Exponentials Simulation: Exploring Growth with Blocks and Graph

Table of Contents

Exponentials Simulation: Exploring Growth with Blocks and Graph 1
1.0 Introduction 2
1.1 Purpose 2
T 2
1.2 Exponentials Simulation 2
2.0 Process Model 3
3.0 Use Cases 4
3.1 Use Case 1: Exploring Linear Growth 4
3.2 Use Case 2: Customizing Exponential Growth 4
3.3 Use Case 3: Long-term Exponent Simulation 5
3.4 Use Case 4: Assessing Learning Outcomes 5
3.5 Use Case 5: Comparing Growth Patterns 6
3.6 Use case 6: Educator Scenario Creation 6
4. UML Model 8
4. 1. Use Case Diagram 9
4.2. Class Diagram 9
4.3. Activity Diagram 10
4.4. Deployment Diagram 10
5.0 Customer Journey Map 11
6.0 Persona 11
7. Testing Strategy 12
7.1. Unit Testing 12
7.2 Sample Test Cases 12
7.3. Integration Testing 16
8.0 Conclusion

1.0 Introduction

1.1 Purpose

The "Exponential Simulation" project's main goal is to help pupils from kindergarten through grade five understand and enjoy the abstract ideas of linear and exponential growth. This project intends to promote a profound understanding of how amounts change and expand over time by combining firsthand activities, dynamic visuals, and interactive exploration. Students will obtain useful problem-solving abilities and the capacity to apply these ideas to real-world settings during this educational adventure, in addition to learning about linear and exponential growth. The project's ultimate goals are to pique kids' curiosity, foster their love of mathematics, and provide them with the fundamental abilities they need to continue learning and solving problems throughout their lives.

1.2 Exponentials Simulation

The idea of growth is central to our comprehension of change and extension in the fields of mathematics and science. This idea is fundamental to many facets of our life, from population growth to the mechanics of disease transmission. Welcome to the "Exponentials Simulation" project, an engrossing learning experience painstakingly designed to reveal the nuances of linear and exponential growth. From kindergarten to grade five, kids can participate in this educational adventure. This project creates an appealing way to make these abstract notions not only accessible but also pleasurable by combining tactile blocks and dynamic graphs.

We begin teaching linear growth to our youngest students by doing a straightforward yet powerful hands-on presentation. Think of a robot painstakingly stacking each brick one at a time. Starting with one block, it steadily adds two and three until a tower of ten blocks is formed. A fundamental idea in mathematics, predictable, linear progression, is laid forth in this interactive activity.

As they advance, students are invited to investigate the idea of growth in more depth by choosing a one-digit number, "k." Then the robot takes over and arranges the blocks in accordance with "k * x," where "x" ranges from 1 to 10. Young minds can understand the idea that growth can occur at various rates and patterns thanks to this dynamic method.

The initiative exposes the fascinating realm of exponential development to pupils in grades 3–5. Students have the option of selecting "2x," "x2," or "3x" here. Based on the selected growth pattern, the robot turns into a mathematical artist and builds complex stacks. These eye-catching patterns operate as a link between students and the conceptually abstract world of mathematics.

The voyage continues past stacks and enters colorful graphs. The 'x' range can be adjusted by students, and they can compare different graphs by superimposing them. This interactive element helps learners comprehend exponential development in greater depth and creates the groundwork for investigating different mathematical functions.

The research explores long-term studies in addition to short-term simulations, giving students the chance to model real-world situations like bacterial development. Students develop understanding of mathematical modeling and its real-world applications by observing and analyzing growth patterns, from rapid spikes to leveling off when capacity is achieved.

Students are urged to ask questions, make predictions, and try out various functions during the tour. Experimenting with growth principles one block or graph at a time, "Exponentials Simulation" is a fascinating and illuminating experience because of its user-friendly interface, extensive educational resources, and exploratory mindset. So be ready to set out on this riveting voyage through the interesting world of growth and don your thinking caps.

2.0 Process Model

It was purposeful and crucial that the Agile Scrum process model be used for the "Exponentials Simulation" project. The iterative and flexible characteristics of Agile Scrum fully match the primary goals of our project. Its importance comes from its capacity to enable continuous feedback, allowing us to quickly respond to the changing demands of educators and students. By ensuring that our efforts to create educational content and software are coordinated, this model promotes a dynamic atmosphere where improvement is not only welcomed but also woven into the very foundation of our project. In summary, Agile Scrum gives us the ability to design a teaching tool that is not only efficient but also constantly responsive, creating the foundation for a dynamic and lasting learning experience.

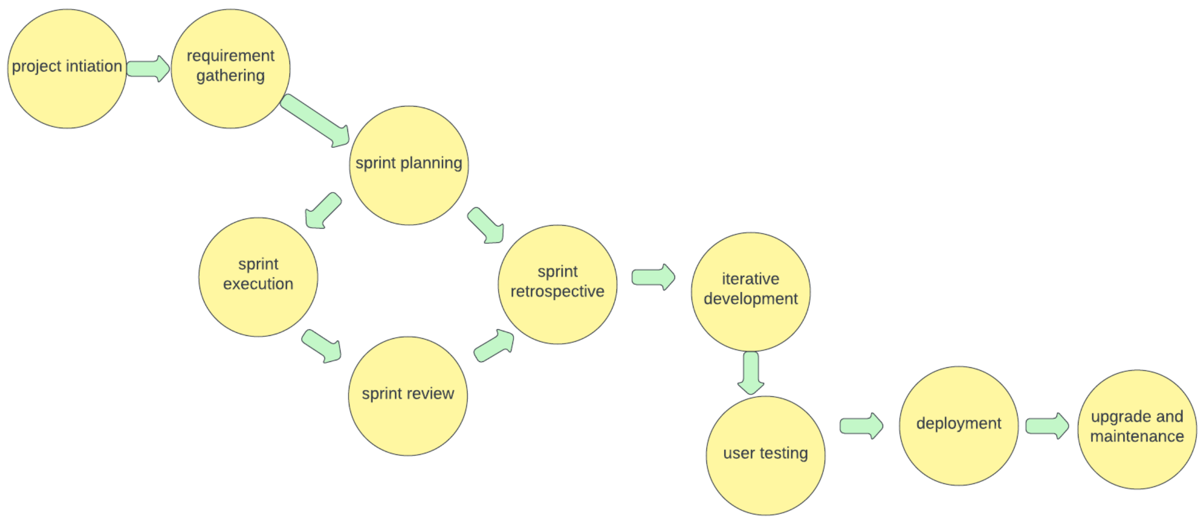


Figure 1. Exponentials simulation process model.

3.0 Use Cases

The following six uses cases were defined by the team and the client as the core system requirements for the delivery of the working prototype. Note, this baseline functionality can be easily extended through the inclusion of more use cases as the project progresses. Each use case lists the name of the use case, primary actors, preconditions, description, and acceptance criteria.

3.1 Use Case 1: Exploring Linear Growth

Primary Actor: Student

Precondition: The simulation is launched and accessible.

Description: The student selects the "Linear Growth" option from the simulation's menu. They are presented with a visual representation of linear growth, starting with a single block, and progressing to multiple blocks, each step clearly displayed.

Acceptance Criteria:

The student can choose to start the linear growth simulation.

The simulation displays the stacking of blocks from 1 to 10, one block at a time.

The student can pause and resume the simulation at any point.

The simulation allows the student to reset and start over.

3.2 Use Case 2: Customizing Exponential Growth

Primary Actor: Educator

Precondition: The educator has access to the simulation's admin panel.

Description: The educator customizes an exponential growth scenario to align with specific learning objectives. They choose the base (e.g., 2x, x^2, 3x) and set the range for 'x,' ensuring it corresponds with the grade level's curriculum.

Acceptance Criteria:

The educator can access the admin panel and choose the "Customize Exponential Growth" option.

The admin panel provides options to select the base (2x, x^2, 3x).

The educator can set the range for 'x' (e.g., 1 to 10).

Once customized, the exponential growth scenario appears in the simulation for students.

The simulation generates graphs that accurately reflect the chosen exponential growth scenario.

Educators can save and share customized scenarios for future use.

3.3 Use Case 3: Long term Exponent Simulation

Primary Actor: Student

Precondition: The student has access to the simulation.

Description: The student chooses the "Long-Term Experiment" option and selects a scenario, such as bacterial growth. They observe the simulation over an extended period, witnessing the growth pattern, leveling off when capacity is reached, and any programmed variations.

Acceptance Criteria:

The student can access the "Long-Term Experiment" feature.

The simulation offers a choice of scenarios (e.g., bacterial growth).

The student can set parameters for the experiment (e.g., initial population, growth rate, carrying capacity).

The simulation accurately represents the chosen scenario's growth patterns.

The student can pause, rewind, and fast forward the simulation.

At the end of the simulation, the student is prompted with questions related to the observed growth pattern for assessment.

3.4 Use Case 4: Assessing Learning Outcomes

Primary Actor: Educator

Precondition: The educator has access to student progress data.

Description: The educator reviews individual student or class progress within the simulation. They analyze data such as the number of simulations completed, time spent, and quiz results to assess learning outcomes and identify areas where additional support may be needed.

Acceptance Criteria:

The educator can access a dashboard displaying student progress data.

The dashboard provides statistics on completed simulations, including the number of linear and exponential growth scenarios explored.

Quiz results and scores are available for each student.

The educator can identify trends in student performance.

The system generates reports highlighting areas where students may need additional guidance or practice.

3.5 Use Case 5: Comparing Growth Patterns

Primary Actor: Student

Precondition: The student has selected the "Compare Growth Patterns" option within the simulation.

Description: The student chooses two different growth models (e.g., 2x and x^2) and observes their respective graphs simultaneously. This use case enables students to visually compare and contrast the growth patterns of distinct functions.

Acceptance Criteria:

The student selects the "Compare Growth Patterns" feature.

The simulation allows the student to choose two growth models.

Two separate graphs are displayed, each representing the selected growth models.

The graphs accurately depict the chosen growth functions.

The student can manipulate the range of 'x' for both graphs.

An explanation of the observed similarities and differences is provided to facilitate learning.

3.6 Use case 6: Educator Scenario Creation

Primary Actor: Educator

Precondition: The educator has access to the simulation's scenario creation tool.

Description: The educator utilizes the scenario creation tool to design custom growth scenarios that align with specific lesson plans. They define the initial conditions, growth patterns, and educational content, tailoring simulations to enhance classroom learning experiences.

Acceptance Criteria:

The educator can access the scenario creation tool from the admin panel.

The tool allows the educator to define initial conditions (e.g., initial population, starting point).

Customizable growth patterns can be created by specifying equations and parameters.

Educational content, including explanations, questions, and quizzes, can be integrated into the scenario.

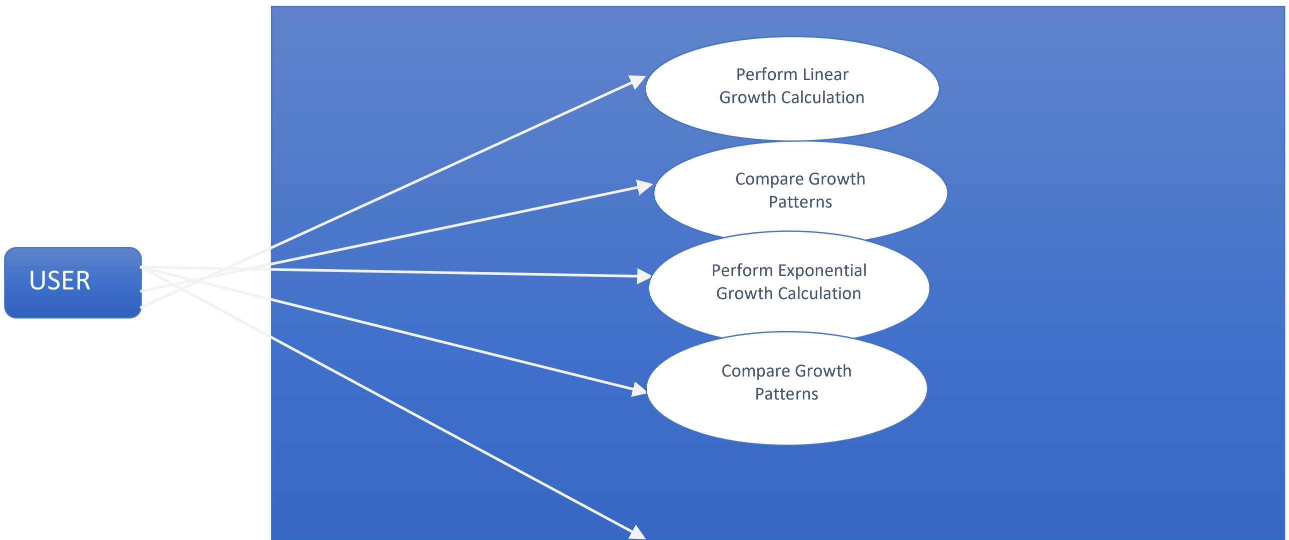
The created scenario seamlessly integrates into the simulation for student access.

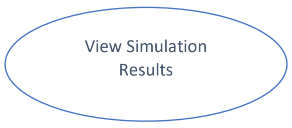
Educators can save and share their custom scenarios with colleagues or the broader educational community.

4. UML Model

4. 1. Use Case Diagram:

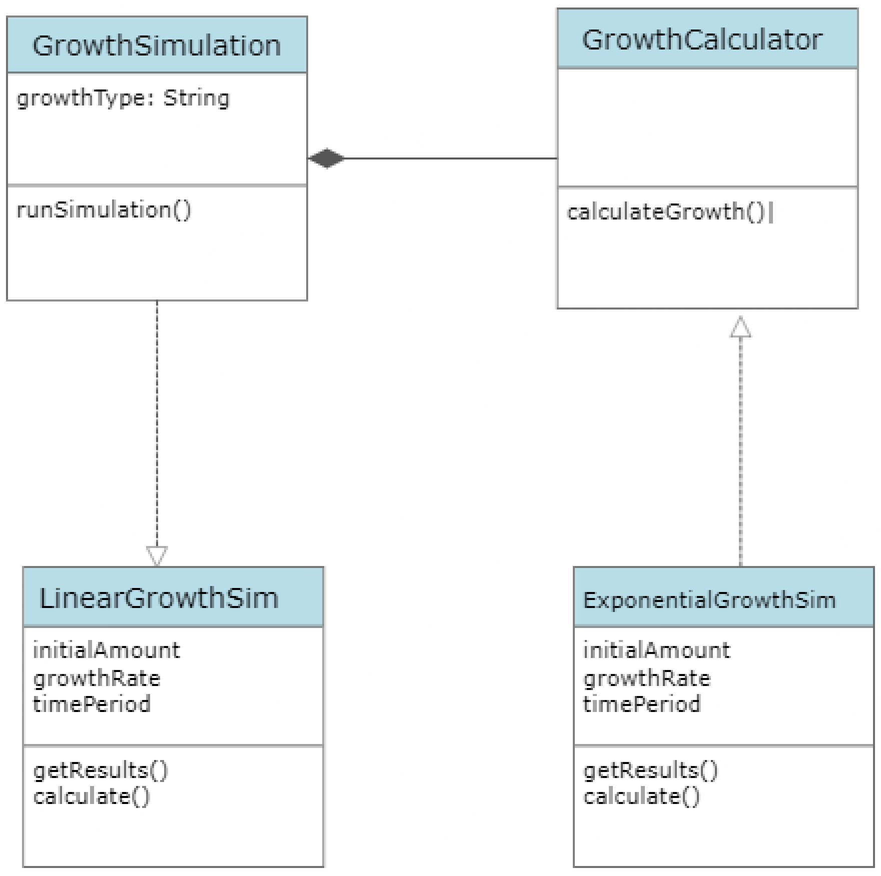
The key features of the project are depicted in a use case diagram. A few of the important use cases represented in the diagram are "Run Linear Growth Simulation," "Run Exponential Growth Simulation," and "Compare Results." Within the simulation system, each use case represents a particular action that a user may carry out. Users can enter parameters into "Run Linear Growth Simulation" to simulate linear growth. Users can simulate exponential growth using the "Run Exponential Growth Simulation" command with certain parameters. Users can examine and contrast the results of both simulations using the "Compare Results" feature, which helps users better understand the distinctions between linear and exponential growth patterns. In the context of the simulation project, the use case diagram provides a clear overview of user interactions and system functionality.


USER
Perform Linear Growth Calculation
Compare Growth Patterns
Perform Exponential Growth Calculation
Compare Growth Patterns



4.2. Class Diagram

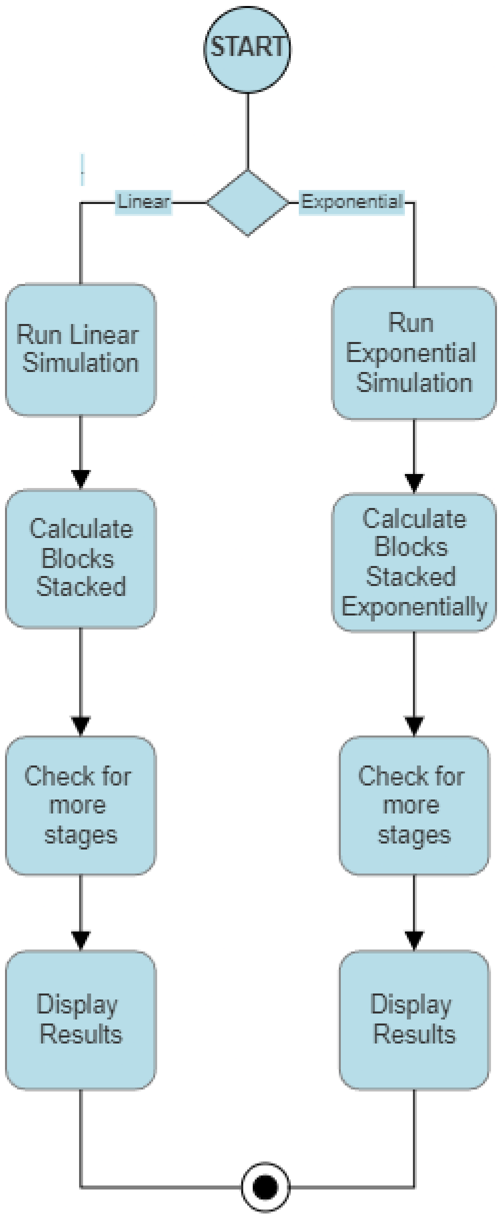
In the below Class Diagram, the common characteristics and techniques of growth simulations are represented by the abstract class Growth Simulation. A class called GrowthCalculator is in charge of computing growth based on specific parameters. A linear growth simulation is represented by the subclass of GrowthSimulation known as LinearGrowthSim. It has characteristics like timePeriod, growthRate, and initialAmount. Additionally, it has the methods calculate() for calculating growth and getResults() for retrieving simulation results. A subclass of GrowthSimulation that represents an exponential growth simulation is called ExponentialGrowthSim. Similar to the LinearGrowthSim class, it has the attributes initialAmount, growthRate, and timePeriod as well as its unique methods calculate() and getResults().



Class Diagram

4.3. Activity Diagram

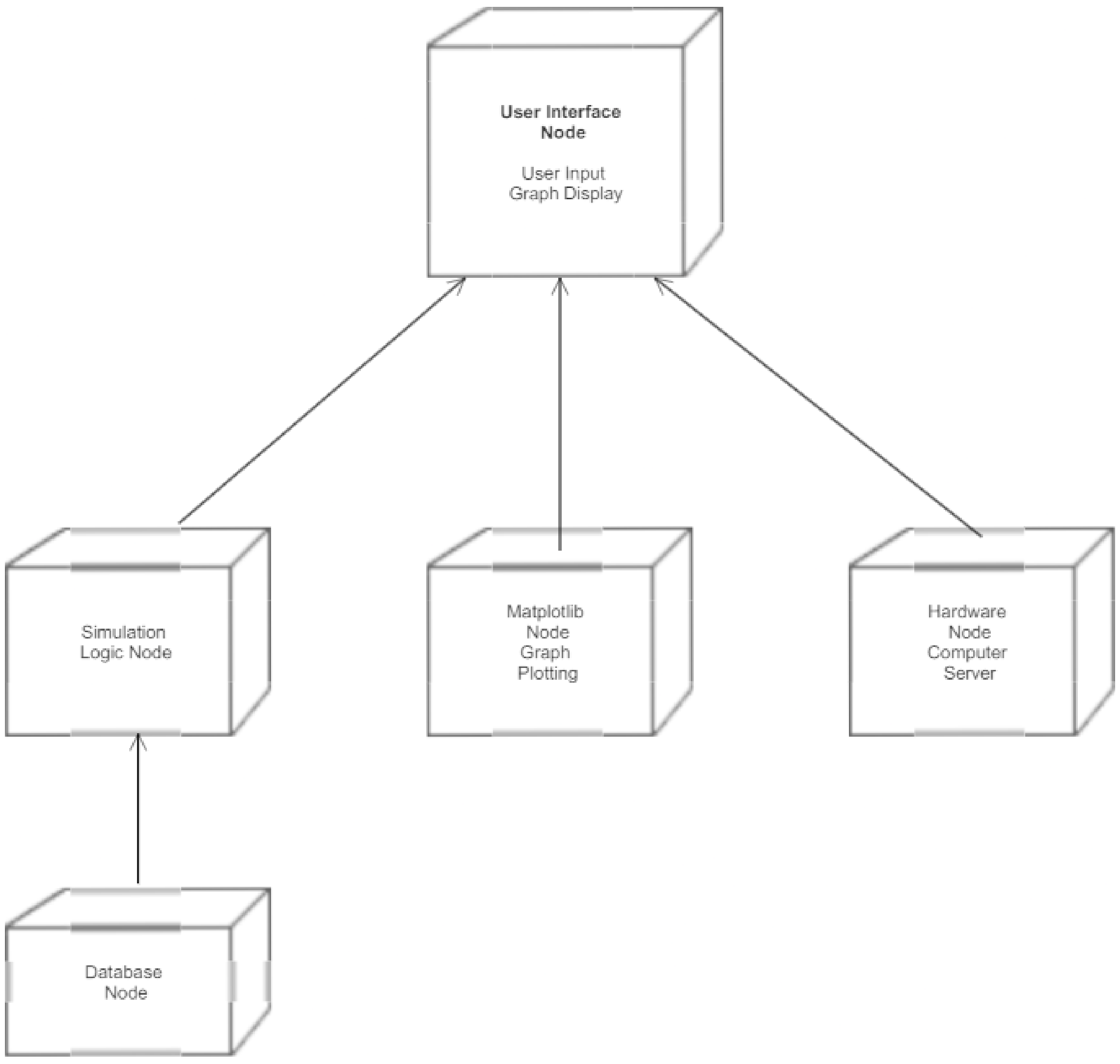
The activity diagram illustrates the sequential steps within the "Linear vs Exponential Growth Simulation" project. It starts with user input for growth type selection and continues through the calculation and display of blocks stacked for both linear and exponential growth scenarios. The diagram emphasizes the user's journey through the simulation, showcasing decision points and interactions with the program.



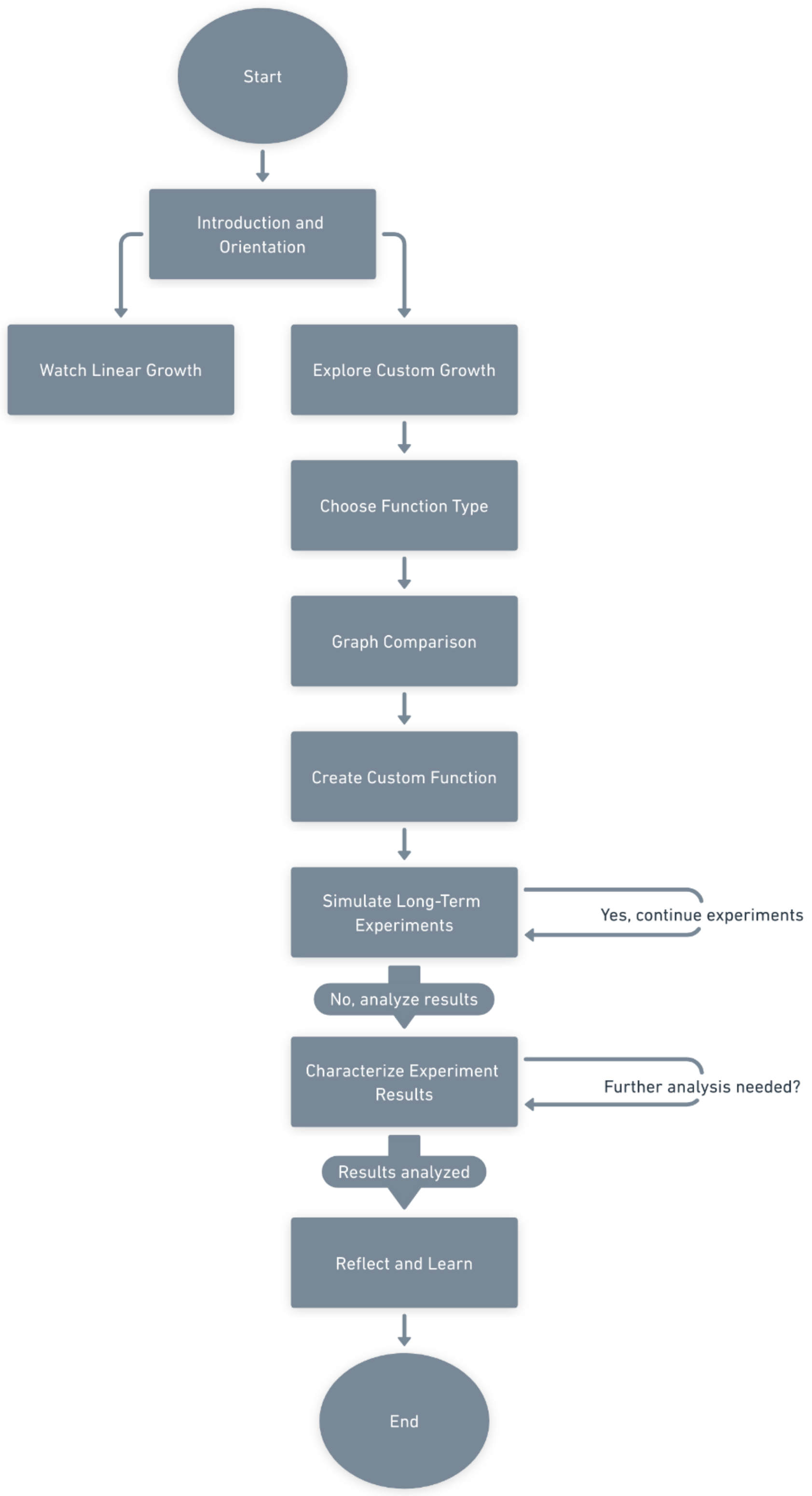
Activity Diagram

4.4. Deployment Diagram

The below deployment diagram illustrates the distribution and interaction of various components within the project. The diagram showcases five main nodes: the User Interface Node responsible for user interaction and graph display, the Simulation Logic Node handling calculations and simulations, the Database Node for storing user data and results, the Matplotlib Node for graph plotting, and Hardware Node(s) representing computers or servers. The User Interface Node communicates with both the Simulation Logic Node and the Matplotlib Node to gather simulation inputs and display graphs. The Simulation Logic Node interacts bidirectionally with the Database Node to store and retrieve data.



5.0 Customer Journey Map:


Start
Introduction and Orientation
Watch Linear Growth
Explore Custom Growth
Choose Function Type
Graph Comparison
Create Custom Function
Simulate Long-Term Experiments
Yes, continue experiments
No, analyze results
Characterize Experiment Results
Further analysis needed?
Results analyzed
Reflect and Learn
End

The Customer Journey Map for the Exponentials Simulation system presents a sequential progression of a user's interactions and experiences. It begins with users being introduced to the simulation's linear growth representation, followed by their engagement with exponential growth concepts by creating and observing stacks of blocks. As they advance, users customize functions, graph various equations, and learn to control graph parameters. The map also outlines the simulation's capability for long-term experiment simulations, encouraging users to analyze and characterize growth patterns. Overall, the map illustrates a step-by-step path where users transition from basic understanding to more advanced concepts, fostering a comprehensive learning journey within the simulation environment.

6.0 Persona

Alex is a motivated high school student who wants to excel in mathematics but occasionally struggles with complex concepts like exponentials. Alex prefers learning through interactive methods, as they find it easier to grasp ideas when they can see visual representations. They have used educational software before and appreciate learning resources that adapt to their pace. However, they often need clear explanations to fully understand abstract mathematical theories. Alex would benefit from the Exponentials Simulation System by using its interactive features to visualize how exponentials work in different scenarios. The ability to manipulate parameters and observe instant outcomes would greatly enhance their comprehension. Additionally, Alex would appreciate explanations that break down the concepts step by step, helping them bridge the gap between theory and application. Providing Alex with an intuitive and engaging learning experience through the simulation system will empower them to overcome challenges and confidently embrace exponential concepts.

Name: Alex **Age: 16**	A high school student who is interested in Mathematics, problem-solving, interactive learning and trying to understand the concept of exponentials and their applications.	He finds abstract math concepts challenging, prefers visual and interactive learning

7. Testing Strategy

The testing strategy for the Exponential Growth Simulation aims to ensure the accuracy, reliability, and user-friendliness of the simulation tool. It involves multiple phases, including unit testing, integration testing, and user testing, to identify and resolve potential issues at various levels of the software development process.

7.1. Unit Testing:

Unit testing focuses on testing individual components or units of the simulation in isolation to ensure they function correctly. Each unit, which could be a function, module, or class, is evaluated to verify that it performs its intended functionality accurately.

Test Case Creation: Create test cases for each function or method within the simulation codebase. Test cases should cover various scenarios and edge cases.

Test Execution: Execute unit tests using a testing framework (e.g., JUnit for Java) to automate the process and ensure consistency.

Input Validation: Test functions with valid, invalid, and boundary inputs to check if they manage input validation correctly.

Expected Output: Compare the actual output of the function to the expected output based on test inputs. Ensure the results match the expected outcomes.

Code Coverage: Aim for high code coverage to ensure that most parts of the code are evaluated. Use tools to measure code coverage and identify untested code paths.

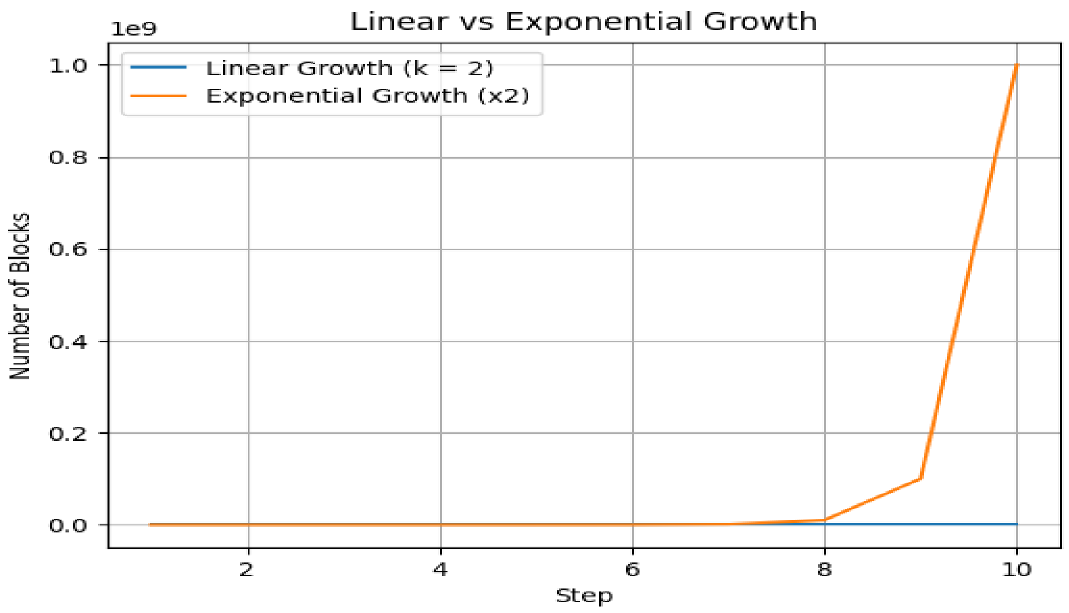
Mocking Dependencies: When a function depends on external components, use mocking or stubbing techniques to isolate the unit being evaluated.

7.2 Sample Test Cases:

Below are the test cases that are illustrated with a graphical representation of the output of code in different task case scenarios.

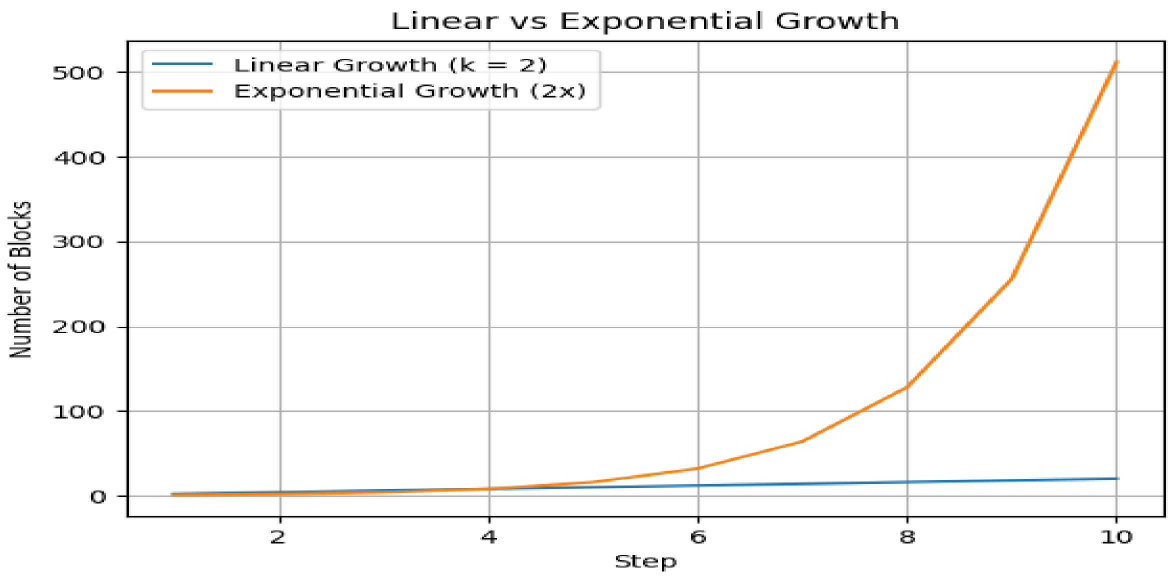
Test Case 1:

Linear Growth: k = 2
Choose graph option: 2x
Exponential Growth (2x)
Output: The exponential growth steps, where each step doubles the number of blocks.



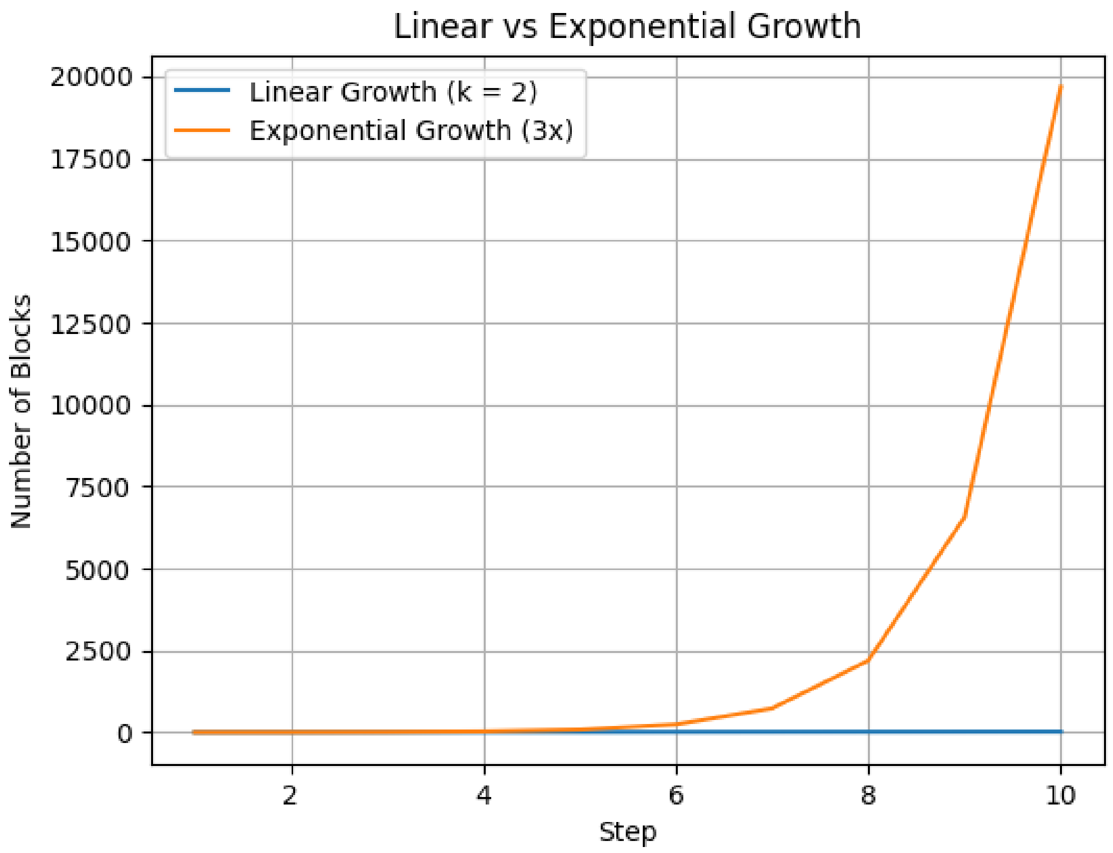
Test Case 2:

Linear Growth: k = 2
Choose graph option: x2.
Exponential Growth (x2)
Output: The exponential growth steps, where each step squares the number of blocks.



Test Case 3:

Linear Growth: k = 2
Choose graph option: 3x
Exponential Growth (3x)
Output: The exponential growth steps, where each step triples the number of blocks.



7.3. Integration Testing:

Integration testing involves evaluating the collaboration and integration of various software components to ensure that they work seamlessly together. The main objectives of integration testing are to identify any inconsistencies, data flow issues, communication problems, and ensure that the system as a whole function as intended.

1. **Component Integration:**

   Components, such as the user interface (UI), growth calculation functions, and visualization modules, will be integrated to ensure they can communicate and exchange data accurately. For instance, the UI should be able to input values, pass them to the growth calculation functions, and display the results obtained from the calculations.

2. **Linear and Exponential Growth Calculation Integration:**

   The integration testing will verify that the linear and exponential growth calculation functions interact correctly with the UI and provide accurate results. The calculations should take the provided input values, compute the growth, and return the expected output.

3. **User Interaction Testing:**

   The integration testing process will include tests to confirm that user interactions, such as clicking buttons, entering data, and triggering calculations, function smoothly and as expected. For example, clicking the "Calculate" button should initiate the calculations and update the UI with the results.

4. **Error Handling Integration:**

   Integration testing will examine how errors are managed and displayed within the integrated system. If invalid input is entered, the system should respond with appropriate error messages without causing crashes or unexpected behavior.

8.0 Conclusion:

We provide a Python program for examining and analyzing linear and exponential growth patterns. The program explores these growth concepts through interactive input, calculations, and graphical representation.

The code consists of three main functions: linear growth, exponential growth, and main. The linear growth function models linear growth by stacking blocks in a linear manner, while the exponential growth function illustrates exponential growth by raising a base to different powers. The main function orchestrates user interactions, calculation calls, and visualization.

The program offers users the ability to input a one-digit number 'k' for linear growth and to choose between three exponential growth options: 2x, x2, or 3x. The user's input influences both the linear and exponential growth calculations. The code then generates and displays both growth patterns graphically using the matplotlib library.

These test cases highlight the adaptability of the code to manage different growth scenarios. The program effectively calculates and presents the linear and exponential growth patterns, while the graphical representation visually demonstrates the growth disparities. The program's interactive nature and ability to visualize complex growth concepts make it a valuable tool for educational purposes, enabling users to intuitively grasp the differences between linear and exponential growth.

References:

- U.Ibarra Hernández, F. J. Álvarez Rodríguez and M. Vargas Martin, "Use processes — modeling requirements based on elements of BPMN and UML Use Case Diagrams," 2010 2nd International Conference on Software Technology and Engineering, San Juan, PR, USA, 2010, pp. V2-36-V2-40, doi: 10.1109/ICSTE.2010.5608758.
- J.A. Pow-Sang, A. Nakasone, R. Imbert and A. M. Moreno, "An Approach to Determine Software Requirement Construction Sequences Based on Use Cases," 2008 Advanced Software Engineering and Its Applications, Hainan, China, 2008, pp. 17-22, doi: 10.1109/ASEA.2008.33.
- C. E. Onime and J. O. Uhomoibhi, "Using interactive video for on-line blended learning in engineering education," 2013 2nd Experiment@ International Conference (exp.at'13), Coimbra, Portugal, 2013, pp. 128-132, doi: 10.1109/ExpAt.2013.6703044.