

CS 320 Course Project Final Report

For

Discrete Probability Calculator

Version 2.0

Prepared by

Group Name: Group 2

Patrick Tsai
Zhicheng Zhou (Z)
Vuochlang Chang (Anna)

011709316
011735296
011706492

patrick.tsai@wsu.edu
zhichengzhou123@gmail.com
vuochlang.chang@wsu.edu

Date:

12/16/2020

Contents

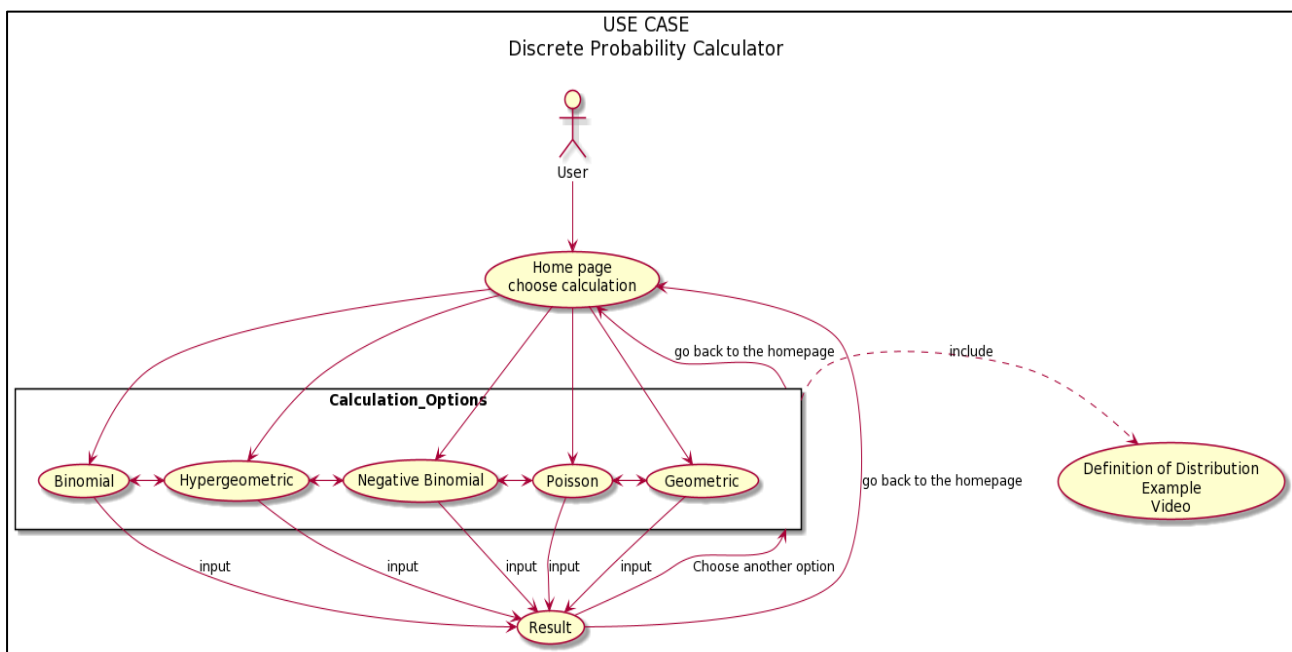
CONTENTS	II
1 INTRODUCTION	1
1.1 PROJECT OVERVIEW.....	1
1.2 DEFINITIONS, ACRONYMS AND ABBREVIATIONS	2
1.3 REFERENCES AND ACKNOWLEDGMENTS.....	4
2 DESIGN	6
2.1 SYSTEM MODELING	6
2.2 INTERFACE DESIGN	20
3 IMPLEMENTATION.....	23
3.1 DEVELOPMENT ENVIRONMENT.....	23
3.2 TASK DISTRIBUTION.....	24
3.3 CHALLENGES	26
4 TESTING	28
4.1 TESTING PLAN	28
4.2 TESTS FOR FUNCTIONAL REQUIREMENTS	29
4.3 TESTS FOR NON-FUNCTIONAL REQUIREMENTS.....	36
4.4 HARDWARE AND SOFTWARE REQUIREMENTS.....	45
5 ANALYSIS.....	47
6 CONCLUSION.....	49
APPENDIX A - GROUP LOG	50

1 Introduction

The Discrete Probability Calculator is a simple website-based application that users can use to calculate the probability of the discrete distribution. There are multiple ways where the user can access the calculations for each distribution. As shown in *Figure 1*, the user can access each distribution mainly from the homepage and directly from the other distributions as well. The overall goals of the system are:

- To provide easy access to the users to calculate discrete probabilities; Users can access DPC website through any browsers whether from their smartphones or computers
- To provide as another option from a TI-84 calculator or other versions
- To help users to understand each probability better with the Help sections
- To improve our understanding of building a web application that interacts with user input.

The DPC system is intended for study purposes, its intended use is for student/academic questions. However, the DPC system is NOT intended to be used as a tool to make any real-life decisions, as that any probabilities derived from the program should not be used to make important life decisions.



(Figure 1- Use Case Diagram)

1.1 Project Overview

Almost every decision humans' make relies on calculating a probability. We make quick probability calculations for simple everyday actions such as crossing the street, we may consider how far away the next car coming toward us is, how wide the street is, how

fast we normally walk, we then quickly analyze these variables before we take our next step. Probability is an integral part of decision making, and a fundamental part of human life. However, most of us do not know the formal way of calculating probability, and therefore there is opportunity for all of us to make better informed decisions.

The Discrete Probability Calculator is a simple website-based application that user can use to calculate the probability of the discrete distribution. This section will cover the general view of the software including its purpose, scope, intended audiences, terms definitions, document conventions for this paper. The overall goals of the system are:

1. To provide easy access to the users to calculate discrete probabilities; Users can access the DPC website through any browsers whether from their smartphones or computers
2. To provide as another option from a TI-84 calculator or other versions
3. To help users to understand each probability better with the Help sections
4. To improve our understanding of building a web application that interacts with user input.

The DPC system is intended for study purposes, its intended use is for student/academic questions. However, the DPC system is NOT intended to be used as a tool to make any real-life decisions, as that any probabilities derived from the program should not be used to make important life decisions.

1.2 Definitions, Acronyms and Abbreviations

TERMS	DEFINITION
Binomial Distribution	Binomial distribution counts the number of successes in some repeated independent experiments, each trial can only consist two possible outcomes: SUCCESS OR FAILURE. The parameters for Binomial distribution are: <ol style="list-style-type: none"> 1. The number of trials. 2. The number of successes. 3. The probability of a success on a given trial (the probability of a success may not vary)
Discrete Probability Distribution	Discrete Probability distribution describes the occurrences of every possible value of a random discrete variable, discrete means that the random variable can only take on some non-negative, non-decimal integer value such as: 0 -> positive infinity.
DPC	Discrete Probability Calculator

Geometric Distribution	<p>Geometric distribution is counting the probability of the number of trials on which the first success can happen. The parameters for Geometric distribution are:</p> <ol style="list-style-type: none"> 1. The number of trials that takes for the first success to happen. 2. The probability of one success can happen.
Hypergeometric Distribution	<p>Hypergeometric distribution is similar to Binomial distribution but with the variance in the probability of a success. In Hypergeometric distribution, the given probability changes for each trial due to the replacement of samples. The parameters for Hypergeometric distribution are:</p> <ol style="list-style-type: none"> 1. The number of items in total. 2. The number of items that are classified as success in total items. 3. The number of items in the chosen sample. 4. The number of items that are classified as success in the chosen sample. 5. The number of success of failure that needs to be found.
Input	The given data from the User
Negative Binomial Distribution	<p>Negative Binomial distribution is looking for the probability of the number of trials takes to produce n successes. The parameters for Negative Binomial distribution are:</p> <ol style="list-style-type: none"> 1. The number of successful trials. 2. The kth trial where the number of successful trials happen on.
Poisson Distribution	<p>Poisson distribution is counting the probability of some number of outcomes can occur during a given time interval (usually represented as μ) The parameters for Poisson distribution are:</p> <ol style="list-style-type: none"> 1. The mean (expected value) of the random variable. 2. The number of outcomes.
SDD	Software Design Document
User	The person who is using the DPC software

1.3 References and Acknowledgments

- [1] “How to Cite References: IEEE Documentation Style”, *IEEEDataPort*, Available: <https://iee-dataport.org/sites/default/files/analysis/27/IEEE%20Citation%20Guidelines.pdf>. [Accessed Oct 27, 2020].
- [2] “Binomial Distribution: Formula, What it is, and how to use it in simple steps,” *Statistics How To*, 06-Jul-2020, Available: <https://www.statisticshowto.com/probability-and-statistics/binomial-theorem/binomial-distribution-formula/>. [Accessed: 10-Dec-2020].
- [3] “Geometric Distribution: Definition & Example,” *Statistics How To*, 16-Sep-2020, Available: <https://www.statisticshowto.com/geometric-distribution/>. [Accessed: 10-Dec-2020].
- [4] “Stat Trek,” *Hypergeometric Distribution*. Available: <https://stattrek.com/probability-distributions/hypergeometric.aspx>. [Accessed: 10-Dec-2020].
- [5] “Negative Binomial Experiment / Distribution: Definition, Examples,” *Statistics How To*, 24-Sep-2018, Available: <https://www.statisticshowto.com/negative-binomial-experiment/>. [Accessed: 10-Dec-2020].
- [6] “Stat Trek,” *Poisson Distribution*, Available: <https://stattrek.com/probabilitydistributions/poisson.aspx>. [Accessed: 10-Dec-2020].
- [7] “Include another HTML file in a HTML file,” *Stack Overflow*, 01-Mar-1961, Available: <https://stackoverflow.com/questions/8988855/include-another-html-file-in-a-html-file>. [Accessed: 10-Dec-2020].
- [8] “Online Mathematics Editor a fast way to write and share mathematics,” *Mathcha*. Available: <https://www.mathcha.io/>. [Accessed: 12-Dec-2020].
- [9] C. W. Helstrom and R. H. Myers, *Probability and statistics for engineers*. New York: Macmillan, 1991.
- [10] “PageSpeed Insights,” *Google*. Available: <https://developers.google.com/speed/pagespeed/insights/>. [Accessed: 12-Dec-2020].
- [11] “HTML,” *W3Schools Online Web Tutorials*. Available: <https://www.w3schools.com/>. [Accessed: 12-Dec-2020].

- [12] "Open-source tool that uses simple textual descriptions to draw beautiful UML diagrams.," *PlantUML.com*. Available: <https://plantuml.com/>. [Accessed: 12-Dec-2020].
- [13] V. Stojanovic, *A typical Model-View-Controller (MVC) software pattern*.
- [14] "How TO - Collapsibles/Accordion," *How To Create an Accordion*. Available: https://www.w3schools.com/howto/howto_js_accordion.asp. [Accessed: 13-Dec-2020].
- [15] "Binomial Distribution: Formula, What it is, and how to use it in simple steps," *Statistics How To*, 06-Jul-2020. Available: <https://www.statisticshowto.com/probability-and-statistics/binomial-theorem/binomial-distribution-formula/>. [Accessed: 13-Dec-2020].
- [16] "An introduction to binomial distribution," *YouTube*. Available: <http://www.youtube.com/embed/qlzC1-9PwQo>. [Accessed: 13-Dec-2020].
- [17] "An introduction to geometric distribution," *YouTube*. Available: <http://www.youtube.com/embed/zq9Oz82iHf0>. [Accessed: 13-Dec-2020].
- [18] "The hypergeometric distribution: An introduction," *YouTube*. Available: <https://youtube.com/embed/BCeFgnh6A1U>. [Accessed: 13-Dec-2020].
- [19] "Introduction to Negative Binomial Distribution," *YouTube*. Available: <https://www.youtube.com/embed/BPlmjp2ymxw>. [Accessed: 13-Dec-2020].
- [20] "An introduction to Poisson Distribution," *YouTube*. Available: <https://www.youtube.com/embed/jmqZG6roVqU>. [Accessed: 13-Dec-2020].
- [21] "Milestone1" Available: https://emailwsu-my.sharepoint.com/:w:/g/personal/vuochlang_chang_ws_u.edu/EamBu1CsiPVAmHXp4PSq2mYBGKNvccD9NnRLi51iqbsvA?email=zhichengzhou123%40gmail.com&e=hyAivS. [Accessed: 13-Dec-2020].

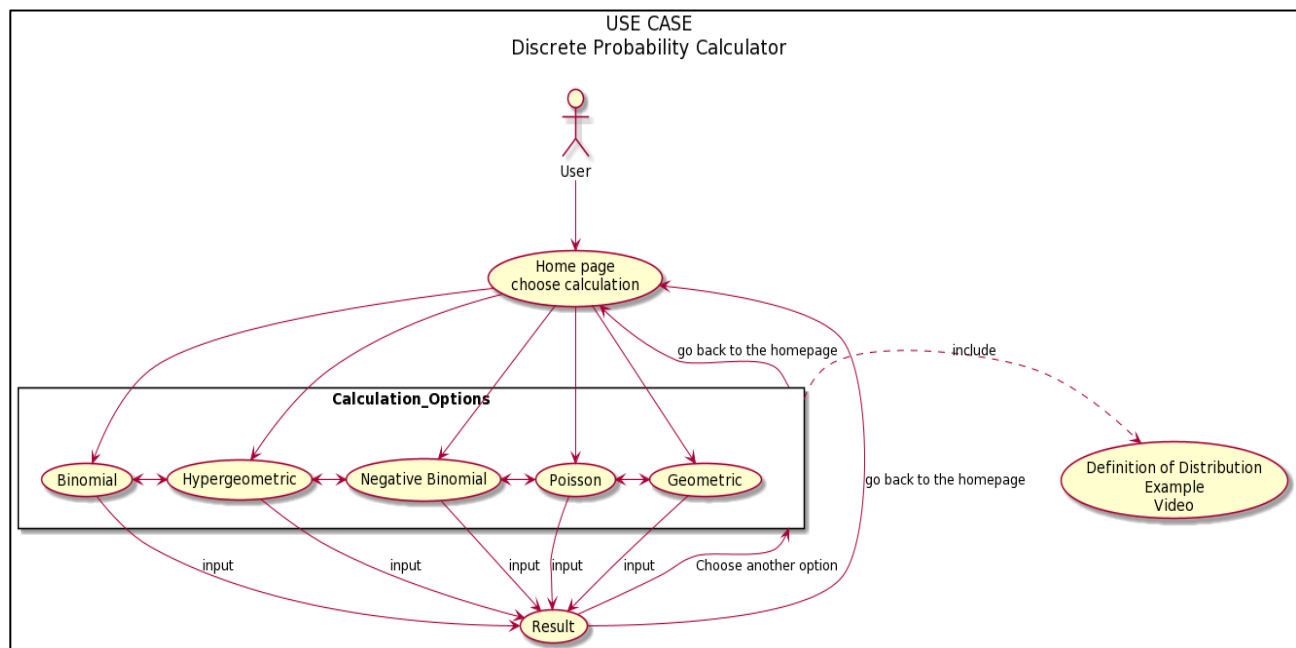
2 Design

2.1 System Modeling

Use-Case:

For our use case diagram, we are taking out the Help-Page section and include the Accordion Menu section that includes the definition of the distribution, an example, and an explanation video for each of the distributions. As shown in the diagram, user will have the option to select one of the distribution calculations directly from the main page (Home page). After the selection,

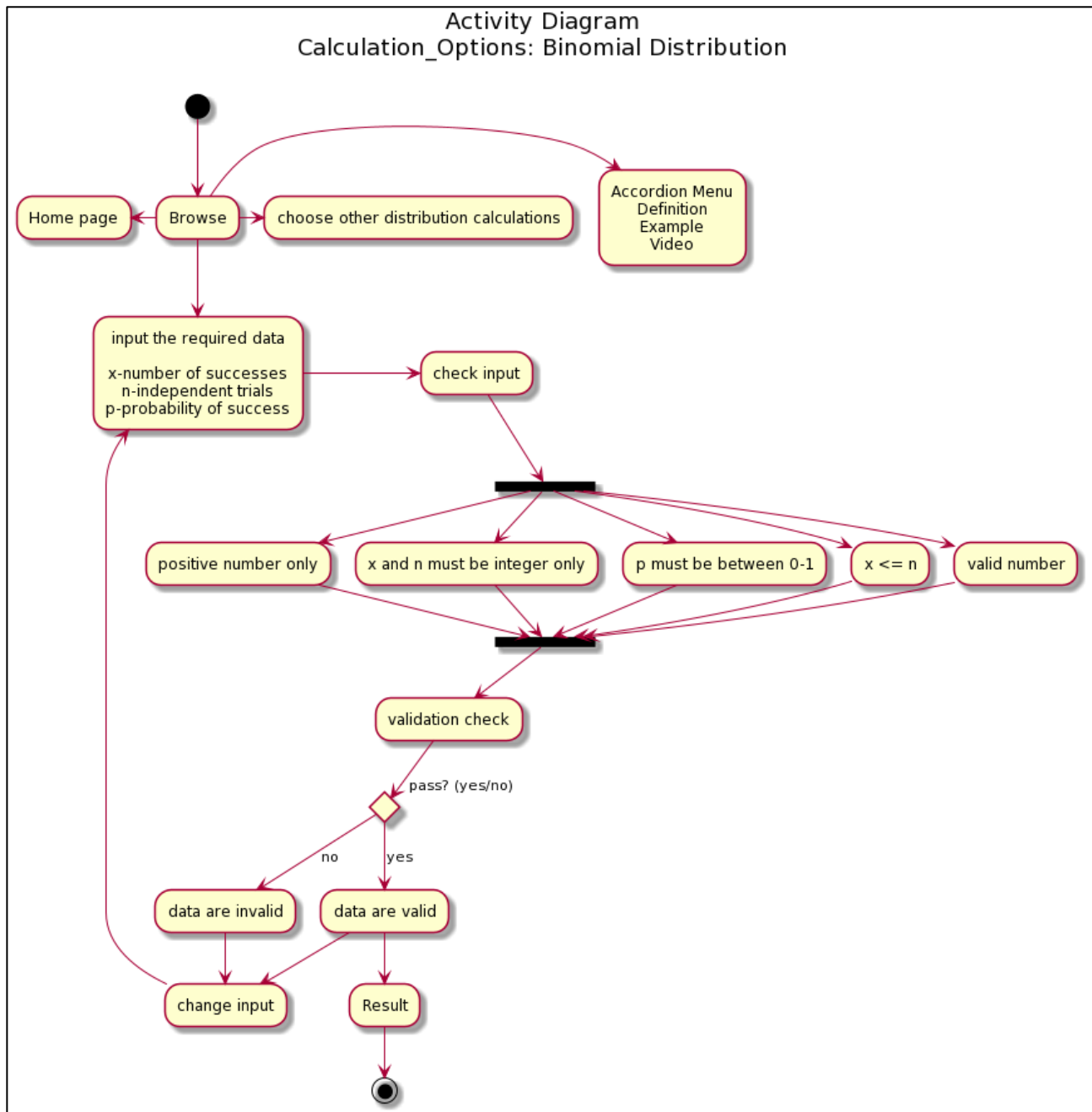
- User can navigate through the subpage (distribution page) and select the followed options:
 - Read the Definition, look at the example and watch a video of that distribution
 - Calculate the probability of the distribution by input the required data
 - After getting the result, user can go back and do more calculations on the same page
- User can go back to the main page
- User can also go to another distribution in that subpage



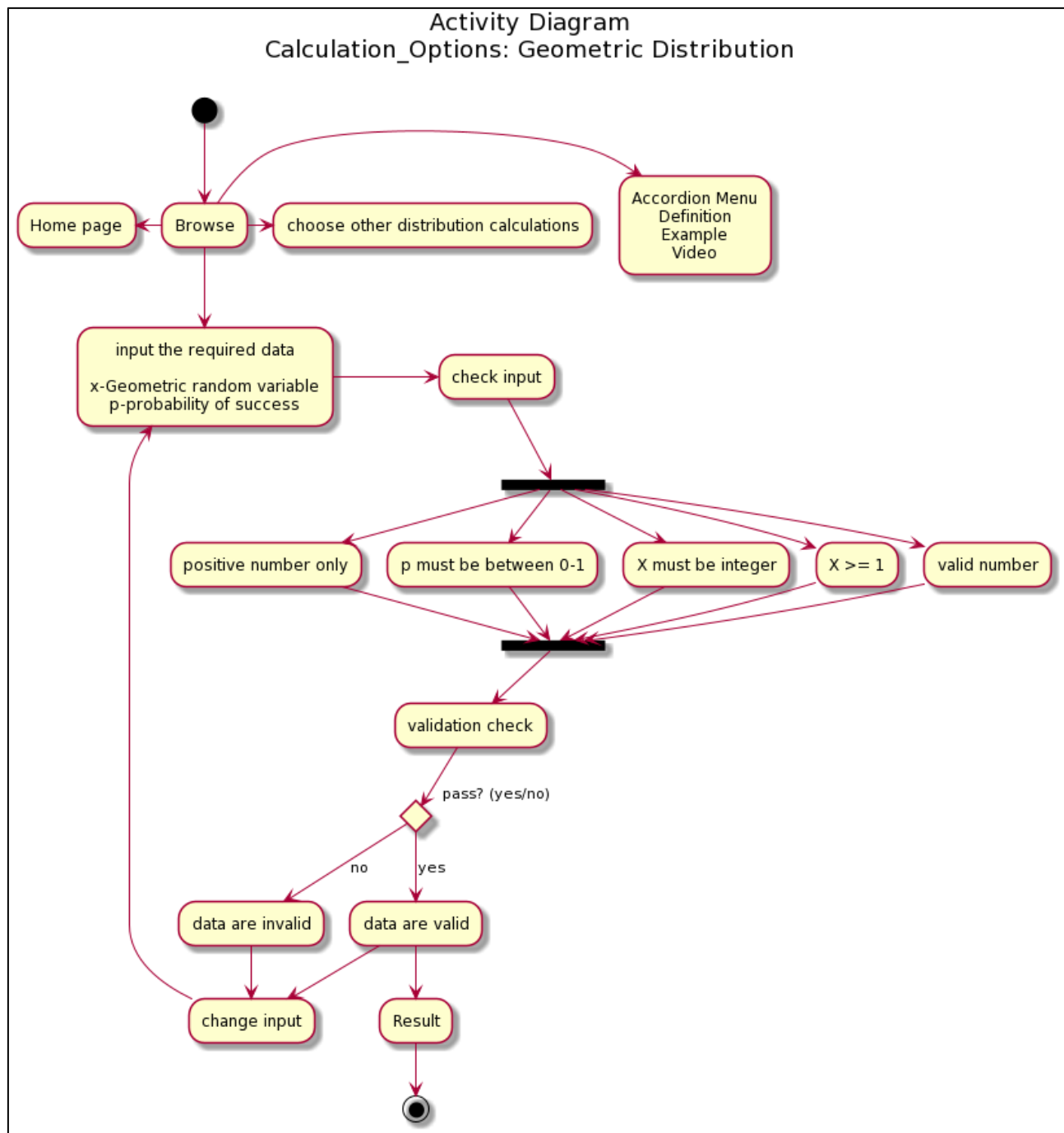
(Figure 1- Use Case Diagram)

Activity-Diagram:

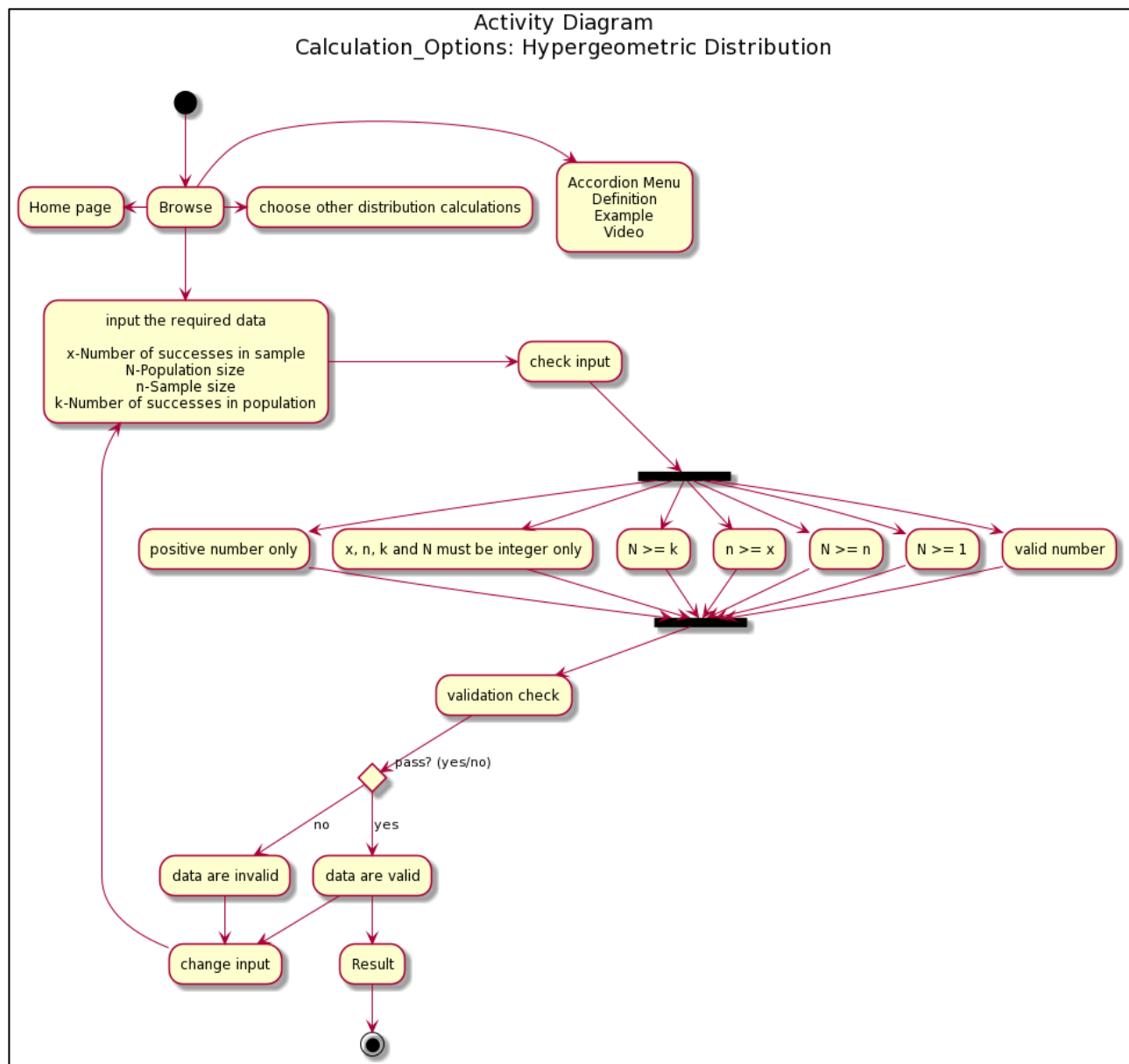
As for the following activity diagrams, we have changed only one thing for all 5 of the diagrams. Instead of using a Help-Page, we are using the accordion menu to include the Definition section, Example section, and the Video section for each distribution.



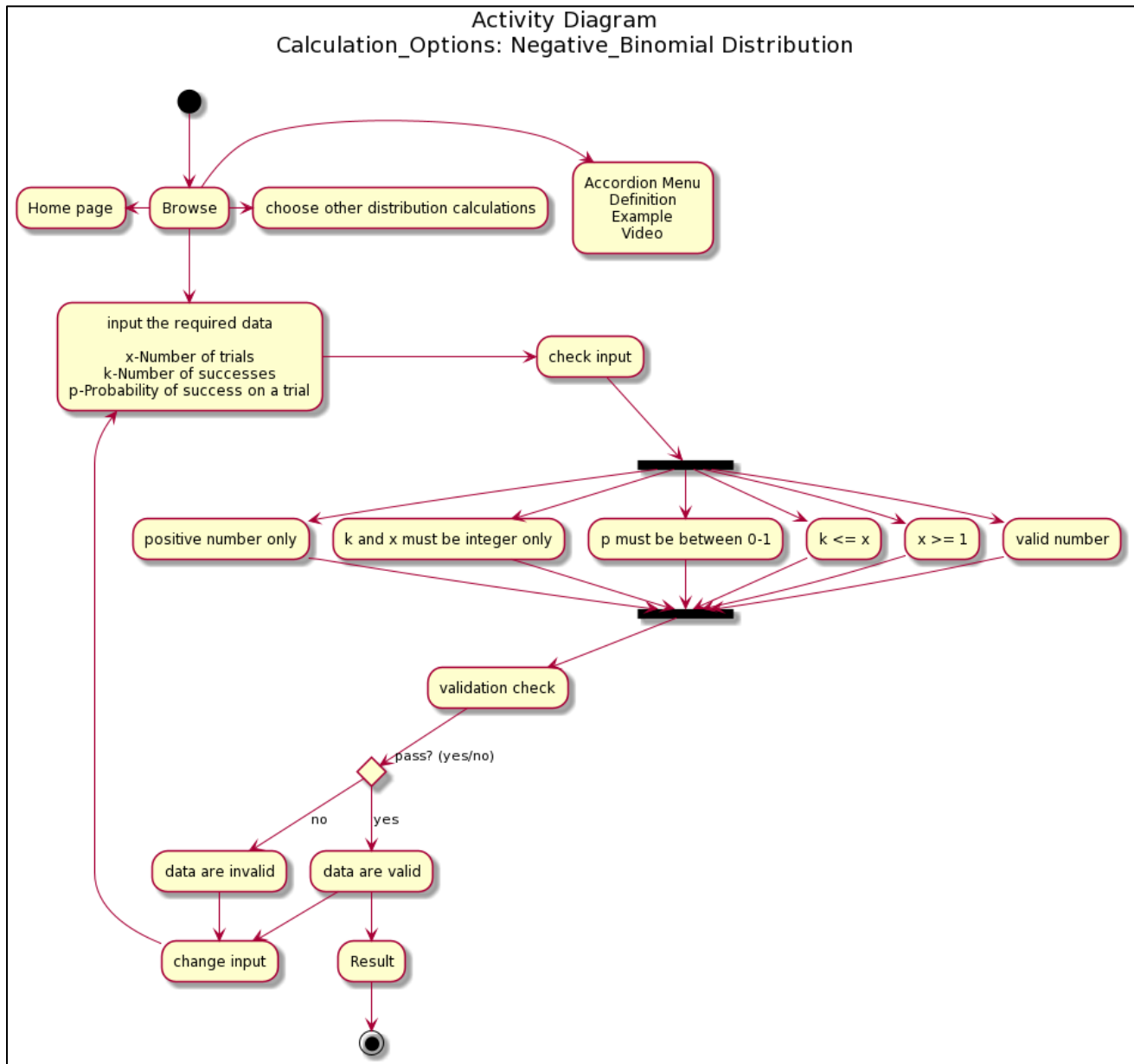
(Figure 2 - Binomial Activity Diagram)



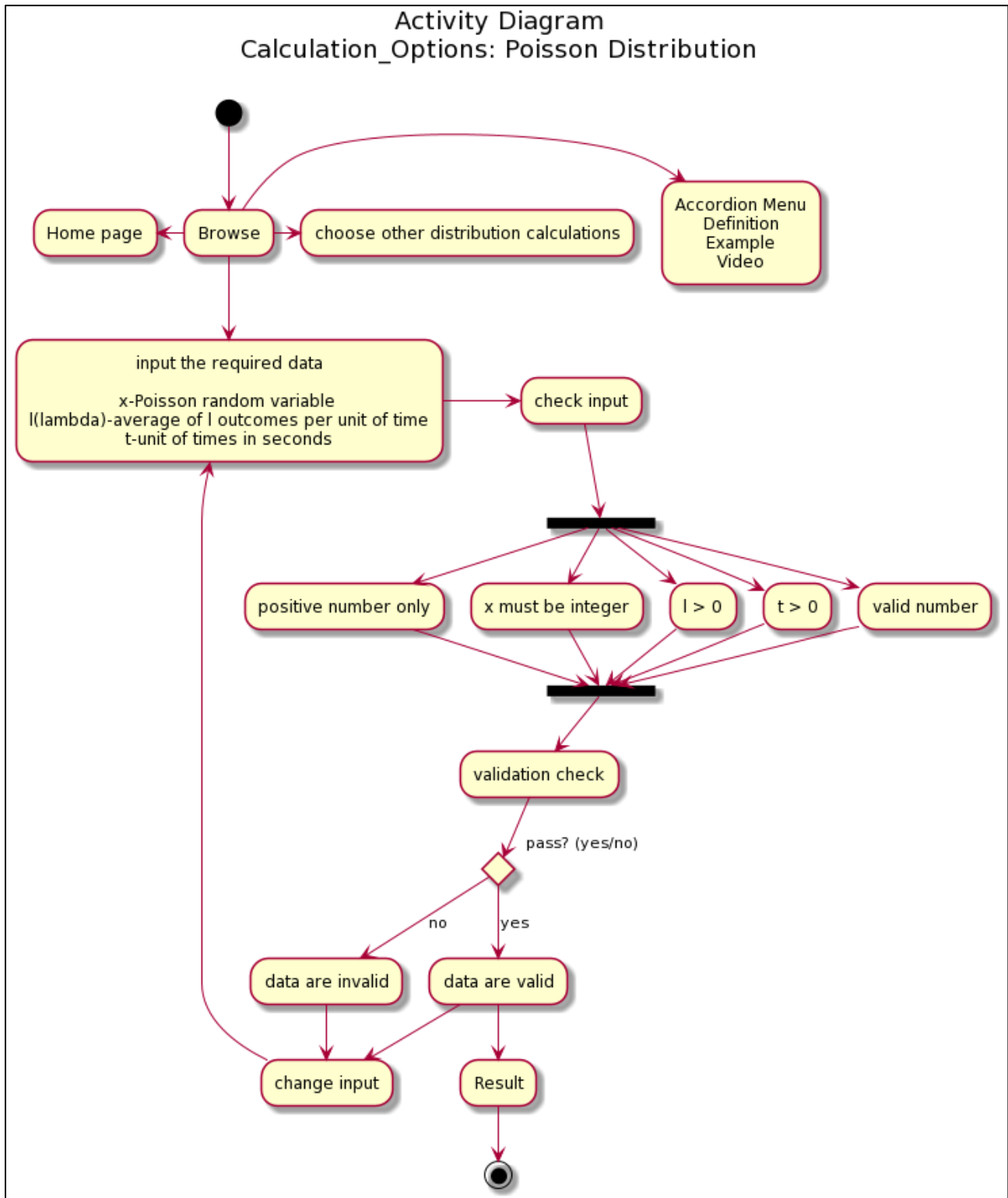
(Figure 3 – Geometric Activity Diagram)



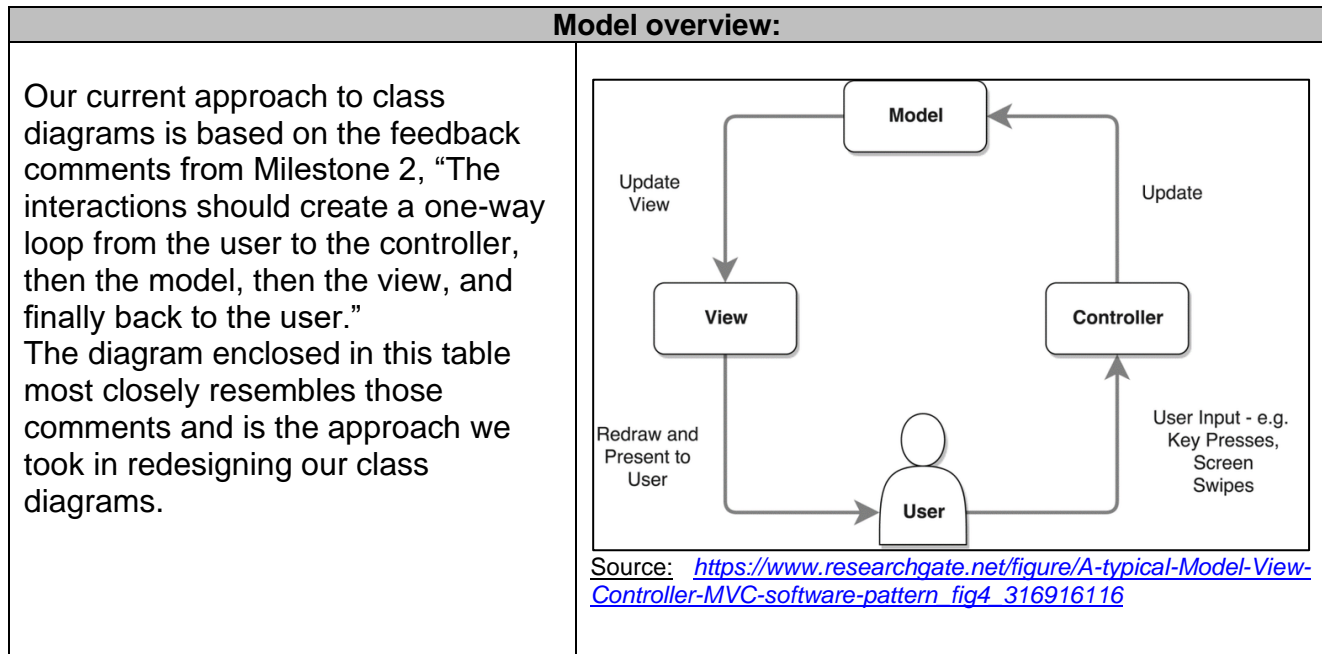
(Figure 4 - Hypergeometric Activity Diagram)



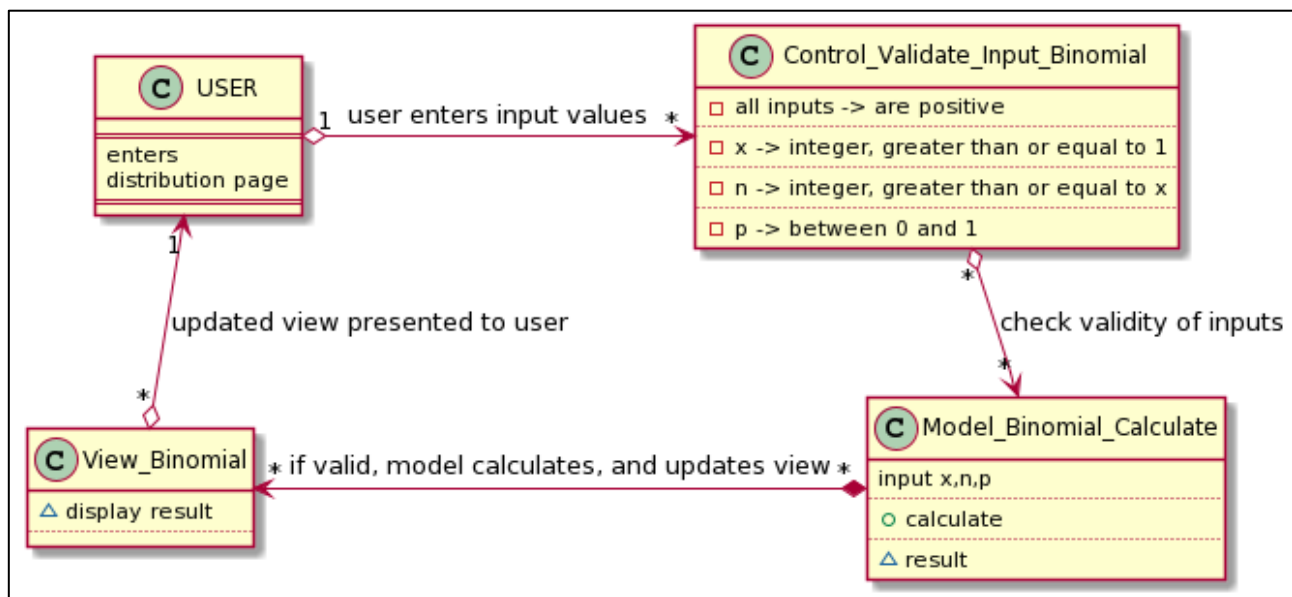
(Figure 5 – Negative-Binomial Activity Diagram)



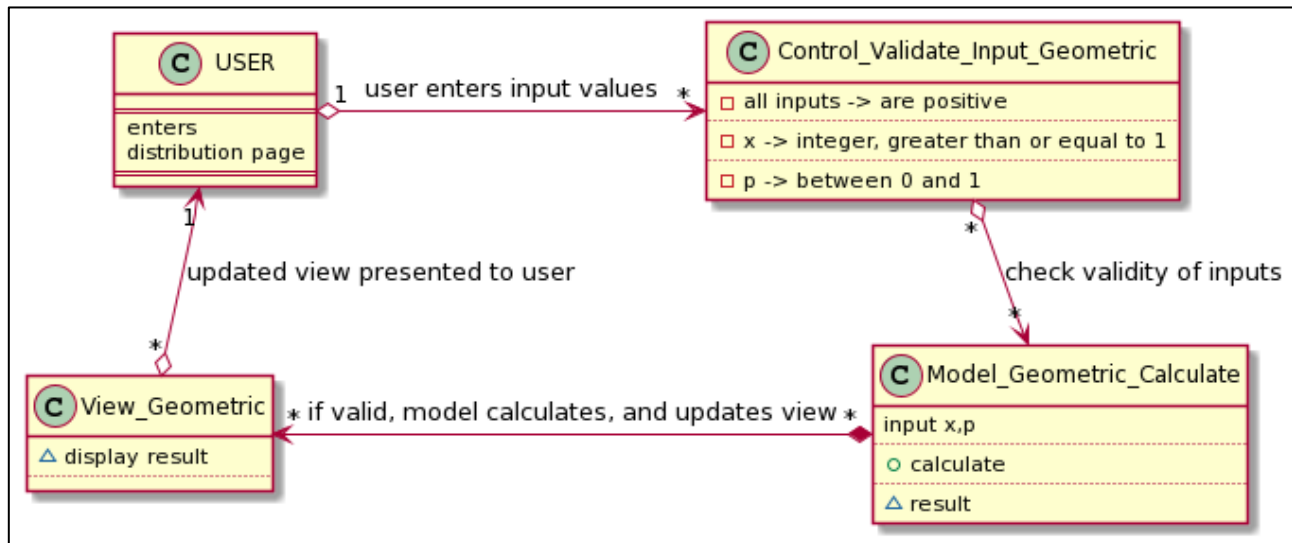
(Figure 6 - Poisson Activity Diagram)

Class Diagrams:

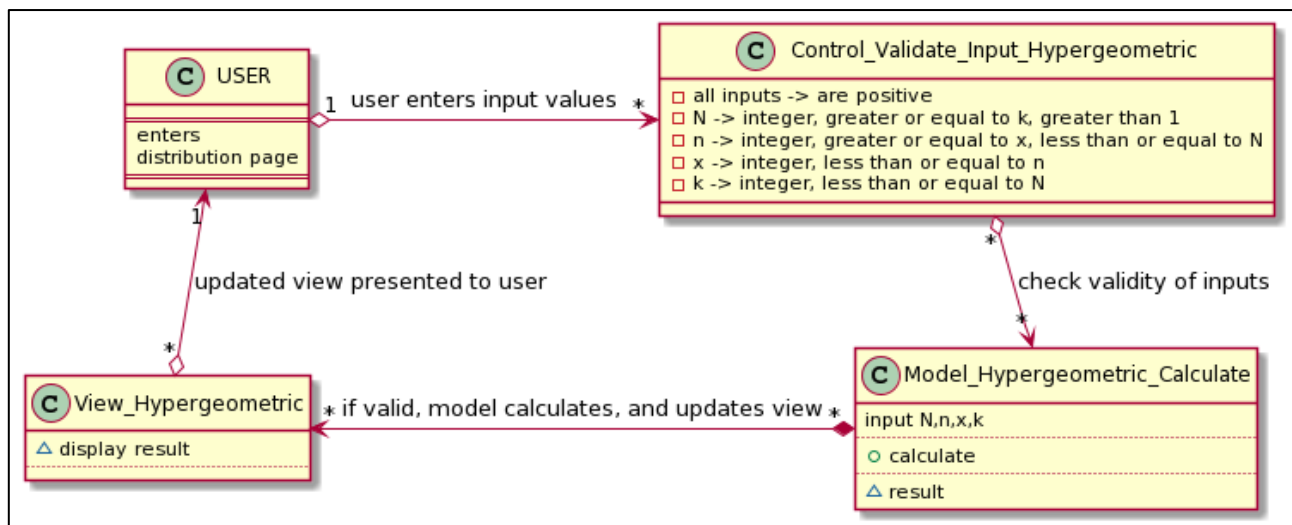
The multiplicity shown in each diagram is the same across all discrete probabilities. It follows a similar pattern from the customer/order example given in class. Just as one customer can make any number of orders, and any order can only be associated with one customer. On our website, any 1 user can make any number of discrete probability calculations, and any 1 calculation is associated with 1 user.

Binomial:

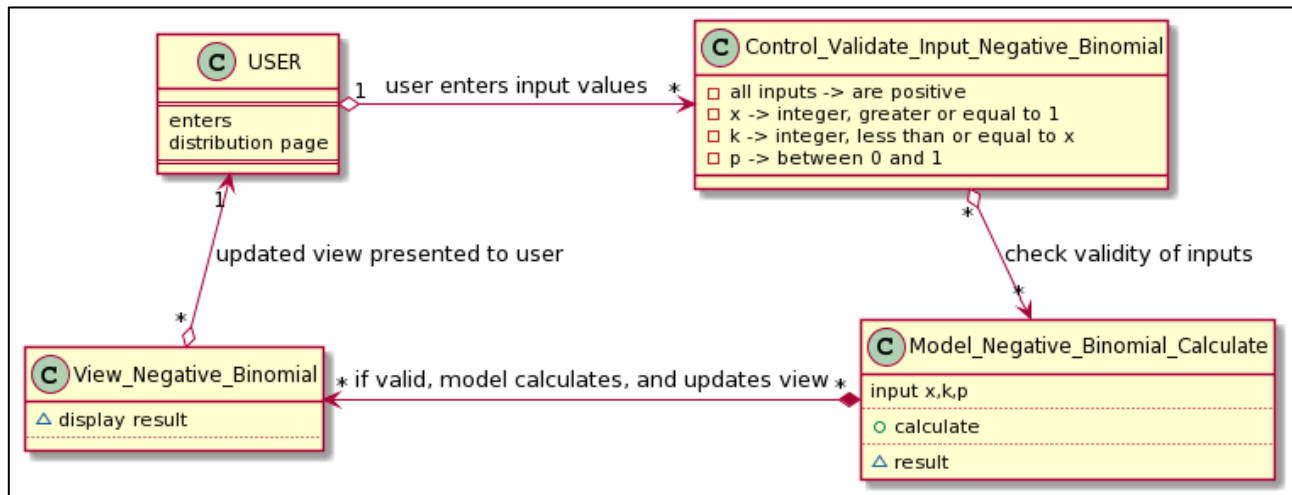
(Figure 7 - Binomial Class Diagram)

Geometric:

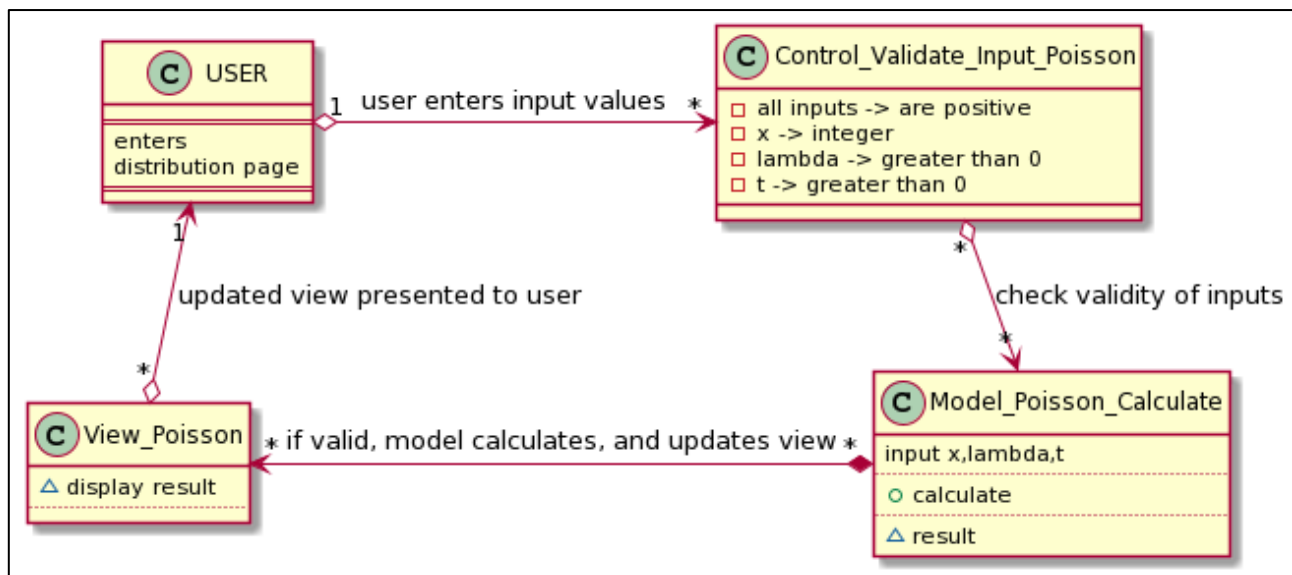
(Figure 8 – Geometric Class Diagram)

Hypergeometric:

(Figure 9 - Hypergeometric Class Diagram)

Negative Binomial:

(Figure 10 – Negative Binomial Class Diagram)

Poisson:

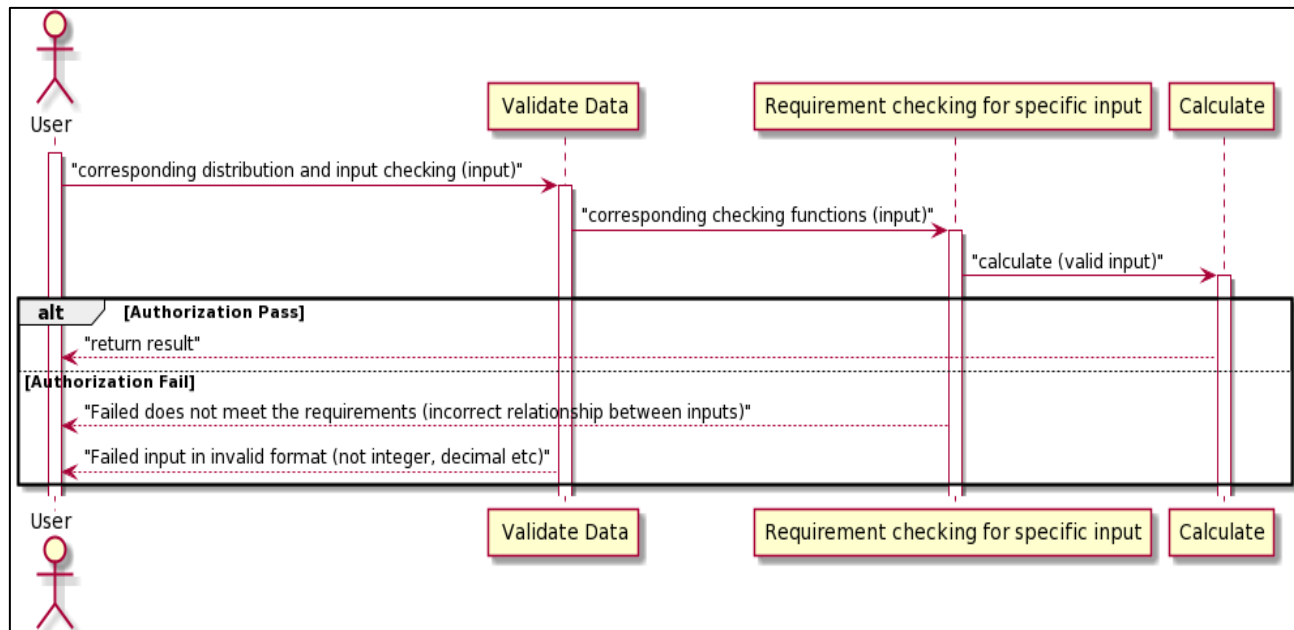
(Figure 11 – Poisson Class Diagram)

Sequence Diagram:**Description:**

First the program will ask the user for some inputs corresponding the distribution they want to calculate.

- The program then read in each input and check if they are numbers only
- if not, it will return false and prevent the user from clicking on the "result" button

- if they are numbers, the program then checks if the numbers are within the correct range and if their relationship is correct, note that for discrete distribution to work, the inputs must be within the range and the relationship must be correct, that is why we have different requirements for different distributions.
- If the numbers follow the criteria, the program will calculate it and return the result.



(Figure 12 – Sequence Diagram)

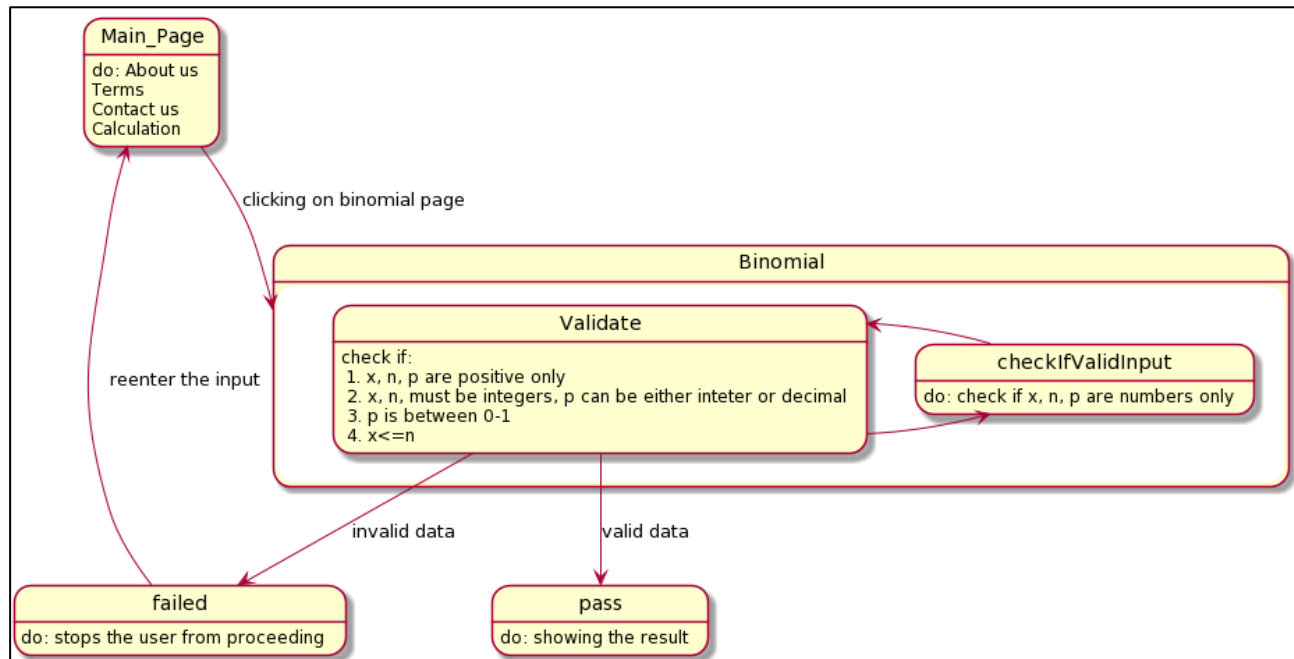
State Diagram for Binomial Distribution:

This is a state diagram for Binomial distribution, our project calculates under 5 different discrete distribution, so **there will be a state diagram for each distribution.**

Description: This state diagram is describing binomial distribution, after the user entered the input, it's in the needing checking state to validate if all the data are correct, we then check if they are:

1. numbers only
2. x, n and p are positive only
3. x, n are integers
4. p can be either integer or decimal
5. p is between 0-1
6. $x \leq n$

We calculate the result only when all the input data is valid, otherwise it will ask the user to reenter the data if anything is incorrect.



(Figure 13 – Binomial State Diagram)

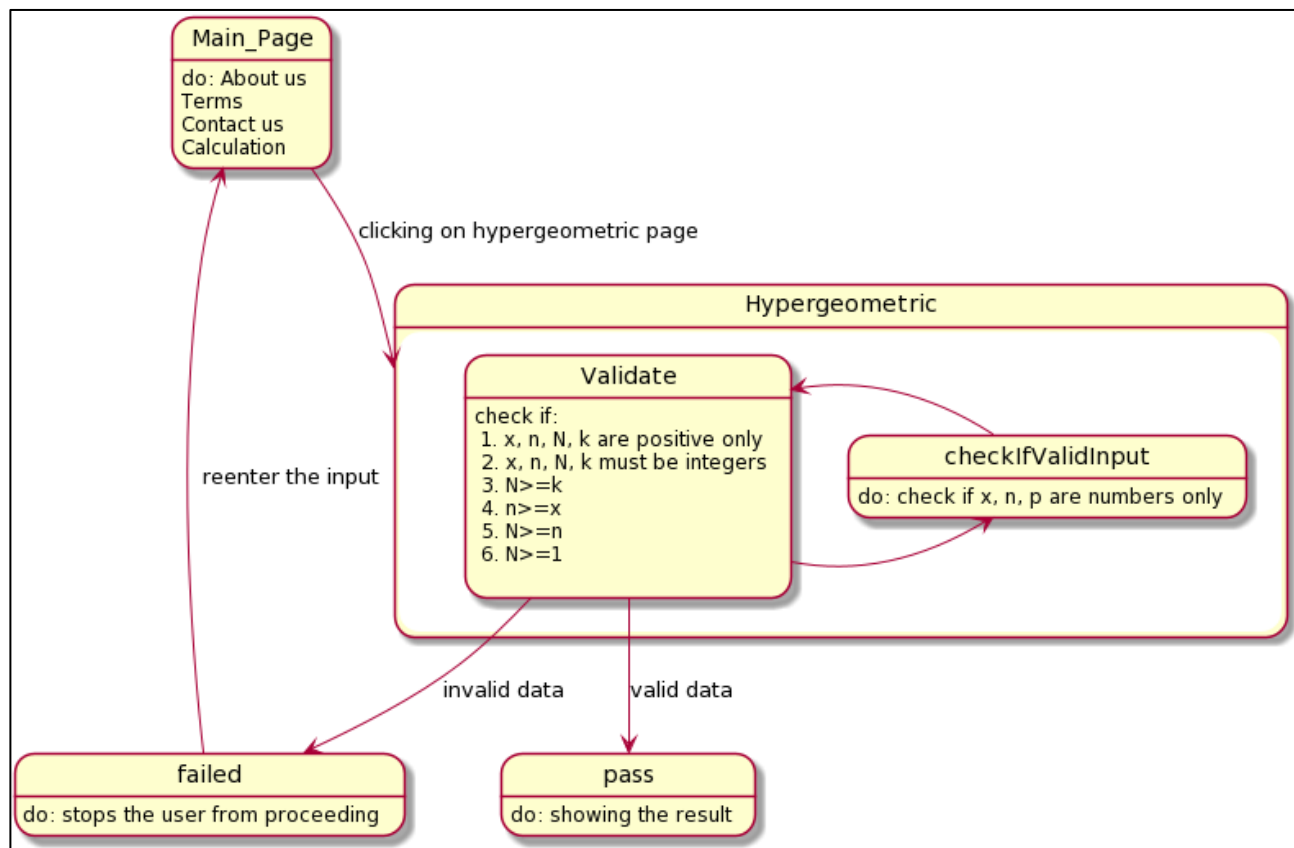
State Diagram for Hypergeometric Distribution:

This is a state diagram for Hypergeometric Distribution.

Description: This state diagram is describing hypergeometric distribution, after the user entered the input, it's in the needing checking state to validate if all the data are correct, we then check if they are:

1. numbers only
2. x, n, N and k are positive only
3. x, n, N and k are integers
4. $N \geq k$
5. $n \geq x$
6. $N \geq n$
7. $N \geq 1$

We calculate the result only when all the input data is valid, otherwise it will ask the user to reenter the data if anything is incorrect.



(Figure 14 – Hypergeometric State Diagram)

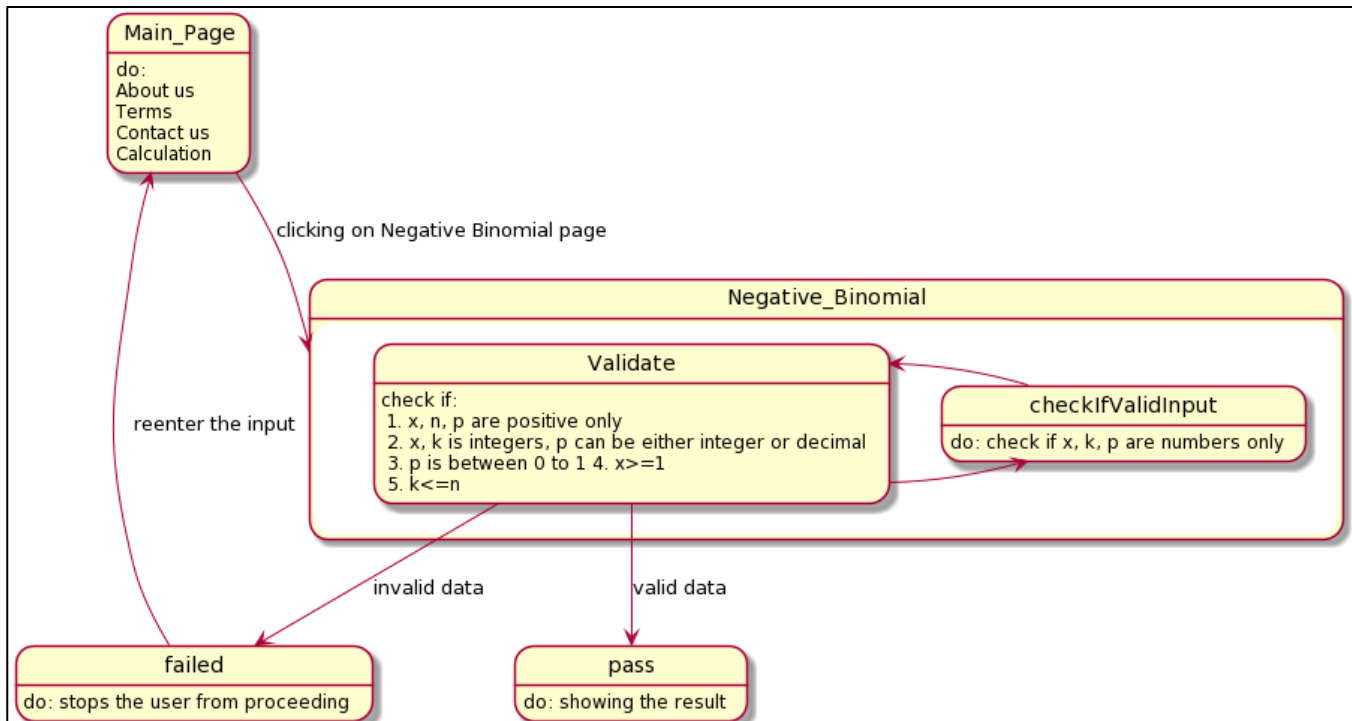
State Diagram for Negative Binomial Distribution:

This is a state diagram for Negative Binomial Distribution.

Description: This state diagram is describing negative binomial distribution, after the user entered the input, it's in the needing checking state to validate if all the data are correct, we then check if they are:

1. numbers only
2. x, k and p are positive only
3. x, k are integers, p can be either integer or decimal
4. p is between 0 to 1
5. $k \leq n$
6. $x \geq 1$

We calculate the result only when all the input data is valid, otherwise it will ask the user to reenter the data if anything is incorrect.



(Figure 15 – Negative Binomial State Diagram)

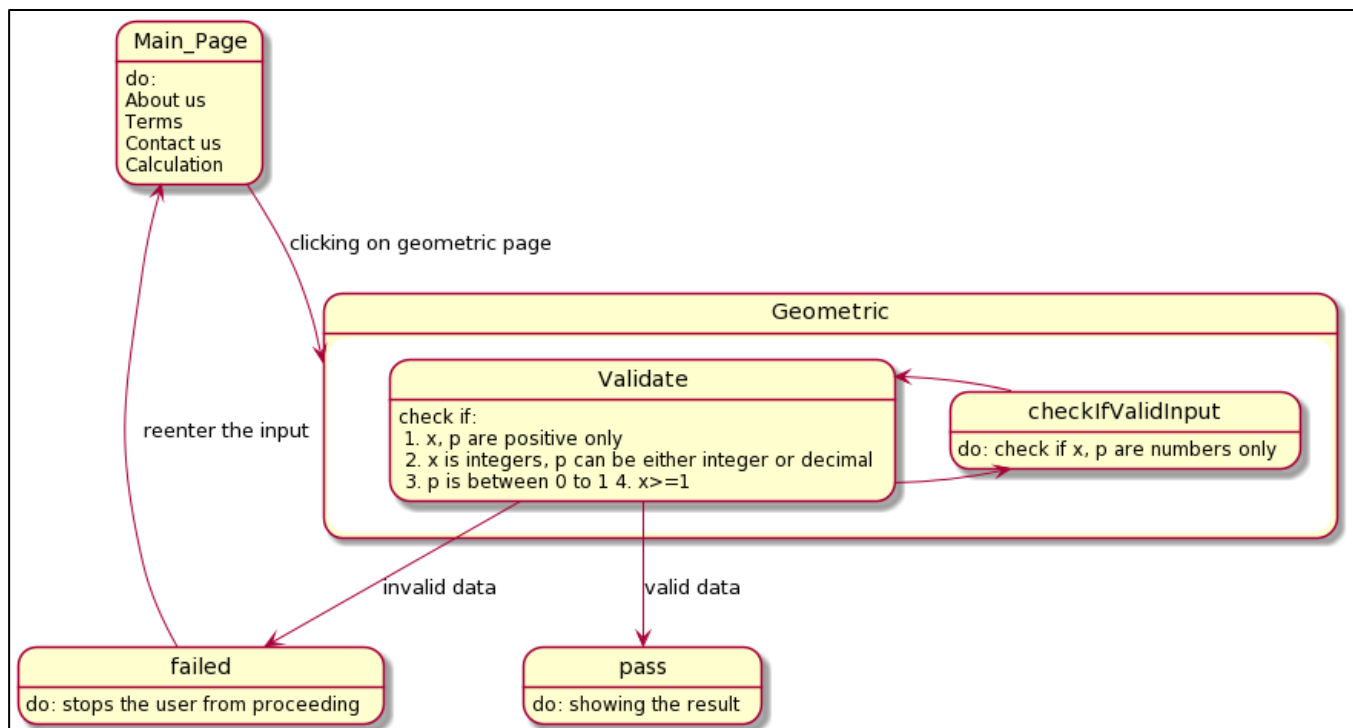
State Diagram for Geometric Distribution:

This is a state diagram for Geometric Distribution.

Description: This state diagram is describing geometric distribution, after the user entered the input, it's in the needing checking state to validate if all the data are correct, we then check if they are

1. numbers only
2. x and p are positive only
3. x is integer
4. p can be either integer or decimal
5. p is between 0-1
6. x>=1

We calculate the result only when all the input data is valid, otherwise it will ask the user to reenter the data if anything is incorrect.



(Figure 16 – Geometric State Diagram)

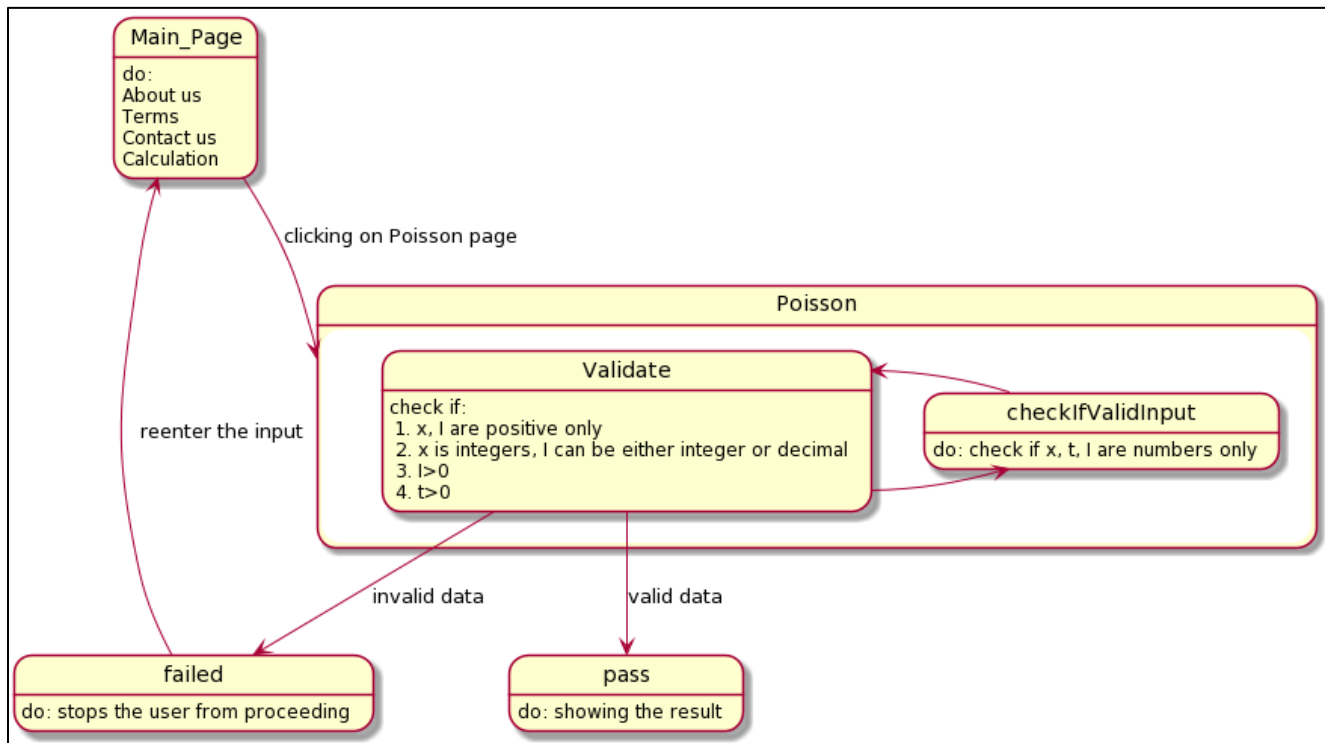
State Diagram for Poisson Distribution:

This is a state diagram for Poisson Distribution.

Description: This state diagram is describing poisson distribution, after the user entered the input, it's in the needing checking state to validate if all the data are correct, we then check if they are:

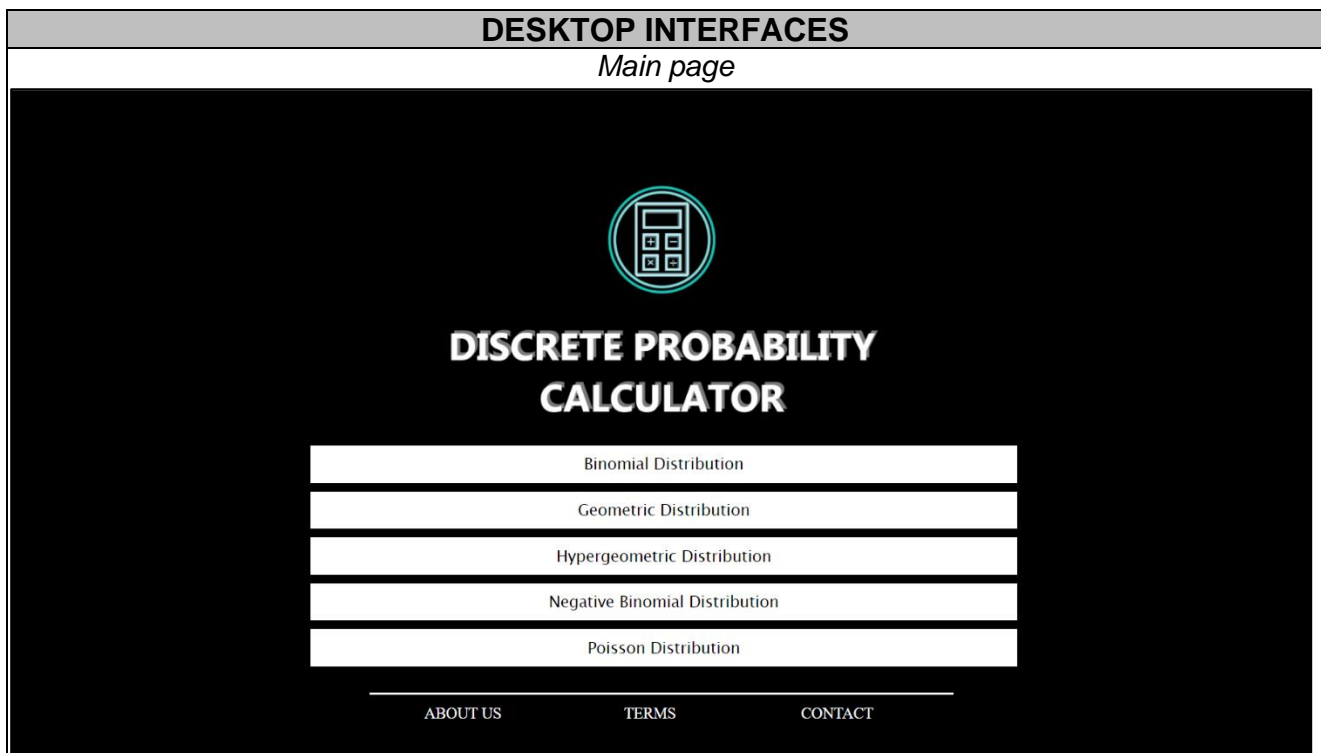
1. numbers only
2. x and l are positive only
3. x is integer, "l" can be either integer or decimal
4. l>0
5. t>0

We calculate the result only when all the input data is valid, otherwise it will ask the user to reenter the data if anything is incorrect.




(Figure 17 – Poisson State Diagram)

2.2 Interface Design



Distribution page



**DISCRETE
PROBABILITY
CALCULATOR**

Binomial Distribution

Geometric Distribution


Hypergeometric Distribution


Negative Binomial Distribution


Poisson Distribution

ABOUT US TERMS CONTACT

Binomial Distribution

Explanation 

Example Problem 

Video 


x: ✓

n: ✓

p: ✓

Result:

About Us page



Discrete Probability Calculator v1.0

[WHO WE ARE:](#)

Team members:

Vuochlang Chang, Patrick Tsai, Zhicheng Zhou

Group information:


Fall 2020, Junior students at Washington state university of Vancouver

Project Information:

The **Discrete Probability Calculator (DPC)** is a group project designed for a class [CS 320 - Fundamentals of Software Engineering](#). It is based on 4 requirements as followed:

- It is a web application
- It has a landing page that helps a user to understand what the application is about
- Besides the landing page, it has multiple other pages
- It has some data to maintain/manipulate

Terms page



**DISCRETE
PROBABILITY
CALCULATOR**

Binomial Distribution

Geometric Distribution


Hypergeometric Distribution

Negative Binomial Distribution

Poisson Distribution

ABOUT US TERMS CONTACT

Terms



**Attribution-NonCommercial-
NoDerivatives 4.0 International
(CC BY-NC-ND 4.0)**

This is a human-readable summary of (and not a substitute for) the [license](#). [Disclaimer](#).

You are free to:

Share — copy and redistribute the material in any medium or format

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

Attribution — You must give [appropriate credit](#), provide a link to the license, and [indicate if changes were made](#). You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

NonCommercial — You may not use the material for [commercial purposes](#).

NoDerivatives — If you [remix, transform, or build upon the material](#), you may not distribute the modified material.


No additional restrictions —
You may not apply legal terms or [technological measures](#) that legally restrict others from doing anything the license permits.

MOBILE INTERFACE

Mobile Home Page

5:41 44%

dpc.patricksai.com/html/mai



**DISCRETE PROBABILITY
CALCULATOR**

Binomial Distribution

Geometric Distribution

Hypergeometric Distribution

Negative Binomial Distribution

Poisson Distribution

ABOUT US TERMS CONTACT

Mobile Distribution Page

5:41 44%

dpc.patricksai.com/html/hyp

DPCv1.0

Hypergeometric Distribution

Explanation +

Example Problem +

Video +

x:

n:

N:

K:

Result:

Reset Result

ABOUT US TERMS CONTACT

Mobile Menu

5:41 44%

dpc.patricksai.com/html/hyp

DPCv1.0

Binomial Distribution

Geometric Distribution

Hypergeometric Distribution

Negative Binomial Distribution

Poisson Distribution

Hypergeometric Distribution

Explanation +

Example Problem +

Video +

3 Implementation

3.1 Development Environment

Our team used many of the development tools used in CS320. Of note, we used IntelliJ, GitHub, JavaScript, Mocha/Chai, HTML and CSS. We wanted to use environments that are common to all our members. Using these tools worked well because we were familiar with them and able to help each other when we ran into an issue. Below is a list of the tools we used and a description of how we used them.

<i>Programming languages, IDEs, Tools</i>	<i>Description</i>
IntelliJ	We used a common IDE – IntelliJ Use of IntelliJ was encouraged for cs320, and because of that it was familiar to all team members.
GitHub	We decided to use GitHub as a distributed version control. It was helpful that we used GitHub for several cs320 class assignments. GitHub helped us keep current code bases to work on individually.
JavaScript	We chose JavaScript for the ease of use and integration with HTML. Our use of JavaScript in cs320 also made it a good choice due to our modest amount of experience using it.
JQuery	We used a JQuery function to pull code from different files onto rendered pages. Specifically, we pulled the menu and footer into rendered pages. By doing so, we modularized our code, only have one file for desktop menu, one file for mobile menu, and one file for the footer.
HTML	We used HTML to render our program as a website.
CSS	We used CSS to customize the styling of our site. Additionally, we created a responsive style framework to handle mobile devices as well as desktop.
Mocha/Chai	Framework for testing JavaScript. We tested code accuracy and code performance.

Server Host	We are using a shared GoDaddy webserver to host our code base.
mathcha.io	Editor for explanation and example problems in accordion menu
Adobe InDesign	Logo images
Creative Commons	Created license for our website We are using the “Attribution – Non-Commercial – No Derivatives 4.0 International (CC BY-NC-ND 4.0) license https://creativecommons.org/licenses/by-nc-nd/4.0/
Google – Analyze and optimize your website with PageSpeed tools	To test the page-speed of each pages of DPC https://developers.google.com/speed

3.2 Task Distribution

We created a table to account for all the tasks assigned and completed for our project. Task were assigned at our weekly Zoom meetings. Zoom meetings occurred every Wednesday at 2 pm. Status updates were recorded on our Discord server, and at our weekly Zoom meetings. We tried to divide work up evenly.

We found it easier to divide written/documentation tasks. However, some coding situations made more sense to assign work to specific team members. The table below outlines how our work was divided.

Project Goal	Section	Description	Team Member
Milestone 0	Sec. 1 – Team members	-	Anna, Z, Patrick
	Sec. 2 – Meeting times	-	Anna, Z, Patrick
	Sec. 3 – Contact methods	-	Anna, Z, Patrick
	Sec. 4 – Concept	-	Anna
Milestone 1	Sec 1. - Intro	1.1 – 1.6	Anna
	Sec. 2 – Overall Desc	2.1 – 2.7	Patrick
	Sec. 3 – Specific Req.	3.1 – 3.3	Z
	Sec. 4 – Other Non-func.	4.1 – 4.3	Patrick, Z
	Appendix A	-	Anna
	Appendix B	-	Anna, Patrick
	Review	Zoom meeting to review all materials. Final draft uploaded by project lead	Anna, Z, Patrick

Milestone 2	Sec. 1 - Intro	1.1 – 1.3	Anna, Z
	Sec. 2 – Activity Diagrams	2.1 – 2.5	Anna, Z
	Sec. 3 – Class Overview	3.1 – 3.6	Patrick
	Sec. 4 - Behavioral	4.1 – 4.6	Z
	Appendix A		Anna, Patrick
	Review	Zoom meeting to review all materials. Final draft uploaded by project lead	Anna, Z, Patrick
Web design	Overall design	-	Anna
	Initial HTML CSS	-	Anna
	Reformat for Mobile	Accordion, CSS, HTML	Patrick
	About us	-	Anna
	Terms, Contact	-	Patrick
	Webserver config and FTP	Setting up a sub domain and FTP	Patrick
Code Integration	Binomial Dist.	.js	Anna
	Geometric Dist.	.js	Z
	Hypergeometric Dist.	.js	Z
	Neg. Binomial Dist.	.js	Anna
	Poisson	.js	Patrick
	Validation Check	.js	Anna
	Dist. descriptions	.txt	Patrick
	Dist. Example probs	.txt	Z
	Dist. Videos	.txt	Z, Patrick
Milestone 3	Sec. 1	Introduction and overview, updated references	Z
	Sec. 2	Use case and state diagram	Anna
	Sec. 2	Mobile images, Class diagrams	Patrick
	Sec. 2	Sequence diagrams, state diagrams	Z
	Sec. 3	3.1 – 3.3	Patrick
	Sec. 4	4.1, 4.3	Anna
	Sec. 4	4.2 testing	Anna, Z, Patrick
	Sec. 4	4.4	Z
	Sec. 5	-	Anna, Z, Patrick
	Sec. 6	-	Anna, Z, Patrick
SRS Revision	Review	Zoom meeting to review all materials. Final draft uploaded by project lead	Anna, Z, Patrick
	Revise and Review	Zoom meeting to revise and review all materials. Final SRS v2 will be uploaded by project lead	Anna, Z, Patrick

3.3 Challenges

Patrick's Challenges

- 1.) HTML/CSS – Our team built and customized the HTML/CSS ourselves. This was more difficult than using premade HTML/CSS templates. I think we all enjoyed the experience of creating the website, but it was a large time sink. A lot of time was spent trying to get the look and formatting correct. Additionally, we implement our own responsive theming for mobile devices. Creating the mobile theming was just as time intensive as the desktop version and I believe more time was spent on theming that could have been better spent on programming algorithms.
- 2.) Testing was difficult for me. I had some trouble using Mocha Chai tests when my function used getElementbyID tags. They would return a null 'style' error. I had to make a separate file and comment out all the getElementID calls in order to test my poisson code. My lack of experience using Mocha/Chai is the main reason for this issue.
- 3.) I really like git, however I'm not an expert at using git. There were a few times I was confused about which version was newest, how exactly to resolve code conflicts and the best practice for git workflow. In the end, my problems seemed to be little more than growing pains, and I am glad I wrestled with a few git issues. I feel better prepared to use git moving forward.

Z's Challenges

1. I'm a novice at HTML and CSS, this is my first time working with them, but my teammates made lots of good template and examples which was helpful to get the hang of it.
2. I've had some trouble with hypergeometric.js (the other geometric.js was easier compared to it) due to its complicated relationship between all the inputs. Four inputs are required for hypergeometric.js and I must check all the previous ones once the new one is entered, so it took me a while to figure out how to manage them.
3. Using GitHub was helpful, but I also have lots of difficulties with it. At the beginning I didn't use git desktop which made all the committing and pushing very hard. It took me a fair amount of time to get used to Version Control System, but I think I've mastered the basic use of git.
4. Testing the source code using Mocha and Chai was another challenge for me, I got lots of "Type error, null value" errors because my original code relied on mostly getldbyelement function to get the input from the user on HTML page, that made it exceptionally hard to write a test case for it, so I had to go back and reformat all the code.

Anna's Challenges

- GitHub App: Before CS320, I never used GitHub before. I used GitLab for my previous CSE classes through its website but using GitHub App was another new thing to learn about. I made mistakes and sometimes messed up with the group code, but we were able to revert the commit and get the original code back. I am still learning new things about GitHub and its functions. I have learned how to use the App but not to the level that I have understood everything completely.
- HTML, CSS, JS: Connecting everything together was challenging for me. Especially how to connect components (each image, paragraph, input box, etc) and use the position function to be hardwired everything together. I have also discovered new functions that I have never learned before such as calling HTML inside an HTML file, using one CSS style for multiple pages, and using js to interact with the user by getting input and display the output and error messages.

4 Testing

4.1 Testing Plan

For the following two sections on testing for Functional requirements and Non-Functional requirements, we are going to follow the listed plans below:

- **Testing for Functional requirements:** We are going to focus on user requirements which are input that will be needed from the user to compute the probability. By focusing on those input, we are going to use Mocha and Chai testing to test if functions return value as expected to decide what to display back to users.
 - **Mocha and Chai testing** will be focusing on the correctness of functions and will be implement by defined classes such as:
 - **ValidateInput** – focus on deciding whether a given input is a valid number
 - **Binomial, NegativeBinomial, Hypergeometric, Geometric and Poisson classes** will focus on deciding whether all input have met the requirements and whether to show error messages or return the result to the user
 - **Requirements** for each defined class are:

Binomial:	<ul style="list-style-type: none"> ○ x and n are integer number ○ p is between 0 to 1 ○ x must be at least 1 ○ $x \leq n$
Geometric	<ul style="list-style-type: none"> ○ x is integer number ○ p is between 0 to 1 ○ x must be at least 1
NegativeBinomial	<ul style="list-style-type: none"> ○ x and k are integer numbers ○ p is between 0 to 1 ○ x must be at least 1 ○ $k \leq x$
Hypergeometric	<ul style="list-style-type: none"> ○ x, n, k and N are integer numbers ○ $N \geq k$ ○ $n \geq x$ ○ $N \geq n$ ○ $N \geq 1$
Poisson	<ul style="list-style-type: none"> ○ x is integer ○ lambda greater than 0

- **Testing for Non-Functional requirements:**
 - We are going to test on speed of functions listed in each class using Mocha and Chai
 - We are going to test on page-speed of each pages (html files) using [Analyze and optimize your website with PageSpeed tools from Google](#)

4.2 Tests for Functional Requirements

The focused objective of Functional testing for DPC is to check the following functionalities of the system:

- **Main functions:** Test the result of the main function for each distribution which is to test the result from the calculation function
- **Error conditions:** Test to see if functions return appropriate value to determine whether it has detected any errors from the user input to determine which error message to display to the user
- **Basic Usability:** This tests the basic usability of the system. We have tested the usability of the system and we can confirm that users that navigate through the website easily and the user can also access the attached video without any glitch.
- **Accessibility:** This tests the accessibility of the system. We have tested and confirmed that our website provides a simple layout that users can navigate through easily and understand what is going on for any specific page. As shown in *section 2.2*, we build the website following a simple layout with simple color combinations which allow users to focus on what they are looking for instead of the design itself.

To test the Main functions and Error conditions, we are going to use Mocha and Chai test to show that functions are returning values as expected. All tests will be divided by each defined class.

CLASS: ValidateInput**Function: validateInput(value)**

- return true if the given data is a valid float numbers, eg: 0.2, 45
- return false if the given data is not a valid float numbers, eg: e42, 34.e, 25., .qw, .34

Function: checkIfVaid(value)

- return true if the given data can be parse into float number, eg: 45, 0.3, 45e
- return false if the given data cannot be parse into float number, eg: we234, num

Function: checkNumberOnly(value)

- return true if the given data is in a valid number format, eg: 12, 12.3
- return false if the given data is not in a valid number format, eg: 98.a, 234.424hi

Test for Correctness

CLASS: ValidateInput**Function: ValidateInput(value)**

- ✓ checkIfValidInput(4.5) = true 1ms
- ✓ checkIfValidInput(2) = true 0ms
- ✓ checkIfValidInput(45.e) = false 0ms
- ✓ checkIfValidInput(25.) = false 0ms

Function: checkIfValid(value)

- ✓ checkIfValid(45) = true 0ms
- ✓ checkIfValid(0.3e) = true 0ms
- ✓ checkIfValid(we234) = false 0ms
- ✓ checkIfValid(num) = false 0ms

Function: checkNumberOnly(value)

- ✓ checkNumberOnly(12) = true 0ms
- ✓ checkNumberOnly(12.3) = true 0ms
- ✓ checkNumberOnly(98.a) = false 0ms
- ✓ checkNumberOnly(.78) = true 0ms

Function: Combination(n, r)

- ✓ Combination(10, 5) = 252 0ms

Function: Factorial(n)

- ✓ Factorial(14) = 87178291200 0ms

CLASS: Binomial
Function: checkX() <ul style="list-style-type: none"> - return false if x is not a valid integer - else - return true
Function: checkN() <ul style="list-style-type: none"> - if n is not valid, return 2 - if n is valid but $n < 1$, return 1 - else return 3 – this means n is integer and greater or equal to 1
Function: checkP() <ul style="list-style-type: none"> - if p is valid number and $p = [0,1]$, return 3 - if p value is not valid number, return 2 - if p is greater than 1, return 1
Function: checkAll() <ul style="list-style-type: none"> - if inputs are valid, return true - else, return false
Function: binomialFormula() <ul style="list-style-type: none"> - return the probability result based on binomial formula

Test for Correctness

CLASS: Binomial

Function: checkX()

- ✓ when $x=4.5$, checkX() should return false -- DISPLAY MESSAGE: Input must be valid Integer 0ms
- ✓ when $x=7$, checkX() should return true -- DISPLAY MESSAGE: ✓ 0ms

Function: checkN()

- ✓ when $n=3.5$, checkN() should return error 2 - n is decimal -- DISPLAY MESSAGE: Input must be valid Integer 0ms
- ✓ when $n=0$, checkN() should return error 1 - n is zero -- DISPLAY MESSAGE: n must be at least 1 0ms
- ✓ when $n = 6$, checkN() should return 3 - n is valid -- DISPLAY MESSAGE: ✓ 0ms

Function: checkP()

- ✓ when $n=3$, checkP() should return error 1 - $p > 1$ 0ms
- ✓ when $n=0.5$, checkP() should return 3 - p is valid 0ms

Function: checkAll()

- ✓ when $n=5$, $x=8$, $p=0.9$, checkAll() should return false -- DISPLAY MESSAGE: x must be less than or equal n 0ms
- ✓ when $n=5.6$, $x=8$, $p=0.9$, checkAll() should return false -- DISPLAY MESSAGE: Invalid Input 0ms
- ✓ when $n=10$, $x=8$, $p=0.9$, checkAll() should return true 0ms

Function: binomialFormula()

- ✓ when $n=10$, $x=8$, $p=0.9$, binomialFormula() should return 0.19371 0ms

CLASS: Hypergeometric**Function: checkX()**

- If n is a valid integer, return 1
- Else n is not a valid integer, return 2

Function: checkn()

- if $n > x$, return 1
- else $n \leq x$, return 2
- If n is not a valid integer, return 3

Function: checkN()

- if $N \geq 1$ and $N \geq n$, return 1
- if $N \geq 1$ but $N \leq n$, return 2
- if $N \leq 1$ and $N \leq n$, return 3
- If N is not a valid integer, return 4

Function: checkK()

- If $K \leq N$, $K \geq x$, return 1
- If $K \leq N$ but $K \leq x$, return 2
- If $K \leq N$ and $K \leq x$, return 3
- If K is not a valid integer, return 4

Function: hyperFormula()

- Testing based on the given input, let $x = 5$; $n = 15$; $N = 20$; $k = 10$, the returned result should be 0.016253869969040248

Test for Correctness

CLASS: Hypergeometric

Function: checkX()

- ✓ when input is 2.63, checkX should return false since it must be a decimal number 1ms
- ✓ when $x=20$, checkX() should return true

Function: checkn()

- ✓ when n is 1.2, checkn should return 3(the false value for not being an integer) because in hypergeometric, n cannot be decimal
- ✓ when n is 0, checkn should return 2(the false value for $n \leq x$)
- ✓ when n is 21, checkn should return 1(the true value for n is integer and $\geq x$) 1ms

Function: checkN()

- ✓ when N is 1.2, checkN should return 4(the false value for not being an integer)
- ✓ when N is 0, checkN should return 3(the false value for $N \geq n$ but $N \leq 1$)
- ✓ when N is 2, checkN should return 2(the false value for $N \geq 1$ but $N \leq n$)
- ✓ when N is 30, checkN should return 1(the true value)

Function: checkK()

- ✓ when k is 1.2, checkK should return 4(the false value for not being an integer)
- ✓ when k is 31, checkK should return 3(the false value for $k \geq x$ but $k \leq N$) 1ms
- ✓ when k is 19, checkK should return 2(the false value for $k \leq N$ but $k \leq x$) 1ms
- ✓ when k is 21, checkK should return 1 1ms

Function: hyperFormula()

- ✓ when $x=5$, $n=15$, $N=20$, $k=10$, the result should be 0.016253869969040248 1ms

CLASS: Geometric
Function: checkX() <ul style="list-style-type: none">• If n is a valid integer, return 1• Else n is not a valid integer, return 2
Function: checkP() <ul style="list-style-type: none">- if n is not a valid decimal number between 1 to 0, return 1- if n is valid number, return 2- If the number is a valid decimal number between 1 to 0, return 3
Function: geometricformula() <ul style="list-style-type: none">• Testing based on the given input, let $p = 0.5$; $x = 5$, the returned result should be 0.03125

Test for Correctness

CLASS: Geometric

Function: checkX()

- ✓ when input is 2.63, checkX should return false since it must be a decimal number 4ms
- ✓ when $x=20$, checkIfValidInput should return true 1ms

Function: checkP()

- ✓ when $p=10$, checkP() should return 1(the false value of not being a decimal
- ✓ when $p=rh43$, checkP() should return 2(the false value of not being a valid number
- ✓ when $p=0.5$, checkP() should return 3(the true value)

Function; geometricFormula()

- ✓ when $x=5$, $p = 0.5$, the result should be 0.03125 1ms

CLASS: Negative Binomial**Function: checkK()**

- return false if k is not a valid integer
- else - return true

Function: checkX()

- if x is not valid, return 2
- if x is valid but $x < 1$, return 1
- else return 3 – this means x is integer and greater or equal to 1

Function: checkP()

- if p is valid number and $p = [0,1]$, return 3
- if p value is not valid number, return 2
- if p is greater than 1, return 1

Function: checkAll()

- if inputs are valid, return true
- else, return false

Function: negativeBinomialFormula()

- return the probability result based on negative binomial formula

Test for Correctness

CLASS: Negative Binomial

Function: checkX()

- ✓ when $x=3.5$, checkX() should return error 2 - x is decimal -- DISPLAY MESSAGE: Input must be valid Integer 1ms
- ✓ when $x=0$, checkX() should return error 1 - x is zero DISPLAY MESSAGE: x must be at least 1 0ms
- ✓ when $x=7$, checkX() should return 3 - x is valid -- DISPLAY MESSAGE: ✓ 0ms

Function: checkK()

- ✓ when $k=4.5$, checkK() should return false - k is decimal -- DISPLAY MESSAGE: Input must be valid Integer 0ms
- ✓ when $k=0$, checkK() should return true -- DISPLAY MESSAGE: ✓ 0ms

Function: checkP()

- ✓ when $p=4$, checkP() should return error 1 - $p > 1$ 0ms
- ✓ when $p=0.75$, checkP() should return 3 - p is valid 0ms

Function: checkAll()

- ✓ when $k=10$, $x=8$, $p=0.9$, checkAll() should return false -- DISPLAY MESSAGE: k must be less than or equal x 0ms
- ✓ when $x=5.6$, $k=8$, $p=0.9$, checkAll() should return false -- DISPLAY MESSAGE: Invalid Input 0ms
- ✓ when $x=10$, $k=5$, $p=0.43$, checkAll() should return true 0ms

Function: negativeBinomialFormula()

- ✓ when $x=10$, $k=5$, $p=0.43$, negativeBinomialFormula() should return 0.1114518037 0ms

CLASS: Poisson**Function: callPoissonInit()**

- create Object poissonModel
- create Object poissonView
- create Object poissonController

Function: poissonController.validateInputX()

- check if x is greater than 0
- check if x is positive
- check if x is an integer
- check if x is less than or equal to 170 (limit)

Function: poissonController.validateInputLambda()

- check if lambda is greater than 0
- check if lambda is positive
- check if lambda is a real number
- check if lambda is less than or equal to 745 (limit)

Function: poissonView.updateXInput()

- returns true if controller response true
- returns false if controller response false

Function: poissonView.updateLambdaInput ()

- returns true if controller response true
- returns false if controller response false

Function: poissonModel.calculate()

- check the model calculate function returns the correct value tested with:
 - $x > \lambda$
 - $x < \lambda$
 - $x = \lambda$

Function: poissonModel.factorial()

- check the model calculates factorials correctly tested with:
 - $n = 0$
 - $n = 1$
 - $n = 10$

passes: 25 failures: 0 duration: 0.02s 100%

Test for Correctness and Performance

CLASS: Poisson

Function: callPoissonInit() - creates poissonModel, poissonView, poissonController Objects

- ✓ Check if the Object poissonModel was created 1ms
- ✓ Check if the Object poissonView was created 0ms
- ✓ Check if the Object poissonController was created 0ms

Function: poissonController - testing validation for x (must be an positive Integer greater than 0)

- ✓ poissonController.validateInputX(0) - value x = 0 should return false 0ms
- ✓ poissonController.validateInputX(0.5) - value x = 0.5 should return false 0ms
- ✓ poissonController.validateInputX(-2) - value x = -2 should return false 0ms
- ✓ poissonController.validateInputX(1) - value x = 1 should return true 0ms
- ✓ poissonController.validateInputX(170) - value x = 170 should return true 0ms
- ✓ poissonController.validateInputX(171) - value x = 171 should return false 0ms

Function: poissonController - testing validation for Lambda (must be an positive Real Number greater than 0)

- ✓ poissonController.validateInputLambda(0) - value lambda = 0 should return false 0ms
- ✓ poissonController.validateInputLambda(0.5) - value lambda = 0.5 should return true 0ms
- ✓ poissonController.validateInputLambda(1.0) - value lambda = 1.0 should return true 0ms
- ✓ poissonController.validateInputLambda(-0.5) - value lambda = -0.5 should return false 0ms
- ✓ poissonController.validateInputLambda(745.1) - value lambda = 745.1 should return false 0ms
- ✓ poissonController.validateInputLambda(745) - value lambda = 745 should return true 2ms

Function: poissonView - testing view will resolve true or false based on Controller response

- ✓ poissonView.updateXInput(true) - should return true 0ms
- ✓ poissonView.updateXInput(false) - should return false 0ms
- ✓ poissonView.updateLambdaInput(true) - should return true 0ms
- ✓ poissonView.updateLambdaInput(false) - should return false 0ms

Function: poissonModel - testing Model will result in the correct answer

- ✓ poissonModel.calculate(1,5) - should return probability 0.033689734995427344 0ms
- ✓ poissonModel.calculate(5,1) - should return probability 0.0030656620097620196 0ms
- ✓ poissonModel.calculate(5,5) - should return probability 0.17546736976785074 0ms

Function: poissonModel - testing Model factorial


- ✓ poissonModel.factorial(0) - should return probability 1 0ms
- ✓ poissonModel.factorial(1) - should return probability 1 0ms
- ✓ poissonModel.factorial(10) - should return probability 3628800 0ms

4.3 Tests for Non-functional Requirements

For Non-Functional requirements, we are going to focus on the page-speed and speed of functions that are running in the background to compute the result.

Response time of functions will test to see the duration of the task that will respond and return value is expected. We are going to use the same tests we have done for functional requirements, but this time, we are looking at the response time which also has included in section 4.2 diagrams.

The following diagrams are the result that we get from running each frame of DPC on <https://developers.google.com/speed>



DISCRETE PROBABILITY CALCULATOR

Binomial Distribution

Geometric Distribution

Hypergeometric Distribution

Negative Binomial Distribution

Poisson Distribution

ABOUT US

TERMS

CONTACT

100

<http://dpc.patricktsai.com/html/main.html>

0-49

50-89

90-100

i

Field Data — The Chrome User Experience Report [does not have sufficient real-world speed data](#) for this page.

Origin Summary — The Chrome User Experience Report [does not have sufficient real-world speed data](#) for this origin.

Lab Data

First Contentful Paint

0.3 s

Speed Index

0.3 s

Largest Contentful Paint

0.3 s

Time to Interactive

0.3 s


Total Blocking Time

0 ms

Cumulative Layout Shift

0

Values are estimated and may vary. The [performance score is calculated](#) directly from these





Discrete Probability Calculator v1.0

WHO WE ARE:

Team members:

Vuochlang Chang, Patrick Tsai, Zhicheng Zhou

Group information:

Fall 2020, Junior students at Washington state university of Vancouver

Project Information:

The **Discrete Probability Calculator (DPC)** is a group project designed for a class CS 320 - Fundamentals of Software Engineering. It is based on 4 requirements as followed:

- It is a web application
- It has a landing page that helps a user to understand what the application is about
- Besides the landing page, it has multiple other pages
- It has some data to maintain/manipulate



<http://dpc.patricktsai.com/html/aboutUs.html>

0-49 50-89 90-100 ⓘ

Field Data — The Chrome User Experience Report **does not have sufficient real-world speed data** for this page.

Origin Summary — The Chrome User Experience Report **does not have sufficient real-world speed data** for this origin.




Lab Data



● First Contentful Paint	0.3 s	● Time to Interactive	0.3 s
● Speed Index	1.1 s	● Total Blocking Time	0 ms
● Largest Contentful Paint	0.3 s	● Cumulative Layout Shift	0

Values are estimated and may vary. The **performance score is calculated** directly from these




DISCRETE PROBABILITY CALCULATOR

- Binomial Distribution
- Geometric Distribution
- Hypergeometric Distribution
- Negative Binomial Distribution
- Poisson Distribution

ABOUT US TERMS CONTACT

Terms



Attribution-NonCommercial-NoDerivatives 4.0 International (CC BY-NC-ND 4.0)

This is a human-readable summary of (and not a substitute for) the [license](#). [Disclaimer](#).

You are free to:


Share — copy and redistribute the material in any medium or format

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

Attribution — You must give [appropriate credit](#), provide a link to the license, and [indicate if changes were made](#). You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

NonCommercial — You may not use the material for [commercial purposes](#).



http://dpc.patricktsai.com/html/terms.html

0-49


50-89

90-100

i

Field Data — The Chrome User Experience Report [does not have sufficient real-world speed data](#) for this page.

Origin Summary — The Chrome User Experience Report [does not have sufficient real-world speed data](#) for this origin.



Lab Data

● First Contentful Paint

0.5 s

● Speed Index

0.7 s

● Largest Contentful Paint

0.5 s

● Time to Interactive

0.5 s


● Total Blocking Time

0 ms

● Cumulative Layout Shift

0

Values are estimated and may vary. The [performance score](#) is calculated directly from these



DISCRETE PROBABILITY CALCULATOR

[Binomial Distribution](#)

[Geometric Distribution](#)

[Hypergeometric Distribution](#)

[Negative Binomial Distribution](#)

[Poisson Distribution](#)

[ABOUT US](#) [TERMS](#) [CONTACT](#)

Contact


DPC v1.0 Developers:

- Vuochlang Chang, Junior - Washington State University Vancouver
- Patrick Tsai, Junior - Washington State University Vancouver
- Zhicheng Zhou, Junior - Washington State University Vancouver

Testing

Mocha Chai Tests for Code Correctness and Performance:

- [Validation Tests](#)
- [Binomial](#)
- [Geometric](#)
- [Hypergeometric](#)
- [Negative Binomial](#)
- [Poisson](#)




<http://dpc.patricktsai.com/html/contact.html>

0-49

50-89

90-100




Field Data

— The Chrome User Experience Report [does not have sufficient real-world speed data](#) for this page.

Origin Summary


— The Chrome User Experience Report [does not have sufficient real-world speed data](#) for this origin.



Lab Data

<div><div></div></div> First Contentful Paint	0.5 s	<div><div></div></div> Time to Interactive	0.5 s
<div><div></div></div> Speed Index	0.6 s	<div><div></div></div> Total Blocking Time	0 ms
<div><div></div></div> Largest Contentful Paint	0.7 s	<div><div></div></div> Cumulative Layout Shift	0

Values are estimated and may vary. The [performance score is calculated](#) directly from these



DISCRETE PROBABILITY CALCULATOR

Binomial Distribution

Geometric Distribution

Hypergeometric Distribution

Negative Binomial Distribution

Poisson Distribution

[ABOUT US](#) [TERMS](#) [CONTACT](#)

Binomial Distribution

Explanation

Example Problem

Video

x:

n:

p:

Result:

79

<http://dpc.patricktsai.com/html/binomial.html>

0-49

50-89

90-100

i

Field Data

The Chrome User Experience Report [does not have sufficient real-world speed data](#) for this page.

Origin Summary

The Chrome User Experience Report [does not have sufficient real-world speed data](#) for this origin.

Lab Data

First Contentful Paint

0.6 s

Speed Index

0.8 s

Largest Contentful Paint

1.7 s

Time to Interactive

2.1 s

Total Blocking Time


380 ms

Cumulative Layout Shift

0

Values are estimated and may vary. The [performance score is calculated](#) directly from these





DISCRETE PROBABILITY CALCULATOR

- Binomial Distribution
- Geometric Distribution**
- Hypergeometric Distribution
- Negative Binomial Distribution
- Poisson Distribution

[ABOUT US](#) [TERMS](#) [CONTACT](#)

Geometric Distribution

Explanation

Example Problem

Video

x:

p:

Result:

76

<http://dpc.patricksai.com/html/geometric.html>

0-49

50-89

90-100

Field Data

The Chrome User Experience Report [does not have sufficient real-world speed data](#) for this page.

Origin Summary

The Chrome User Experience Report [does not have sufficient real-world speed data](#) for this origin.

Lab Data

First Contentful Paint

0.5 s

Time to Interactive

2.2 s

Speed Index

0.8 s

Total Blocking Time

420 ms


Largest Contentful Paint

1.8 s

Cumulative Layout Shift

0

Values are estimated and may vary. The [performance score is calculated](#) directly from these



DISCRETE PROBABILITY CALCULATOR

[Binomial Distribution](#)[Geometric Distribution](#)[Hypergeometric Distribution](#)[Negative Binomial Distribution](#)[Poisson Distribution](#)

[ABOUT US](#)[TERMS](#)[CONTACT](#)

Hypergeometric Distribution

Explanation

Example Problem

Video

x:

n:

N:

K:

Result:

Reset

Result

82

http://dpc.patricktsai.com/html/hypergeometric.html

0-49

50-89

90-100

i

Field Data — The Chrome User Experience Report [does not have sufficient real-world speed data](#) for this page.

Origin Summary — The Chrome User Experience Report [does not have sufficient real-world speed data](#) for this origin.

Lab Data

● First Contentful Paint

0.6 s

● Time to Interactive

2.1 s

● Speed Index

0.7 s

■ Total Blocking Time

270 ms


■ Largest Contentful Paint

1.9 s

● Cumulative Layout Shift

0

Values are estimated and may vary. The [performance score is calculated](#) directly from these

**DISCRETE
PROBABILITY
CALCULATOR**

[Binomial Distribution](#)
[Geometric Distribution](#)
[Hypergeometric Distribution](#)
[Negative Binomial Distribution](#)
[Poisson Distribution](#)

[ABOUT US](#) [TERMS](#) [CONTACT](#)

Negative Binomial Distribution

Explanation

Example Problem

Video

x:

k:

p:

Result:

84

http://dpc.patricktsai.com/html/negative_binomial.html

0-49

50-89

90-100

i

Field Data — The Chrome User Experience Report [does not have sufficient real-world speed data](#) for this page.

Origin Summary — The Chrome User Experience Report [does not have sufficient real-world speed data](#) for this origin.



Lab Data

First Contentful Paint

0.5 s

Speed Index

1.1 s

Largest Contentful Paint

1.8 s

Time to Interactive

2.0 s


Total Blocking Time

230 ms

Cumulative Layout Shift

0

Values are estimated and may vary. The [performance score is calculated](#) directly from these



**DISCRETE
PROBABILITY
CALCULATOR**

Binomial Distribution

Geometric Distribution

Hypergeometric Distribution

Negative Binomial Distribution

Poisson Distribution

ABOUT US TERMS CONTACT

Poisson Distribution

Explanation


Example Problem

Video

X:

λ :

Result:



<http://dpc.patricksai.com/html/poisson.html>

0-49 50-89 90-100 ⓘ

Field Data — The Chrome User Experience Report [does not have sufficient real-world speed data](#) for this page.

Origin Summary — The Chrome User Experience Report [does not have sufficient real-world speed data](#) for this origin.

Lab Data

● First Contentful Paint	0.5 s	● Time to Interactive	2.1 s
● Speed Index	0.7 s	▲ Total Blocking Time	430 ms
■ Largest Contentful Paint	1.7 s	● Cumulative Layout Shift	0

4.4 Hardware and Software Requirements

- This web application will be supported on Windows, Macintosh, IOS and Android.
- This web application will require user to use mouse to make clicks to interact and the keyboard for typing in inputs for distribution calculation. No other hardware is required.

- This web application will also require monitor to display all the reactions between the user and the application itself.
- This application will be supported on most the popular browsers, for example: Safari, Chrome, Firefox, IE etc.
- This application will be supported on Windows, Mac OS, Linux, etc.
- JavaScript will be used to handle the calculations, interactions between functions/classes and any back-end process. Html and CSS will also be used for designing the appearance and formatting the outputs/messages.
- This application will identify data shared across software components:
 - Inputted data from user
 - Message from system for invalid data / warnings
 - Data from program to the monitor

5 Analysis

Vuochlang Chang - Anna

Milestone 1: Software Requirement System

I worked on section 1 - Introduction to the DPC. It took me about 2 to 3 hours to finished it because it was asking only for descriptions. However, since my section wasn't required that much time, I started the code for the website to get everything running.

Milestone 2: Software Design Document

I worked on section 2 - the activity diagram. It took me about an hour to finished those 5 diagrams, but then, later on, I changed the diagram a few times to make it look better.

Milestone 3: Final Report

I worked on section 4 - testing. This milestone took the most effort because we have to make sure that everything is presentable. Even we have divided parts of codework for the website and parts of the section for the milestone to finish, there are still more background works that needed to be done.

Patrick Tsai

Milestone 1: Software Requirement System

Section 2 Overall Description and Section 4 Other Non-Functional Requirements

I spent about 6 to 8 hours on Milestone 1. Most of my time was spent thinking about how the site may possibly act, an example is the section about who will use the site. This was time intensive because the project was new and didn't have a defined state (at the time) so it was easy to speculate. Another time sink was a formatting issue with Word 365. I had to copy paste everything into a new file section by section before submitting the document.

Milestone 2: Software Design Document

Section 3 Class Diagrams

I spent about 5 to 7 hours on Milestone 2. I spent the least amount of time on this Milestone. My only issue was getting familiar with plantUML. I spent most of my time tinkering with the diagram flow. In the end, my diagram was wrong, but it did show the basic connectivity within our site.

Milestone 3: Final Report

I spent about 8 to 10 hours on the Milestone 3. I spent most of my time creating the Mocha Chai tests for the Poisson distribution, remaking the class diagrams from Milestone 2, and finalizing the website.

Zhicheng ZhouMilestone 1: Software Requirement System

I spent about 5 to 6 hours to work on Milestone 1. I was assigned to do Section 3: Specific Requirements.

For this section I had to be very careful and specific about what I write because the main idea of this section is how our calculator would require user to interact with it and what hardware, software requirements there are in order to use this calculator, so I spent lots of time to plan out the how the initial interface would look like and what specific actions we would expect the user to take to interact with the calculator. All we had was a picture of how the landing page looks like when I was doing this section, so details of what inputs should be entered for a specific distribution was not provided in this section. I also had some problems with the hardware and software requirements, but the professor mentioned in class that all we need to write for that is mouse, keyboard if our project does not require anything else, so that clarified my confusions.

Milestone 2: Software Design Document

I spent about 3 to 4 hours to work on Milestone 2. I was assigned to do the Behavioral Diagrams, considering our project is both event-driven and data-driven so I made one sequence diagram for the entire structure of the calculator and five state diagrams for all the distributions. Even though Plant UML is not as complicated as other coding languages, but it still took me a fair amount of time to get everything set up correctly.

Milestone 3: Final Report

I spent about 4 to 5 hours to work on Milestone 3. I was assigned to do the introduction, overview, reference, behavioral diagrams and testing for the distributions I wrote. I updated the reference because we used lots of materials from stattrek.com to learn more about all the distributions and examples. I also updated the sequence and state diagrams as the TA commented on our Milestone 2, I corrected some typos, added more states as for our final product.

6 Conclusion

Vuochlang Chang - Anna

While building this DPC, I have learned a lot about HTML, CSS, and JS coding styles. I have never been taught about those HTML and CSS coding style before this CS320 class, so everything is new to me. I have also discovered <https://www.w3schools.com/> that has helped me to understand what I was doing. It is the best source that I can refer to when I want to learn and understand about combining different codes to make the website looks nicer.

On the other hand, I also learn that teamwork and group effort are the most important thing in this project. I really appreciate the teamwork and my teammates. Everything has been going smoothly and we are able to communicate and help each other without any conflict.

Patrick Tsai

As CS students, we usually work alone. I find group projects important for improving communication skills and diplomacy. At the start of this project, I had a very different website in mind, but working within the group helped open my mind to something completely different, and something that I'm very proud to have worked on. It would not have happened without the great work of my teammates Anna and Z.

I can't put into words everything I learned because it would take up too much space. But I want to emphasize how much I enjoyed working with git. I'm not a master at using it, but I appreciate the flexibility of version control and the power of seamlessly linking the code bases of 3 students located in three different locations.

Additionally, learning about Mocha and Chai has been enriching. I like to code, but understanding code accuracy, performance and maintenance are essential to bettering myself as a developer.

Zhicheng Zhou - Z

This is my first semester at University and the first time doing such a big group project with other people which has been an incredible experience for me. I've learned so much about JavaScript; HTML; CSS and Git, before came to this class and this project, I had no idea how to use any or those, now I can proudly say that I can write some program using JavaScript and how to implement some website using HTML and CSS. On the other hand, using what we've learned in class has also helped me a lot, such as Unit Testing, Automated Testing and ESLint coding style, those professional tools have helped me solve lots of problems.

I wanted to mention that I couldn't have completed this project without two of my teammates. I've had lots of trouble to get started but my teammates were just so nice to walk me through anything that I needed help for, I realized that collaboration is a big part of programming and I hope I will bring the collaborative skills to future workplaces.

Appendix A - Group Log

CS320 – Meeting Notes

These are the Meeting Minutes for FINAL REPORT ONLY – Previous meeting minutes are found in the Group Log Appendix in Project Milestone 1 and 2

11/30/2020 - Zoom 11:55am to 12:20pm

- this was a quick meeting to go over some responsive themeing code
- Patrick worked on for the site, he wanted to get the OK to push it to the main repo. Also, went over some git stuff

12/02/2020 - Zoom meeting from 11:00am to 12:30pm

- Anna went over the callBinomial JavaScript code for validating inputs and the calculation
- Z showed us the hypergeometric code and we went over some git stuff
- Patrick went over respsnive themeing code, accordion javascript, and some css stuff.

Next meeting Saturday Dec 5, 11am to talk about Milestone 3

12/05/2020 - Zoom meeting from 11:00am to 12:35pm

- Z went over the hypergeometric/geometric code
- Anna and Patrick went through Project milestone 3 and separated the work.

```
Project Milestone 3
Section 1
Z
- this will be copy and paste from Milestone 1, 2
- update Reference
Section 2
Patrick
- add images of web design (section 2.2)
- mobile phone screen shots
Anna
- change the use case
- activity diagram
Patrick
- change class diagram
Z
- change sequence and state diagrams
Section 3
Patrick
- will do section 3.1, 3.2 and 3.3
Everyone Section 3.2
- review what is in table make changes and add descriptions
Section 4
Anna
will do 4.1 - 4.3
Z
will do 4.4
Section 5
- we will do this together at a meeting before Dec. 16th
Section 6
Everyone
- Write 1 paragraph about what they learned during the project (learn code, programs)
```

12/07/2020 - Zoom Meeting

- Monday, Dec 7 from 12:00pm - 12:25 pm
- Decided on a time for the project presentation [Tuesday, Dec. 15th 10:40am]
- Anna talked about changes she made to the diagrams for Milestone 3

Next meeting Wed. Dec 9, 2pm - this will be similar to today, a quick check in, where we will talk more about Milestone 3

12/9/2020 - Zoom meeting

- from 12pm - 1:30 pm
- Anna has done a lot for Milestone 3. She went over her sections.
- Z and Patrick will create Mocha/Chai tests for their code and will add to Anna's section.
- Talked about Milestone 2 diagrams and what needs to be changed for the next report. We went over Milestone grader notes.
- We realized we also need to update the SRS and turn it in with the Final Report.
- Divided up some last task: Anna will make the About page, Patrick will handle the Terms and Contact pages.

12/12/2020 - Zoom meeting

- Dec. 12, 11am - 12:30pm
- We went over the Final report section by section. Not much left to do. Only things left - references, Meeting log, git log. Yay.
- We also went over the SRS revision;
 - left to do - Patrick needs to redo the UML for the project overview (take out help pages), reread the SRS and take out any extra stuff, the comments on the SRS basically said it was too long.
- We also went over the demonstration and assigned everyone's parts.

12/14/2020 - Zoom meeting

- Dec. 14, 11am - 12:00pm
- We went over the Final report and SRS to see what is left to change
- We practiced for the project presentation
- Anna will change some test cases for Binomial test
- next meeting Wed. Dec. 16 11:00 am
- Presentation: Dec. 15, 10:40 am

12/16/2020 - Zoom meeting

- Dec. 16, 11am - 12:00pm
- Review SRS v2 and Final Report
- Patrick will submit them on BlackBoard