Readme!

Please feel free to edit this template by adding / editing and removing slides. I've tried to cover most of what you might want to include but there are many ways of providing evidence so don't be afraid to delete slides that you don't need.

Slides with a green background should be used to show evidence of planning / project management. The slide with a purple background should be duplicated and used to show 'relevant implications' evidence.

The instructions in the grey boxes should be deleted once they have been read. Copies of the instructions are underneath the slides in the speaker notes (just in case they are needed at a later stage).

91906 & 91907 Complex Programming

Trello board / Project Management: https://tinyurl.com/2w654axz **It is public but you will have to login**

The project on Github: https://tinyurl.com/bdz2ujy2

Final Project Filename: app.py, static/js/script.js, static/css/styles.css, templates/index.html | To use, run app.py

Final Testing Video: https://tinyurl.com/bddz8x5y

Project Management

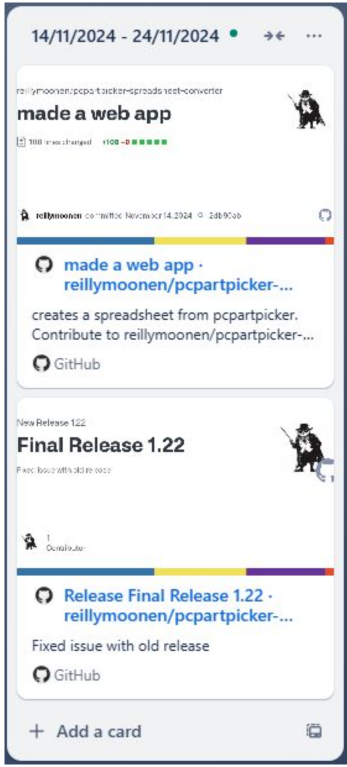
Incremental + Evolutionary Prototyping

I chose a combination of Evolutionary Prototyping and Incremental Development in this project because I had a general goal in mind but not a definitive set of features to start with. Rather than over-engineer at the start, I prefer to build a simple working version first and then develop it further based on real use and ideas that arose along the way. This approach enabled me to stay flexible and open to changes while keeping a solid codebase after each addition. It suited the exploratory nature of the project and avoided premature complexity at the start.

Project Management - Sprint Tracking

Task	Sprint Number	Start Date	End Date
Create UI and bring it to a release	1	14/11/2024	24/11/2024

What are you working on?	What did you achieve?	What are the next steps?
Making the UI more usable and user friendly. And separating the HTML, CSS and JS into their own separate files.	Created a full web ui using HTML, CSS and JS in the same file. I added an animated background gradient and a dark mode toggle. I also registered this project with Creative Commons using [this licence.]()	Separating the components so that the user can use project without initially putting in a file name. Adding a help menu. Adding item numbers. Adding a way to edit items location and be able to delete them.
Evaluation - what worked well & what did not?		
What has been achieved at this point is far better already than using a terminal environment. But a lot could be improved with the UI usability.		



14/11/2024 - 24/11/2024

reillymoonen/pcpartpicker-spreadsheet-converter

made a web app

108 lines changed +108 −0

reillymoonen committed November 14, 2024 2db90ab

made a web app · reillymoonen/pcpartpicker-...

creates a spreadsheet from pcpartpicker. Contribute to reillymoonen/pcpartpicker-...

GitHub

New Release 1.22

Final Release 1.22

Fixed issue with old release

1 Contributor

Release Final Release 1.22 · reillymoonen/pcpartpicker-...

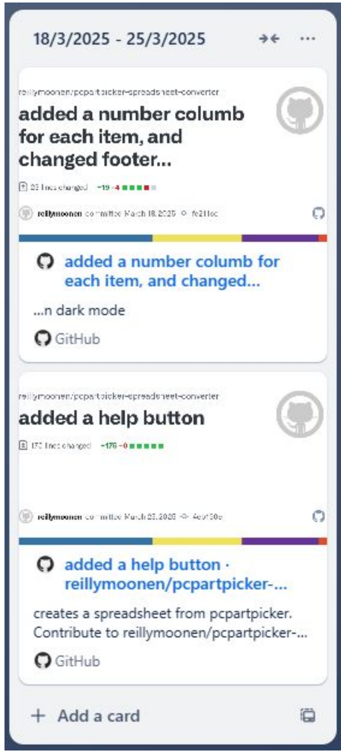
Fixed issue with old release

GitHub

Project Management - Sprint Tracking

Task	Sprint Number	Start Date	End Date
Remake the whole UI	2	18/3/2025	25/3/2025

What are you working on?	What did you achieve?	What are the next steps?
Improving the help window ui and fixing scroll bugs with the help window	Restructured the UI so that the user just has to put in a link and can choose a filename later Added sort functionality Added drag and drop functionality Added a paste button Added placeholder text for input fields Fixed dark mode bug Add numbered items Add delete button Added a total count Moved CSS and JS to separate files Added a text cut off for long names Added a subheading	Improving the help window ui and fixing scroll bugs with the help window Maybe a link button to take the user to the parts website for more info


18/3/2025 - 25/3/2025

reillymoonen/pcpartpicker-spreadsheet-converter

added a number columb for each item, and changed footer...

23 lines changed +19 -4

reillymoonen committed March 18, 2025 · [illegible]

added a number columb for each item, and changed...

...n dark mode

GitHub

reillymoonen/pcpartpicker-spreadsheet-converter

added a help button

176 lines changed +176 -0

reillymoonen committed March 23, 2025 · [illegible]

added a help button · reillymoonen/pcpartpicker-...

creates a spreadsheet from pcpartpicker. Contribute to reillymoonen/pcpartpicker-...

GitHub

+ Add a card

What are you working on?	What did you achieve?	What are the next steps?
	Added a Date/Time button for filename Changed link and delete button colours in dark mode Changed footer link colour in dark mode Added a help button and window Added hover effects to items Update URL validation to allow country codes	

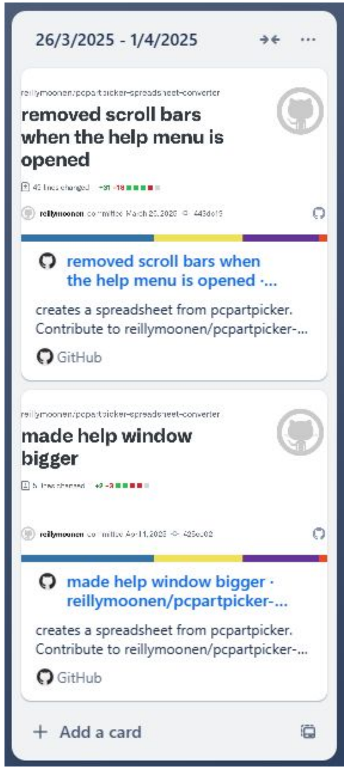
Evaluation - what worked well & what did not?
The UI has been completely remade, the product so far is for more usable than it was before, especially with the data and making customizing it far more usable. So far the help window is incredibly buggy, so that will have to be fixed later on.

Project Management - Sprint Tracking

Task	Sprint Number	Start Date	End Date
Polish the help window	3	26/3/2025	1/4/2025

What are you working on?	What did you achieve?	What are the next steps?
Adding link buttons Going over the ui one last time just to make it a little more usable	Removed main website scroll bar when opening the help menu Fixed scaling issue for help window when window size is scaled down Fixed help menu in dark mode Added image into help menu Added proper instructions to the help menu Added a Favicon Fixed sort menu buttons in dark mode	Going over the ui one last time just to make it a little more usable

Evaluation - what worked well & what did not?
Adding the image to the help menu really did make some difference, as well as hiding the scroll bar. During this period I also tried to add aria labels which did not work at all.


26/3/2025 - 1/4/2025

reillymoonen/pcpartpicker-spreadsheet-converter

removed scroll bars when the help menu is opened

49 lines changed +31 -18

reillymoonen committed March 26, 2025 · 443d015

removed scroll bars when the help menu is opened ·...

creates a spreadsheet from pcpartpicker. Contribute to reillymoonen/pcpartpicker-...

GitHub

reillymoonen/pcpartpicker-spreadsheet-converter

made help window bigger

5 lines changed +2 -3

reillymoonen committed Apr 1, 2025 · 425e[illegible]02

made help window bigger · reillymoonen/pcpartpicker-...

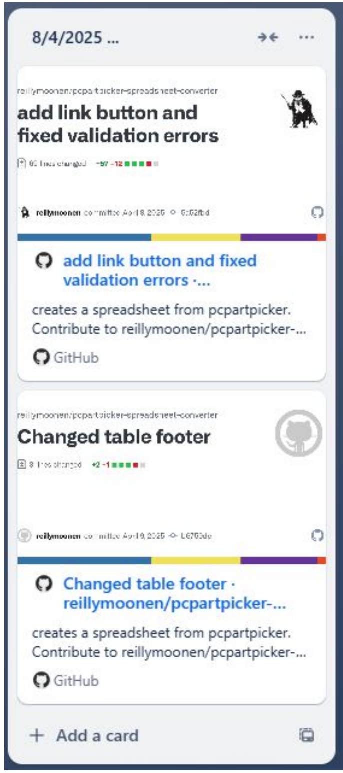
creates a spreadsheet from pcpartpicker. Contribute to reillymoonen/pcpartpicker-...

GitHub

Project Management - Sprint Tracking

Task	Sprint Number	Start Date	End Date
Polish ui and add link button	4	8/4/2025	…

What are you working on?	What did you achieve?	What are the next steps?
This documentation	Fixed table footer Updated the subheading size Added link buttons for each item Fixed link icons	This documentation
Evaluation - what worked well & what did not?		
The link buttons are great, as well as redoing the table footer		


8/4/2025 ...

reillymoonen/pcpartpicker-spreadsheet-converter

add link button and fixed validation errors

[illegible] lines changed +57 -12

reillymoonen committed Apr 18, 2025 · [illegible]

add link button and fixed validation errors ·...

creates a spreadsheet from pcpartpicker. Contribute to reillymoonen/pcpartpicker-...

GitHub

reillymoonen/pcpartpicker-spreadsheet-converter

Changed table footer

3 lines changed +2 -1

reillymoonen committed Apr 19, 2025 · [illegible]

Changed table footer · reillymoonen/pcpartpicker-...

creates a spreadsheet from pcpartpicker. Contribute to reillymoonen/pcpartpicker-...

GitHub

Relevant Implications

Functionality: Functionality is the ability for it to work as intended and achieves the original purpose, this can be achieved through thorough testing and by having users testing. I have addressed this by adding in user input error handling and I have attempted to use the program as incorrectly as possible while testing to insure that it functions correctly and as expected. I also got my Dad, Aunt and another couple of students to test the final program.

Usability: Usability is the ease of use and considering the end user and their requirements. Making the outcome easy to interact with and navigate, if this wasn't addressed it could frustrated people and force them to have a bad experience or give up. The program is made of different sections that are labeled and in a clear and logical order with a clear hierarchy for people to navigate and understand how to use. Doing this should be easier for the user to navigate. I put in instructions to help inform people how to use the program.

Relevant Implications

Legal: If copyright laws have been broken or intellectual property is being used without the owners permission then the outcome is not legal, also accessing unauthorised or private information can violate privacy rights and data protection regulations. I have addressed this by making the program is only able to access public information from pcpartpicker links, any private information is not able to be accessed. All libraries used in the program are free and open source, so there is no copyrighted material or intellectual property being used. In addition to this I have registered this project and github repo under CC BY-SA 4.0 meaning that the users are free to:
Share - copy and redistribute the material in any medium or format for any purpose, even commercially.
Adapt - remix, transform, and build upon the material for any purpose, even commercially.

Continue on next slide…

Relevant Implications

However users will have to provide:
Attribution - They must give appropriate credit , provide a link to the license, and indicate if changes were made. They may do so in any reasonable manner, but not in any way that suggests the licensor endorses them or they're use.
ShareAlike If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.
However they do not have to comply with the license for elements of the material in the public domain or where they're use is permitted by an applicable exception or limitation.

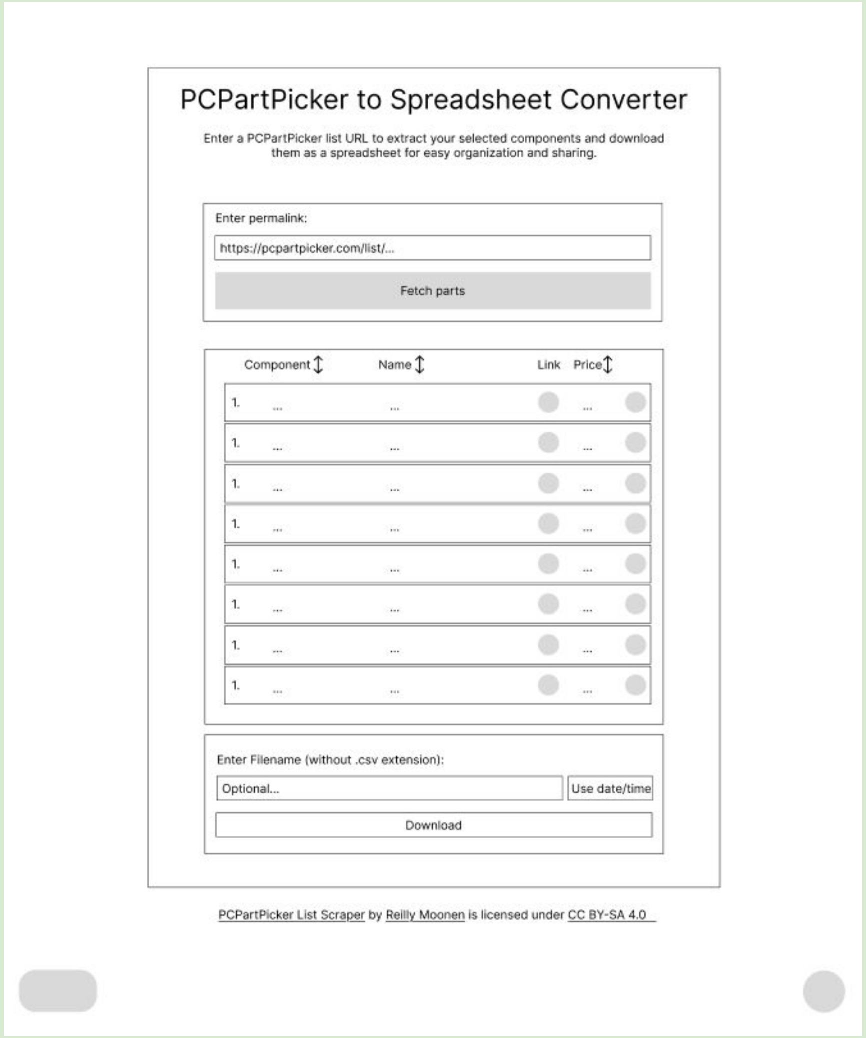
Relevant Implications

Accessibility: Accessibility aims to make programs usable and inclusive for all individuals, no matter their abilities or disabilities. This includes using color-blind friendly color schemes, implementing dyslexia-friendly fonts, and ensuring compatibility with the majority or all devices. My program should be fairly easy for users to use, it uses a wide ranged color palette because of the background gradient, however I did chuck a screenshot into a colour blindness simulator and made sure that everything was usable, and it was, so users with various forms of colour blindness should be able to use the program without any issues. There are currently some accessibility issues; the program currently requires users to have the ability to run Python programs, which needs some prior technical knowledge. Additionally, it can only be operated on devices capable of running Python, which excludes most mobile phones and tablets. One thing that could be done to address this in the future is building it into a website, so then all users with access to the internet will be able use it easily

Relevant Implications

End user considerations: End user considerations involve understanding the target audience and tailoring the product to meet their specific needs, expectations, and level of expertise. This includes using appropriate terminology, and ensuring the product aligns with the users' goals and technical abilities. The program addresses this by recognizing that the primary users are likely to be building their own computers who want to track component shipments and manage their build process. Given this audience, I’ve used relevant terminology that should be familiar to people around that expected age range. The program assumes a basic level of technical ability, including the ability to run Python scripts, which aligns with the typical skill set of our expected users.

Graphical User Interface Design...



PCPartPicker to Spreadsheet Converter

Enter a PCPartPicker list URL to extract your selected components and download them as a spreadsheet for easy organization and sharing.

Component	Name	Link	Price
1. ...	...	●	... ●
1. ...	...	●	... ●
1. ...	...	●	... ●
1. ...	...	●	... ●
1. ...	...	●	... ●
1. ...	...	●	... ●
1. ...	...	●	... ●
1. ...	...	●	... ●

Program Structure

https://tinyurl.com/2w654axz

Problem Decomposition

Provide evidence showing that you have decomposed the task. This can be in the form of a trello screenshot or a list of components. If necessary, you may revisit this slide and add to it / edit it as you create your outcome. When you make changes to this slide, please do so in a different colour, date the changes and explain why they were made (this can help provide evidence for M / E grades).

Hint: Use the structure you developed earlier to work out what components you need. For each function, you should have at least one component.

Problem Decomposition

Problems	Description	Dependencies
1. Link Input Field	UI component to allow user to paste a PCPartPicker link	HTML/CSS/JS
2. URL Validation Logic	Verifies the structure of the submitted URL and checks if it matches known PCPartPicker formats	JS & Python
3. Web Scraper / Parser	Retrieves part data from the given PCPartPicker list URL	Python (Flask backend)
4. Data Cleaning Module	Filters and structures the data (names, prices, links)	Python
5. Spreadsheet Exporter	Converts the parsed data into a .csv spreadsheet file	Python
6. UI Table Renderer	Dynamically displays the extracted parts in a table	JS/HTML
7. Delete Button Functionality	Allows users to remove individual parts from the list	JS
8. Drag-and-Drop Sorting	Enables reordering of parts in the table visually	JS
9. Dark Mode Toggle	Switches UI appearance for light/dark themes	JS/CSS

Problem Decomposition

10. Help Menu UI	Displays instructions for users unfamiliar with the tool	HTML/JS/CSS
11. Filename Input Generator	Allows user to define a filename or auto-generate one using date/time	JS
12. Spreadsheet Download Trigger	Starts the download process with correct formatting and naming	JS + Python
13. Input Error Handling	Ensures user feedback is given if links or actions fail	JS

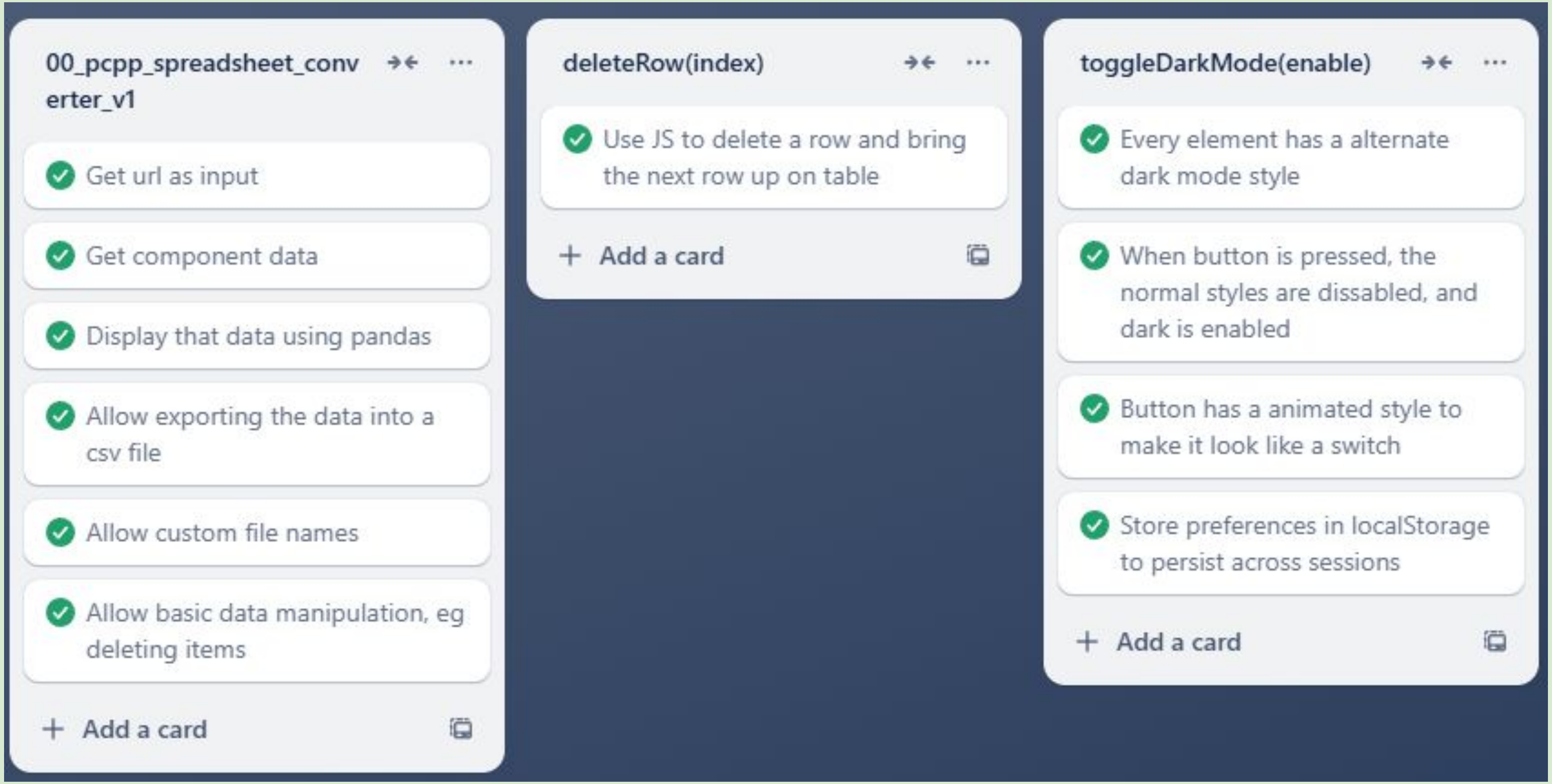
Develop your Components!!

Make as many copies of the next three slides as you need. For each component you should have…

- A trello screenshot / evidence of ongoing use of your project management tools
- A test plan for the component that has been created BEFORE you start coding. Your plan should allow you to test all logical pathways for this component. It should also include test cases for relevant boundary and unexpected values.
- Screenshot evidence showing that you have worked through the test plan you developed.

You should also provide evidence of trialling multiple components & techniques. Please make slides as needed for that evidence.

Component Planning...



00_pcpp_spreadsheet_converter_v1

- ✓ Get url as input
- ✓ Get component data
- ✓ Display that data using pandas
- ✓ Allow exporting the data into a csv file
- ✓ Allow custom file names
- ✓ Allow basic data manipulation, eg deleting items

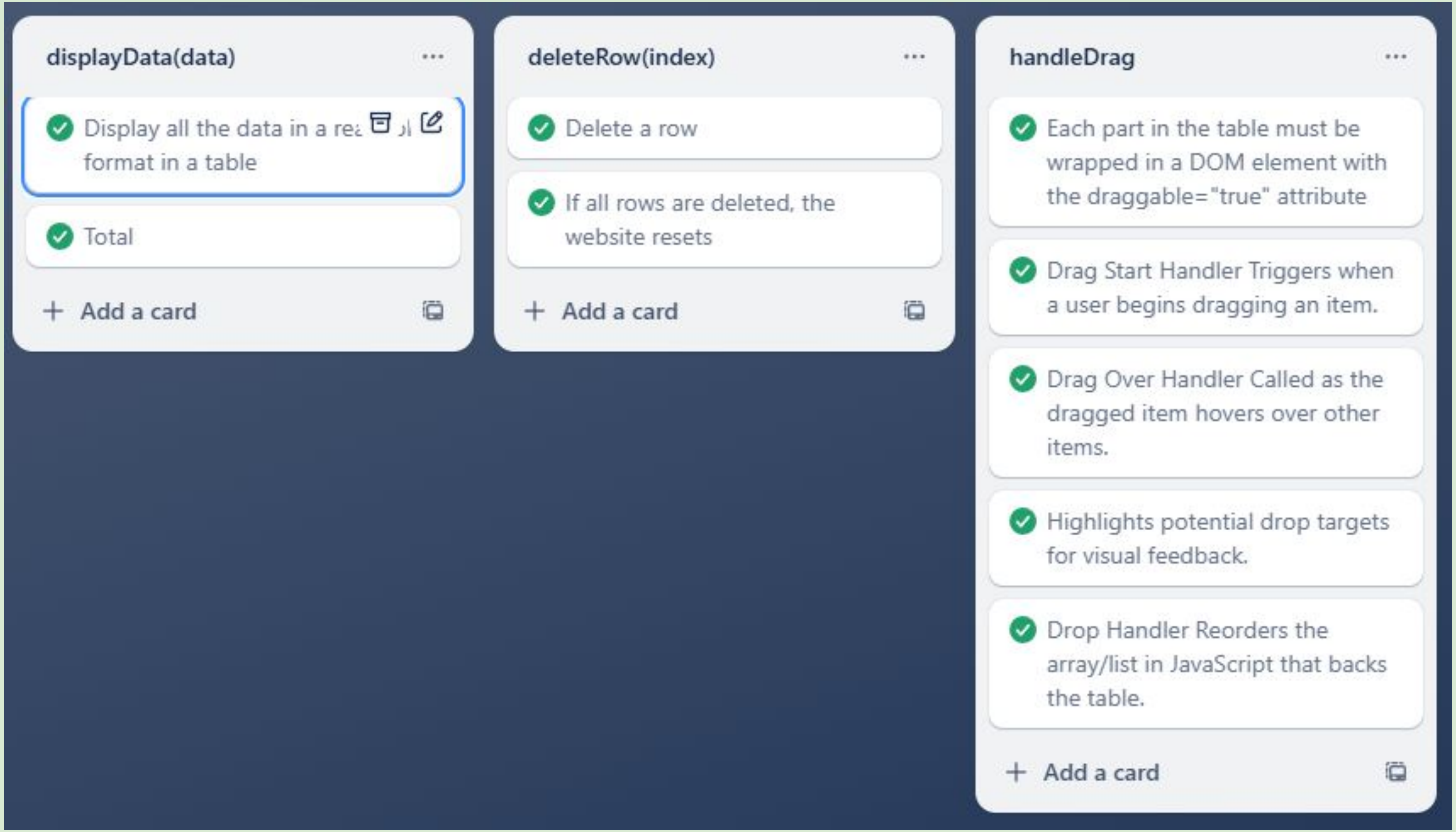
deleteRow(index)

- ✓ Use JS to delete a row and bring the next row up on table

toggleDarkMode(enable)

- ✓ Every element has a alternate dark mode style
- ✓ When button is pressed, the normal styles are dissabled, and dark is enabled
- ✓ Button has a animated style to make it look like a switch
- ✓ Store preferences in localStorage to persist across sessions

Component Planning...



displayData(data)

- ✓ Display all the data in a rea ɔl format in a table
- ✓ Total

+ Add a card

deleteRow(index)

- ✓ Delete a row
- ✓ If all rows are deleted, the website resets

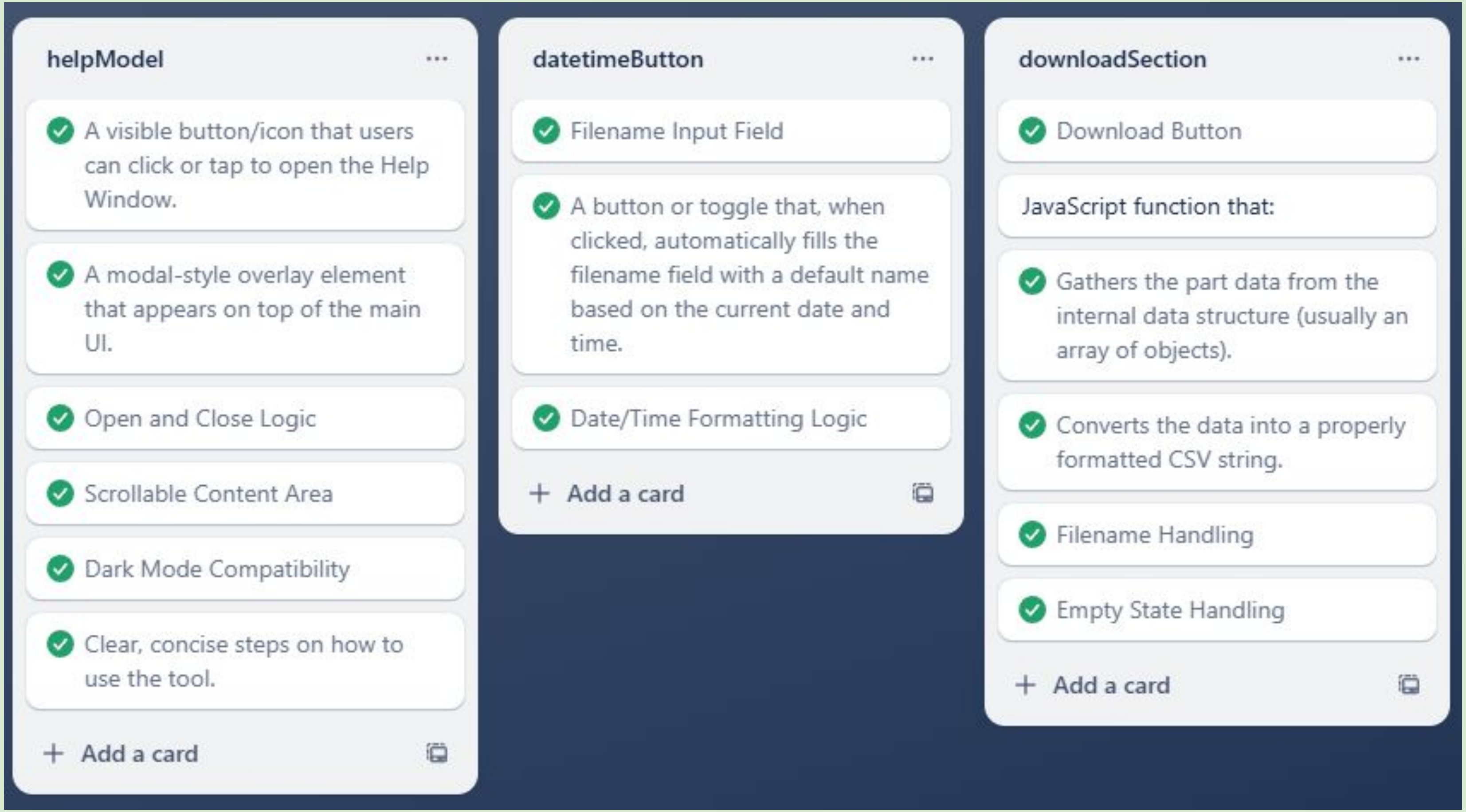
+ Add a card

handleDrag

- ✓ Each part in the table must be wrapped in a DOM element with the draggable="true" attribute
- ✓ Drag Start Handler Triggers when a user begins dragging an item.
- ✓ Drag Over Handler Called as the dragged item hovers over other items.
- ✓ Highlights potential drop targets for visual feedback.
- ✓ Drop Handler Reorders the array/list in JavaScript that backs the table.

+ Add a card

Component Planning...



helpModel ···

- ✓ A visible button/icon that users can click or tap to open the Help Window.
- ✓ A modal-style overlay element that appears on top of the main UI.
- ✓ Open and Close Logic
- ✓ Scrollable Content Area
- ✓ Dark Mode Compatibility
- ✓ Clear, concise steps on how to use the tool.

datetimeButton ···

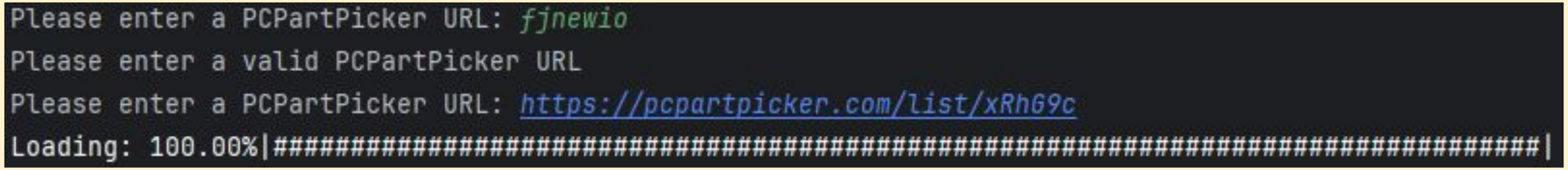
- ✓ Filename Input Field
- ✓ A button or toggle that, when clicked, automatically fills the filename field with a default name based on the current date and time.
- ✓ Date/Time Formatting Logic

downloadSection ···

- ✓ Download Button
- JavaScript function that:
- ✓ Gathers the part data from the internal data structure (usually an array of objects).
- ✓ Converts the data into a properly formatted CSV string.
- ✓ Filename Handling
- ✓ Empty State Handling

ask_user_for_url - Test Plan (?and screenshot)

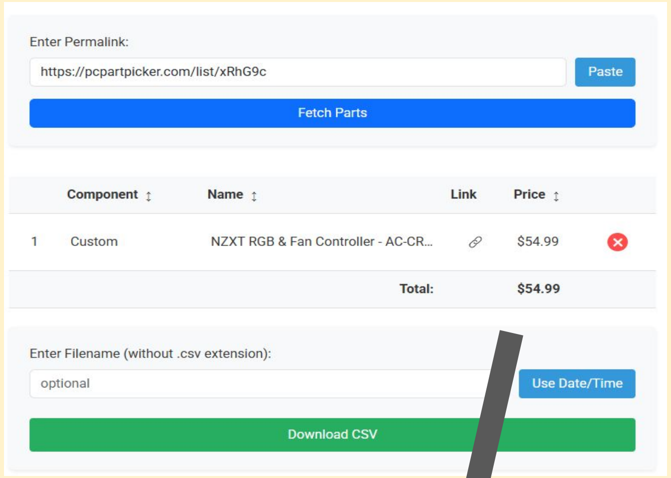
Link	Expected Values
Invalid Link	Display error
Valid Link	Display the data



```
Please enter a PCPartPicker URL: fjnewio
Please enter a valid PCPartPicker URL
Please enter a PCPartPicker URL: https://pcpartpicker.com/list/xRhG9c
Loading: 100.00%|##################################################################################|
```

deleteRow(index) - Test Plan (?and screenshot)

Delete items	Expected Values
User deletes item	Deletes item row from table
User deletes all items	Program goes back to default, so that the user cant download a black file



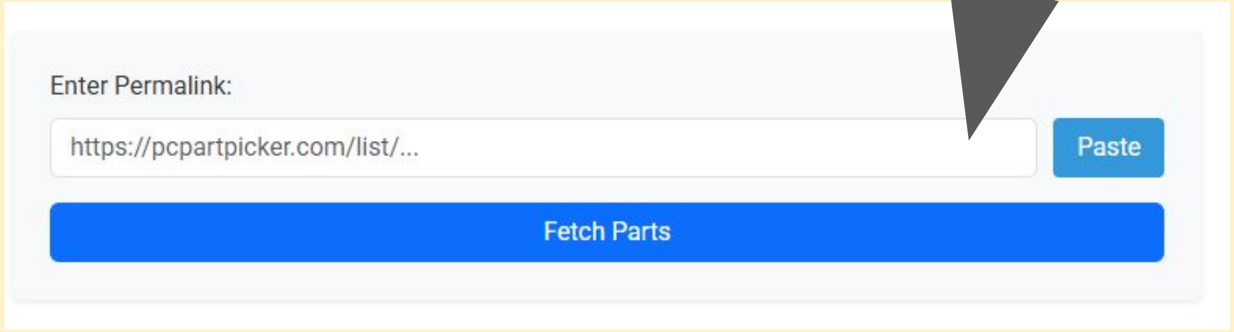
Enter Permalink:

https://pcpartpicker.com/list/xRhG9c

	Component	Name	Link	Price	
1	Custom	NZXT RGB & Fan Controller - AC-CR...		$54.99	
			Total:	$54.99	

Enter Filename (without .csv extension):

optional

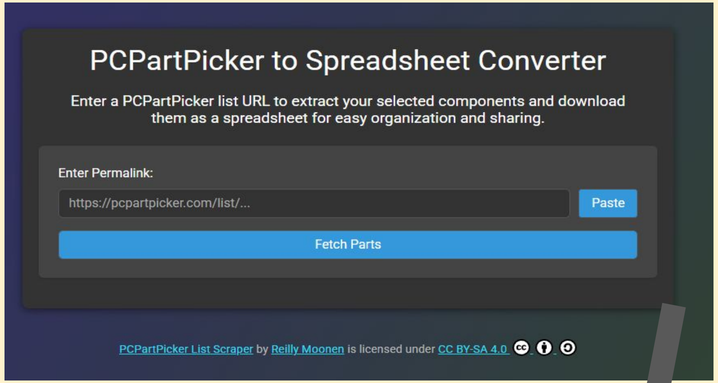


Enter Permalink:

https://pcpartpicker.com/list/...

toggle Dark Mode(enable) - Test Plan (?and screenshot)

Dark Mode	Expected Values
User clicks toggle in light mode	Uses classList.toggle() to safely apply/remove classes, and Stores preference in localStorage to persist across sessions
User clicks toggle in dark mode	Uses classList.toggle() to safely apply/remove classes, and Stores preference in localStorage to persist across sessions

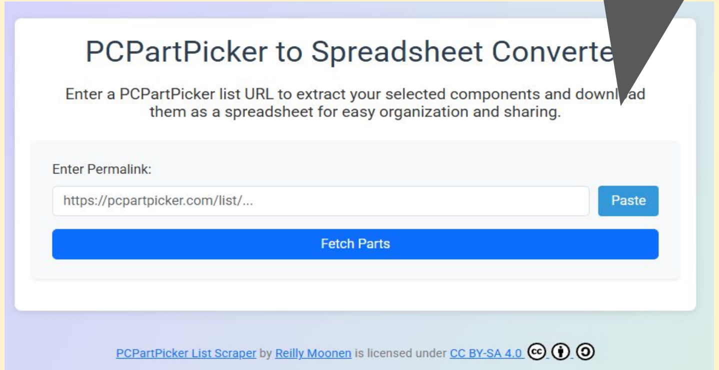


PCPartPicker to Spreadsheet Converter

Enter a PCPartPicker list URL to extract your selected components and download them as a spreadsheet for easy organization and sharing.

Enter Permalink:

https://pcpartpicker.com/list/...



PCPartPicker to Spreadsheet Converte

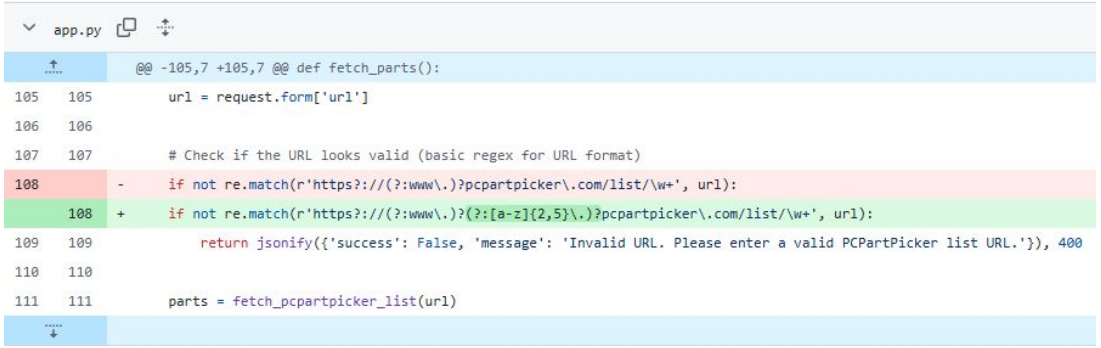
Enter a PCPartPicker list URL to extract your selected components and downl ad them as a spreadsheet for easy organization and sharing.

Enter Permalink:

https://pcpartpicker.com/list/...

Debugging Evidence <URL country code validation>

Use this slide to provide evidence that you have identified any bugs that prevent certain functionality and how you fixed them.



app.py

```diff
@@ -105,7 +105,7 @@ def fetch_parts():
     url = request.form['url']
 
     # Check if the URL looks valid (basic regex for URL format)
-    if not re.match(r'https?://(?:www\.)?pcpartpicker\.com/list/\w+', url):
+    if not re.match(r'https?://(?:www\.)?(?:[a-z]{2,5}\.)?pcpartpicker\.com/list/\w+', url):
         return jsonify({'success': False, 'message': 'Invalid URL. Please enter a valid PCPartPicker list URL.'}), 400
 
     parts = fetch_pcpartpicker_list(url)
```

The website was marking urls with country codes as invalid, so I made it so that it will ignore that part of the url.

Debugging Evidence <Help window scaling issue>

Use this slide to provide evidence that you have identified any bugs that prevent certain functionality and how you fixed them.

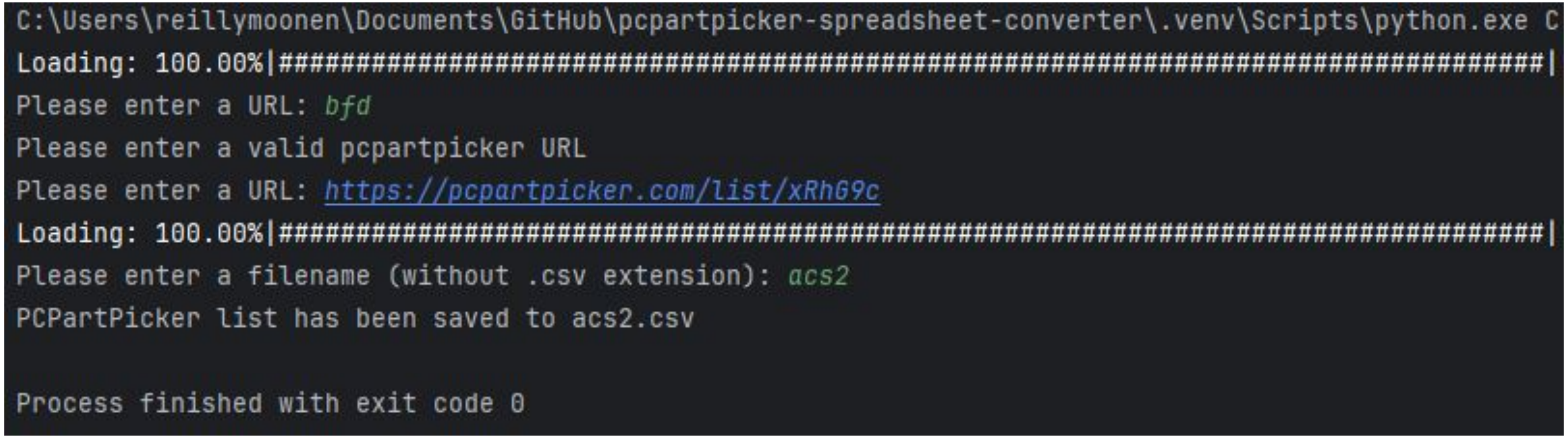
https://github.com/reillymoonen/pcpartpicker-spreadsheet-converter/commit/3b03a35c654800a416d546103f6a2f66a0c62fb2

When the window was scaled down to a mobile ratio, the help menu would move down the page, instead of staying in the center and just scaling to fit.
I did this by:

- Switching to a flex layout
- Applying opacity transitions
- Separating visibility (.show class) from display
- Preventing interaction when hidden
- Ensuring modal is fully hidden on load

00_pcpp_spreadsheet_converter_v1 (I added screenshot after)

Data input	Expected Values
Invalid Link	Display error
Valid Link	Display the data

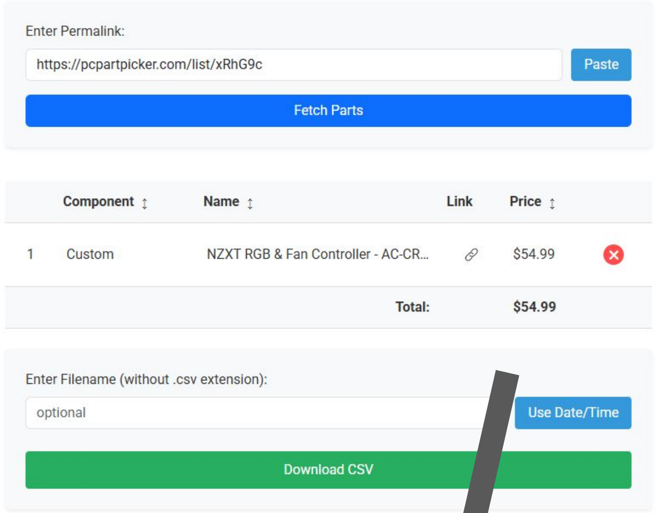


```
C:\Users\reillymoonen\Documents\GitHub\pcpartpicker-spreadsheet-converter\.venv\Scripts\python.exe C
Loading: 100.00%|################################################################################|
Please enter a URL: bfd
Please enter a valid pcpartpicker URL
Please enter a URL: https://pcpartpicker.com/list/xRhG9c
Loading: 100.00%|################################################################################|
Please enter a filename (without .csv extension): acs2
PCPartPicker list has been saved to acs2.csv

Process finished with exit code 0
```

deleteRow(index) (I added screenshot after)

Data input	Expected Values
User deletes item	Deletes item row from table
User deletes all items	Program goes back to default, so that the user cant download a black file



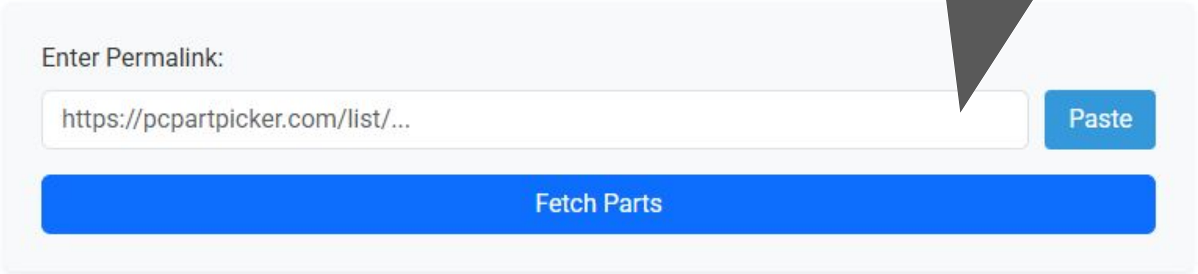
Enter Permalink:

https://pcpartpicker.com/list/xRhG9c

	Component ↕	Name ↕	Link	Price ↕	
1	Custom	NZXT RGB & Fan Controller - AC-CR...	🔗	$54.99	✖
			Total:	$54.99	

Enter Filename (without .csv extension):

optional

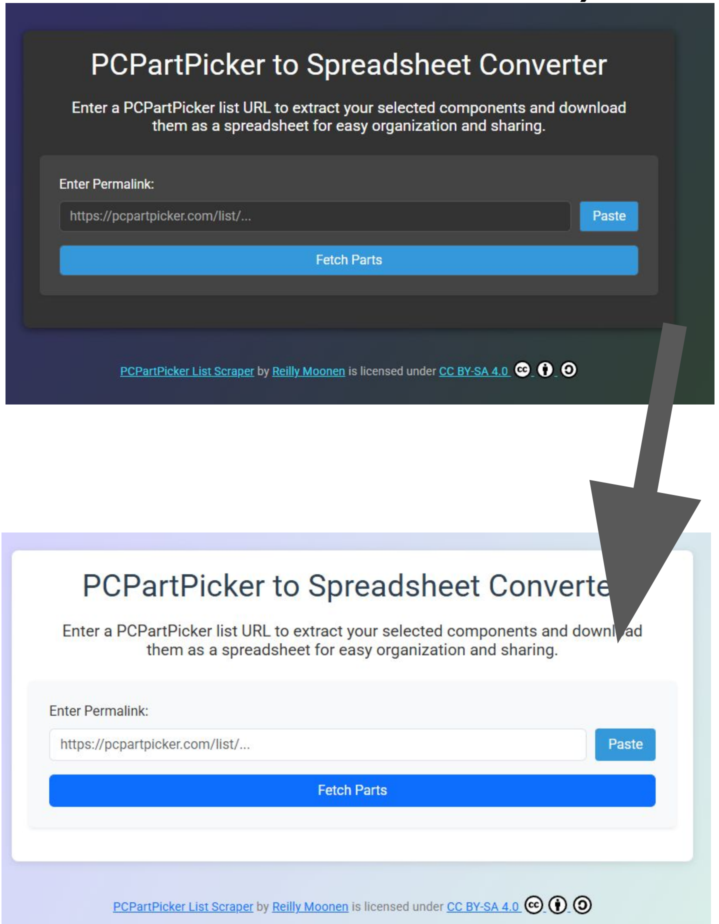


Enter Permalink:

https://pcpartpicker.com/list/...

toggleDarkMode(enable) (I added screenshot after)

Data input	Expected Values
User clicks toggle in light mode	Uses classList.toggle() to safely apply/remove classes, and Stores preference in localStorage to persist across sessions
User clicks toggle in dark mode	Uses classList.toggle() to safely apply/remove classes, and Stores preference in localStorage to persist across sessions


PCPartPicker to Spreadsheet Converter

Enter a PCPartPicker list URL to extract your selected components and download them as a spreadsheet for easy organization and sharing.

Enter Permalink:

https://pcpartpicker.com/list/...

PCPartPicker to Spreadsheet Converte

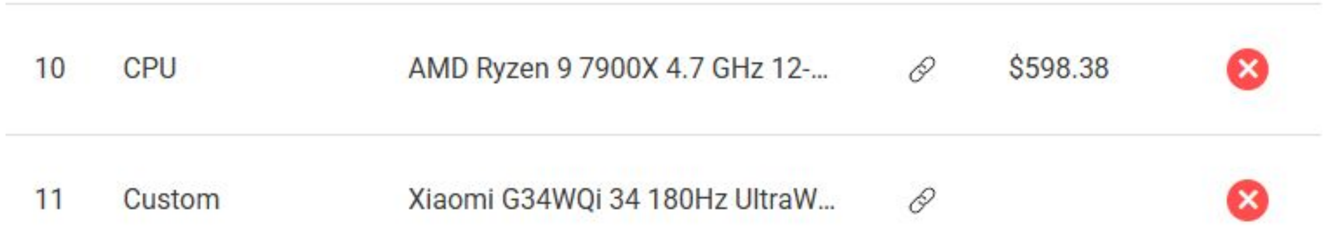
Enter a PCPartPicker list URL to extract your selected components and downl ad them as a spreadsheet for easy organization and sharing.

Enter Permalink:

https://pcpartpicker.com/list/...

displayData(data) (I added screenshot after)

Data input	Expected Values
Normal data	Table shows 5 columns with number, component type, part names, prices, and clickable links
Missing data, eg price from discontinued item	Leaves that area blank



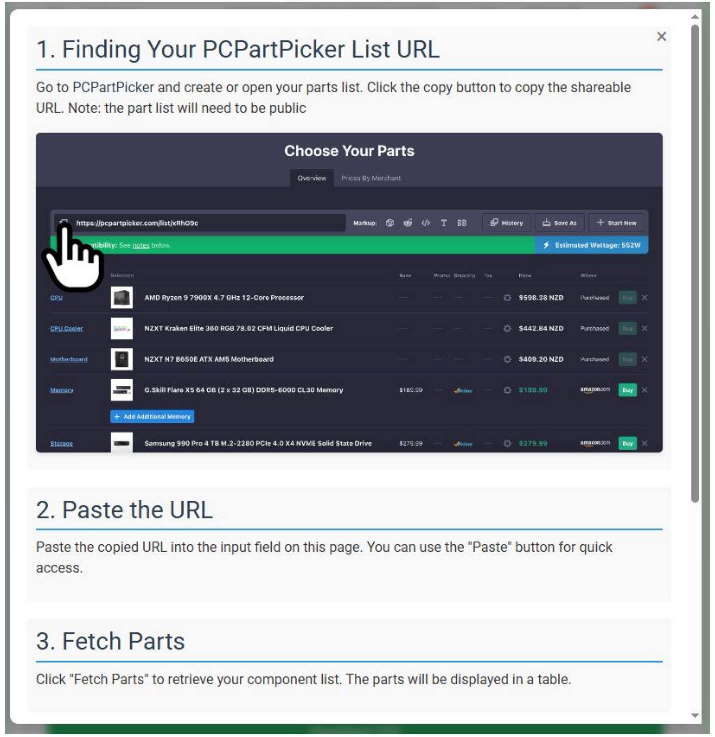
10	CPU	AMD Ryzen 9 7900X 4.7 GHz 12-...	🔗	$598.38	✖
11	Custom	Xiaomi G34WQi 34 180Hz UltraW...	🔗		✖

handleDrag

Data input	Expected Values
Normal item drag	Items position in the table is changed to where the user let the item “drop”
There is no way to bug this out that I have found, just one way to do it	

helpModel (I added screenshot after)

Data input	Expected Values
Open via button click	Modal becomes visible
Close via close button or by clicking outside	Modal hides
Resize browser when modal is open	Content resizes properly; text remains readable; no overflow issues
Scrollable content	Internal scroll bar appears; modal does not overflow the screen


1. Finding Your PCPartPicker List URL

Go to PCPartPicker and create or open your parts list. Click the copy button to copy the shareable URL. Note: the part list will need to be public

Choose Your Parts

Overview | Prices By Merchant

https://pcpartpicker.com/list/xRhG9c

…tibility: See notes below.

Estimated Wattage: 552W

	Selection	Base	Promo	Shipping	Tax	Price	Where
CPU	AMD Ryzen 9 7900X 4.7 GHz 12-Core Processor	—	—	—	—	$598.38 NZD	Purchased
CPU Cooler	NZXT Kraken Elite 360 RGB 78.02 CFM Liquid CPU Cooler	—	—	—	—	$442.84 NZD	Purchased
Motherboard	NZXT N7 B650E ATX AM5 Motherboard	—	—	—	—	$409.20 NZD	Purchased
Memory	G.Skill Flare X5 64 GB (2 x 32 GB) DDR5-6000 CL30 Memory	$189.99	—	✓Prime	—	$189.99	amazon.com
Storage	Samsung 990 Pro 4 TB M.2-2280 PCIe 4.0 X4 NVME Solid State Drive	$279.99	—	✓Prime	—	$279.99	amazon.com

2. Paste the URL

Paste the copied URL into the input field on this page. You can use the "Paste" button for quick access.

3. Fetch Parts

Click "Fetch Parts" to retrieve your component list. The parts will be displayed in a table.

datetimeButton (I added screenshot after)

Data input	**Expected Values**
User clicks Date/Time button when filename input is empty	Default name “parts_list”
User already entered a filename and clicks button	Add number “parts_list (1)”
Date/time button is pressed	“pcparts_2025-05-30_13-16-58”

downloadSection (I added screenshot after)

Data input	Expected Values
Valid data present, user clicks "Download"	Uses classList.toggle() to safely apply/remove classes, and Stores preference in localStorage to persist across sessions
No parts are loaded	Download is prevented and displays an error message, but because of the delete items component this should never happen initially

Component Testing Evidence <Edit Me>

Use this slide to provide evidence that you have tested your component in accordance with the test plan you developed earlier.

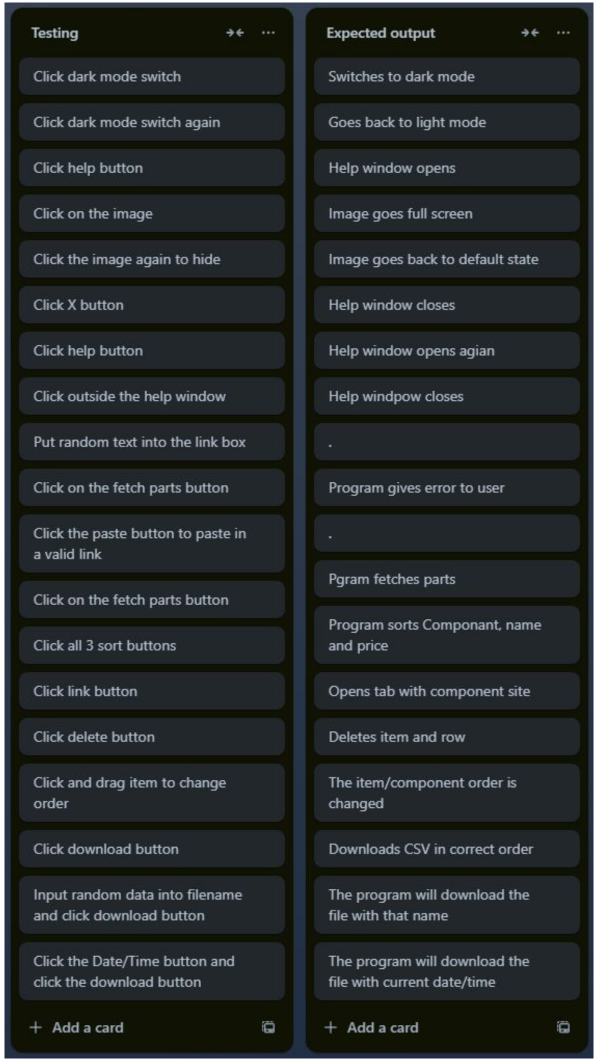
The previous screenshots

Complete Program...

Assemble your components into a working program. On the slides that follow, please provide a test plan and evidence that your program works as expected.

If you are going for M / E, you also need to create extra slides showing how you have used your testing to improve the functionality of your program. This could mean having multiple test plans (and screenshots) showing several iterations of the assembled program.

Complete Program <test plan>



Testing	Expected output
Click dark mode switch	Switches to dark mode
Click dark mode switch again	Goes back to light mode
Click help button	Help window opens
Click on the image	Image goes full screen
Click the image again to hide	Image goes back to default state
Click X button	Help window closes
Click help button	Help window opens agian
Click outside the help window	Help windpow closes
Put random text into the link box	.
Click on the fetch parts button	Program gives error to user
Click the paste button to paste in a valid link	.
Click on the fetch parts button	Pgram fetches parts
Click all 3 sort buttons	Program sorts Componant, name and price
Click link button	Opens tab with component site
Click delete button	Deletes item and row
Click and drag item to change order	The item/component order is changed
Click download button	Downloads CSV in correct order
Input random data into filename and click download button	The program will download the file with that name
Click the Date/Time button and click the download button	The program will download the file with current date/time

Complete Program <testing evidence>

Create as many slides as needed to provide evidence of testing that your outcome works for expected, boundary and unexpected cases. If you wish, you are welcome to submit a *brief* video showing your testing.

https://tinyurl.com/bddz8x5y

Complex Processes - Discussion

Discuss how you used (and combined) information gained from the planning, testing and trialling of your components to improve the quality of your program. Note that **synthesising information** from planning, testing and trialling of components and then discussing how this lead to a high quality outcome are needed for an E grade.
To synthesise means to “bring together” to create something new.

To synthesise information, you need to look at the testing/trialling information from different components, or different stages of development, or from different kinds of testing, and bring together this information.

- You might need to compare contradictory information and work out the best solution.
- You could show how different tests all led to one unified conclusion, or led to new ideas
- You might need to come up with whole new ideas not originally tested

Complex Processes - Discussion

Reflect on how using the processes helped to develop the outcome through effectively testing and trialling various components to refine and enhance the design and functionality.
When you discuss how information led to a high quality outcome you could write about:

- Two (or more) sources of information that disagreed with each other, and say how you decided which to trust
- How your idea changed over time as new testing/trialling evidence was gained and what effect this had
- How you tried different approaches and decided which was best and why

This program was developed to solve a common limitation faced by users of PCPartPicker, a well-known website used for planning and managing custom PC builds. PCPartPicker helps users create part lists by allowing them to select compatible components such as CPUs, GPUs, RAM, storage, power supplies, and more. One of its most useful features is that it automatically compares prices across multiple suppliers based on the user's country, helping people build PCs that match both their performance needs and their budget. Because of this, PCPartPicker is widely used by hobbyists, students, IT professionals, and even businesses that need to build machines for specific tasks.

Despite its usefulness, PCPartPicker has one major drawback: it does not offer a native option for exporting the part list as a spreadsheet. This becomes a problem for users who want to take their part lists offline, include them in reports or documentation, or simply manage them in a tool like Microsoft Excel or Google Sheets. For example, students may want to include their PC build in a portfolio, or IT staff may need to track parts across multiple builds using spreadsheets. Without the ability to export, users are forced to manually copy and paste each item, which is time-consuming and prone to formatting errors.

My program directly addresses this issue by providing a simple way to convert a PCPartPicker list into a downloadable spreadsheet file. Users only need to paste the link to their parts list into the interface, and the program will automatically extract the relevant information including component names, prices, and links and generate a clean, formatted spreadsheet file that can be saved and reused. This significantly improves workflow, saves time, and reduces frustration, especially for users working with large or multiple builds.

Additionally, the program includes a user-friendly graphical interface with features such as dark mode, drag-and-drop functionality, input validation, and the ability to customise file names. These enhancements make the tool accessible even to users with limited technical experience, while still being flexible enough for more advanced users. By turning a previously manual and error-prone task into a quick and reliable process, this program adds real value to the PC building workflow and helps users get more out of an already powerful platform.

Throughout the development of my program, I continually combined information from planning, testing, and trialling to improve the quality and functionality of the final outcome. My use of an evolutionary prototyping approach meant that I was constantly evaluating what worked and what did not, and bringing together findings from different components and development stages to guide improvements.

One key example of this came from testing URL validation. At first, I believed that all user links would follow the format "pcpartpicker.com/user/saved/...". However, during testing I discovered that most actual links used a different structure, such as "pcpartpicker.com/list/...". This conflict between my original assumption and real-world data led me to change my validation logic. Instead of expecting only one link format, I modified the program to allow both formats by checking for either "items" or "alt_items" in the page content. This reduced user errors and significantly improved usability by making the program more forgiving and flexible.

When implementing features like deleting and sorting items, I tested different approaches. I found that if the user deleted all items, the interface became unusable unless the page was refreshed. After trialling alternatives, I chose to reset the interface to its default state when all items were deleted. This approach was clearer for users and prevented errors, which was confirmed through testing.

Another case of combining information came during the design and testing of the Help window. Early versions had issues with scrollbars appearing unnecessarily, layout problems when resizing the window, and inconsistent behavior in dark mode. I gathered this feedback from multiple sources including personal testing, family members, and other students. I then implemented a series of improvements, such as hiding the main page scroll bar when the Help window was open, fixing scaling for smaller screens, and adjusting dark mode styles. These changes were based on both user feedback and trialling under different conditions. Although my attempt to include ARIA labels for accessibility did not work.

In summary, the high quality of the final program came from combining insights from multiple rounds of testing, from different components, and from a range of users. The outcome was significantly improved by responding to conflicting data, adapting to real-world use cases, and trying different approaches before choosing the most effective solution.

I also experimented with visual and interactive design elements such as animated backgrounds, hover effects, and dark mode. User testing showed that hover effects made the interface easier to navigate, and that placeholder text helped guide input. By reviewing this feedback and comparing different trial versions, I made design choices that increased clarity and engagement.

Another important way I combined information to improve the final product was through how I managed the separation of concerns in the codebase. Originally, all the HTML, CSS, and JavaScript were placed in a single file. During early testing, this made editing and debugging difficult. After trialling a modular approach where each type of code had its own file, I found that separating these components made the project easier to manage and made future changes much more efficient. This change also helped with performance and maintainability, which became clear when updating styles or adding new interactive elements like the dark mode toggle.

Feedback on the user interface also led to design changes that were not part of the original plan. For example, testers found that entering a filename right away was confusing or unnecessary at the start. Based on this, I changed the flow so that users could input a link first and choose the filename later. This adjustment came from combining usability testing with observation of user behavior, leading to a smoother workflow that better aligned with user expectations.

Another example of combining information came from refining the sort functionality. When first implemented, sorting caused layout issues in dark mode due to mismatched button styling. Rather than fixing just the styling, I reviewed the entire interaction and redesigned the sort feature to work better across themes and devices. This broader improvement was the result of combining visual trialling, logic testing, and feedback from people using the feature in different ways.

I also discovered during testing that long item names could break the layout. To address this, I introduced text truncation with ellipses. Although this had not been originally planned, it was a solution that came from reviewing component-level testing (table structure and responsiveness) and combining it with usability goals. It not only preserved the layout but also made the overall appearance cleaner.

Throughout development, I was also comparing how different browsers and screen sizes affected performance and display. Small differences between Chrome and Firefox highlighted inconsistencies that led me to adopt more standard CSS practices and improve cross-browser compatibility. This was an example of synthesising testing results from multiple environments and aligning on a solution that worked consistently.

These kinds of improvements many of which were not part of the initial scope were possible only by continuously comparing planned ideas with real-world feedback, interpreting contradictory test results, and merging technical fixes with usability goals. This approach helped create a final program that was reliable, visually consistent, easier to use, and better suited to its intended audience.