Research Article

Unveiling Area Estimation through Newton's Forward Divided Difference Method

Nurzalina Harun[1,], Zubaidah Sadikin[2], Che Nurul Hanna Che Sulaiman[3], Nur Hamiza Basri[4], Mohd Agos Salim Nasir[5] and Sidik Rathi[6,*]

1 Universiti Teknologi MARA Shah Alam; nurzalina@tmsk.uitm.edu.my; 0000-0003-0477-580X
2 Universiti Teknologi MARA Shah Alam; zubaidah1590@uitm.edu.my; 0000-0003-3967-5243
3 Universiti Teknologi MARA Shah Alam; 2023148909@isiswa.uitm.edu.my; 0009-0003-8235-7619
4 Universiti Teknologi MARA Shah Alam; 2023115405@isiswa.uitm.edu.my; 0009-0003-4696-6492
5 Universiti Teknologi MARA Shah Alam; mohdagos066@uitm.edu.my; 0000-0002-8761-4595
6 Universiti Teknologi MARA Shah Alam; sidik8423@uitm.edu.my; 0000-0002-6408-3920
* Correspondence: sidik8423@uitm.edu.my; +60193693140

Abstract: *In this era of globalization and technological advancements, Malaysia stands out as a vibrant hub for innovations and technology, along with an entrepreneurial spirit. However, it appears that there will be occasions when the creation of software demands resources such as money, time, and workforce to provide quality to the user, but it is unlikely to be able to reach the ideal evolution. In this research study, a potentially novel technological interface named AppleMath will use a mathematical method called Newton's Forward Divided Difference Interpolation as a framework for area estimations. Nonetheless, there have been a few difficulties in developing this operating system: The technological industry is always evolving and there are numerous enhancements available to keep up with the newest software developments. It is difficult to monitor safety breaches to secure all user's information. This is due to different malicious programs and viruses growing increasingly damaging, leaving it incredibly difficult for designers to find vulnerabilities in the application's safety breaches. Requests from users of this prospective software are constantly shifting, making it difficult to keep track of their personal preferences. Fortunately, there are answers to these problems: The program designer must stay up to date with the latest developments in the current programming advances. Additionally, creators of software must keep current with the most recent security patches to guarantee all users' information has been safeguarded. Software engineers must continually study evolving technological difficulties since needs change swiftly over time. The invention of this virtual interface has significance for the community since it provides practical applications for individuals amongst learners, particularly in the architectural and building professions, allowing them to explore while performing effectively in everyday situations. AppleMath can give a prominent platform for the public to use for educative reasons, hence achieving commercialization interest eventually.*

Keywords: Graphical User Interface; Area Approximation; Newton's Forward Divided Difference Interpolation.

DOI: 10.5281/zenodo.10924859

1. INTRODUCTION

As the globe grows progressively driven by innovative technologies, the Malaysian government acknowledges the critical role that innovation plays in guiding economic development and has made important measures to create an optimal setting for investors in technology and business owners to boost the rise of the digital economy, according to Communications and Digital Minister

Fahmi Fadzil. However, there are still challenges in developing software, and it is improbable that the perfect creation will be achieved. Additionally, the computer software industry is currently faced with pressure to implement safety precautions and eliminate vulnerabilities in software. Money is often cited as an obstacle, but research indicates an absence of understanding about the necessary resources for achieving a certain degree of confidentiality assurances (Venson et al., 2023).

Modern technology programs have been steadily influencing every aspect of our lives, providing advantages to all aspects of human interactions. The most notable addition is the Graphical User Interface (GUI), which enables users to easily discover, comprehend, and employ visual elements such as icons, controls, and menu items. In this study of research, an inventive graphical user interface, or GUI, called AppleMath will be constructed to help users determine the area of an item. The instance that will be implemented is rectangle-shaped baggage. We chose this product as a model since the user can arrange and reorganize spaces in this luggage bag, which has a maximum capacity. Using this instance, the user may also apply it to different situations, such as placing many items in a room or constructing various architectures for a building based on its measurements.

Without a user-friendly interface, users find themselves at competitive challenges when resolving technological issues linked to software programs, regardless of their skills or degree of understanding. Garnier et al. (2021) found that engaging technologies, such as GUIs, are beneficial in higher learning. Hence, this experiment will be beneficial to all sectors, particularly the engineering industry, allowing them to explore while performing effectively in everyday situations. Besides, AppleMath software can estimate an item's area using a mathematical method known as the Newton Forward Divided Difference Approach. In general, manually computing an item's area requires both patience and meticulous work. Consequently, this digital interface named AppleMath is intended to assist users in correctly approximating the computations they require.

2. METHOD & MATERIAL

When developing a computerised calculation system, we must first select an appropriate mathematical technique for attaining the desired result. In the current research, Newton's Forward Divided Difference Interpolation is one of several mathematical methods used to compute the area of an item. Newton's Divided Difference is the process of generating an extrapolated formula that also uses the Finite Divided Difference process (Biswajit Das & Dhritikesh Chakrabarty, 2016). What's more, Newton's Forward Divided Difference method is a traditional interpolation of polynomials technique that is used extensively in the field of numerical analysis. The Newton-type interpolated polynomial approach has several benefits, including a straightforward mathematical formulation and easy calculation (Zou, L. et al., 2020). Newton's Interpolating technique uses a collection of identified points, with values like x and y, to construct a function of length, $P_n(x)$ according to the Finite Divided Difference table. As stated by Arora, J. et al. (2023), the primary goal of Newton's extrapolation approach is to generate a more complex polynomial formula, which greatly reduces the possibility of approximated mistakes. Therefore, Newton's Interpolation technique is employed to regenerate the polynomial equation at the interface site at a lower computing cost.

The Newton's Forward Divided Difference Interpolation formula can be expressed mathematically as:
$P_n(x) = f[x_0] + f[x_0, x_1](x - x_0) + \cdots + f[x_0, x_1, \dots, x_n](x - x_0) \dots (x - x_{n-1})$

With correspond to the figure for the k^{th} divided difference: $f[x_0], f[x_0, x_1], \dots, f[x_0, x_1, \dots, x_n]$

Referred to: $f[x_i, x_{i+1}, \dots, x_{i+k-1}, x_{i+k}]$

Defined by the Divided Difference formula: $\frac{f[x_{i+1},x_{i+2},\dots,x_{i+k}]-f[x_i,x_{i+1},\dots,x_{i+k-1}]}{x_{i+k}-x_i}$

Furthermore, there ought to be additional programming available to assist in the development of a programmed calculating process. This study will employ MATLAB, a computer language, to support AppleMath in acquiring the beneficial attributes needed for effective development. MATLAB additionally serves as a language for coding and computer framework that is widely used by professionals, academics, and organisations globally (Reis, E. M. et al., 2022). Since MATLAB has several applications like the MATLAB App Designer, it is often viewed as an advanced tool instead of a preferred programming syntax. It allows users to build sophisticated applications with no need to have experience as an expert in software development. Consequently, the MATLAB software system will be implemented for designing AppleMath's graphical interfaces (GUIs), as well as acquiring strong data examination, control, and visualisation capabilities.

3. FINDINGS

3.1 The portrayal and data of the item: rectangle-shaped baggage.

The following section of results will concentrate on a specific item, the rectangle-shaped luggage, before proceeding to use it to provide an applicable instance to demonstrate area estimations applying the Newton Forward Divided Difference Interpolation scheme. The graphic following displays a rectangle-shaped luggage (see Figure 1), which will then be mapped onto graphing paper as shown in Figure 2 to get coordinates data at a dimension of 10cm length and 5cm width (see Table 1).

Figure 1. A rectangle-shaped baggage.

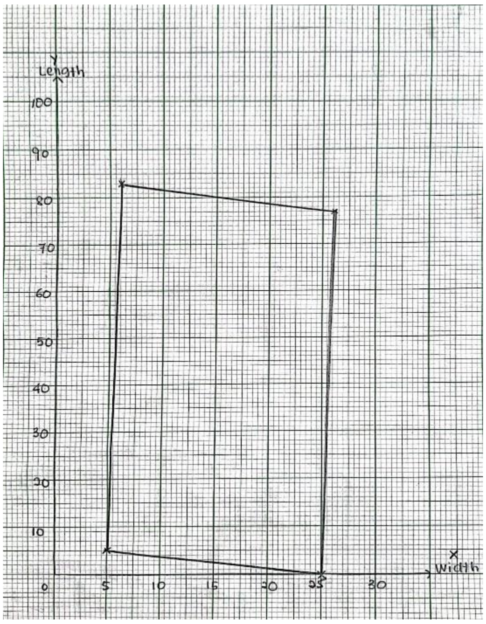

Figure 2. The mapped rectangle-shaped baggage onto graphing paper.

x	5	6	25	26
y	5	83	0	77

Table 1. The coordinates data of a mapped rectangle-shaped baggage from the graphing paper

3.2 The computations of the item using Newton's Forward Divided Difference Interpolation application.

The computations following demonstrate the way Newton's Forward Divided Difference Interpolation works when calculating the polynomial function as well as area of a rectangle-shaped items drawn on graphing paper.

x	5	6	25	26
y	5	83	0	77

Table 1.

x	y	*First Divided Difference*	*Second Divided Difference*	*Third Divided Difference*
5	5			
		78		
6	83		-4.1184	
		-4.3684		0.3898
25	0		4.0684	
		77		
26	77			

Table 2. The Divided Difference Table.

From Table 2., let x_0 = 5.

By Newton's Forward Divided Difference Interpolation Formula correspond to the figure for the k^{th} divided difference:

$f[x_0]$ = 5

$f[x_0, x_1]$ = 78

$f[x_0, x_1, x_2]$ = -4.1184

$f[x_0, x_1, x_2, x_3]$ = 0.3898

Then, the equation of a rectangle-shaped baggage:

P_n(x) = 5 + 78 (x-5) + (-4.1184) (x-5) (x-6) + 0.3898 (x-5) (x-6) (x-25)

∴ P_n(x) = 0.3898x^3 – 18.1512x^2 + 242.1914x – 800.9020

Where $Area = \int_a^b f(x)\,dx$

$$Area = \int_5^{26} 0.3898x^3 - 18.1512x^2 + 242.1914x - 800.9020\,dx$$

$$Area = \left[\frac{0.3898x^4}{4} - \frac{18.1512x^3}{3} + \frac{242.1914x^2}{2} - 800.9020x\right]_5^{26}$$

$$Area = -772.2780 - (-1672.5113)$$

$$Area = -772.2780 + 1672.5113$$

$$Area = 900.2333$$

$$\therefore Total\ approximated\ area\ of\ a\ rectangle - shaped\ baggage = 900.2333\ cm^2$$

3.3 The flow sheet and interface design of AppleMath.

The subsequent sections will demonstrate the AppleMath diagram, containing its user interfaces, as a demonstration for users to review the application's workflow and anticipate what they might require performing.

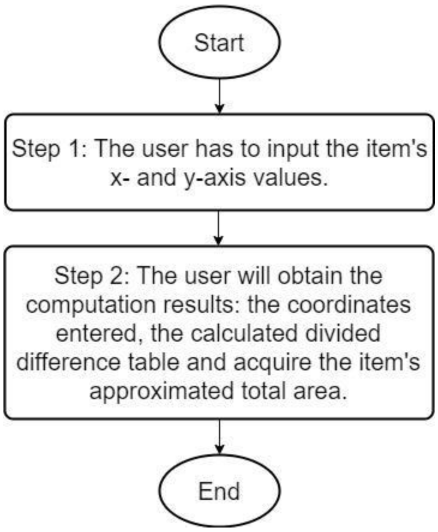

Figure 3. Flowchart of AppleMath.

Figure 4. First Designed Interface: The user will click the 'Start' button to begin.

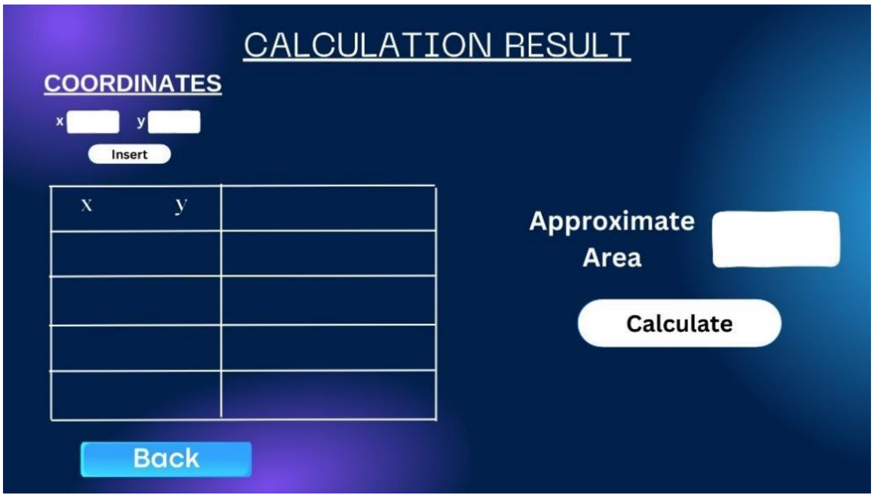

Figure 5. Second Designed Interface: The user will need to type the item's x- and y-axis coordinates into the given spaces. Then, the user will receive the mathematical results, including the coordinates that were inputted, the calculated dividing difference table, and the item's predicted total area.

The AppleMath-designed screens will subsequently be converted into interactive user interfaces (GUIs) by employing MATLAB Application Designer and related software code. We will also be looking over the area prediction algorithm with Newton Forward Division Difference, along with the MATLAB app structure and coding design.

4. Discussion

In this section, we will dive into a thorough exploration of the outcomes obtained by employing an intriguing simulator known as AppleMath to gauge an approximation of an item's area specifically a rectangle-shaped baggage. AppleMath is a simulator that was designed using the App Designer features and used C++ language for the programming in MATLAB R2023b software (*Using MATLAB With Other Programming Languages*, n.d.). Furthermore, the visual components in the graphical user interface, the implementation of the Newton Forward Divided Difference interpolation method and the callback function of each button will also be discussed to determine the functionality of the system.

4.1 Visual Components in Graphical User Interface

The visual elements were dragged from the comprehensive component library and dropped into the design view layout to align with the project's exacting specifications. The attributes such as shape, size, position and colour can be effortlessly modified to achieve optimal visual impact based on the creator's preference. The visual components of the project are shown in the Figure 6 and Figure 7.

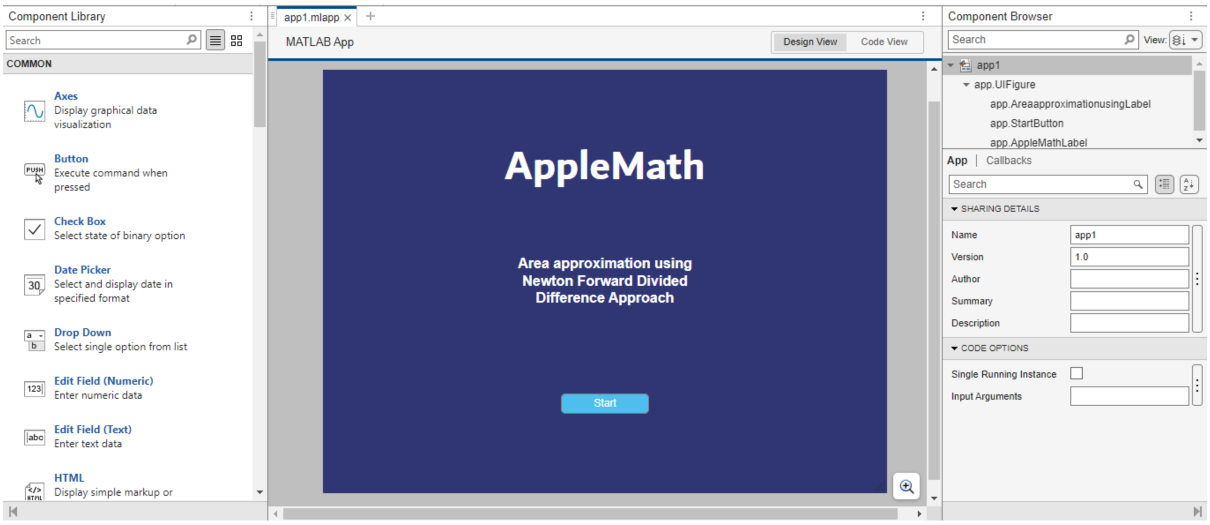

Figure 6. Visual Components in Graphical User Interface for app1.

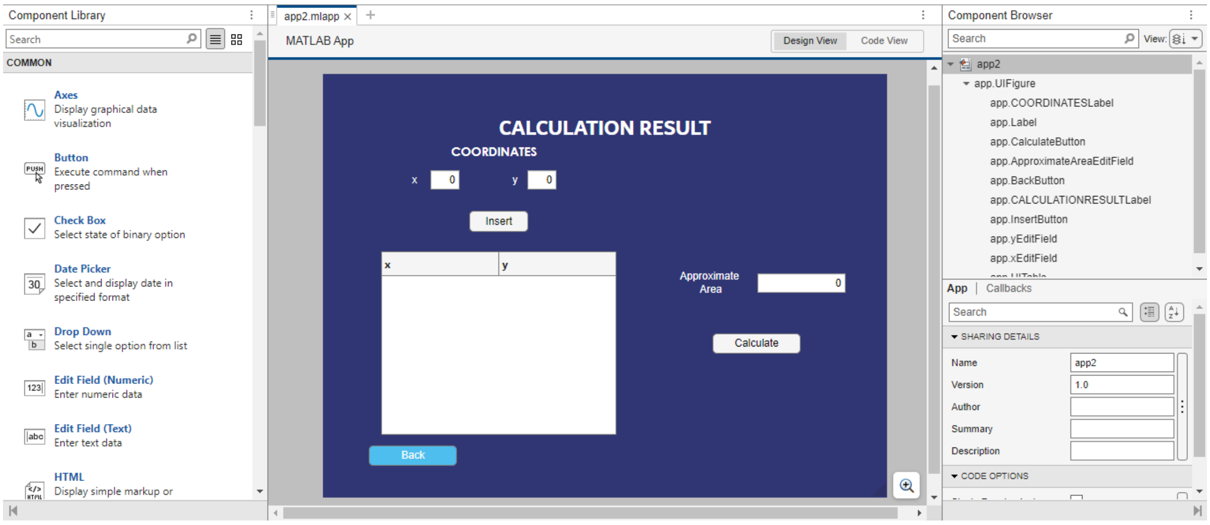

Figure 7. Visual Components in Graphical User Interface for app2.

The component library on the left side is where components are dragged based on the figure above. There are several types of components provided to choose depending on the project requirement. The components are then dropped on the design view and the label for each component will appear in the component browser on the right side of the visual. Furthermore, there are properties and a few options like adding a callback function for each component by right clicking the component label. The coding for each component will be visible in the code view. Figure 8 and Figure 9 show the coding properties for the app components.

```
% Properties that correspond to app components
properties (Access = public)
    UIFigure                      matlab.ui.Figure
    AreaapproximationusingLabel   matlab.ui.control.Label
    StartButton                   matlab.ui.control.Button
    AppleMathLabel                matlab.ui.control.Label
end
```

Figure 8. Components Command for app1.

```
% Properties that correspond to app components
properties (Access = public)
    UIFigure                   matlab.ui.Figure
    COORDINATESLabel           matlab.ui.control.Label
    Label                      matlab.ui.control.Label
    CalculateButton            matlab.ui.control.Button
    ApproximateAreaEditField   matlab.ui.control.NumericEditField
    ApproximateAreaLabel       matlab.ui.control.Label
    BackButton                 matlab.ui.control.Button
    CALCULATIONRESULTLabel     matlab.ui.control.Label
    InsertButton               matlab.ui.control.Button
    yEditField                 matlab.ui.control.NumericEditField
    yEditFieldLabel            matlab.ui.control.Label
    xEditField                 matlab.ui.control.NumericEditField
    xEditFieldLabel            matlab.ui.control.Label
    UITable                    matlab.ui.control.Table
end
```

Figure 9. Components Command for app2.

Properties are used to determine how components look, behave and function. After establishing the variable name for a component, properties are defined by explicitly adjusting their values in the code. The adjustment can occur within the component's callback functions or other pertinent sections of the code. Figure 10 shows the public properties code in this system.

```
properties (Access = private)
    DataTable % To store the data
end
```

Figure 10. Public Properties code for app2.

4.2 Defining Methods

a) Start Button

The 'Start' button in Figure 11 is programmed to switch from app1 to app2 when users click it based on the code in Figure 12.

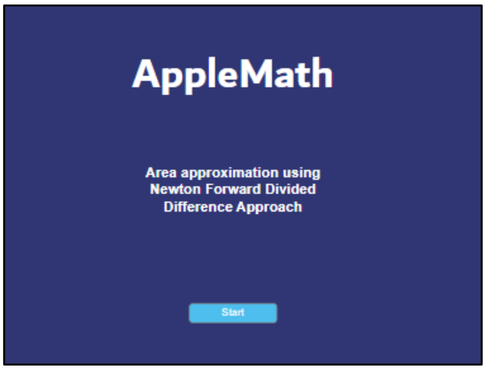

Figure 11. Main Interface.

```
% Button pushed function: StartButton
function StartButtonPushed(app, event)
    % To switch from app1 to app2
    app2
end
```

Figure 12. App1 Code View for Start button.

b) Startup Function

The startup function is a custom method where users can add to their app's code to handle initialization tasks. The code begins by assigning a number to each property and the prefix "app." is used before the field's name in the code. Figure 13 shows the startup function utilized in the project.

```
methods (Access = private)
    % Code that executes after component creation
    function startupFcn(app)
        app.DataTable = [];
    end
end
```

Figure 13. Startup Function Code for app2.

c) Insert Button

The 'Insert' button is used to insert the data for coordinate x and y based on the graphical sketch into the table and the callback has been programmed on the button. The value x entered will be stored in the app.xEditField column similarly for the y column. The data entered will automatically display in the table after clicking the 'Insert' button. Figure 14 shows the 'Insert' button and the data table in the graphical user interface. The coding for 'Insert' button is displayed in Figure 15.

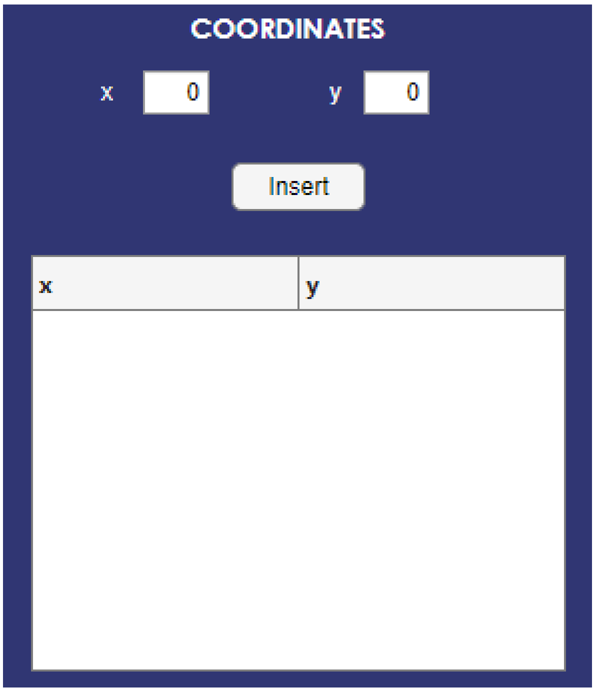

Figure 14. Insert Button and Data Table in app2.

```
% Callbacks that handle component events
methods (Access = private)

    % Button pushed function: InsertButton
    function InsertButtonPushed(app, event)
        x = app.xEditField.Value;
        y = app.yEditField.Value;
        nr = [x y];

        app.UITable.Data = [app.UITable.Data;nr]; % New row
        app.DataTable = app.UITable.Data;
    end
```

Figure 15. Coding for Insert Button in app2.

d) Calculate Button

The 'Calculate' button will show the Divided Difference table and area approximation using Newton Forward Divided Difference formula according to data entered by the user. Figure 16 and Figure 17 display the graphical user interface of Approximate Area and the code for the function.

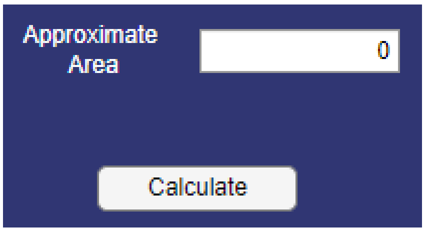

Figure 16. Calculate Button in app2.

```
% Button pushed function: CalculateButton
function CalculateButtonPushed(app, event)
    % Get the data from the table
    x = app.UITable.Data(:, 1);
    y = app.UITable.Data(:, 2);

    % Initialize the divided difference table
    n = length(x);
    dd = zeros(n, n + 1);
    dd(:, 1) = x;
    dd(:, 2) = y;

    % Calculate the divided difference table
    for j = 3:n + 1
        for i = j - 1:n
            dd(i, j) = (dd(i, j - 1) - dd(i - 1, j - 1)) / (dd(i, 1) - dd(i - j + 2, 1));
        end
    end

    % Display the divided difference table
    app.UITable.Data = dd(:, 1:n + 1);

    % Initialize the approximate area using the Newton forward divided difference
    approximate_area = 0;

    % Calculate the approximate area using the Newton forward divided difference
    for i = 2:n
        term = dd(i, 1);  % Use the first order divided difference from the ith row
        for j = 2:i - 1
            term = term * (app.UITable.Data(i, 1) - x(j));
        end
        approximate_area = approximate_area + term;
    end

    % Display the approximate area
    app.ApproximateAreaEditField.Value = approximate_area;
end
```

Figure 17. Coding for Calculate Button in app2.

e) Back Button

The 'Back' button in Figure 18 is programmed to switch from app2 to app1 when users click it based on the code in Figure 19.

Figure 18. Back Button in app2.

```
% Button pushed function: BackButton
function BackButtonPushed(app, event)
    % To switch from app2 to app1
    app1
end
```

Figure 19. App2 Code View for Back button.

4.3 Functionality Testing

Figure 20 illustrates that the phase commences upon the user's activation of the 'Start' button triggers the opening of another layout known as 'app2'.

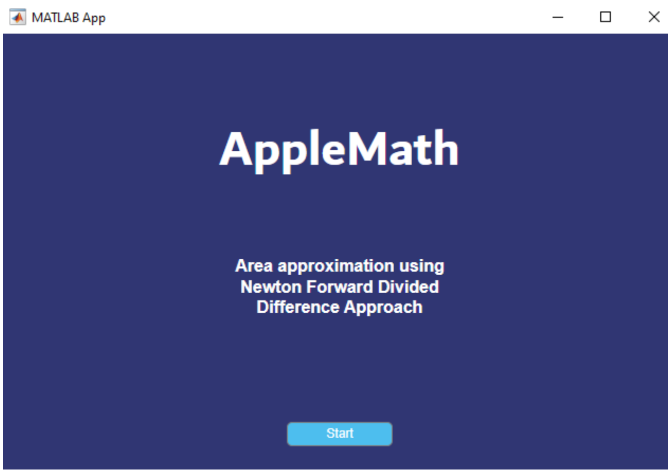

Figure 20. Layout of app1.

To evaluate the capability of app2, a set of value x and y has been entered by the user and it will be shown in the table after they press the 'Insert' button. (see Figure 21)

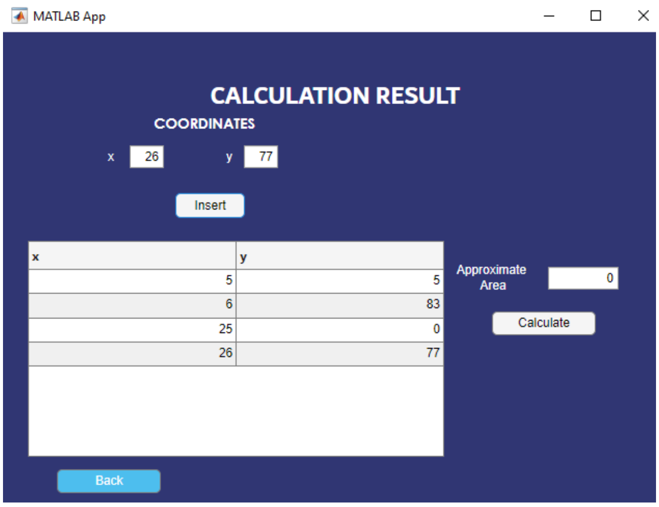

Figure 21. Data of x and y values.

Subsequently, the computation of the Divided Difference table will be executed, yielding the calculated approximation of the area using the Newton Forward Divided Difference method. This outcome appears when the user clicks the 'Calculate' button. Lastly, users can smoothly return to the previous layout by simply pressing the 'Back' button. (see Figure 22)

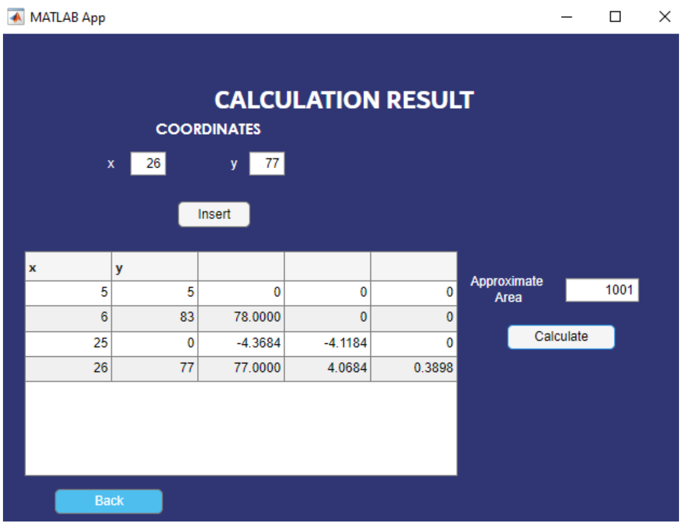

Figure 22. Calculation result of Divided Difference table and Approximation area.

5. Conclusion

In conclusion, it is critical that continual study be conducted by many researchers, with an emphasis on improving and expanding faultless software development processes. The creation of this breakthrough system, AppleMath, would give an organised result from user-friendly computing of an item's area utilising Newton's Forward Divided Difference Interpolation algorithm. In future versions, AppleMath will act as an initial foundation for advancements in software that allow for different types of mathematical analysis in the setting of practical problem-solving. Furthermore, this prospective software may render assignments accessible to regular individuals, allowing even those with less advanced knowledge to operate basic apps using a user interface (GUI). The findings from the study will widen the Newton Forward Divided Difference Interpolation techniques for applications with a focus on learning and understanding acquisition.

Acknowledgments: We are determined to acknowledge our respected lecturers for assisting us in understanding more about this subject matter, whether in person or via the Internet. We also appreciate other pupils enrolled in this educational endeavour for their assistance and encouragement. Additionally, we wish to express our gratitude to the competition's organizers for allowing us the opportunity to participate in this Virtual Innovation Competition to foster the values of creative thinking and craftsmanship in every industry as well as showcase our inventions and concepts via the provided digital medium.

References

7 Reasons Why Software Development is So Hard. (2021, October 1). GeeksforGeeks. https://www.geeksforgeeks.org/7-reasons-why-software-development-is-so-hard/

Areas by integration. (n.d.). https://www.rit.edu/academicsuccesscenter/sites/rit.edu.academicsuccesscenter/files/documents/math-handouts/C7_AreasbyIntegration_BP_9_22_14.pdf

Arora, J., R, R. K., Sathiyaraj R, & G S Pradeep Ghantasala. (2023). Securing web documents by using piggybacked framework based on Newton's forward interpolation method. Journal of Information Security and Applications, 75, 103498–103498. https://doi.org/10.1016/j.jisa.2023.103498

Das, B., & Chakrabarty, D. (2016). Newton's Divided Difference Interpolation formula: Representation of Numerical Data by a Polynomial curve. International Journal of Mathematics Trends and Technology, 35(3), 197–203. https://doi.org/10.14445/22315373/ijmtt-v35p528

Finding areas by integration. (n.d.). https://www.mathcentre.ac.uk/resources/uploaded/mc-ty-areas-2009-1.pdf

Garnier, H., Gilson, M., Muller, H., & Chen, F. (2021). A new Graphical User Interface for the CONTSID toolbox for Matlab. IFAC-PapersOnLine, 54(7), 397–402. https://doi.org/10.1016/j.ifacol.2021.08.392

Malaysia a dynamic hub for technology and innovation - Fahmi. (n.d.). The Star. https://www.thestar.com.my/business/business-news/2023/07/10/malaysia-a-dynamic-hub-for-technology-and-innovation---fahmi

Reis, E. M., Gralha, C., & Monteiro, M. P. (2022). Surveying communities of users of MATLAB and clone languages. Journal of Computer Languages, 73, 101170–101170. https://doi.org/10.1016/j.cola.2022.101170

Reuters. (2023). Malaysia – Driven by digital evolution. Reuters. https://www.reuters.com/plus/malaysia-driven-by-digital-evolution

Santiago, V. A. de, Özcan, E., & Balera, J. M. (2022). Many-objective test case generation for graphical user interface applications via search-based and model-based testing. Expert Systems with Applications, 208, 118075. https://doi.org/10.1016/j.eswa.2022.118075

Stackoverflow. (2022). What is MATLAB good for? Why is it so used by universities? When is it better than Python? https://stackoverflow.com/questions/179904/what-is-matlab-good-for-why-is-it-so-used-by-universities-when-is-it-better-th

Using MATLAB with Other Programming Languages. (n.d.). MATLAB & Simulink. https://www.mathworks.com/products/matlab/matlab-and-other-programming-languages.html

Venson, E., Boehm, B., & Clark, B. A. (2023). The effects of required security on software development effort. Journal of Systems and Software, 111874–111874. https://doi.org/10.1016/j.jss.2023.111874

Wen Jun Hong, Tao Li Liu, & Bi Feng Guo. (2020). Design and Development of Innovative Software for Student Union. Procedia Computer Science, 166, 93–98. https://doi.org/10.1016/j.procs.2020.02.027

Zou, L., Song, L., Wang, X., Weise, T., Chen, Y., & Zhang, C. (2020). A New Approach to Newton-Type Polynomial Interpolation with Parameters. Mathematical Problems in Engineering, 2020, e9020541. https://doi.org/10.1155/2020/9020541