApp web application.

2.3 Accessing the Application

- RShiny Application

  To access the application, open either the *server.R* or *ui.R file* in RStudio and click the Run App button located at the top-right corner of the source pane. Alternatively, assuming the shiny package is installed, you may open the application through the terminal using the following commands:

  ```
  > setwd(dirname(rstudioapi::getSourceEditorContext()$path))
  > library(shiny)
  > runApp()
  ```

- shinyApp Web Application

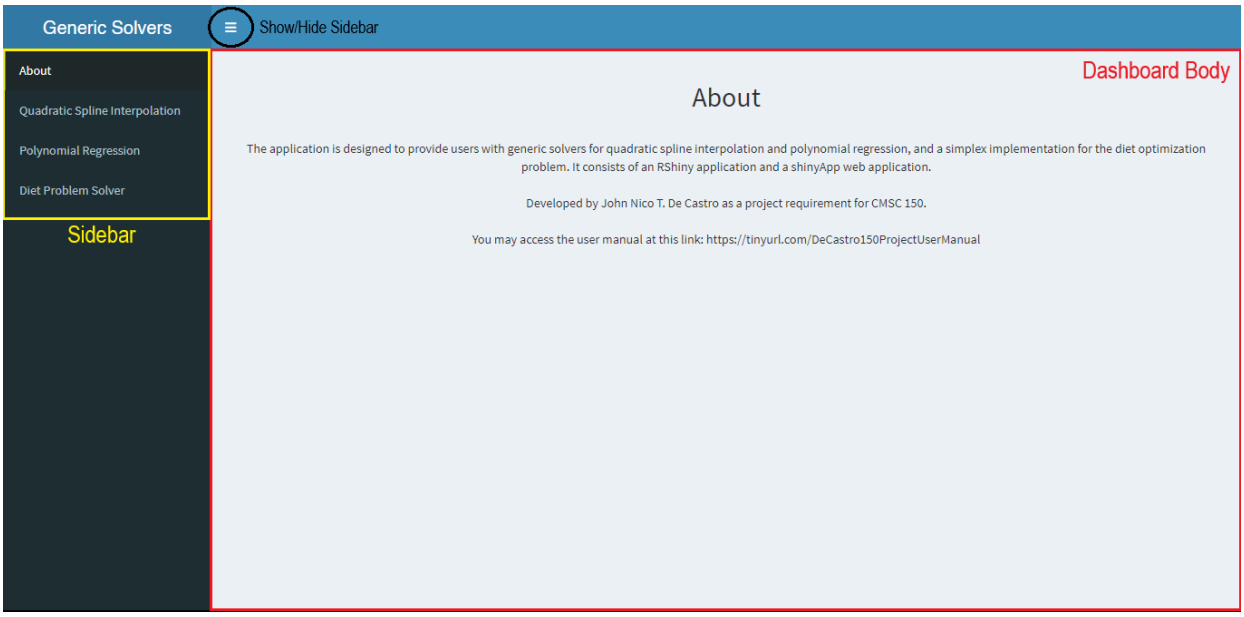
  The shinyApp web application can be accessed at the link below:

  - https://okkinn.shinyapps.io/CMSC150Project/

User Manual

2.4 Parts of the Application

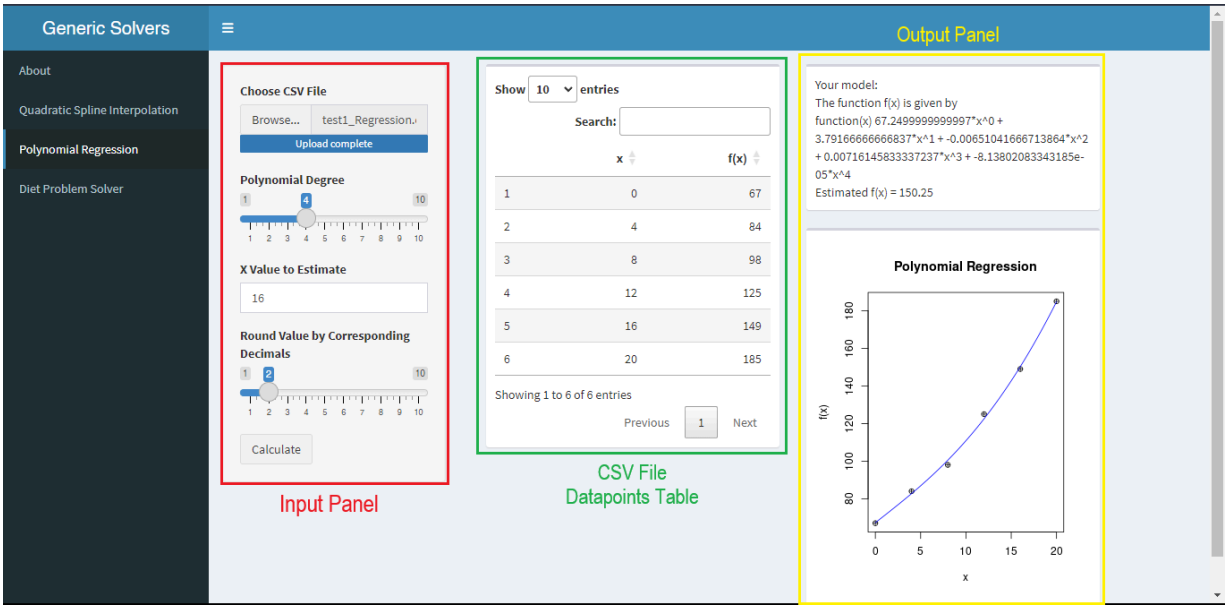
- The following parts of the application applies both to the RShiny application and shinyApp web application.
    - Sidebar: Contains the clickable About, Quadratic Spline Interpolation, Polynomial Regression, and Diet Problem Solver tabs
    - Dashboard Body: Shows information about the tab and the inputs and outputs for the solvers.
    - Show/Hide Sidebar: Toggles the sidebar to be shown or hidden



User Manual

2. Getting Started

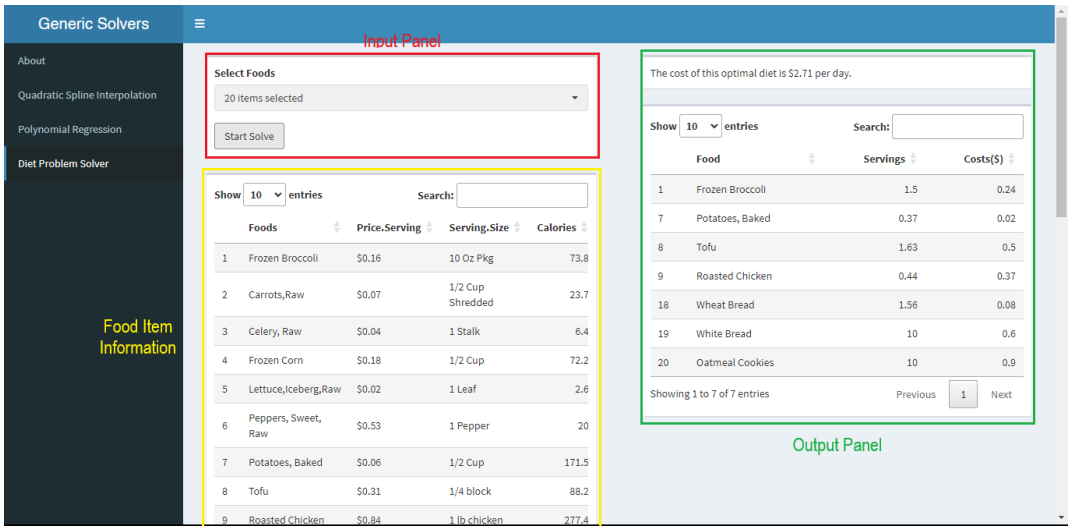
- Generic Solvers: Quadratic Spline Interpolation & Polynomial Regression
    - Input Panel: Contains the input parameters for the specific solver
    - CSV File Data Points Table: Data points from the csv file in tabular form
    - Output Panel: Contains the outputs for the specific solver

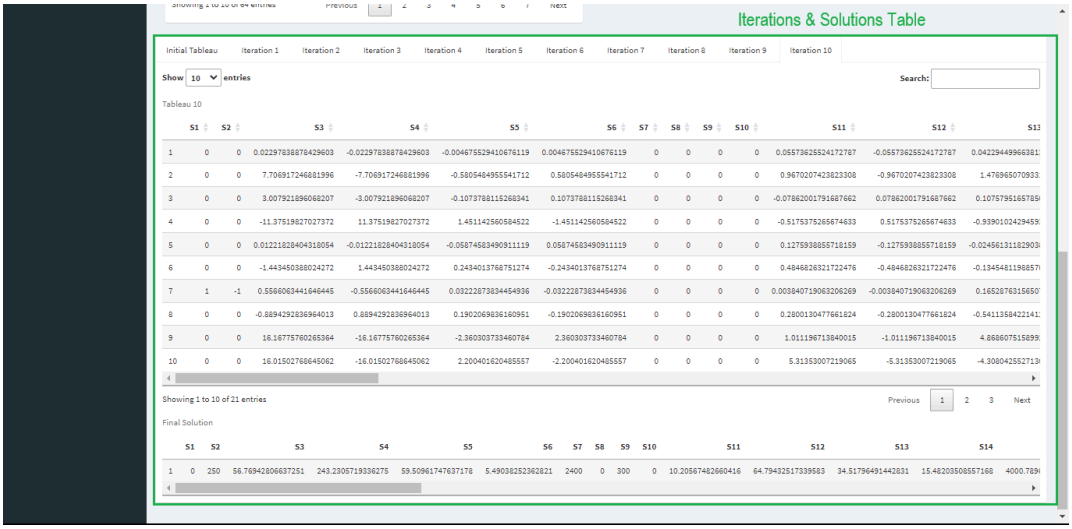


User Manual

2. Getting Started

- Simplex Implementation: Diet Problem Solver
  - Input Panel: Contains the select food input parameter for the solver
  - Food Item Information: Contains information of the food items that can be selected
  - Output Panel and Iterations & Solutions Table: Contains the outputs for the diet problem solver





User Manual

3. Generic Solver: Quadratic Spline Interpolation & Polynomial Regression

3.1 Overview

The Generic Solvers in this application are designed for Quadratic Spline Interpolation and Polynomial Regression respectively. Both solvers use an input .csv file containing the data points and an input x value to estimate to find their respective solutions. For Quadratic Spline Interpolation, this includes the functions per interval, estimate f(x) given the input x, the output function, and the graph of the functions and points. For Polynomial Regression, the solution includes the output function that fits the data given, the estimate f(x) given the input x, and the graph of the function and points.

3.2 Usage

Follow these steps to use the Generic Solvers:

1. Open the application and navigate to the “Quadratic Spline Interpolation” or “Polynomial Regression” tab found at the sidebar.
2. Upload the .csv file containing the data points and input the X Value to estimate. A table in the middle will be shown containing the data points. For the polynomial regression, you may also adjust the polynomial degree (default = 4) by moving the slider. For both the solvers, you may also adjust the number of decimals to round.
3. Click the “Calculate” button to start the solver.
4. View the results on the right side of the dashboard body.

3.3 Input Parameters

The generic solvers require the following as the input parameters:

- .csv file containing the data points
- x value to estimate

User Manual

3.4 Output Results

Upon running the solver, the application will provide the following outputs:

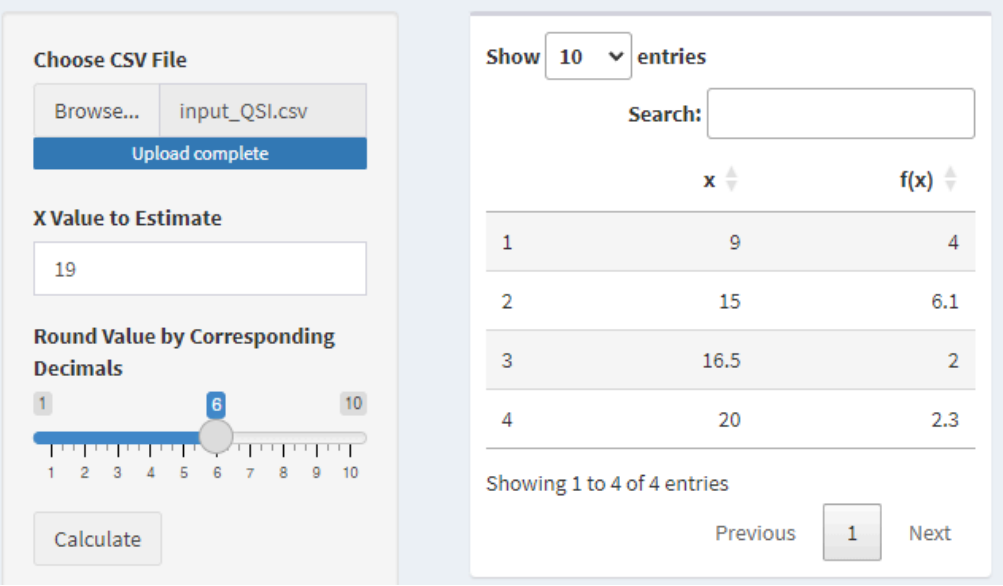
- Quadratic Spline Interpolation:
  - Functions per interval
  - Estimate f(x) given the input x
  - Output function
  - Graph of the functions and points

- Polynomial Regression:
  - Output function that fits the data given
  - Estimate f(x) given the input x
  - Graph of the function and points.

3.5 Example

Quadratic Spline Interpolation Example

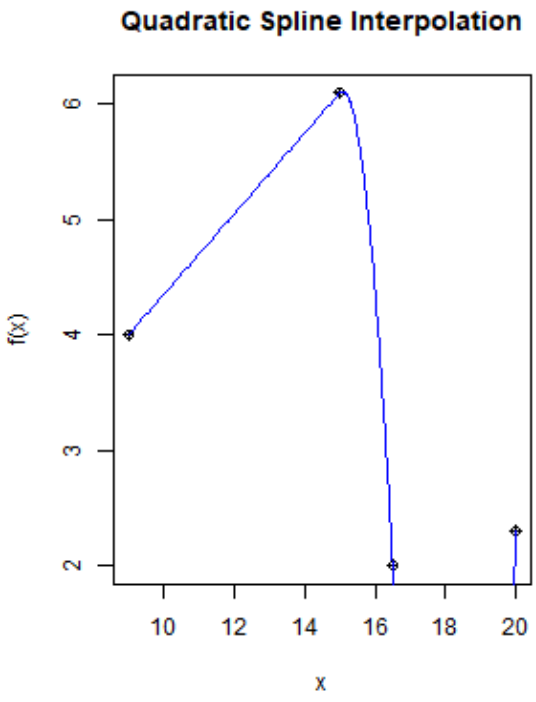
Consider the following sample input below:



User Manual

After running the solver, the application will output:

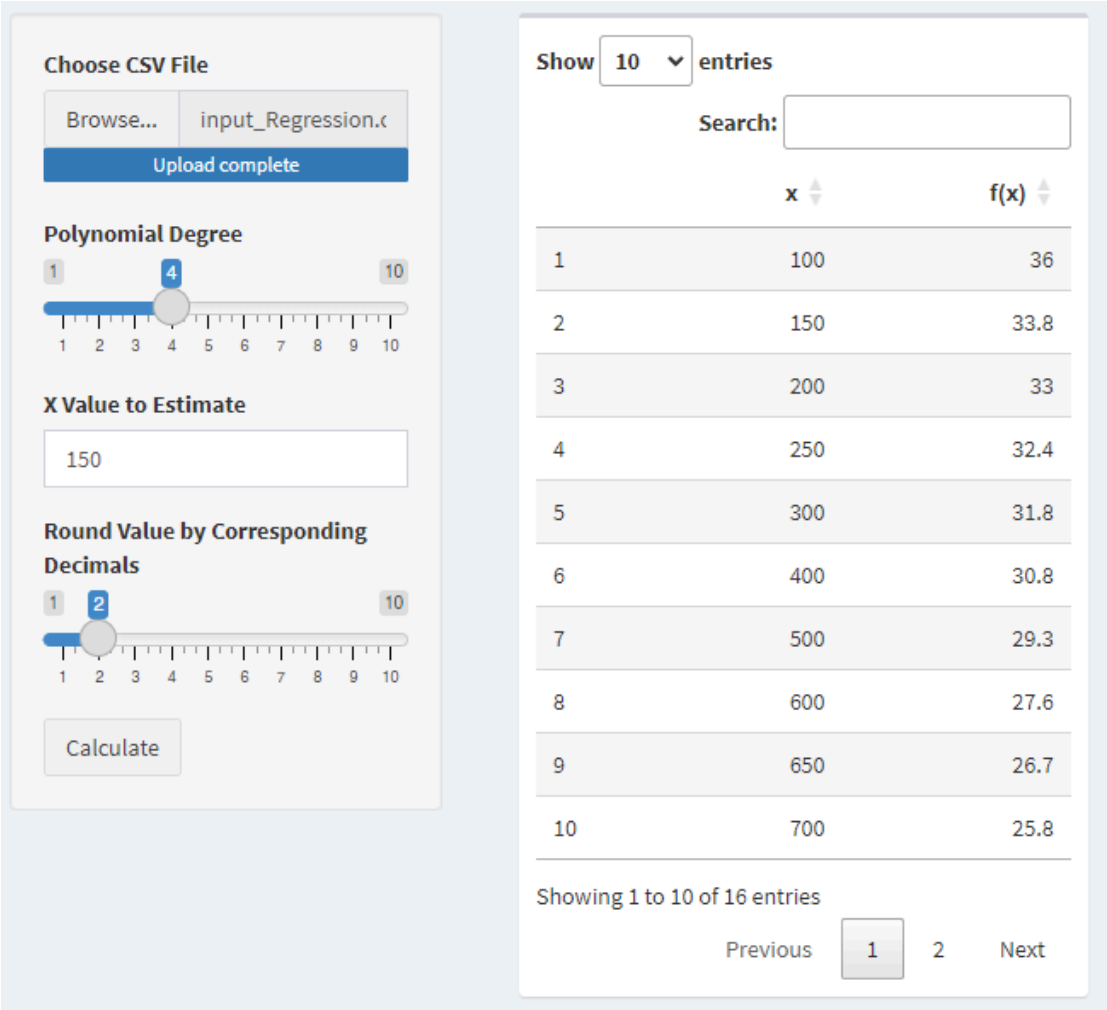
Your model:
The f(x) per interval is given by
[1] function(x) 0.35*(x^1) + 0.850000000000001
[2] function(x) -2.05555555555557*(x^2) +
62.016666666667*(x^1) + -461.650000000002
[3] function(x) 1.68639455782314*(x^2) +
-61.4676870748303*(x^1) + 557.09591836735
Estimated f(x) = -2.001701
The output function is given by:
function(x) 1.68639455782314*(x^2) +
-61.4676870748303*(x^1) + 557.09591836735



User Manual

Polynomial Regression Example

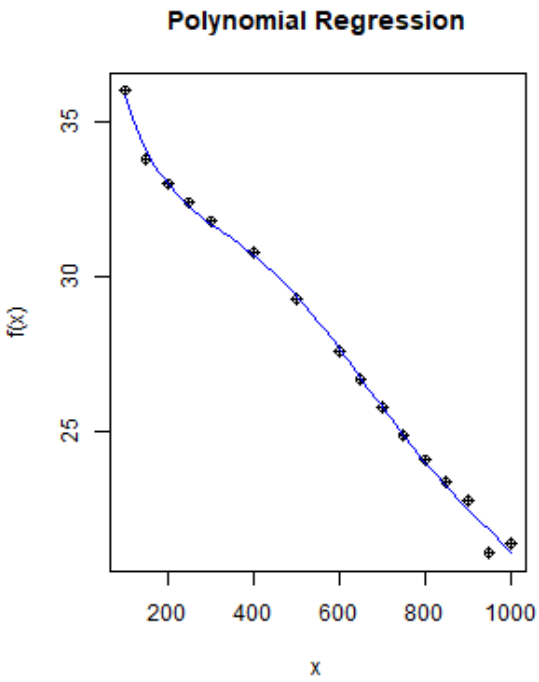
Consider the following sample input below:



User Manual

After running the solver, the application will output:

Your model:
The function f(x) is given by
function(x) 42.357283113739*x^0 +
-0.09374645828336*x^1 + 0.00034420769871560 9*x^2
+ -6.48388139275411e-07*x^3 + 5.48372836623295e-
10*x^4 + -1.71708850732662e-13*x^5
Estimated f(x) = 34.12



User Manual

4. Simplex Implementation: Diet Problem Solver

4.1 Overview

The Simplex Implementation in this application is designed to solve the diet optimization problem. It uses the simplex method, a linear programming method, to find the cheapest and most nutritious combination of foods that will satisfy all the daily nutritional needs of an individual while minimizing the cost. Each food item is limited to 0-10 servings. The daily nutritional requirements are given by the *nutritional_values.csv* file and can be seen in the table below.

Table for Daily Nutritional Requirements

Minimum	Nutrient	Maximum
2000	Calories	2250
0	Cholesterol	300
0	Total Fat	65
0	Sodium	2400
0	Carbohydrates	300
25	Dietary Fiber	100
50	Protein	100
5000	Vitamin A	50000
50	Vitamin C	20000
800	Calcium	1600
10	Iron	30

User Manual

4.2 Usage

Follow these steps to use the Diet Problem Solver:

5. Open the application and navigate to the “Diet Problem Solver” tab found at the sidebar.
6. Select the possible food items to be included in the diet computation.
7. Click the “Start Solve” button to start the simplex algorithm.
8. View the results on the right side and at the bottom of the dashboard body.

4.3 Input Parameters

The Simplex Implementation requires only the selected food items as the input parameter.

4.4 Output Results

Upon running the solver, the application will provide the following outputs:

- At the right side of the page, two boxes will appear, the first indicating the daily cost of the computed diet and the box below listing the food items, servings, and the total costs of each food item.
- At the bottom part of the page, two tables show the tableau and basic solution for each iteration. Each iteration can be accessed through the iteration tabs.

4.5 Example

Consider the following example:

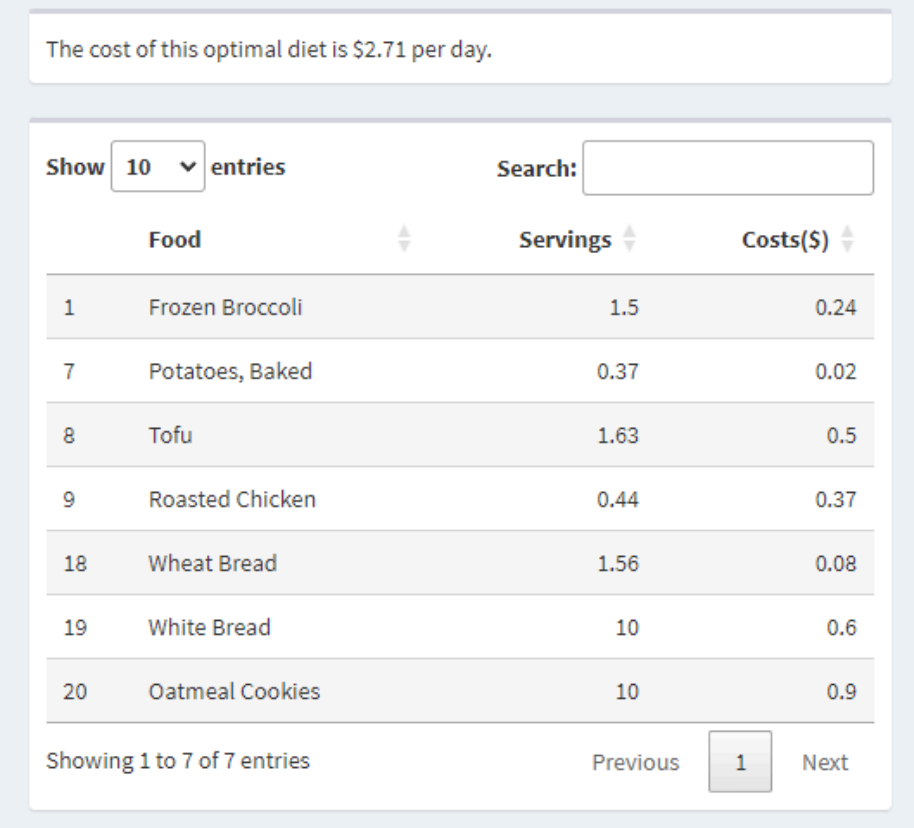
- Selected Food Items:

Frozen Broccoli	Carrots,Raw	Celery, Raw	Frozen Corn
Lettuce,Iceberg,Raw	Roasted Chicken	Potatoes, Baked	Tofu
Peppers, Sweet, Raw	Spaghetti W/ Sauce	Tomato,Red,Ripe,Raw	
Apple,Raw,W/Skin	Banana	Grapes	Kiwifruit,Raw,Fresh
Oranges	Bagels	Wheat Bread	White Bread
Oatmeal Cookies			

User Manual

After running the solver, the application will output:

- Cost of diet per day and table showing the food items, their servings, and total cost for each.



The cost of this optimal diet is $2.71 per day.

Show 10 entries Search:

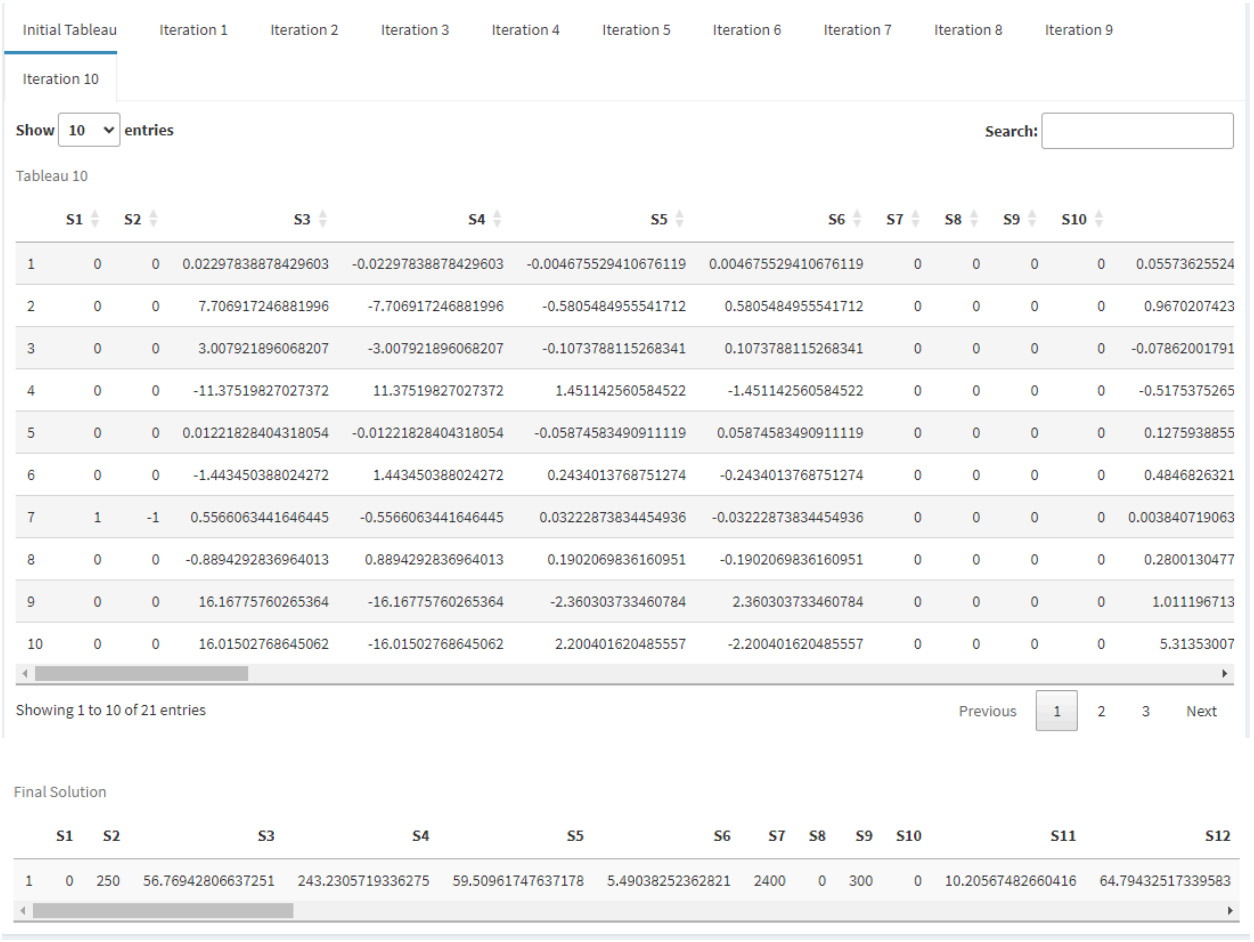
	Food	Servings	Costs($)
1	Frozen Broccoli	1.5	0.24
7	Potatoes, Baked	0.37	0.02
8	Tofu	1.63	0.5
9	Roasted Chicken	0.44	0.37
18	Wheat Bread	1.56	0.08
19	White Bread	10	0.6
20	Oatmeal Cookies	10	0.9

Showing 1 to 7 of 7 entries Previous 1 Next

User Manual

4. Simplex Implementation

- Tableaus and basic solution per iteration (shows final solution for the last iteration)



Initial Tableau | Iteration 1 | Iteration 2 | Iteration 3 | Iteration 4 | Iteration 5 | Iteration 6 | Iteration 7 | Iteration 8 | Iteration 9 | Iteration 10

Show 10 entries Search:

Tableau 10

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	
1	0	0	0.02297838878429603	-0.02297838878429603	-0.004675529410676119	0.004675529410676119	0	0	0	0	0.05573625524
2	0	0	7.706917246881996	-7.706917246881996	-0.5805484955541712	0.5805484955541712	0	0	0	0	0.9670207423
3	0	0	3.007921896068207	-3.007921896068207	-0.1073788115268341	0.1073788115268341	0	0	0	0	-0.07862001791
4	0	0	-11.37519827027372	11.37519827027372	1.451142560584522	-1.451142560584522	0	0	0	0	-0.5175375265
5	0	0	0.01221828404318054	-0.01221828404318054	-0.05874583490911119	0.05874583490911119	0	0	0	0	0.1275938855
6	0	0	-1.443450388024272	1.443450388024272	0.2434013768751274	-0.2434013768751274	0	0	0	0	0.4846826321
7	1	-1	0.5566063441646445	-0.5566063441646445	0.03222873834454936	-0.03222873834454936	0	0	0	0	0.003840719063
8	0	0	-0.8894292836964013	0.8894292836964013	0.1902069836160951	-0.1902069836160951	0	0	0	0	0.2800130477
9	0	0	16.16775760265364	-16.16775760265364	-2.360303733460784	2.360303733460784	0	0	0	0	1.011196713
10	0	0	16.01502768645062	-16.01502768645062	2.200401620485557	-2.200401620485557	0	0	0	0	5.31353007

Showing 1 to 10 of 21 entries Previous 1 2 3 Next

Final Solution

	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12
1	0	250	56.76942806637251	243.2305719336275	59.50961747637178	5.49038252362821	2400	0	300	0	10.20567482660416	64.79432517339583

User Manual

5. About the Developer

5.1 Developer Background

John Nico T. De Castro

John Nico T. De Castro is a passionate and driven student with a strong foundation in programming logic. He is currently studying at the University of the Philippines Los Baños as a 3rd-year undergraduate computer science student. With interests in artificial intelligence, machine learning, and data science, and proficiency in several programming languages such as C, Python, and R, Nico is eager to apply his skills and knowledge to create impactful and innovative technologies..

5.2 Contact Information

For any inquiries, feedback, or technical support, feel free to contact Nico De Castro:

- *Email:* nicodecastro09@gmail.com / jtdecastro@up.edu.ph
- *LinkedIn*: https://www.linkedin.com/in/nico-de-castro
- *GitHub:* https://github.com/okkinn

User Manual

6. References

The following references were used by the developer in creating the application:

- R Documentation (through the RStudio Application)
- shinyapps.io (https://shiny.posit.co/r/articles/share/shinyapps/)
- shiny Layout Guide (https://mastering-shiny.org/action-layout.html)
- shiny Layout Guide (https://shiny.posit.co/r/articles/build/layout-guide/)
- shinydashboard Tutorial (https://www.youtube.com/watch?v=41jmGq7ALMY)
- shinydashboard (https://rstudio.github.io/shinydashboard/structure.html#header)
- DataTables Manual (https://datatables.net/manual/)
- shinyWidgets (https://dreamrs.github.io/shinyWidgets/index.html)
- shinyWidgets pickerInput (https://stackoverflow.com/questions/50218614/shiny-selectinput-to-select-all-from-dropdown)
- Shiny Dashboard Dynamic TabPanel (https://stackoverflow.com/questions/53050339/shinydashboard-dynamic-tabpanel)