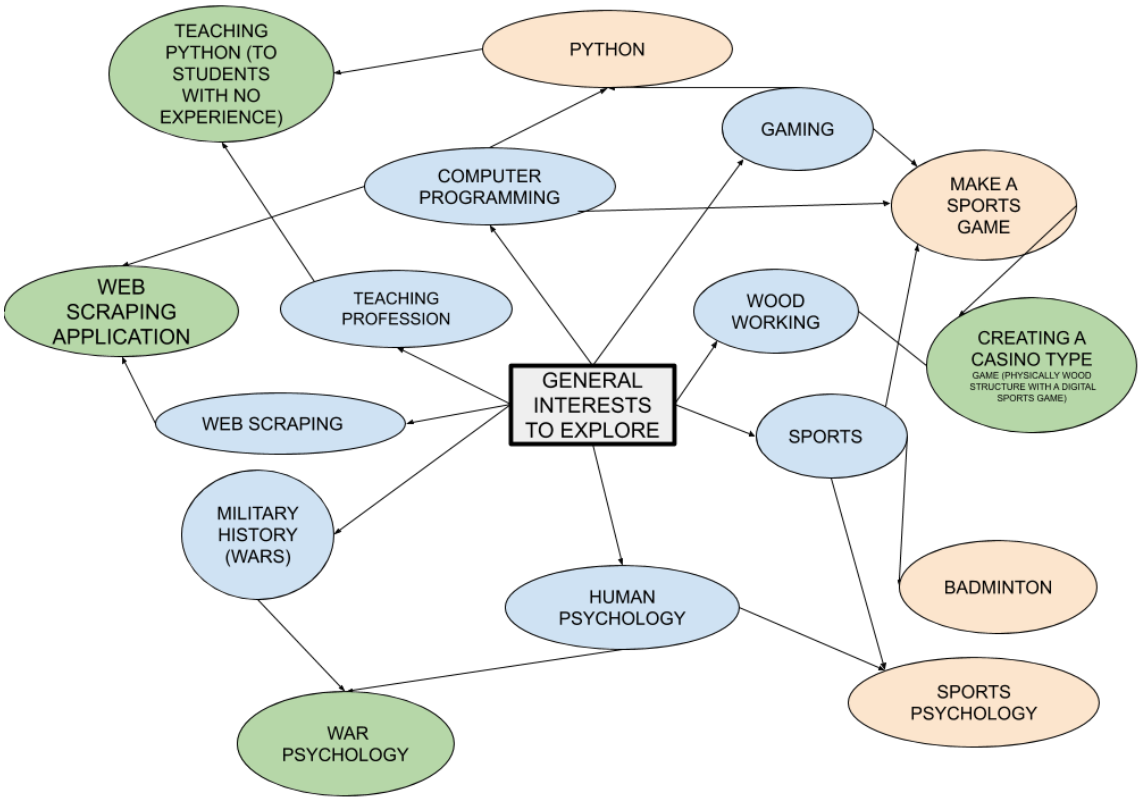
20/07/2019

This the day I began my personal project. After having had an introduction about the Personal Project, the first thing that crossed my mind was to make a brainstorming chart (mind map), to further explore my interests and maybe decide one to pursue in the PP. So here is the brainstorming that I did:

KEY FOR THE BRAINSTORMING

	General Interests (Something I have been doing or wanting to do.)
	More specific interests (Maybe combining of the general interests)
	Possible final ideas/products (With combinations of ideas/interests)



Now that I have brainstormed all my general and explored deeper into possible final ideas for my product/goal, I must decide on one 'possible final idea' and provide reasons as to why.

But first here are explanations of the 'possible final solution:

Casino Type Game	War Psychology	Making a web scraper

• Combines
  ○ Woodworking
  ○ Programming
  ○ Gaming
• Physical structure (made of wood) with a monitor and accessories to play a game created by me
• Great way to learn woodworking and game development, two usually unrelated topics but both interesting and complex.
• The programming languages needed for this is very limited so it is not challenging in that manner. But it will be challenging to build the physical structure from scratch.
• This project will only be challenging physically because making games is nothing out of the ordinary for teenagers in computer programming. It is highly publicized in the internet and made easy with simple modules in python.
	• Combines
	  ○ Psychology
	  ○ Military(war) history
	• In-depth research (using reliable internet sources and books) on the thinking behind prominent wars in history like WW2 and the mental states(tricks) of infamous persons such as Hitler or
	• This is challenging intellectually because it requires a change in the way you reason with events that seem illogical. For example, the events in WW2 have no clear reason behind them, and to understand them you have to think in the manner that the perpetrator of genocides thinks (For example Hitler-WW2 & Pol Pot-Cambodian genocide).
	• However, in my previous school, this topic was briefly covered in history class so it will not be extremely challenging for me personally.
		• Combines
		  ○ Programming
		  ○ Python
		  ○ Website Building
		• Extremely hard to make as in order to scrape(extract information from) specific websites, it requires a deep understanding of the language in which the website is made and the programming language used to extract it.
		• Excellent opportunity to test and improve my python knowledge attained in 5 month Python club + MYP 5 Design class.
		• This is very challenging because it is very hard to retrieve information from large e-commerce websites like Amazon, and it proves how websites like Trivago (web scraper for hotel prices) are extremely hard to create and maintain. Even though to the public eye, it is made to look easy and consumer-based. To make one by yourself is very challenging as it requires a deep understanding of how websites behave and Python's syntax

CHALLENGING Only Physically	CHALLENGING Intellectually	VERY CHALLENGING Very Hard to make
Teaching Python to absolute beginners		

• Combines
  ○ Computer Programming
  ○ Python
  ○ Teaching Profession
• It is a great way to pass on beneficial knowledge to beginners. And also to relearn and master the basics of a very powerful programming language such as Python. This can benefit me and the students by equipping us with the knowledge that is highly beneficial in order to understand the constantly improving digital world, which by some great minds will surpass or be equal to humans in the future. So knowing the basics of a such a powerful language will help us in the future to branch out to any profession, which will all be in some way or form aided by digitization or automation, and we will be able to understand it to a certain extent.
• This is very challenging because teaching requires a thorough understanding of the subject, inferring the fact that I will need to master the syntax of python in order to teach it to others. It requires this level of understanding because it is inevitable that my classmates will have doubts, or misunderstanding of the subject when I teach it, and to resolve them I need to be able to explain it at an extremely complex or easy level, fitting to what their doubt is.

VERY CHALLENGING Very Hard to make

DECISION OF FINAL IDEA:

Above are all my possible final ideas for the personal project, and none of them is extremely challenging (which is how it should be, as the project is over months not weeks). So in order to make my final idea extremely challenging, I will combine the two 'very challenging ideas'. This means combining the idea of a 'Web Scraper' and 'Teaching Python to absolute beginners', I will combine this into a project that equally incorporates both these ideas, and makes it extremely challenging.

VERY CHALLENGING Very Hard to make	+	VERY CHALLENGING Very Hard to make
Electronic product price web scraper (to help consumers learn about best prices)	&	Teaching basics of python to absolute beginners through the product price scraper
EXTREMELY CHALLENGING		

NOTE: When I refer to this project as a website, I mean that it is a web application, as it's role is not just to only display writing and information but to and scrape(mine information from other web bases or websites) with the command of the user. Making it more of a web application than a regular website.

The official definition for a web application is: "*An application program that is stored on a remote server and delivered over the Internet through a browser interface.*"

What is the final project going to be?

My project now will be divided into two phases:

1. Making a web scraper that scrapes electronic product prices to help customers get value for their money & best option
2. Teaching absolute beginners basics of python using the product price web scraper as application-based learning.

In this way when my project is finished it can be displayed as a physical product online (as an accessible website to the public) and with it a large poster presenting all the python lessons taught with pictures and possibly videos. Making the web scraper will better the teaching as well because there is a large difference between theoretical learning and application-based learning. The IB revolves around practical and application-based learning, so I would like to do the same so that voluntary and students and I both learn for the sake of applying effectively later in our life and not for the sake of memorizing.

Why do I want to achieve this?

I want to achieve this for two main reasons:

1. Firstly it is to get experience with application-based programming because I feel that this experience will be important for me to better apply my programming knowledge in the future because I am passionate about computer science-oriented career path. Web scraping in specific will also give me some experience with harder more complicated side of programming which deals with issues like legality (because making money from data scraping some websites is illegal). In addition it will help me better myself in thinking like a programmer, using logic and reasoning for every step taken along the way.
2. Secondly, it will help the voluntary students understand how fascinating a programming language like Python is, in addition to enabling us (including me, because I am re-learning with the voluntary students) with valuable skills of python programming for the future, because even though knowing a programming language thoroughly might be a rare skill, in the future, it might be a necessity for most jobs and career paths. In addition, it will also give us an insight into how large websites like Amazon, Flipkart etc. behave helping us scrape information from them.

Where do I want to do this?

I will be doing the first phase of my project by myself, mostly at the comfort of my house. I predict that it will take me 1.5 - 2 months to complete.
As soon as I am done with the making of the web scraper I will make a plan for teaching it with the basics of python, which I estimate will take me 1 week to do.

The second phase will be entirely in school, as I will be teaching the basics of python and how the web scraper was made in an empty classroom (with laptop screen connected to TV). The teaching part will take around 2-3 weeks to complete depending on how complex much my classmates (students) want to learn the basics of python through web scraping.

So in entirety, the primary half of my project will be done at my house and the remaining will be done at school. The total time taken should be 3 and a half months.

When do I want to do this?

I will be doing the first phase including the planning of teaching, during free time at my house and the second phase will be done during free periods or Zero Periods at school for 2-3 weeks. However, the specifics of the time required and scheduled will be further discussed in criterion B.

What makes this personal project "Personal"?

Ever since I was a kid, I was overly curious about how the technology worked, how one machine connected to another, how did the xerox machine in my house magically print objects ours? how did that website work? Etc. And 1-2 months ago I started experimenting with Python and HTML, in order to resolve the fascinations from my childhood, and I was instantly hooked, however, I never got really far on my own, as I only learnt parts of the theory and never applied it functionally. So, I feel that finally making a programming project on my own will answer some of my questions from a young age, and also provide me with experience in addition to helping me hone a valuable skill for future utilization, in my career.

GLOBAL CONTEXT:

The most relevant global context for my project of making a web scraper and teaching python using it as an application example is Scientific and Technological innovation. As I will be answering the question: How do we understand the world in which we live? Through focusing on exploring the impact of scientific and technological advances on communities and the environment, such as the wonder of acquiring materials (shopping) through a web-application on the internet.

What do I want others to understand through my work?

Firstly, I want them to understand the work that goes behind creating web-applications that use several data points, different programming languages (one for the application itself (python), and another for displaying that application online (HTML & CSS)). I want them to appreciate and comprehend the complexity of making web applications on their own (learning multiple languages), without using the assistance of prebuilt templates.

What impact do I want my project to have?

Specifically, I want to enable the voluntary students and myself with a valuable skill, programming and logical thinking. I sincerely believe that even though this skill is something out of the normal or special at the moment, it will be a necessity in the future. So teaching ourselves and others before is preparing us to cope with bigger challenges and changes in the future.

How can a specific context give greater purpose to my project?

The specific context of 'scientific and technological innovation', will enable me to better focus my energy and time into learning the concepts and way of thinking behind programming applications, while also considering human interactions with it, such as replicating average human behaviour when they look for items to purchase in a real shop. This specificity will help me greatly focus my work on the important elements and ignore the unessential such as dwelling too much on the internal aesthetics of the website, which the consumers will not see.

Prior Learning and subject-specific knowledge

NOTE: Know nothing of the essentials needed / Know something

Know	Want	Learn
PROGRAMMING		
The basics of Python	A more complex understanding of the language and it's syntax through an application-based approach of making the web-application (scraper) 1. Module Beautifulsoup, which enables Python users to scrape HTML LXML (two web-programming languages) websites. 2. Advanced concepts in python: a. Classes & Objects b. Lambda functions c. Utilizing modules	*Will be filled after learning*
I know nothing about HTML, but the fact that they are used to make websites	An intermediate understanding of web design, using the backbone language HTML, which will enable me to make a functional website on my own. 1. Basics of HTML 2. Few advanced concepts of HTML: a. Embedding other programs b. Implementing CSS 3. The coherence of HTML with CSS	*Will be filled after learning*

I know nothing about CSS, except that they are used to improvise on the HTML and make it more usable + aesthetically appealing. I also know nothing about how to connect databases from Python to the front end web application (ie: MySQL)	A basic understanding of CSS, so I can make the interface of my website, usable by the shopping consumers and aesthetically appealing 1. Basics of CSS 2. A basic course of web aesthetics through CSS 3. How to implement CSS into HTML files	*Will be filled after learning*
E-COMMERCE EDUCATION		
I am aware of the popular e-commerce business in my country: 1. Amazon 2. Flipkart 3. Snapdeal	I wish to learn anatomy, of theses eCommerce websites: 1. Supply chain management 2. Shipping and returns 3. Client Relationship management. 4. Catalogue and product display 5. Marketing and Loyalty programs	*Will be filled after learning*
I am unaware of any legality or marketing aspects in getting information from eCommerce businesses	I want to learn the exact legality of theses eCommerce websites so: 1. I can legally obtain information from them 2. Use this information compare product prices between Amazon, Flipkart and Snapdeal. 3. Publish the website online, for customers to use (no profits for me, referencing back to the supplier).	*Will be filled after learning*
TEACHING		
I am not aware of anything about teaching	I want to learn about the basic techniques of teaching: 1. The behaviour of a good teacher	*Will be filled after learning*

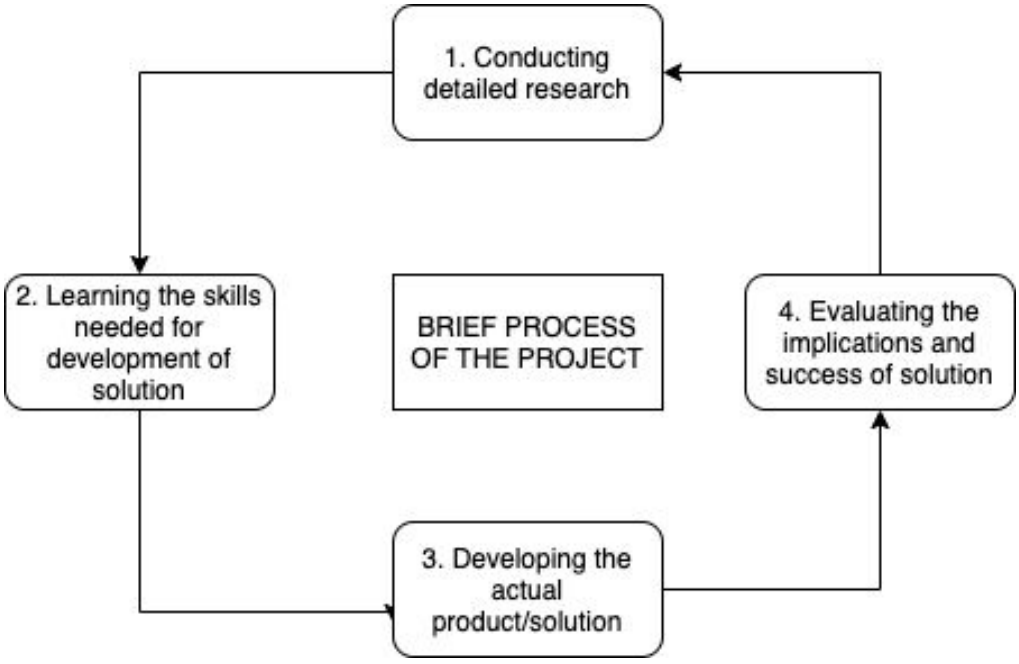
	2. Planning for the teaching 3. How to make engaging and concise lessons 4. Knowledge of the topic 5. etc.	

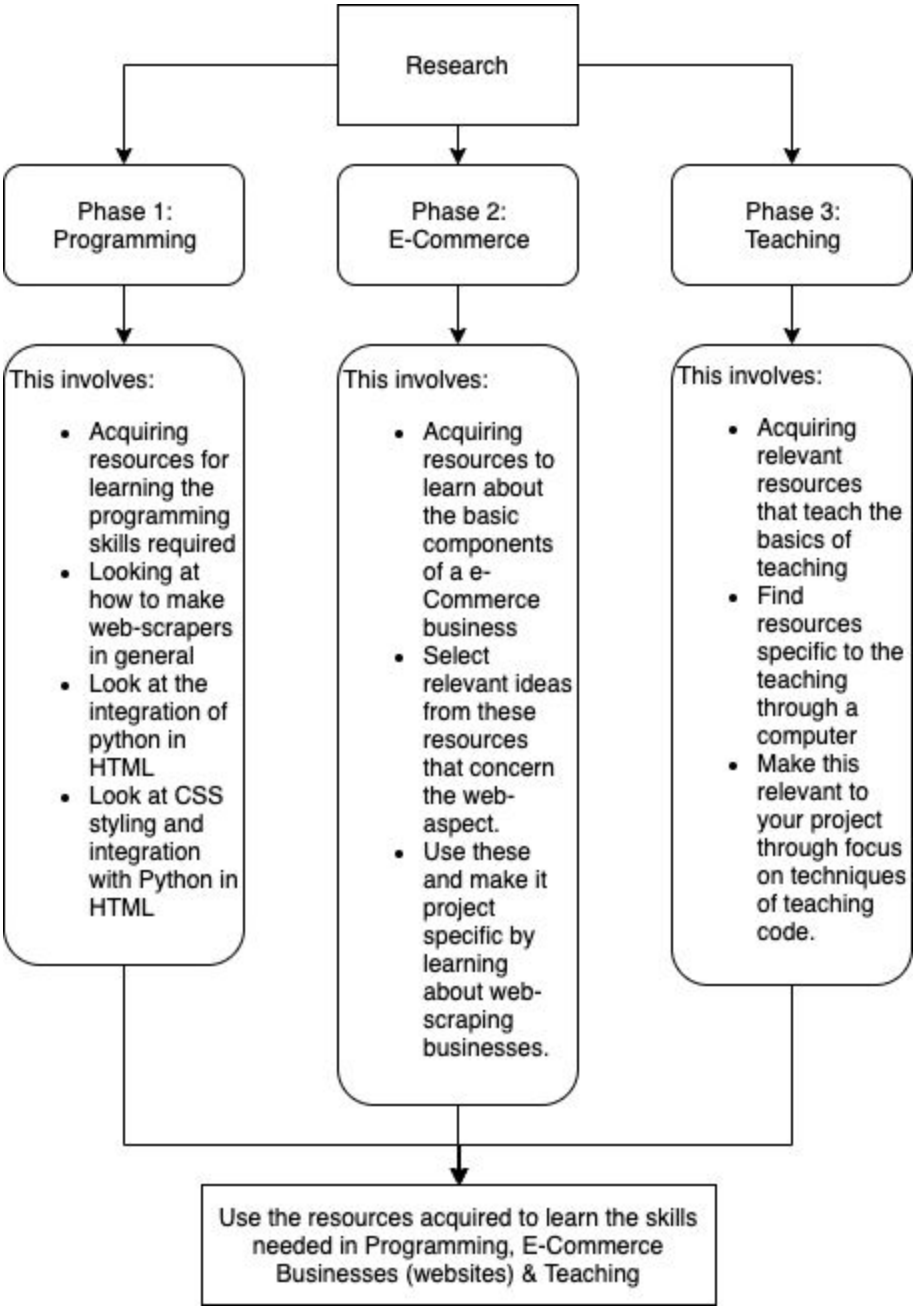
Validation of ideas (using the SMART method):

SPECIFIC What? Where? How?	In this project, there will be two major phases. Phase involves me developing a python programme with the ability to scrape e-commerce websites through a search bar and provide accurate referrable results, this programme will be uploaded online on the website, making it a web application. The second phase involves me teaching my peers (who volunteered for 2 weeks) the basics of python, utilizing my python web-application as an application based example to aid and simplify the understanding of the language. Phase one will be conducted entirely on my own, during my free time at home or otherwise, whereas the second phase will be completed during zero-period in school, and this phase will last exactly 2 weeks. I will be doing the first phase through first equipping myself with the correct programming knowledge needed: Learning advanced Python concepts, HTML and CSS for the styling of the website. In addition, I will also study briefly, a few e-commerce guides, which explain the aspects of the websites and businesses, assisting me in making the web-application more accurate and appealing, after this I will do the project itself. For phase two I will need to equip myself with the basic techniques of teaching, through a free & credible online course, and then teach the web-application with basics.
MEASURABLE From - To	I am now a very novice python programmer, who is aware and in some instances can apply the basic knowledge into making sub-intermediate programmes and applications. Through this project, I want to get to an intermediate level of python programming, where I am proficient with the language and able to apply it to any intermediate task required. For web-design, knowledge about e-commerce and teaching techniques, I wish to get from zero to a basic level, where I am able to understand and apply novice principles.
ASSIGNABLE Who?	My target audience for the product of phase one will be the average e-commerce consumers, who are specifically looking **for electronic products.** However, my website will be uploaded on the internet for anyone to access. In addition, my target audience, the people who wish to learn the basics of python will be 5 peers from Grade 10 who have volunteered for this opportunity. The specifics of the exact people involved in the teaching phase will be discussed furthermore in the planning phase of the project.

REALISTIC Feasible?	It is is a feasible project, as it involves a very self-sustained goal in phase one, as it's success completely relies on my time management and determination to complete learning the different tools and making the application itself. Phase two is a little less reliable, it also depends on my peers' willingness to follow through, however, I am sure that at least 2-3 people (who are close friends) will show up to my class.
TIME-BASED When?	The implication of the project, which is making the web-scraper: **Phase one** and teaching my peers: **Phase two,** will take around 2 ½ months. However, I require 2 more months before that so that I can learn the skills I need for making the web application. And 2 weeks after making the application to learn about teaching techniques and making a plan for the lesson I will have over two weeks time. Which extends the time of my entire project to about 5 ½ -6 months, including some buffer time in between phases

Developing a brief plan for the project:





Selecting sources for research:

No	Origin	Purpose	Value	Limitations

		Phase 1: Programming		
Secondary				
1.	Youtube video, by certified developer tutorials website/channel, freecodecamp.org https://www.youtube.com/watch?v=kMT54MPz9oE The origin of this video is quite trustworthy as the channel releasing it has been giving tutorials to developers since 2014, with no complaints.	The purpose of this source is to teach absolute beginners in web development, the basics of HTML and CSS. HTML is the language that is responsible to the displaying of text, images, tables and other functions, where as CSS deals with making the HTML look more presentable and organized.	This source provides two main values to me: ● It teaches me the absolute basics of HTML ● It teaches me the absolute basics of CSS Which is what I require to know in order to make the web-application The source also gives my information on how a website might look like from a developers point of view, in addition to exploring some advanced concepts that can be utilized by beginners to make their websites look advanced.	The main limitation of this 2 hour long tutorial it these factors: ● Even though the video time is around 2 hours, it will take me almost 2 or 3 times that in order to completely comprehend the lesson. So in reality I will be spending about 5-6 hours spread over a week to learn the basics. ● This tutorial is very basic, so any complex concepts I need to include in my web-app, I will need to learn externally
2.	Python works based off of modules. Modules in python are tools to perform tasks in a more efficient manner. The major module i will be utilizing is the BeautifulSoup module, which allows developers to data	Firstly, this source is made by the individuals who created BeautifulSoup, and they have created a place where people can come and learn all about what they have made. The purpose of this source is to teach	For me this source provides: ● A basic understanding of how the syntax of Python works in BeautifulSoup. ● Give tips and techniques on how to scrape	The main limitations of this source include: ● Because this source is made by the same people who made the module, they will not address the problems with it.

	mine/scrape other websites. The origin of this source is: https://www.crummy.com/software/BeautifulSoup/bs4/doc/ This source can be trusted as it is made by the same people who made the module.	absolute beginners, how to scrape other websites using Python and the module BeautifulSoup.	e-commerce businesses. ● Legality of web-scraping businesses. ● Parsing specific elements of other websites. ● Extracting only text from various elements on a web page.	● Only the benefits and working parts of the module are taught and advertised. ● It is a very large database of tutorials in this source, so it is difficult to find information relevant to my project.
3.	Python can also be displayed online, like in the case of my project, which makes it a web application. So this source will teach me how to upload my python code online so people can use it as a web-app. The origin of this source is: https://www.youtube.com/playlist?list=PLzMcBGfZo4-n4vJJybUVV3Un_NFS5EOgX	The purpose of this source is to teach me how to upload my python code onto the internet, so it can be utilized by other people to find the best price for electronic products on e-commerce websites, In essence it will teach me all I need to know for this project about using a flask server (web server), to host my python application on the internet.	This source will provide me thee values: ● Making web servers using the framework flask ● Using HTML on this web server. ● Using CSS, to style the HTMl on this server. ● Uploading python scripts and executing it through an input and button method on this server	The limitations are: ● The tutorials are very broad, and mostly concern the making of servers displaying simple text. So I will need to learn that in order to learn how to upload python scripts onto the server. ● It is a very long tutorial series and will take me around 2 weeks, because of the extra server-HTML aspect I need to learn.
Primary				
4.				

		Phase 2: E-Commerce		
Secondary				
5.	The origin of this source: https://netonomy.net/2013/10/16/most-important-components-ecommerce-stores-ignored/	It outlines the most major aspects of an e-commerce business, so I can understand the complexity of whose data I will be mining, in addition to selecting key aspects to mine/scrape.	This source provides me these values: ● What goes into making an ecommerce business ● Give me an idea of the legality of the e-commerce business ● Understand what aspects of the business are available from scraping ● Selectively decide on what aspects of the e-commerce business I will mine and display on my website. My understanding of this needs to be very shallow, as I am not trying to build a business. I am just trying to understand who and what I am minding data from.	Limitations: ● It has a lot of irrelevant aspects, that are directed to other people who are trying to create a business themselves. ● I only need 10% of what is being discussed, so finding it is relatively difficult.
		Phase 3: Teaching		
Primary				
6.	Origin of the source: [redacted]	The purpose of this [redacted]sign teacher in my school,	The values of this interview: ● Because she is experienced in teaching code,	There are no limitations to this source.

		Indus International School. I will be interviewing her with a set of 5 basic questions on how to teach other people code. This will provide many values to Phase 3 of my research.	and has been certified with a degree in computer science, It will help me understand best how to teach others using code. • This will inturn, provide with the best knowledge on how to effectively teach my peers how I made the web-applicatio n through the basics of Python.	

Here are the actual contents of the research:

[redacted] email concerning, why a video is the best possible way of teaching code:

[redacted]

My answers are interspersed in your email below in bold.

What method of teaching code is most efficient?
I think it would be by video, however, it may not be the most efficient for all students, however for students who are self learners, online tutorial is very efficient as they can manage their time on their own and practice while they are learning through online media.

As you are a qualified Design teacher, what are some essential tips in teaching code that contains languages like Python, HTML & CSS?
The online tutorial must give a framework for students to practice the code right there, starting from basic syntax to use of statements and commands and to develop logic building through simple programming exercises.

How to approach all levels of programmers in a video tutorial?
You could segregate the topics into various levels as beginner, intermediate and advanced, so programmers can choose topics based on their abilities.

In which order should a web application be taught to students, ex: front end first or backend first etc.?
I have learnt backend first and then the front end part, but there is no convention or standard for

this. It depends on the programmers skill and interest. So if the tutorial clearly explains what are the functions of the front end and back end software are, then the student would have a clear understanding of deciding which one to pick up and learn. For a full stack developer, learning all these side by side is important. So if your tutorial is designed to offer that for students with clear understanding, it would be a great plus.

Should concepts in the language be thoroughly explained prior to showing examples in the web-application, or be explained as the video progresses?
Again it depends on the level of the course. eg. for a beginner level I would explain the concepts and then the examples, for an advance level I would throw them examples to see if they can get to the concepts themselves, may be some hints or guide could be shown. Intermediate would be a mix of these.

How long should a typical coding lesson be?
hmn...again there is no standard rule for this. But in my experience, not too long, not too short is a good starting point. So novice programmers do not get overwhelmed and lose focus by the length of the video, it is good to keep the time to an average attention span of the audience. So may be short for beginners like 10 mins or so and then examples to get hands on work. For intermediate, it could span across 20-30 mins and for advance level it could go from 40 -60 minutes? Also if the framework can track the students progress, so they could pick up from where they left, it would help them to come back to the course.

[redacted]r survey had a good set of questions. I hope my answers will help you build a quality work that you are satisfied with.

Enjoy!

[redacted]

Moving on, here is more research concerning, what I want to compare concerning the product in my solution:

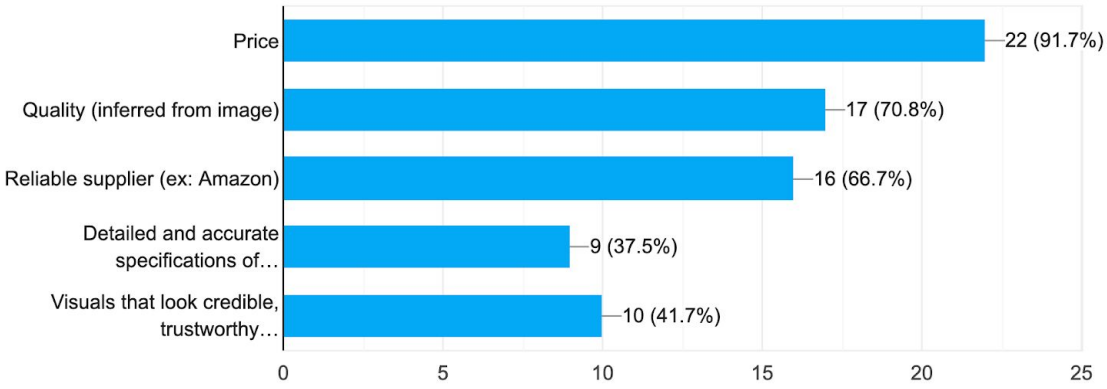
1) I conducted a survey to understand what aspects of a product people would most like to see when comparing products between different suppliers/retailers/companies; in this case Amazon, Flipkart & Snapdeal (link to survey:)

   Link to the survey: https://forms.gle/U8FUmPhuhruUQBUq6

   Here is the data from the above survey that around 25 people answered, and they were mostly peers from my old school, new school and a few family members:

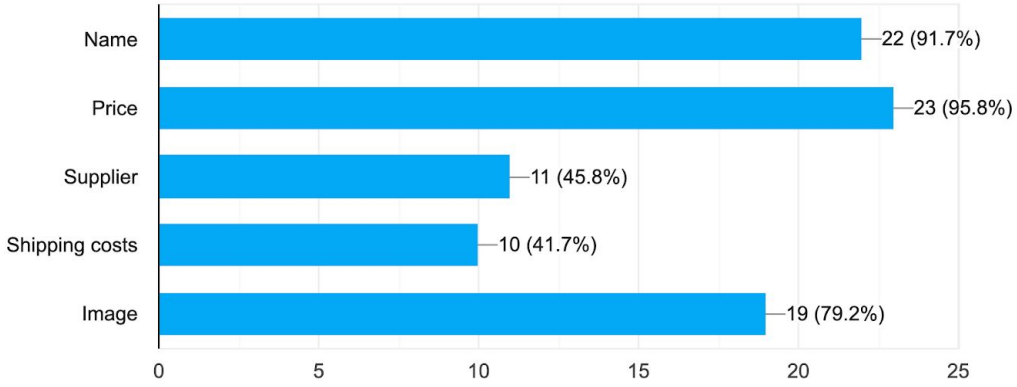
What are the influential factors that affect your purchase decision online?

24 responses

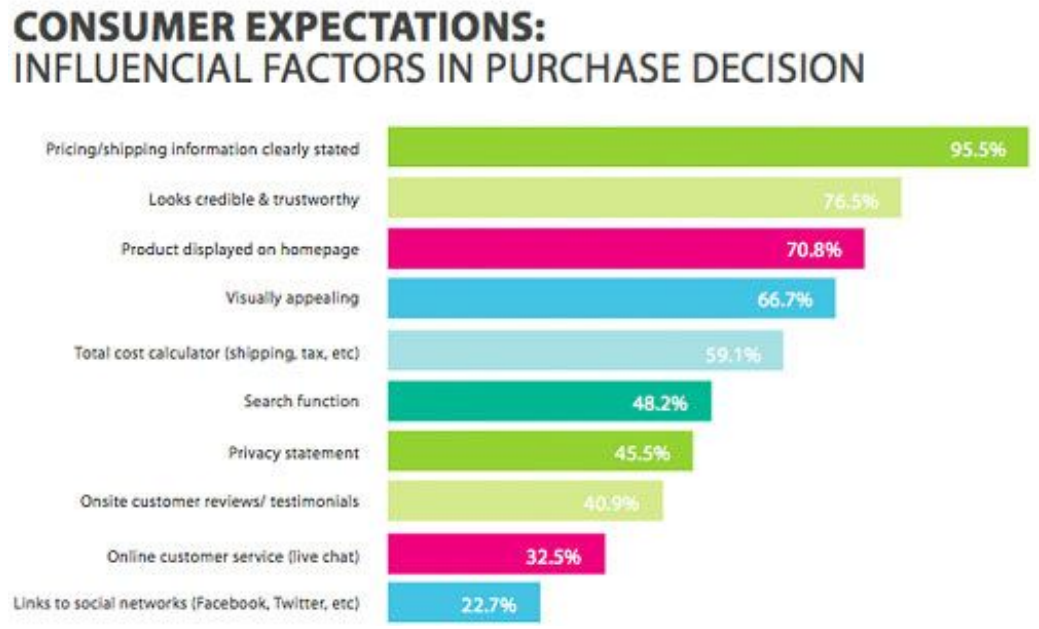


If there was an application that compared products from different eCommerce suppliers like Amazon, Flipkart and Snapdeal, what factors of the product would you compare?

24 responses



2) Second is secondary source which talks about what makes products appealing:



https://www.getelastic.com/customer-expectations

Develop success criteria for the product:

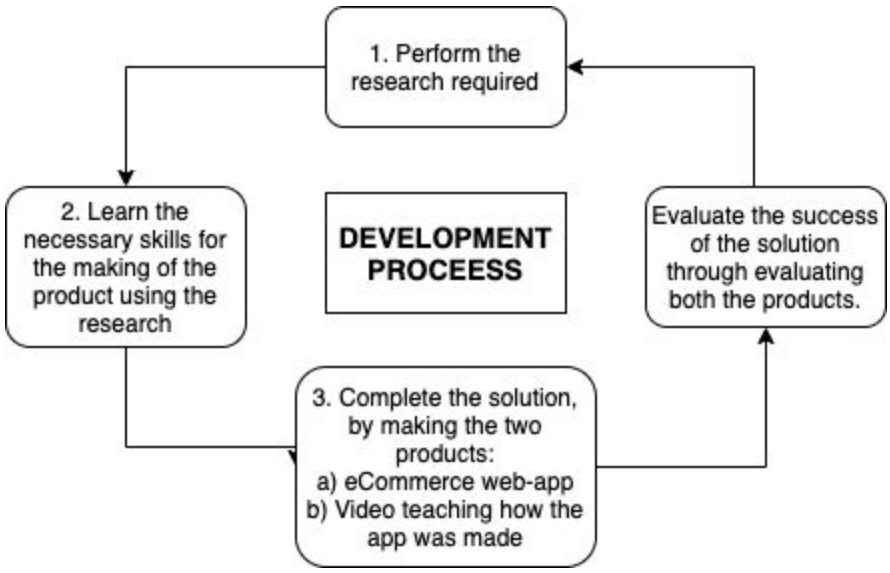
Specification	Results
Product 1 - Web application to assist eCommerce consumers make better product purchases in terms of price and quality	
Aesthetics: The product must be minimalistic in design	✔ This criteria is fulfilled as the layout of the web app is dark and contrasted only with red warnings and a white search bar, making it only a three colour scheme. In addition there is no irrelevant information on the web app, and the results from your search only give you back three tables one from Amazon, Flipkart and Snapdeal results. This is further substantiated from this graph from the survey wherein 17 of my peers/family members where asked if my web app was minimalistic in design and over 70% said that it was
Accessibility: The product must be accessible by anyone with the internet	✔ This product is a web application, and I am only using Amazon, Flipkart and Snapdeal information for comparison and help of users, not for profit, thus this web app can be published into

	a web domain and anyone with an internet connection can access it.
Development/Cost: The product should not cost anything to make	✔️ This criteria was fulfilled by the product, because I spent no money in the process of making, I used Python to code the back end and HTML, CSS to code the front end, and none of these were edited with programmes that cost money, and all the code was written by me.
Development: The product should be made in the span of 3-4 weeks	❌This success criteria was not fulfilled by my product, because it took one week longer than anticipated to complete the product, as there was a problem with scraping data from Amazon, due to their Captcha software, which bypassed by making it clear to their detection that I was only going to use this data for comparison, and I would not make any profits by making my browser header visible and giving all ownership to them. (**Note:** I did not do anything illegal because I am only comparing information and not making any profits myself)
Development: The product should be made only using digital assets (like programming language Python)	✔️This criteria was fulfilled by my web application, as it was entirely made on my laptop, and did not use any physical effort. It was made using the programming language Python for the back end scripting languages HTML and CSS on the front end, comparing product results from Amazon, Flipkart and Snapdeal.
Function: All information provided by the product should be reliable and accurate	✔️This criteria was fulfilled by my product as all the information displayed on the website from national and world renowned eCommerce companies Amazon, Flipkart and Snapdeal, and as my product compared the data from these (unchanged) it was as reliable as these three companies → which is very. This is also further substantiated from this graph from my survey wherein 17 of my peers were asked about if the information from my app was reliable, and 86.2% answered yes
Function: The product must help eCommerce customers make better purchase decisions	✔️According to a survey that I conducted amongst my peers assessing the success of my web app, the results → “” clearly showed that almost all of them believed this product would

	help them make better purchase decisions online. This is further substantiated from this graph from my survey wherein 17 of my peers were asked if my app will help them make better online purchases and 94.1% said yes
Product 2 - A comprehensive video tutorial that explains the programming behind the web application, helping viewers improve their knowledge in programming and eCommerce.	
Accessibility: The product should be accessible by anyone with an internet connection	✔️The product is a video on YouTube, so it is accessible to anyone with an internet connection, thus fulfilling this criteria.
Development: The product should be made free of cost	✔️This product was made using free software available on macbook called Imovie and uploaded to a free streaming website, YouTube, thus it fulfilled the criteria.
Function: The product must teach others programming in Python and HTML of the web application in product one.	✔️This product teaches my product 1(web application) from beginning to start, which contains Python for the back and HTML and CSS for the front end, meaning it does cover programming in Python and HTML.

Development Process:

The broad process of the project, proceed in this manner:



But more specifically, this was the nuances of the big four sub-processes mentioned above:

1. Perform the research required:
    1.1. Technological research, where I acquire information about:
        - Python modules BeautifulSoup (for scraping information from Amazon, Flipkart & Snapdeal), Requests (for accessing web pages of three mentioned) and Flask

(for making the back end networking of my web-application)(***Section 1 → Appendix 4***)

- HTML(scripting) and CSS(styling) for the front end of the website, and MySQL for connecting a database from the Flask back-end to the front end, to store and display information scraped for comparison of products form three eCommerce websites (namely: product name, price, etc.)(***Section 2 → Appendix 4***)

1.2. eCommerce oriented research wherein I (***Section 3 → Appendix 4***):

- Obtain information from previously held surveys, on what customers of an eCommerce website want when comparing products (ie: correctly displayed price info, quality image, verified reseller, etc.)
- Conduct my own survey to people whom I know, go/went to school with, where I broadly cover specific questions on the influential factors that decide what product they purchase when faced with choices, in addition to what factors of the product I must include in my web-application that compares these prices.
- Obtain information on the legality and marketing in the eCommerce businesses, and if I am permitted to publish the web-application online with no strings attached with Amazon, Flipkart or Snapdeal.

1.3. Teaching oriented research wherein I: (***Section 4 → Appendix 4***)

- Interview my MYP 5/Grade 10 Design teacher regarding the most efficient way of teaching others code in addition to getting tips on how to make my teaching of my web-app, the most optimum for learning.

2. Learn the necessary skills for the product through research: (evidence of learning → ***Appendix 4***)
   2.1. Learn the python modules and HTML, CSS and MySQL from the information from the research.
   2.2. Analyse and finally deduce the exact factors that eCommerce customers wish to see in comparison of products utilizing the primary and secondary survey from the research.
   2.3. Decide on and learn the most efficient way of teaching code, from the information obtained through the interview with the Design teacher.

3. Complete the execution of the solution by making the two products:(***Appendix 6***)
   3.1. Product (1): Make web-application which provides users with a comparison between similar products, obtained through the same search query information from Amazon, Flipkart and Snapdeal, executed with the python module, HTML & CSS, MySQL skills obtained in development phase 2
   3.2. Product (2): Make a comprehensive online tutorial going through the entire tech/programming process of the web-application made (recommended by design teacher)

4. Evaluate the success of the solution by assessing the product(s):
   4.1. Perform primary surveys on how helpful the web application and the YouTube programming tutorial is. (***Appendix 7***)

4.2. Compare the final two product solutions against their respective success criteria, to determine the level of success. (***Appendix 5***)

I WOULD LIKE TO FOCUS ON SELF MANAGEMENT, AND HOW I IMPROVED ON THIS:

Self and time management has been a great weakness for me throughout the Middle Years programme, however making a plan for this process journal gave me an opportunity to develop the skills that I had been lacking for a long time. More specifically, through making the development process, I was able to improve my organizational skills, namely simplifying tasks in order to make them easier to separate and get a quicker sense of starting and finishing each task, to finally complete the whole project. Furthermore, I developed great skills in prioritizing research bringing them together to develop my skills, whereas before I used to be lost in selecting the relevant details. In addition, I also developed a far better perseverance due to the betterment of my planning skills, the personal project at first seemed extremely time consuming and borderline impossible, but as I broke down into four processes, and further into sub-sections, it seemed doable and I could focus on it for the entire duration. Lastly, I improved greatly in reflecting, as I had to maintain a process journal wherein I would discuss problems I encountered with myself, or sometimes others making it far easier to get the solutions, and through these repetitive reflection I could improve my whole project overall.

I FINISHED THE PLANNING PHASE, NOW I MOVE ON TO THE ACTION:
First step is doing all the research and learning all the skills required:

Beautiful Soup:

```
from bs4 import BeautifulSoup  → # importing module in
```

```
Soup = BeautifulSoup("html_page", 'setting')
```

this will be the keyword used to parse the HTML

this page will be received by the module requests.

#The setting can be either 'lxml' or 'HTML.parser' depending on web page ~~(xxx)~~.

Now there are the most common methods of parsing the file for information

```
Soup.find('tag', {'class': 'class_name'})
```

This is the easiest way to identify specific tags and their information in a large websites like:

- → Amazon
- → Flipkart
- → Snapdeal

Soup (find_all)
find All } can be used to find all ~~tags~~ information within a website that contain the same tags.

Eg	Amazon	Flipkart	Snapdeal
product name	('div' {'class': 'product_name'})	('div', {'class': '_1_W02 lx'})	.select('product-title')
price	('span' {'class': 'a-price-whole'})	('div', {'class': '_2 clol|3cP'})	.select ('l-float product-price')
image link	('img' {'class': 's-image'})	('div', {'class': '_31iAHj'})	.select ('.picture-elem')

All these information will come with a lot of unwanted information, so we need to do,

for x in their_list:
    (new_list).append (x, text)
else:
    print ("already text)
    pass

→ This will have all the product names, prices, & images in a tuple format [ ]

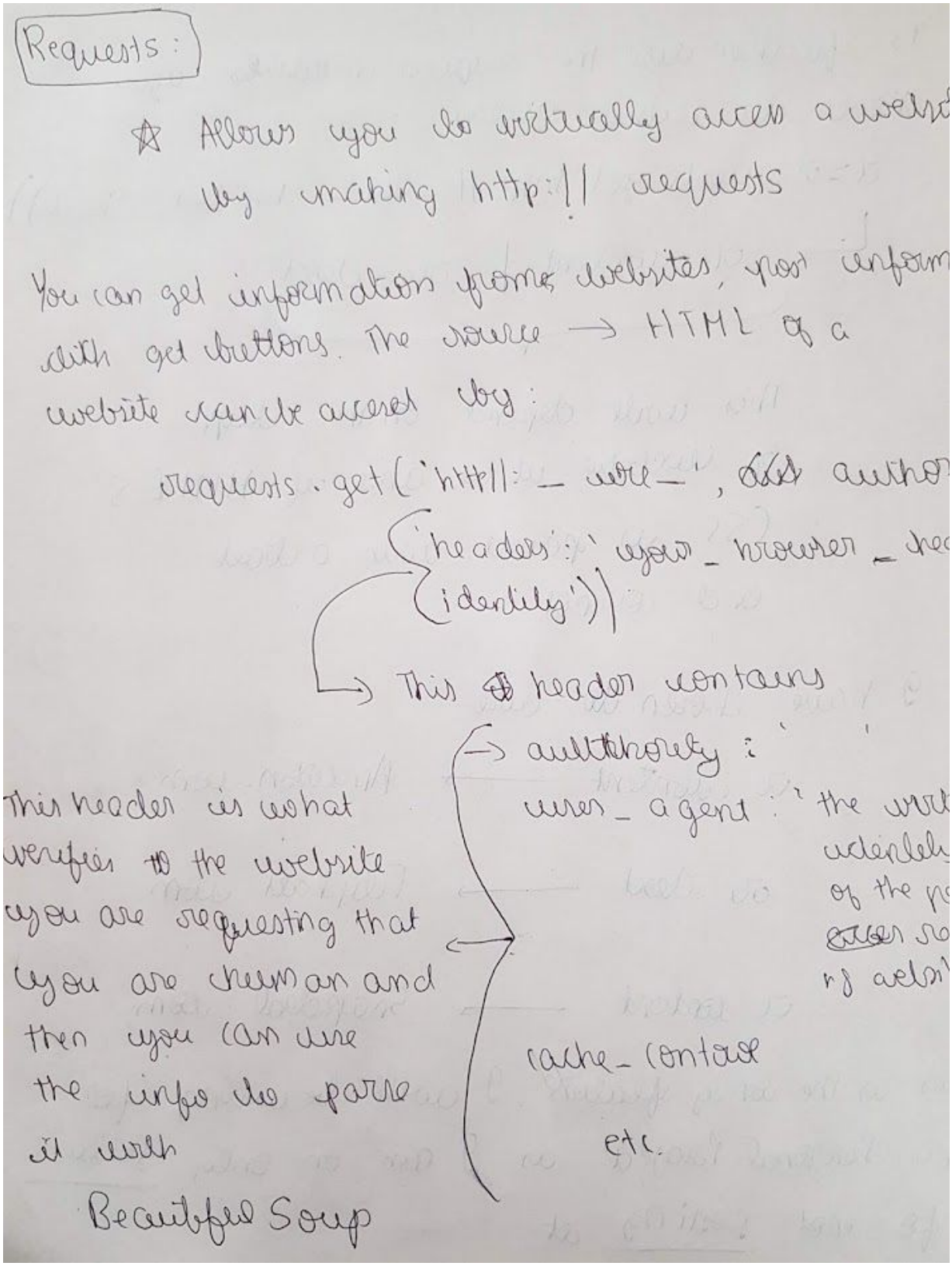
list = [(name, price link), (name, price, link). (name, p

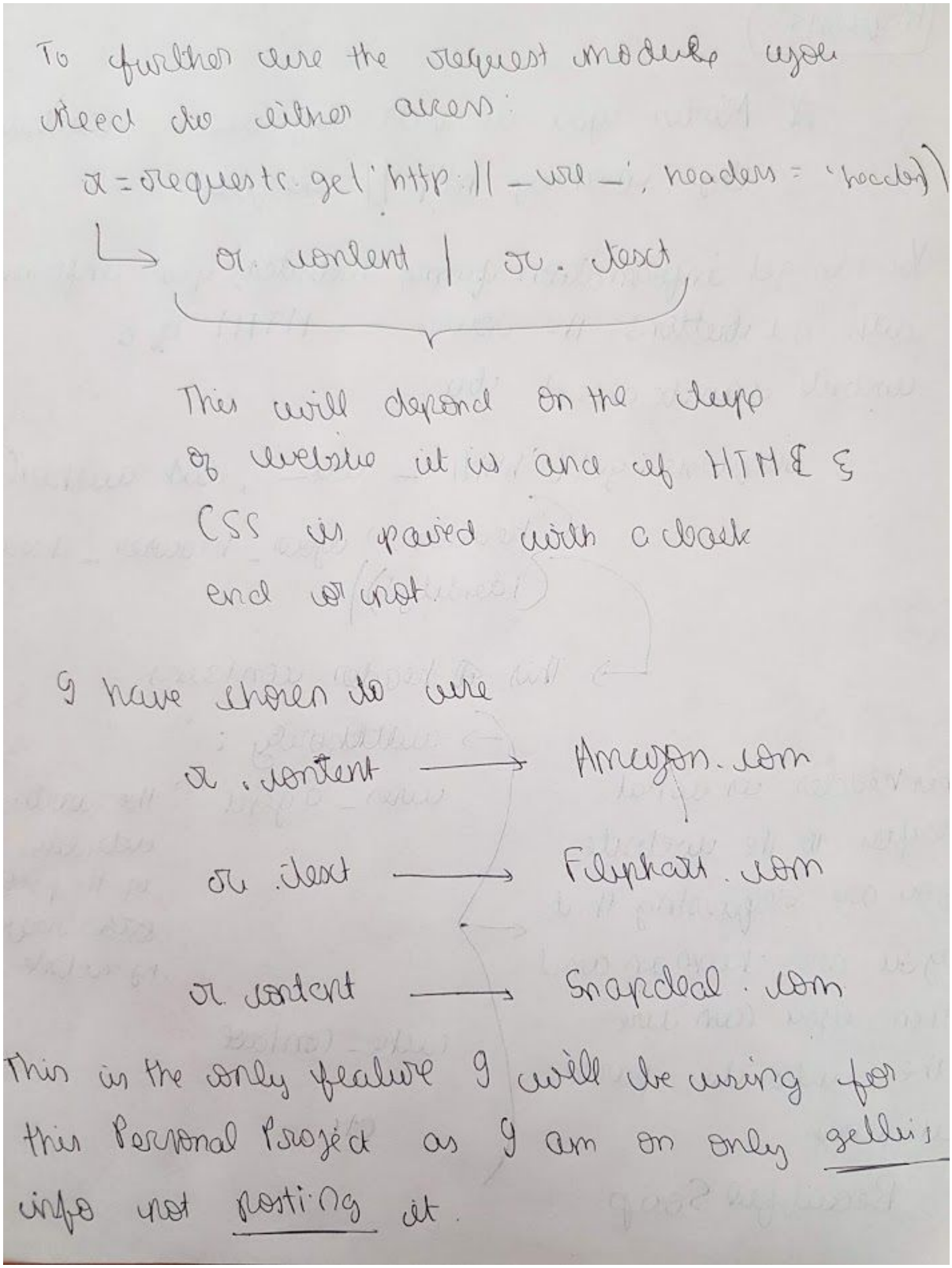
In this case it in two steps:

→ select container of the tags

→ Then get specific tag an their info

Learning Requests module (used for making http requests to amazon etc. and getting source text)





Learning Flask (the framework used to set up the back end of my web-app, which connect the web page itself to the BeautifulSoup/Requests code, and a database):

Flask :

Flask will be used to handles make the back end of my web-application

There are concepts in Flask are of interest to my web-application:

→ @ app.route('/'_page name_)

methods:
→ Post
→ Get

learning:

making the initialization:

app = Flask(__name__)

end the back end

making a home page

@ app.route('/' or '/home')

this will make the home page however an HTML file should be attached to make it work.

@app.route('/' or '/home' etc.)
render_template('file_name.html')

This will only work if there are not methods of getting or posting information into or out of the website

GET

@app.route('/', methods = ['POST'])
render_template('file_name.html')

POST

@app.route('/some_name', methods = ['GET
render_template('file_name.html',
value_that_you_want
to_post = x)

In the HTML FILE while posting → get the input by

<p> The value being received is {{ value_
-you-wa
to_post}
<p

Running the app:

```
if __name__ == "__main__":
    app.run()
```

In order to learn HTML and CSS, I watched this video: [HTML and CSS course] and made the website myself:
This is the HTML & CSS I wrote after researching and learning from that video, and made the static website below, to understand, experiment and familiarize concepts I learned in the video.

The following are some additional interests of mine, besides gaming and computers:

- *Badminton*
- *Movies*
- *Writing Articles (Basic Journalism)*

The following are some of my favourite fruits:

1. *Mangoes*
2. *Jack Fruit*
3. *Oranges*
   - *Bananas are tied with Oranges as I have them several times throughout the day, for energy.*
4. *Strawberries*
5. *Other berries in general*
6. *etc.*

Seasonal or Regional restrictions for my favourite fruits:

Mangoes
They are only available in Southern Asia, during the summer.
Jack Fruit
No seasonal or geographical restrictions that I am aware of.
Oranges
Available all throughout the year.
Bananas
No restrictions whatever, however Bananas in southern India are marginally smaller (Just a fun fact).
Berries
Available all throughout the year.

First thing that came up when I googled "Complicated table", so plagarized it to make myself look smart.

Random data that makes me look smart

Polygon categories	*Data struct*	*Max clique*	*Total*
Random	*-1.24%*	*31.48%*	*10.78%*
Random with holes	*-3.19%*	*30.69%*	*10.72%*
Orthogonal	*-1.21%*	*21.39%*	*5.86%*
Orthog. with holes	*-15.46%*	*24.27%*	*-0.15%*
Random high	*-51.82%*	*87.72%*	*85.76%*
Total	*-29.42%*	*86.83%*	*80.99%*

Please login to your google account.

Don't worry I cannot see it because I haven't learnet JavaScript just yet.

Enter your username

Now, please enter all other usernames and passwords that you have made.

Username:
Password:

Username:
Password:

Username:
Password:

Username:

This is the research and learning of MySQl, which is the database I used to connect the Flask back end of my web application to the HTML & CSS front end:

I chose writing to learn code, as I believe it is the best way of remembering!

MySQL:

A server can be established in your local host by downloading MySQL and making a new server inside system preferences.

Initializing the connection to the server from Python

```
import mysql.connector
mydb = mysql.connector.connect(
    host = "localhost",
    user = "root"
    passwd = "password",
    database = "database name")
```

Then we have to create tables and edit them:

```
cursor = mydb.cursor
establish cursor
```

TABLES

creating :

```
cursor.execute("CREATE TABLE
    _name_(value TEXT,
           value INT(20),
           imag TEXT))
```

inserting data :

→ "INSERT INTO _name_(value, value, image)
VALUES (%s, %s, %s)

place holders

getting data :

print("SELECT * database_name.table_name

This represents all but can be singulars or multiple (column names).

I finished the research segment and started utilizing the skills I learnt to make my actual web app:
Link to GitHub [redacted]
The above is a link where all the code I wrote for this project lies!!!

But more specifically here is the python code:
This is everything that I wrote, and ended up with over 13 days, below it is a brief explanation of everything I did, on each of those 13 days.

Python

```
import mysql.connector
from flask import Flask, render_template, url_for, redirect, request
from bs4 import BeautifulSoup
import requests
import time
import random
from tabulate import tabulate
import pandas as pd
import datetime

search_quereies = []

Snapdeal_Names = []
Snapdeal_Prices = []
Snapdeal_imagelinks = []
Snapdeal_Prices_2 = []

Flipkart_Names = []
Flipkart_Prices = []
Flipkart_Prices_2 = []
Flipkart_ImageLinks = []

Amazon_Names = []
Amazon_Prices = []
Amazon_Prices_2 = []
Amazon_ImageLinks = []


def clear(list):
    list.clear()


clear(search_quereies)

clear(Snapdeal_Names)
```

```
clear(Snapdeal_Prices)
clear(Snapdeal_Prices_2)
clear(Snapdeal_imagelinks)

clear(Flipkart_Names)
clear(Flipkart_Prices)
clear(Flipkart_Prices_2)
clear(Flipkart_ImageLinks)

clear(Amazon_Names)
clear(Amazon_Prices)
clear(Amazon_Prices_2)
clear(Amazon_ImageLinks)


time_variable = random.uniform(1,1.5)

mydb = mysql.connector.connect(host="localhost",
                        user="root",
                        passwd="dinky2004",
                        database="scrape_database")
print(mydb)
cursor = mydb.cursor()

app = Flask(__name__)

@app.route('/')
def home():
   return render_template("index.html")


@app.route('/', methods=['POST'])
def my_form_post():
   text = request.form['u']
   processed_text = text.lower()

   search_quereies.append(processed_text)

   if len(search_quereies) >= 1:

      Product_search = search_quereies[0]

      def Integrated():
         def Amazon():
            time.sleep(time_variable)
            URL = "https://www.amazon.in/s?k=" + str(Product_search) + "&ref=nb_sb_noss_2"
            headers = {
```

```
            'authority': 'www.amazon.in',
            'cache-control': 'max-age=0',
            'rtt': '50',
            'downlink': '10',
            'ect': '4g',
            'upgrade-insecure-requests': '1',
            'user-agent': 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_4) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/79.0.3945.88 Safari/537.36',
            'sec-fetch-user': '?1',
            'accept':
'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/
signed-exchange;v=b3;q=0.9',
            'sec-fetch-site': 'same-origin',
            'sec-fetch-mode': 'navigate',
            'referer': 'https://www.amazon.in/',
            'accept-encoding': 'gzip, deflate, br',
            'accept-language': 'en-GB,en;q=0.9,en-US;q=0.8,nl;q=0.7',
            'cookie': 'session-id=258-1636592-3456905; i18n-prefs=INR;
ubid-acbin=257-5529898-5487600;
x-wl-uid=1ir5E8+OGhMOBpYNk5vAaB/JiH6qK69EwafO54hquG79/1zQlrhpNsM5nmNrkgP7e/m69
DA9SWNY=; lc-acbin=en_IN;
session-token=RNXDMxPntpb6YB4qLb/SPv+B2D0zCLft5u0EuG4qsBl5C7QyxS8Vu28Sm2iu9j1LS
73JtkQsBpHnu6bYxStohPe6gNbEvpgHsJ7m8ld188mgFVDm8Wtjri7Iaq9R5TvjF4zgFmwEP21zD9hf
8zmSmODV/8yDxYZ5lTS5McKbQossGXMLNLHZxSopMuq3jN4A; visitCount=28;
session-id-time=20827584011;
csm-hit=tb:s-D9EHD2TB6FW1FS29NDVB|1577817064939&t:1577817066038&adb:adblk_yes',
        }
        time.sleep(time_variable)

        try:
            recieve = requests.get(URL, headers=headers, timeout=10)
        except:
            "One or more of the websites are unresponsive, please retry later."
            exit()

        soup = BeautifulSoup(recieve.content, 'html.parser')

        def initial_viability_test():
            test_count = 0
            The_Whole_Page = soup.prettify()
            while test_count < 2:
                print(The_Whole_Page)
                test_count += 1
                time.sleep(time_variable)

        def name_scrape():
            time.sleep(time_variable)
```

```
            outlines = soup.find_all("span", {"class": "a-size-medium a-color-base a-text-normal"})

            for outline in outlines:
                name = outline.text
                Amazon_Names.append(name)

            if len(Amazon_Names) is 0:
                outlines_2 = soup.findAll("span", {"class": "a-size-base-plus a-color-base
a-text-normal"})

                for outline_2 in outlines_2:
                    name_2 = outline_2.text
                    Amazon_Names.append(name_2)

            # print(Amazon_Names)

        def price_scrape():
            time.sleep(time_variable)

            outlines = soup.findAll("span", {"class": "a-price-whole"})

            for x in range(len(outlines)):
                outline = outlines[x]
                price = outline.text
                Amazon_Prices.append(price)

            # print(Amazon_Prices)

        def image_scrape():
            time.sleep(time_variable)
            outlines = soup.findAll("img", {"class": "s-image"})

            for x in outlines:
                image_link = x['src']
                Amazon_ImageLinks.append(image_link)

            # print(Amazon_ImageLinks)

            # print(outlines)

        # initial_viability_test()
        name_scrape()
        price_scrape()
        image_scrape()

    def Snapdeal():
```

```
        try:
            retrieve = requests.get(
                "https://www.snapdeal.com/search?keyword=" + str(Product_search) + "&sort=plrty",
timeout=2)
        except:
            "One or more of the websites are unresponsive, please retry later."
            exit()

        retrieve = retrieve.text

        data = BeautifulSoup(retrieve, 'lxml')

        def initial_viability_test():
            time.sleep(time_variable)
            test_count = 0
            The_Whole_Page = data.prettify()
            while test_count <= 100:
                print(The_Whole_Page)
                test_count += 1

        def name_scrape():
            time.sleep(time_variable)
            names = data.select('.product-desc-rating')
            for name in names:
                product_identification = name.select('.product-title ')
                product_name = product_identification[0].getText()

                Snapdeal_Names.append(product_name)
            # print(Snapdeal_Names)

        def price_scrape():
            time.sleep(time_variable)
            prices = data.select('.product-desc-rating')
            for price in prices:
                price_identification = price.find_all('span', 'lfloat product-price')
                price_values = price_identification[0].getText()

                Snapdeal_Prices.append(price_values)

            for x in range(len(Snapdeal_Prices)):
                Snapdeal_Prices[x].strip()
            # print(Snapdeal_Prices)

        def image_scrape():
            time.sleep(time_variable)
            images = data.select('.picture-elem')
```

```
                for x in images:
                    image = x.find_all('img')
                    for y in image:
                        image = y.get('data-src')
                        Snapdeal_imagelinks.append(image)

                # print(Snapdeal_imagelinks)

            name_scrape()
            price_scrape()
            image_scrape()

            # for x in range(len(Snapdeal_Names)):
            # output = "Name:" + str(Snapdeal_Names[x]) + ", Price:" + str(Snapdeal_Prices[x]) + ",
Image:" + str(Snapdeal_imagelinks[x])
            # print(output)

        def Flipkart():
            headers = {
                'Connection': 'keep-alive',
                'Cache-Control': 'max-age=0',
                'Upgrade-Insecure-Requests': '1',
                'User-Agent': 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_4) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/79.0.3945.88 Safari/537.36',
                'Sec-Fetch-User': '?1',
                'Accept':
'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/
signed-exchange;v=b3;q=0.9',
                'Sec-Fetch-Site': 'same-origin',
                'Sec-Fetch-Mode': 'navigate',
                'Accept-Encoding': 'gzip, deflate, br',
                'Accept-Language': 'en-GB,en;q=0.9,en-US;q=0.8,nl;q=0.7',
            }

            URL = "https://www.flipkart.com/search?q=" + str(
                Product_search) +
"&otracker=search&otracker1=search&marketplace=FLIPKART&as-show=on&as=off"

            try:
                recieve = requests.get(URL, headers=headers)
            except:
                "One or more of the websites are unresponsive, please retry later."
                exit()

            time.sleep(3)
            recieve = recieve.text
```

```
soup = BeautifulSoup(recieve, 'lxml')

def initial_viability_test():
    test_count = 0
    The_Whole_Page = soup.prettify()
    while test_count <= 100:
        print(The_Whole_Page)
        test_count += 1

def name_scrape():
    time.sleep(time_variable)
    outlines = soup.findAll("div", {"class": "_1UoZlX"})

    for x in range(len(outlines)):
        outline = outlines[x]
        identify = outline.find("div", {"class": "_3wU53n"})
        name = identify.text
        Flipkart_Names.append(name)

    if len(Flipkart_Names) is 0:
        outlines_2 = soup.findAll("div", {"class": "_3liAhj"})

        for y in range(len(outlines_2)):
            outline_2 = outlines_2[y]
            identify_2 = outline_2.find("a", {"class": "_2cLu-l"})
            name_2 = identify_2.text
            Flipkart_Names.append(name_2)
    else:
        pass
    # print(Flipkart_Names)

def price_scrape():
    time.sleep(time_variable)
    outlines = soup.findAll("div", {"class": "_1UoZlX"})

    for x in range(len(outlines)):
        outline = outlines[x]
        identify = outline.find("div", {"class": "_1vC4OE _2rQ-NK"})
        price = identify.text
        Flipkart_Prices.append(price)

    if len(Flipkart_Prices) is 0:
        outlines_2 = soup.findAll("div", {"class": "_3liAhj"})

        for y in range(len(outlines_2)):
            outline_2 = outlines_2[y].find("div", {"class": "_1vC4OE"})
            price_2 = outline_2.text
```

```
                Flipkart_Prices.append(price_2)

            else:
                pass
            # print(Flipkart_Prices)

        def image_scrape():
            time.sleep(time_variable)
            outlines = soup.findAll("div", {"class": "_1UoZlX"})

            for x in range(len(outlines)):
                outline = outlines[x]
                identify = outline.find("div", {"class": "_3BTv9X"})
                image = identify.find("img")
                image_link = image['src']
                Flipkart_ImageLinks.append(image_link)

            if len(Flipkart_ImageLinks) is 0:
                outlines_2 = soup.findAll("div", {"class": "_3liAhj"})

                for y in range(len(outlines_2)):
                    outline_2 = outlines_2[y]
                    identify_2 = outline_2.find("div", {"class": "_3BTv9X"})
                    image_2 = identify_2.find("img")
                    image_link_2 = image_2['src']
                    Flipkart_ImageLinks.append(image_link_2)

            else:
                pass
            # print(Flipkart_ImageLinks)

        name_scrape()
        price_scrape()
        image_scrape()
        # initial_viability_test()

    Amazon()
    Flipkart()
    Snapdeal()

Integrated()

def Data():
    for price in Snapdeal_Prices:
        updated_price = price.strip("Rs. ")
        updated_price_2 = updated_price.replace(",", "")
        Snapdeal_Prices_2.append(updated_price_2)
```

```
    """
    Data_Table_Snapdeal = pd.DataFrame(
        {'Name': Snapdeal_Names,
         'Price(Rs.)': Snapdeal_Prices_2,
         'Image(link)': Snapdeal_imagelinks, }
    )
    file = open("Integration_Data", "w")

    file.write(str(Data_Table_Snapdeal))
    """

    for price in Flipkart_Prices:
        price_2 = price.strip("₹")
        price_3 = price_2.replace(",", "")
        Flipkart_Prices_2.append(price_3)
    """
    Data_Table_Flipkart = pd.DataFrame(
        {'Name': Flipkart_Names,
         'Price(Rs.)': Flipkart_Prices_2,
         'Image(link)': Flipkart_ImageLinks,
         }
    )

    file = open("Integration_Data", "a")

    file.write(str(Data_Table_Flipkart))

    upper_limit = [len(Amazon_Names), len(Amazon_Prices), len(Amazon_ImageLinks)]
    upper_limit.sort()
    """

    for price in Amazon_Prices:
        comma_less = price.replace(",", "")
        Amazon_Prices_2.append(comma_less)
    """
    Data_Table_Amazon = pd.DataFrame(
        {"Name": Amazon_Names[0:upper_limit[0]],
         "Price(Rs.)": Amazon_Prices_2[0:upper_limit[0]],
         "Image(link)": Amazon_ImageLinks[0:upper_limit[0]],
         })
    file = open("Integration_Data", "a")

    file.write("\n")
    file.write(str(Data_Table_Amazon))
    """
def SQL():
    def create_tables():
```

```
            # If you want to change the settings of a table you must use --> "ALTER TABLE x

            try:
                cursor.execute("CREATE TABLE snapdeal (name TEXT, price INTEGER(20), image
TEXT)")
            except:
                "Table already created"

            try:
                cursor.execute("CREATE TABLE flipkart (name TEXT, price INTEGER(20), image
TEXT)")
            except:
                "Table already created"

            try:
                cursor.execute("CREATE TABLE amazon (name TEXT, price INTEGER(20), image
TEXT)")
            except:
                "Table already created"

        def insert_data():

            def snapdeal():
                sqlFormula = "INSERT INTO snapdeal (name, price, image) VALUES (%s, %s, %s)"
                data = []

                for x in range(len(Snapdeal_Names)):
                    data_installment = (Snapdeal_Names[x], Snapdeal_Prices_2[x],
str(Snapdeal_imagelinks[x]))
                    data.append(data_installment)

                try:
                    cursor.execute("TRUNCATE TABLE snapdeal")
                except:
                    print("Table already empty")

                print(data)

                try:
                    cursor.executemany(sqlFormula, data)
                except:
                    print("YOU FAILED")
                mydb.commit()

            def flipkart():
                sqlFormula = "INSERT INTO flipkart (name, price, image) VALUES (%s, %s, %s)"
```

```
            data = []

            for x in range(len(Flipkart_Names)):
                data_instance = (Flipkart_Names[x], Flipkart_Prices_2[x], str(Flipkart_ImageLinks[x]))
                data.append(data_instance)

            try:
                cursor.execute("TRUNCATE TABLE flipkart")
            except:
                print("Table already empty")

            print(data)
            try:
                cursor.executemany(sqlFormula, data)
            except:
                print("YOU FAILED")
            mydb.commit()

        def amazon():
            sqlFormula = "INSERT INTO amazon (name, price, image) VALUES (%s, %s, %s)"
            data = []

            upper_limit = [len(Amazon_Prices_2), len(Amazon_ImageLinks), len(Amazon_Names)]
            upper_limit.sort()

            limit = min(upper_limit)

            for x in range(limit):
                data_instance = (Amazon_Names[x], Amazon_Prices_2[x],
str(Amazon_ImageLinks[x]))
                data.append(data_instance)

            try:
                cursor.execute("TRUNCATE TABLE amazon")
            except:
                print("Table already empty")

            print(data)
            try:
                cursor.executemany(sqlFormula, data)
            except:
                print("YOU FAILED")

            mydb.commit()

        snapdeal()
        flipkart()
```

```
        amazon()

    create_tables()
    insert_data()

Data()
SQL()

def clear(list):
    list.clear()

clear(search_quereies)

clear(Snapdeal_Names)
clear(Snapdeal_Prices)
clear(Snapdeal_Prices_2)
clear(Snapdeal_imagelinks)

clear(Flipkart_Names)
clear(Flipkart_Prices)
clear(Flipkart_Prices_2)
clear(Flipkart_ImageLinks)

clear(Amazon_Names)
clear(Amazon_Prices)
clear(Amazon_Prices_2)
clear(Amazon_ImageLinks)

flipkart_upload = []
amazon_upload = []
snapdeal_upload = []

cursor.execute("SELECT * FROM scrape_database.flipkart")
my_results = cursor.fetchall()

for result in my_results:
    flipkart_upload.append(result)

cursor.execute("SElECT * FROM scrape_database.amazon")
my_results_2 = cursor.fetchall()

for result in my_results_2:
    amazon_upload.append(result)

cursor.execute("SELECT * FROM scrape_database.snapdeal")
my_results_3 = cursor.fetchall()
```

```
        for result in my_results_3:
            snapdeal_upload.append(result)

        #print(flipkart_upload)
        #print(amazon_upload)
        #print(snapdeal_upload)

        return render_template("result.html", flipkart=flipkart_upload, amazon=amazon_upload,
snapdeal=snapdeal_upload)

    else:
        print("The user messed something up.")
        return render_template("index.html")


if __name__ == "__main__":
    app.run()
```

HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Scrape.it()</title>
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css') }}">
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css"
integrity="sha384-Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9I
fjh" crossorigin="anonymous">

</head>
<body>
    <div class="indexbg">
<form method="POST">

    <nav class="navbar navbar-expand-lg navbar-dark bg-dark">
        <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarText"
aria-controls="navbarText" aria-expanded="false" aria-label="Toggle navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarText">
          <ul class="navbar-nav mr-auto">
            <li class="nav-item">
              <a class="nav-link" href=[redacted]
target="_blank">MY GIT HUB</a>
```

```
          </li>
          <li class="nav-item" >
            <a class="nav-link" href="http://127.0.0.1:5000/">HOME</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="https://pypi.org/project/beautifulsoup4/"
target="_blank">DOWNLOAD MODULES USED</a>
          </li>
        </ul>
      </div>
    </nav>

  <hr>


  <div class="container">
    <div class="result_text">
      <h1>Scrape.it()</h1>
      <div class="warning">
        <h3>WARNING: Due to some issue with the scraping software utilized in this
web-application, the flipkart images will not be rendered. </h3>
      </div>
      <hr>
      <p align=center>↓Here are the results from your scrape query↓</p>
    </div>
  </div>


  <div align="center">

    <br>
    <br>
    <div class="tables">
    <div class="white">
    <h2>AMAZON DATA</h2>
    </div>
    <table class="table">
      <thead class="thead-dark">
        <tr>
          <th>Name</th>
          <th>Price(₹)</th>
          <th>Image(link)</th>
        </tr>
      </thead>
        {% for x in amazon %}
      <tbody>
        <tr>
```

```
            <td>{{x[0][0:60]}}</td>
            <td>{{x[1]}}</td>
            <td><img src="{{x[2]}}" style=" max-width:100px; max-height:100px;"></td>
        </tr>
    </tbody>

    {% endfor %}

</table>

<br>
<br>

<div class="white">
<h2>FLIPKART DATA</h2>
</div>
<table class="table">
    <thead class="thead-dark">
        <tr>
            <th>Name</th>
            <th>Price(₹)</th>
            <th>Image(link)</th>
        </tr>
    </thead>
        {% for x in flipkart %}
    <tbody>
        <tr>
            <td>{{x[0]}}</td>
            <td>{{x[1]}}</td>
            <td><img src="{{x[2]}}" style=" max-width:100px; max-height:100px;"></td>
        </tr>
    </tbody>

    {% endfor %}
</table>

<br>
<br>

<div class="white">
<h2>SNAPDEAL DATA</h2>
</div>
<table class="table">
    <thead class="thead-dark">
        <tr>
            <th>Name</th>
            <th>Price(₹)</th>
```

```
            <th>Image(link)</th>
          </tr>
        </thread>
          {% for x in snapdeal %}
            {% if x[2] == "None" %}
        <tbody>
          <tr>
            <td>{{x[0][0:60]}}</td>
            <td>{{x[1]}}</td>
            <td><p>Image Unavailable</p></td>
            {% else %}
          <tr>
            <td>{{x[0][0:60]}}</td>
            <td>{{x[1]}}</td>
            <td><img src="{{x[2]}}" style=" max-width:100px; max-height:100px;"></td>
          </tr>
        </tbody>

          {% endif %}
        {% endfor %}
      </table>
       </div>
   </div>
</form>
</div>
</body>
</html>
```

First day of coding:
The first day, I had no problems, I was just experimenting with the module BeautifulSoup and all my requests to snapdeal was getting accepted

Fourth day of coding:
This is where I finished the implementation of scraping for snapdeal, and this was very easy, being that they do not have any captcha software at all

Day 5:
I have encountered a major problem with Amazon, as I have just realised that, they include a captcha software that blocks the user if it detects a certain amount of requests at the same time

Day 7:
Still have the same problem with Amazon, compulsively looked through the internet for solutions but did not find one

Day 8:
Problem solved, with the inclusion of headers:

headers = {
'authority': 'www.amazon.in',
'cache-control': 'max-age=0',
'rtt': '50',
'downlink': '10',
'ect': '4g',
'upgrade-insecure-requests': '1',
'user-agent': 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_4) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.88 Safari/537.36',
'sec-fetch-user': '?1',
'accept': 'text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9',
'sec-fetch-site': 'same-origin',
'sec-fetch-mode': 'navigate',
'referer': 'https://www.amazon.in/',
'accept-encoding': 'gzip, deflate, br',
'accept-language': 'en-GB,en;q=0.9,en-US;q=0.8,nl;q=0.7',
'cookie': 'session-id=258-1636592-3456905; i18n-prefs=INR; ubid-acbin=257-5529898-5487600; x-wl-uid=1ir5E8+OGhMOBpYNk5vAaB/JiH6qK69EwafO54hquG79/1zQlrhpNsM5nmNrkgP7e/m69DA9SWNY=; lc-acbin=en_IN; session-token=RNXDMxPntpb6YB4qLb/SPv+B2D0zCLft5u0EuG4qsBl5C7QyxS8Vu28Sm2iu9j1LS73JtkQsBpHnu6bYxStohPe6gNbEvpgHsJ7m8ld188mgFVDm8Wtjri7Iaq9R5TvjF4zgFmwEP21zD9hf8zmSmODV/8yDxYZ5lTS5McKbQossGXMLNLHZxSopMuq3jN4A; visitCount=28; session-id-time=2082758401l; csm-hit=tb:s-D9EHD2TB6FW1FS29NDVB|1577817064939&t:1577817066038&adb:adblk_yes',

This solved the problem by letting the captcha software that it is trusted, personal computer accessing the data/website and not an automated computer that is mining the data for malicious activity. In addition to conveying that all the rights for the website still remain to them.

Day 10:
This was the final day, I figured out and became, well versed with headers and bypassing captchas and finished flipkart with the same method:

Day 11:
This is the day I made my website, the front end of my website, I used the my HTML tutorial skills, that I learnt 15 days, back to style the website to the best of my capability, and finally configured it so that it can be revisited as many times as possible

Day 12:
Though I was done with the product, it turns out that there was a problem that occurs when I press enter

or search too many times in the search bar, making it show faulty results. I tried to fix this for the rest of the day, but realised that it was a problem with my MySQL database that I cannot really fix it, so I put up a warning right above the search bar on the web app warning them about the faulty results, hoping the user only press either the search button or enter once.

Day 13:
I wrapped up everything I need for the implementation of the project, today.

Reflecting on my implementation, research, and everything that happened before this point:

Here is the updated Know want learn chart, with the learn part filled in, to illustrate the exact things I learnt in each of the main parts of my project:

Know	Want	Learn
PROGRAMMING		
The basics of Python	A more complex understanding of the language and it's syntax through an application based approach of making the web-application (scraper) 1. Module Beautifulsoup, which enables Python users to scrape HTML LXML (two web-programming languages) websites. 2. Module Requests that allows you to virtually access websites without opening it. 3. Module Flask, which makes the back end of a web application enabling you to connect the website to the python code in the back.	• *I learnt the module BeautifulSoup which parses websites for specific information* • *I learnt the module Requests which gets the html source pages by making virtual http requests without manually going to websites* • *I learnt the module Flask, which makes the back end of a web application*
I know nothing about HTML, but the fact that they are used to make websites	An intermediate understanding of web design, using the backbone language HTML, which will enable me to make a functional website on my own. 1. Basics of HTML	*I learnt HTML completely:* • *Tables* • *Text* • *Lists* • *Etc.*

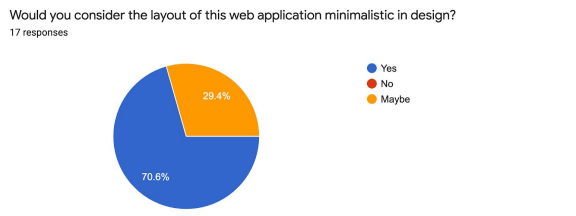
	2. Few advanced concepts of HTML: a. Embedding other programs b. Implementing CSS 3. Coherence of HTML with CSS	*And I learnt how to get and post with connection to Flask inside the HTML code.*
I know nothing about CSS, except that they are used to improvise on the HTML and make it more usable + aesthetically appealing. I also know nothing about how to connect databases from Python to the front end web application (ie: MySQL)	A basic understanding of CSS, so I can make the interface of my website, usable by the shopping consumers and aesthetically appealing 1. Basics of CSS 2. A basic course of web aesthetics through CSS 3. How to implement CSS into HTML files	*I learnt the basic colour and format styling in CSS.* *I learnt MySQL connector which makes the database for the back end and then connects it to the front end.*
ECOMMERCE EDUCATION		
I am aware of the popular ecommerce business in my country: 1. Amazon 2. Flipkart 3. Snapdeal	I wish to learn the aspects of eCommerce products that customers of my web application would like to compare to make the best decision in quality and in price.	-------
I am unaware of any legality or marketing aspects in getting information from eCommerce businesses	I want to learn the exact legality of theses eCommerce websites so: 1. I can legally obtain information from them 2. Use this information to compare product prices between Amazon, Flipkart and Snapdeal. 3. Publish the website online, for customers to use (no profits for me, referencing back to the supplier).	*I learnt that it is inherently illegal to scrape information of website, if you wish to use it to make profit, however I also learnt that my project only helps the customers, and does not make profit, and at the end of they day, those customers will return to the very websites that I scraped information from*
TEACHING		
I am not aware of anything about teaching	I want to learn about the basic techniques of teaching:	*I learnt from my design teacher that the most optimum way of*

	1. Behaviour of good teacher 2. Planning for the teaching 3. How to make engaging and concise lessons 4. Knowledge of the topic 5. etc.	*teaching code is through interactive videos, and realised that you must you use the work that you have done, in your code, and have a concrete understanding of the langage to teach comprehensively/effectively*

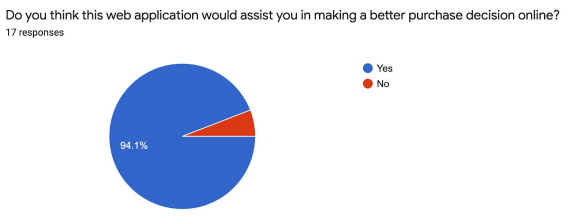
I conducted a survey with my peers and family members concerning the success of my solution, in order to rate my success overall:

Results of the survey I conducted amongst my peers and family members, about how successful/helpful my first product, the web application is:
Link to the survey → https://forms.gle/WJVW5iZs5Y6fYLFm8

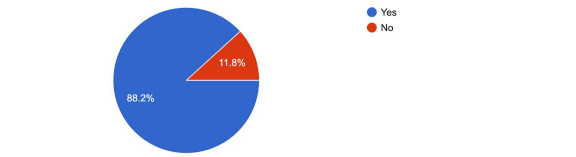
1)



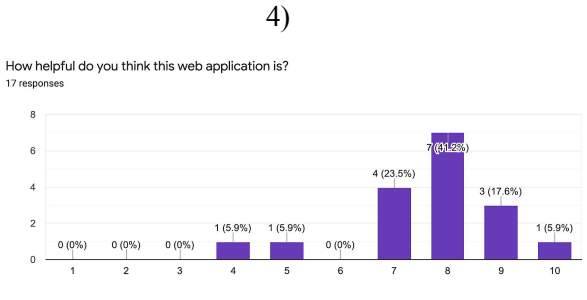
2)

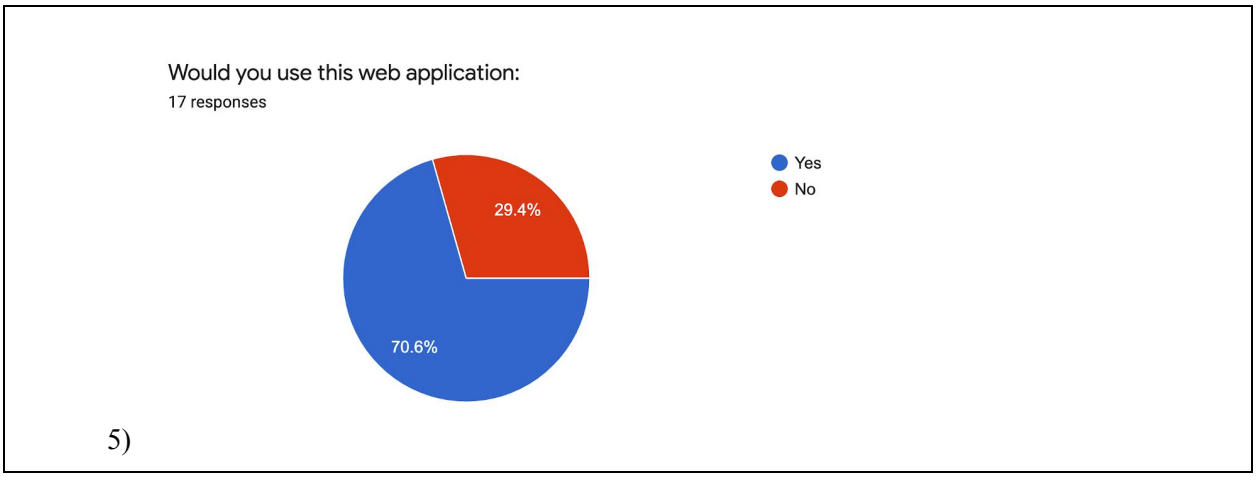


3)



4)





5)

And personally, I believe I developed in these aspects of an IB Learner throughout the course of this project.

My development as an IB Learner:

My school has taught me that the IB learner profile is an integral part of the characteristics of leader of tomorrow, and through the course of this project, I grew a significant amount in each of the learner profiles:

Thinker: I thought critically when I had to solve any errors in my code, and I evaluated all the possibilities before I made progress, and made decisions throughout my programme, which also helped me bypass a large problem with scraping of Amazon (does not allow repetitive scrape attempts, through activating captcha, however bypasses by me using headers (making my online identity seen))

Risk Taker: Because I was tackling a problem statement which concerns the simplification of e-commerce shopping, involving taking information from large companies like Amazon, there were a lot of parameters of legality, and if I had taken the wrong steps I would have broken the law, but because I did not use the product information for my own benefit, but for the benefit of others, while crediting back to the very companies that sold the product, in addition to the fact that there was no advertisements on my website, I did not break any laws.

Open Minded: Prior to this project, I was very underdeveloped in my programming skills, and truly did not believe in my ability to tackle a problem such as this one, however the only thing that changed this belief, was my open mindedness throughout the journey, where I took advice from my peers and parents, who are more well versed with technology and software engineering.

Balanced: In addition to open mindedness, the only other thing that helped me achieve my goal was a structure and balance in my plan. Due to the well spaced time plan from the planning

phase of this project, which meticulously identified and prioritized each nuance of this project, I was able to spend minimal time in planning the research, and programming, and most time actually doing it.

Knowledgeable: I used to be under the impression that programming is a skill, that is extremely difficult to master, but as I discovered the countless resources online that aided me in expanding my knowledge in Python, Web Development, and Web Scraping, I realised that it is the way you utilize the resources available to you that determines your success in learning new topic, like programming.

Inquirer: I also developed a great deal as an inquirer, because all the skills that I learnt to make my product were sourced from research I conducted online, wherein I learnt the valuable skill determining the reliability of a source, and prioritizing research for maximum learning.

BIBLIOGRAPHY

Soup, Beautiful. "Beautiful Soup Documentation¶." Beautiful Soup Documentation - Beautiful Soup 4.4.0 Documentation, 18 Oct. 2016, www.crummy.com/software/BeautifulSoup/bs4/doc/#making-the-soup.

"Testing Flask Applications¶." Testing Flask Applications - Flask Documentation (1.1.x), flask.palletsprojects.com/en/1.1.x/testing/#the-application.

Schafer, Corey. "Requests Python Course." Youtube.com, YouTube, 26 Feb. 2019, www.youtube.com/watch?v=tb8gHvYlCFs.

Freecodecamp.org, director. HTML CSS Course. HTML and CSS Full Course, YouTube, 15 Oct. 2018, www.youtube.com/watch?v=kMT54MPz9oE.

SQL, My. "MySQL Connector/Python Developer Guide :: 5.1 Connecting to MySQL Using Connector/Python." MySQL, Mysql.com, 12 July 2017, dev.mysql.com/doc/connector-python/en/connector-python-example-connecting.html.

Bustos, Linda. "Top 10 Things Customers Expect from Your Online Store." Get Elastic Ecommerce Blog, Get Elastic Ecommerce Blog, 3 July 2019, www.getelastic.com/customer-expectations.

Kho, Julia. "How to Web Scrape with Python in 4 Minutes." Medium, Towards Data Science, 11 Oct. 2019, towardsdatascience.com/how-to-web-scrape-with-python-in-4-minutes-bc49186a8460.

Useful, Make Data. “Python Web Scraping Past Captcha.” YouTube, Youtube.com, 28 Oct. 2019, www.youtube.com/watch?v=auDx0aVWn-s&t=409s.

Roberts, Edward. “Is Web Scraping Illegal? Depends on What the Meaning of the Word Is: Imperva.” Blog, Imperva, 27 Sept. 2019, www.imperva.com/blog/is-web-scraping-illegal/.