White-Simulation

Abdulrahman Shashtari & Mahmoud Barmawi

January 2025

Acknowledgment

All praise and gratitude are due to Allah, the Lord of the Worlds, and may peace and blessings be upon the most honorable of prophets and messengers. By the grace and mercy of Allah, we have successfully and distinctively completed our first graduation project. We extend our heartfelt thanks to Allah Almighty for the countless blessings He has bestowed upon us. This project's success is a testament to His generosity and favor.

We express our profound gratitude to our beloved parents for their unwavering support and encouragement throughout this journey. Their motivation and steadfast belief in us gave us the strength to persevere, work diligently, and remain hopeful in achieving our dreams. Your love resides forever in our hearts.

We also extend our sincere thanks to our esteemed supervisor, Dr. Asmaa Afeefi, for her invaluable support and guidance throughout every stage of this project. Her insightful advice and constructive feedback were instrumental in the project's success and development. Words cannot fully convey the depth of our gratitude for her unwavering assistance, which has greatly enriched our knowledge and experience. Thank you, Dr. Asmaa, for your continuous encouragement and dedication.

Contents

Abstraction

Simulation plays a key role in education as it offers interactive place to explore and understand complex concepts. It allows learners to visualize and experiment with concepts to make them more understandable. This will make learning easier and more enjoyable, especially in fields like physics, mathematics, and computer science. So, we developed a desktop application designed for interactive simulations, note-taking, and sharing notes.

The application includes a group of physics and math simulations, such as a pendulum, additive wave functions (sine and cosine), and gravity, along with algorithm visualizations like the bubble sort algorithm. In addition to simulations, the application offers a whiteboard where users can draw, annotate ideas, and elaborate on simulations. Users can bring simulations onto the whiteboard, add notes, and make visual connections between concepts. also, the app supports saving and sharing whiteboard notes with other users, so it's like a community for sharing and explaining concepts. The main goal is to develop an application that makes understanding concepts easier through simulations and visualizations.

For development, we used Electron.js to build the desktop application, PostgreSQL for the database, and Node.js with Express.js for the backend.

Introduction

Interactive tools play an important role in modern education, creating a better learning environment by making complex concepts more intelligible and accessible. Simulations offer effective ways of visualization and experimentation. This allows students to understand complex abstract ideas through direct handling. This has been particularly more useful in subjects like physics, mathematics, and computer science, where the understanding of a theoretical concept has to be made clear with dynamic presentations and experimentation.

To this end, we have developed a desktop application that integrates interactive simulations, note-taking, and collaboration into one platform. This application will include a variety of simulations, including pendulum motion, additive wave functions, gravitational interactions, algorithm visualizations such as bubble sort. These let users play with parameters, observe results, and intuitively understand how things work.

One of the unique selling points of our application is the inclusion of a digital whiteboard in addition to the simulations. The whiteboard is useful for freehand drawing, annotation, and explanation. One can place the simulations on the whiteboard, explain with notes, and connect visuals for better understanding. This integration enables active learning wherein simulations and notes coexist in one single workspace, promoting deeper and cohesive learning.

It further allows saving and sharing of whiteboard notes, enabling the user to share in collaboration and community through shared knowledge and insight. Sharing annotated simulations and notes makes the platform a very useful tool for educators and students alike, promoting collaborative learning and discussion.

From a technical perspective, the application was developed with several modern technologies. It used Electron.js to build the desktop interface for cross-platform use and Node.js, with Express.js, for server-side operations. The application makes use of PostgreSQL as a database to securely store user data and notes shared by them.

Our aim is to provide a comprehensive tool that empowers learners for exploration, visualization, and sharing of knowledge, thus making education more interactive and joyful. With this application, we are trying to fill the gap in theoretical knowledge and practical understanding through an appealing collaborative learning environment.

Literature Review

Some of the most useful tools in education are simulations and whiteboards. Simulations help people understand difficult concepts visually and actively, especially in subjects like physics, math, and computer science. Digital whiteboards, on the other hand, are great for brainstorming, taking notes, writing explanations, and sharing ideas in real time. While many tools focus on either simulations or whiteboards, our research found no examples of a system that combines both. Most tools make users switch between running a simulation and using a whiteboard, which can interrupt learning and note-taking.

Our project solves this problem by combining simulations and a whiteboard in one place. This lets users interact with simulations while also taking notes, doing calculations, or writing their own explanation about a simulation. This new approach makes learning more engaging and helps users experiment and document their work at the same time, creating a more effective educational tool.

Our whiteboard proves to be more feature-rich and flexible than any other existing tools. OpenBoard is one such platform that focuses mainly on basic annotations and fails to customize it well for the user. Our whiteboard comes equipped with dynamic features like customizable pencils, precision erasers, and color palettes. Most of the competitors would allow static shapes and drawings but live-simulated environments are built directly into our application's canvas, so elements can easily be dragged and dropped to create an interactive visual element on the same stage or canvas. Some of the sites allow exporting to a number of formats like PDF, but this one focus on saving work as scalable vector graphics (SVG), display documents in the best quality possible. Standard whiteboards have navigation limited enough for a user to keep scrolling a huge area, because of this, we developed in the first place this feature, "Back to Content", which enables realignment to an active area in an instant. All of these, compounded against improvements made in zoom as well as undo and redo features, create a state-of-the-art whiteboard not just for academic use but also for other professional purposes.

Whereas other similar platforms are often characterized by less precision, speed, and effectiveness, like Apps on Physics, OPhysics, our tools for simulation distinguish themselves because of the high illustrative accuracy, the fast rate of knowledge application, and the strong focus on knowledge improvement. Despite the fact that numerous web resources offer simulations, which include pendulums, waves, and inclined planes, they do not possess the level of accuracy and interactivity of our implementations. For example, in the case of the projectile simulation, we not only show motion but also included the calculations.

Another feature that we provide and which is not available in other sites is the MOSFET simulation. Other examples include bubble sort and projectile motion; here, simulation design aims with identify key characteristics that would sustain student attention. For example, our projectile motion example not only depicts the corresponding trajectory but also enables user to control initial velocity, angle or gravity and more excitingly, it is a practical tutorial with calculation behind. our simulations combine science with simplicity together with ease to make sure users get correct outcomes and ease. Thus including these tools into our system, we offer the unique space for learning and forade and experimentation, which would be difficult to find in the same quality and depth elsewhere.

Methodology

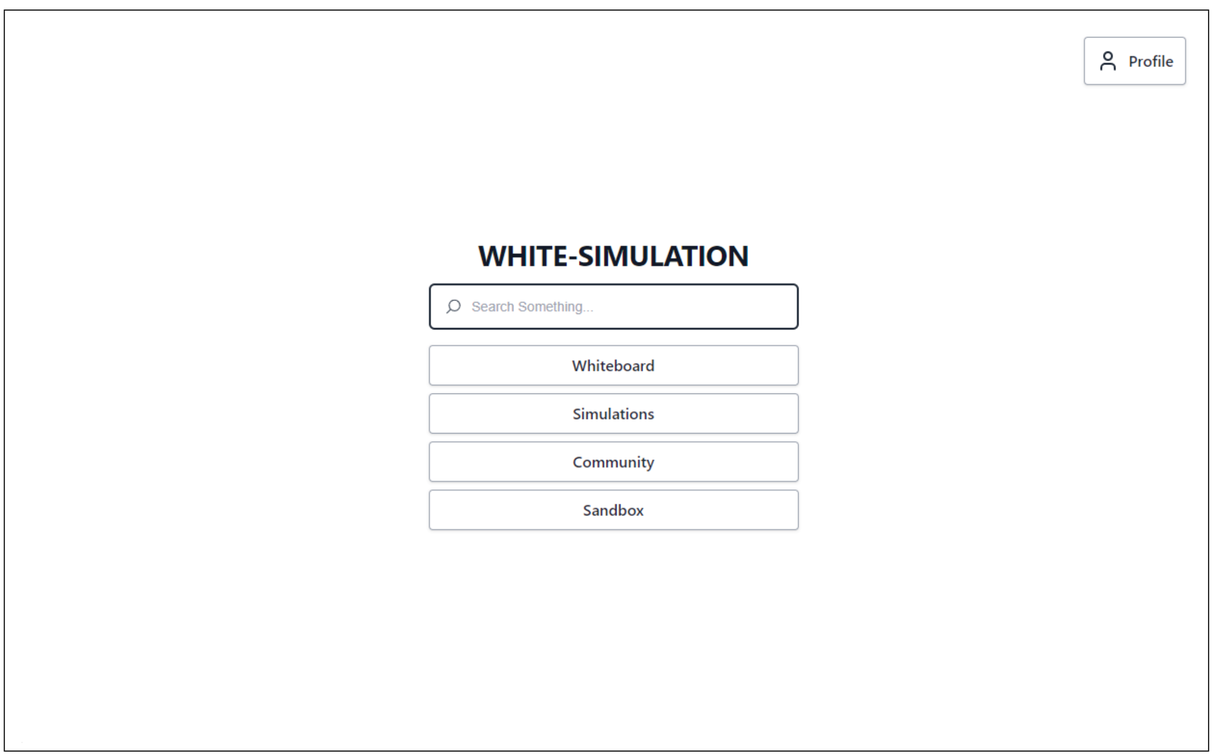
4.1 Main Page

The main page of White-Simulation is the starting point to reach the main sections of the app, and is designed to be simple and easy to use, especially for new users. The application has four main sections: **Whiteboard**, **Simulations**, **Community**, and **Sandbox**. Each section has its own button on the main page, making it easy to navigate and access the different parts of the app.

To make finding simulations quicker, we included a **search bar**. Users can type the name of a simulation, and the app will help them find it right away. This feature makes it simple to get to exactly what users are looking for.

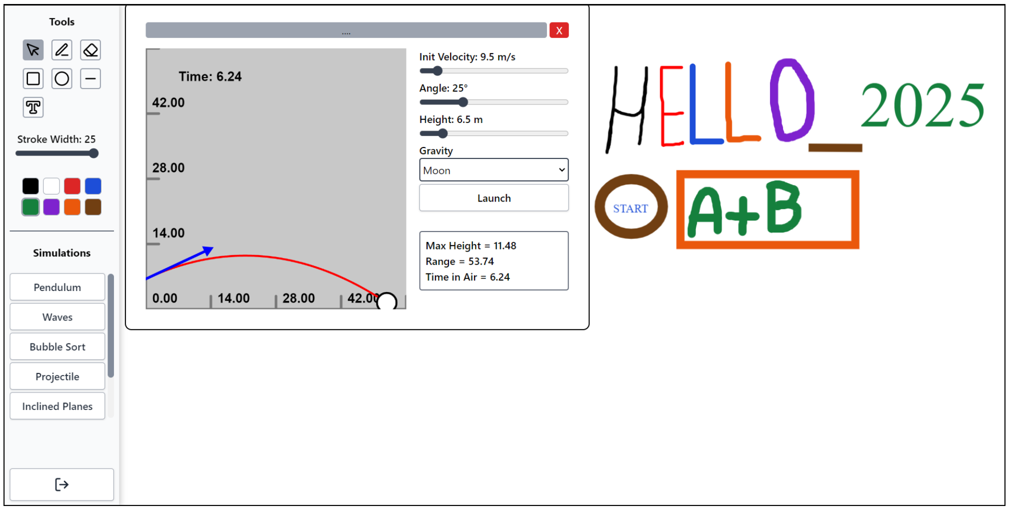
We also added a **sign-in/sign-up button**, which allows users to log in, create an account, or go to their profile with just one click.

The main page is clean and straightforward, making it easy to move between the app's features. It's designed to help the users to get started quickly and use the app as they want.



4.2 Whiteboard

The whiteboard is a core component of our application, providing users with a versatile tool for drawing, annotation, and visualizing simulations. This module was developed using **Fabric.js**, a powerful library for creating and manipulating canvas-based drawings. The following sections detail the features and functionality implemented within the whiteboard module.



Features and Functionalities

4.2.1 Object Selection

The whiteboard makes it possible to pick an item and move it around the canvas display area. This functionality allows activity such as moving, resizing, or deleting elements. The object selection was implemented using Fabric.js's built-in capabilities for object detection and manipulation.

4.2.2 Free Drawing

Using a pencil-like tool, the user has full range of movement and the color, thickness and style of the line drawn can be adjusted. It is meant to provide fluid working and speedy leading for diverse uses characterized by drawing and markup.

4.2.3 Eraser Tool

One of the most significant custom features of the whiteboard is the eraser tool. Since Fabric.js does not provide a built-in eraser, we designed and implemented this functionality from scratch. Our approach involved analyzing the JSON structure that Fabric.js uses to represent the canvas. By understanding how individual objects and paths are stored, we developed a mechanism to selectively remove parts of a drawing without affecting other elements on the canvas. This custom-built eraser tool integrates seamlessly with other drawing features, providing a smooth and intuitive user experience. The development of this tool required extensive effort to ensure compatibility with all supported drawing types, highlighting the innovative solutions applied in this project.

4.2.4 Shapes and Lines

The available shapes of the whiteboard are those that include circles, rectangles and straight lines. Such shapes can be placed, resized, rotated and colored directly on the canvas. This functionality is good for making structural diagrams.

4.2.5 Text Input

A new text tool was provided which allowed users to directly write text on any point on drawing area. The text color can be changed to their taste.

Color Customization

All tools, including free drawing, shapes, lines, and text, support color customization. It's up to users whether to choose from the carefully chosen palette of colors.

4.2.6 Simulation Integration

The whiteboard integrates with the simulation feature, allowing users to click on a simulation from a side panel. Upon selection, the simulation appears on the canvas, enabling interactive visualizations. This integration was implemented using **dnd-kit**, a drag-and-drop library that simplifies the process of moving elements between components.

4.2.7 File Management

- Save Locally and on Cloud: Concerning convenience, users can store their work permanently locally or in cloud for remote retrieval. And another feature is documents can be saved in SVG format.
- Cloud Privacy Options: When storing in cloud, users can decide to work with private files or public files depending on the discretion required.
- Load Files: Documents and media files created locally are easily retrievable and can be imported back into the whiteboard for purposes of continuity of use by the user.

4.2.8 Undo Redo

An undo-redo functionality was incorporated to enable users to reverse or redo the actions taken on work done. This feature also increases people's productivity because they have some safety harness to work with.

4.2.9 Navigation and Zoom

- Zoom In and Zoom Out: The canvas can also be zoomed in and out in order to capture features on the whiteboard or have an aerial view of the board as a whole.
- Pan Canvas: It is also possible to move the view of the canvas using the space button with the right mouse button.

4.2.10 Dynamic Context Features

If a user moves the cursor outside the content area of the whiteboard, then a back to content button is displayed. This feature helps the user to get back to the area they work most is a simple way without having to look for it yourself.

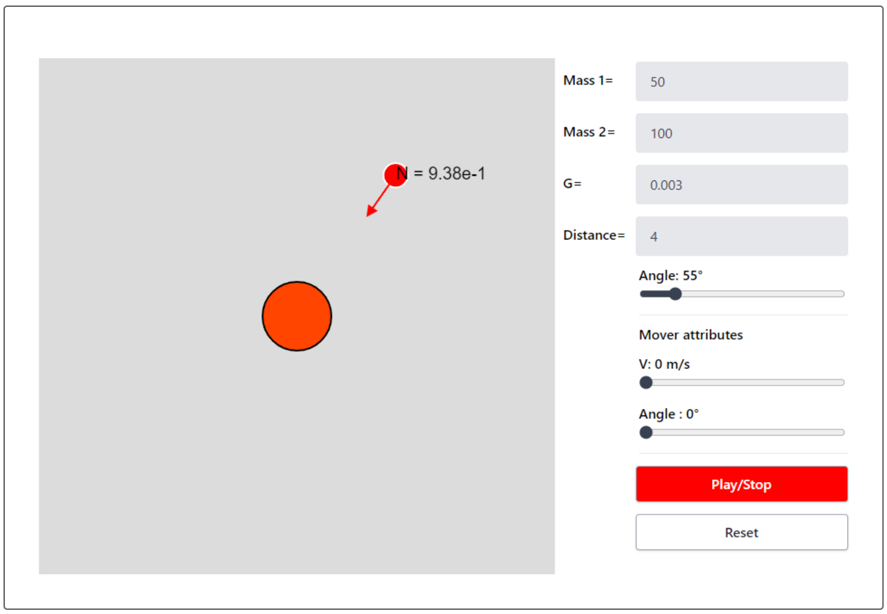
4.3 Simulations

In the White-Simulation project, simulations are important part for enabling users to visualize and interact with complex concepts in physics, mathematics, and computer science. users can experiment with real-world scenarios, enhancing their understanding and problem-solving skills.

To build these simulations, we used **p5.js** as the rendering tool, ensuring a smooth and visually engaging experience. All simulations are fully interactive, allowing users to manipulate variables and observe the effects in real time.

4.3.1 Gravity Simulation

The gravity simulation helps users understand the fundamental principles of gravitational force and how it influences the motion of objects. By visualizing gravitational interactions, making understanding the Newton's law of gravitation easy.



Features

Users can set the mass for both the attractor object and the mover object (which gets attracted). They can also configure the gravitational constant (G) and define the starting position of the mover based on its distance and angle relative to the attractor. Additionally, users have the option to assign an initial velocity to the mover, specifying both its magnitude and direction, which somehow can simulate the basic solar movement.

Implementation

The simulation is based on Newton's law of universal gravitation:

$$F = G\frac{m_1 m_2}{r^2}$$

To determine the initial position of the mover object based on its distance (d) and angle (θ) relative to the attractor object, the following formulas are used:

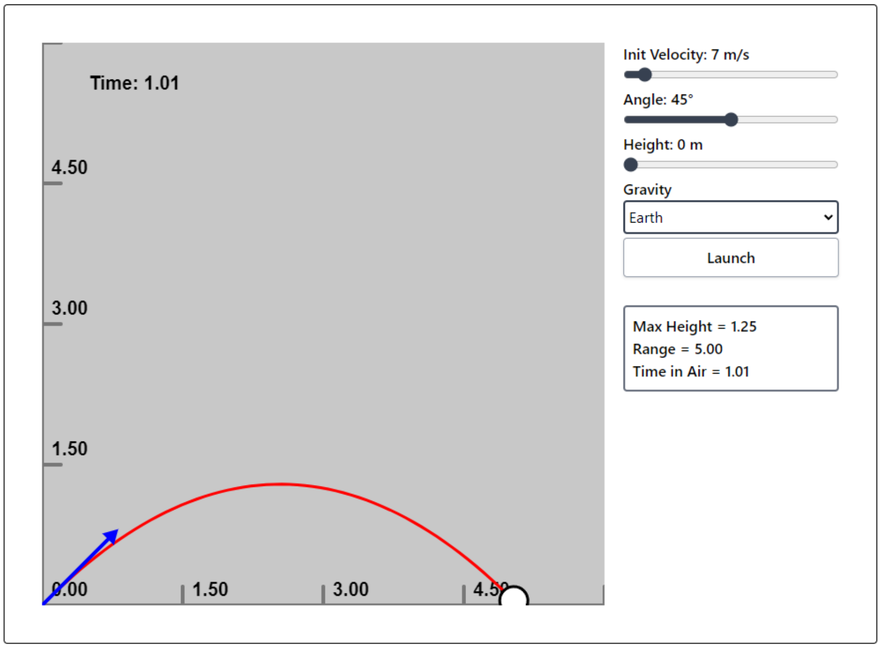
$$x_{\text{mover}} = d \cdot \cos(\theta) + x_{\text{attractor}}$$

$$y_{\text{mover}} = d \cdot \sin(\theta) + y_{\text{attractor}}$$

For the simulation purposes, we add a scale to set that 1 meter is equal to 50 pixels. Therefore, the distance d is multiplied by 50 to convert real-world units to the pixel-based coordinate system used in the simulation.

4.3.2 Projectile Motion Simulation

The projectile motion simulation helps users understand the principles of projectile dynamics, showing how initial velocity, angle, and gravitational force affect the trajectory of a moving object. This simulation makes it easier to visualize the parabolic path of projectiles.



Features

Users can set the initial velocity and angle of the projectile, as well as the height from which it is launched. Additionally, users can choose the gravitational force based on different celestial bodies: Earth, Mars, or the Moon. The simulation then calculates and displays the key outputs, such as the maximum height, the range (horizontal distance), and the time the projectile spends in the air.

Implementation

The simulation is based on the equations of motion for projectile dynamics. The horizontal and vertical components of the motion are treated independently, and gravity is applied only to the vertical component. The key formulas used in the simulation are:

For horizontal motion:

$$x = v_0 \cdot \cos(\theta) \cdot t$$

For vertical motion:

$$y = h + v_0 \cdot \sin(\theta) \cdot t - \frac{1}{2} g t^2$$

Where: - v_0 is the initial velocity, - θ is the launch angle, - t is the time of flight, - g is the gravitational acceleration (dependent on the celestial body chosen), - h is the initial height of the projectile.

The values of g are set according to the selected celestial body:

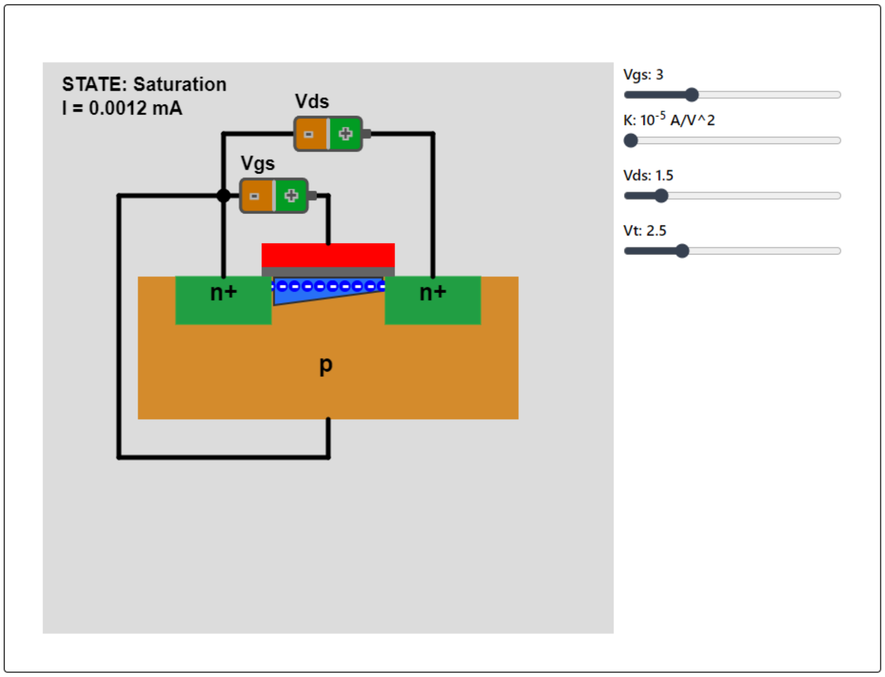
- For Earth: $g = 9.81\,\text{m/s}^2$,
- For Mars: $g = 3.71\,\text{m/s}^2$,

- For the Moon: $g = 1.62\,\mathrm{m/s^2}$.

The results of the simulation include the **maximum height**, **range**, and **time in the air**, calculated using standard projectile motion equations. These outputs provide users with an intuitive understanding of how the projectile behaves under different conditions.

4.3.3 NMOS MOSFET Simulation

The NMOS MOSFET simulation helps users to explore and understand the behavior of an NMOS transistor under different operating states. By visualizing the channel formation and the movement of electrons between the source and drain. and shows MOSFET operation states: cutoff, linear, and saturation.



Features

Users can configure the following transistor parameters: Gate-to-source voltage (V_{GS}), Drain-to-source voltage (V_{DS}), Threshold voltage (V_T), Transconductance parameter (K)

Based on these inputs, the simulation determines The operating state of the MOSFET: cutoff, linear, or saturation. and The drain current (I_D) in milliamps.

Additionally, The simulation shows the drain source channel and current flow:

- linear region: conductive channel forms, allowing current flow between the source and drain.
- saturation region: pinch-off point occurs in the channel near the drain.
- Electron movement between the source and drain is animated for every states (in cut off the current stop).

Implementation

The simulation relies on the standard current and state equations of NMOS transistor. The region of operation is determined as follows:

1**Cutoff Region** ($V_{GS} < V_T$):

$$I_{DS} = 0$$

In this state, the transistor is off, and no current flows between the source and drain.

2. **Linear Region** ($V_{GS} > V_T$ and $V_{DS} < V_{GS} - V_T$):

$$I_{DS} = \frac{K}{2} \cdot \left(2V_{DS}(V_{GS} - V_T) - V_{DS}^2\right)$$

A conductive channel forms, and the drain current increases linearly with V_{DS}.

3. **Saturation Region** ($V_{GS} > V_T$ and $V_{DS} \geq V_{GS} - V_T$):

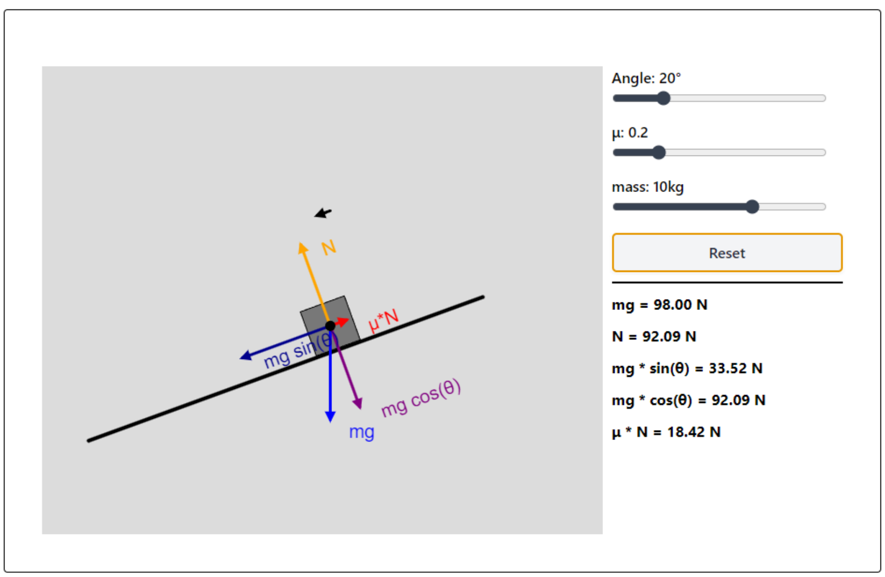
$$I_{DS} = \frac{K}{2} \cdot (V_{GS} - V_T)^2$$

In this state, the channel is pinched off near the drain, and the current is mostly independent of V_{DS}.

For visualization: the channel is displayed according to the operating region: a linear channel in the linear region and a pinch-off channel in saturation. For electron flow is only shows only on non-cut off states.

4.3.4 Inclined Plane Simulation

This simulation helps users to understand the forces acting on a box sliding on inclined surface. By visualizing forces, such as gravitational force, normal force, and frictional force, users can explore how these forces changes with different parameters.



Features

Users can configure: The angle of the inclined plane (θ), The coefficient of friction (μ) and The mass of the sliding box (m). Based on these inputs, the simulation calculates and displays: Gravitational force (mg), Normal force (N), Downward force along the plane ($mg\sin(\theta)$), Perpendicular force ($mg\cos(\theta)$), Frictional force (μN).

Implementation

The simulation is based on the following physics equations:

1.Gravitational Force:

$$F_g = mg$$

2.Normal Force:

$$N = F_g \cdot \cos(\theta) = mg\cos(\theta)$$

3.Force Along the Plane:

$$F_g \cdot \sin(\theta) = mg\sin(\theta)$$

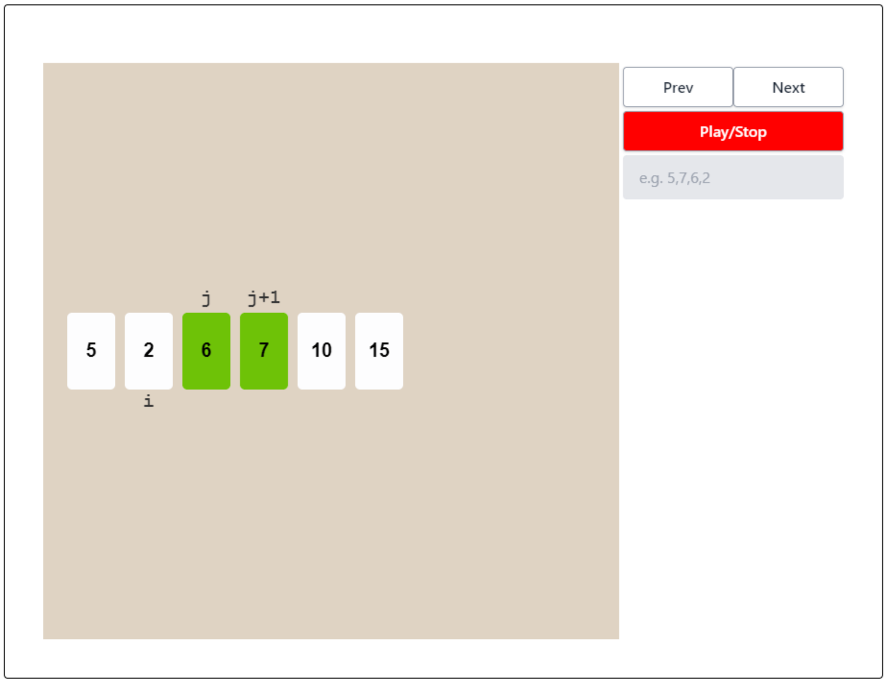
4.Frictional Force:

$$F_{\text{friction}} = \mu N = \mu \cdot (mg\cos(\theta))$$

The simulation provides real-time visual feedback, including the force vectors acting on the box. this is done in update draw loop. By playing with parameters, users can try diffrent scenarios such as low friction or steep angles, making understanding of inclined plane mechanics easy.

4.3.5 Bubble Sort Simulation

The bubble sort simulation allows users to visualize the step-by-step process of sorting an array of numbers using the bubble sort algorithm. By providing an interactive and animated representation of the sorting process.



Features

Users can input a custom array of numbers to sort then by pressing play or stop to start or stop an animation that visualizes the sorting process with two pointers: i (current iteration) and $j, j+1$ (current elements being compared).

The simulation includes visual colors to indecates the current operation:

- Red: Indicates the elements being compared need to be swapped.

- Purple: Highlights elements that have been swapped.
- green: Shows elements that do not need to be swapped.

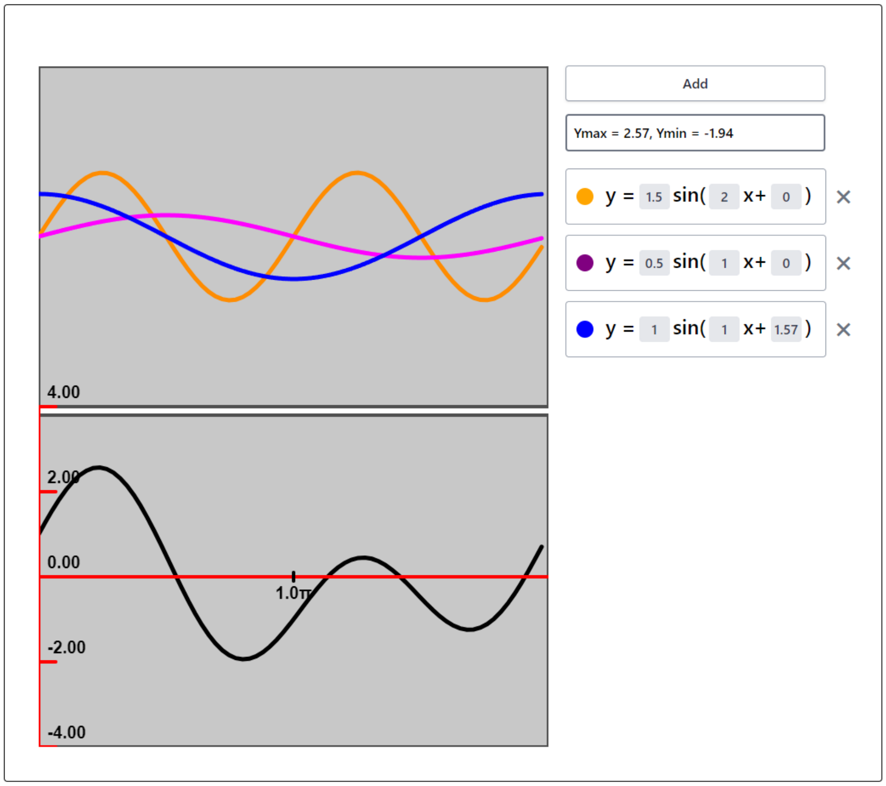
Also there is "Next" and "Prev" buttons to manually step through the sorting process instead of playing the animation. by the end of the sorting all cards will be colored green.

Implementation

The simulation uses the bubble sort algorithm with animations techniques. we made slowed the speed of animation by change the frame rate per seconds which was sufficient to make the steps readable for the user.

4.3.6 Wave Simulation

This simulation allows users to visualize the behavior of sinusoidal functions and their summation by showing the input waveforms and their sums output. This simulation helps in understanding concepts such as wave superposition, and harmonic analysis.



Features

Users can add sinusoidal functions with their parameters such as: Frequency (f), Amplitude (A), Phase shift (ϕ).

The simulation shows the visualization for each individual wave in a unique color, allowing for easy identification of each function and its drawn wave. and you can see the sum of all input waveforms as the output wave, dynamically calculated and displayed on the screen. and some results like the maximum and minimum Y-values of the output wave.

Implementation

The simulation calculates the output wave by summing all input waveforms. Each waveform is defined as:

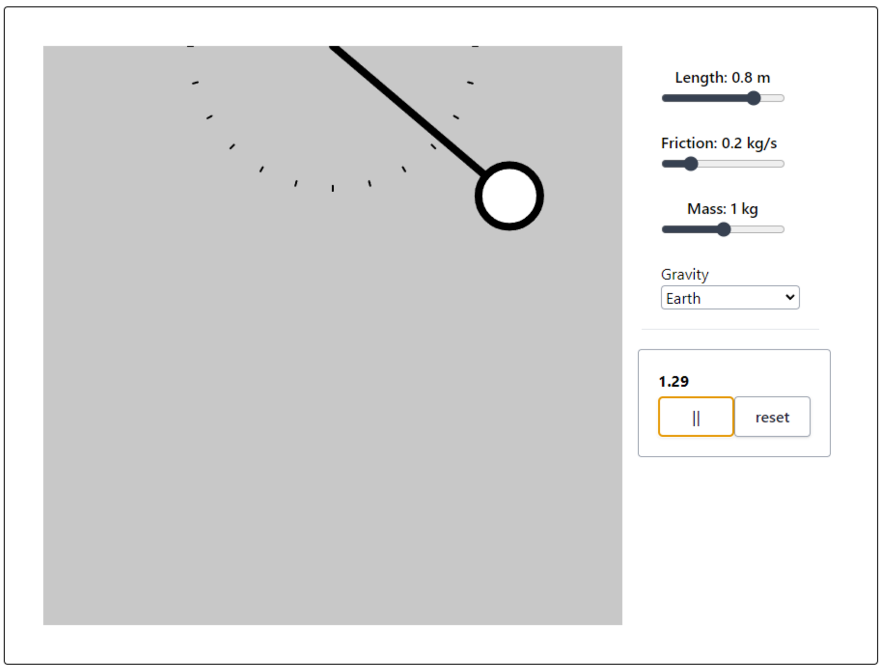
$$y(t) = Amp \cdot \sin(freq \cdot x + \phi)$$

The resultant wave is computed as:

$$y_{\text{output}}(t) = \sum_{i=1}^{n} y_i(t)$$

4.3.7 Pendulum Simulation

The pendulum simulation gives users an interactive way to understands the dynamics of a pendulum, by the principles of classical mechanics. It provides an engaging way to understand how various parameters affect the motion of the pendulum.



Features

Lets users to Adjust the pendulum parameters like modify the **length** of the pendulum rope, the **air friction** (in kg/s), and the **mass** of the pendulum bob. Also Gravity can be set to one of three options: **Earth**, **Mars**, or **Moon**, to observe the effects of different gravitational environments.

Additionally, we added interactive Control that the users could drag The pendulum manually by the mouse pointer to a specific angle and then release it.

Also there is a built-in timer allows users to **start**, **pause**, or **reset** the time. This feature can be used to measure the period of the pendulum (time taken for a full oscillation).

Implementation

The pendulum simulation is based on the dynamics of a simple pendulum, incorporating gravitational force, air friction, and user interaction.
1. Pendulum Force: The force on the pendulum is calculated using the following equation:

$$F = -\frac{g \cdot \sin(\theta)}{L}$$

where g is the gravitational acceleration, θ is the angular displacement, and L is the length of the pendulum.

2. Damping Force (Air Friction): The damping force, representing air friction, is modeled as:

$$F_{\text{damping}} = \frac{b}{m} \cdot \omega$$

where b is the air friction coefficient, m is the mass of the pendulum, and ω is the angular velocity.

3. Angular Acceleration: The angular acceleration α is determined by the net force (gravity and damping) applied to the pendulum:

$$\alpha = F - F_{\text{damping}}$$

4. Iterative Update of Angular Velocity and Displacement: The angular velocity and angular displacement are updated iteratively using the following equations:

$$\omega = \omega + \frac{\alpha}{\text{FPS}}, \quad \theta = \theta + \frac{\omega}{\text{FPS}}$$

where FPS is the frame rate.

5. Pendulum Position: The position of the pendulum in Cartesian coordinates is determined by:

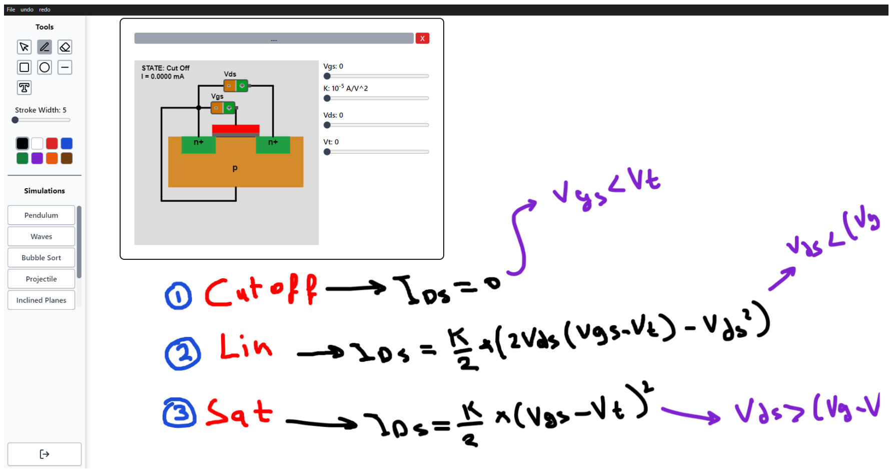
$$x = L \cdot \sin(\theta) \cdot \text{METER\_SCALE} + x_{\text{origin}}, \quad y = L \cdot \cos(\theta) \cdot \text{METER\_SCALE} + y_{\text{origin}}$$

where METER_SCALE converts meters to pixels, and $(x_{\text{origin}}, y_{\text{origin}})$ is the anchor point of the pendulum.

These formulas allow the pendulum to be dragged while accounting for friction, gravity, and user interaction. The time for a full oscillation can be tracked using a timer that can be paused and reset during the simulation.

Combining Whiteboard with Simulations

In the White-Simulation project, the integration of the whiteboard and simulations provides users with a dynamic and interactive learning environment. On the left side of whiteboard there is a list of all available simulations, When a user clicks on one of the simulations, a draggable component appears on top of the whiteboard, seamlessly launching the simulation.



This integration allows the simulation to run while the user can continue to use the whiteboard simultaneously. The whiteboard offers a space to draw and add notes, while the active simulation provides real-time feedback on the user's input, enhancing the learning experience.

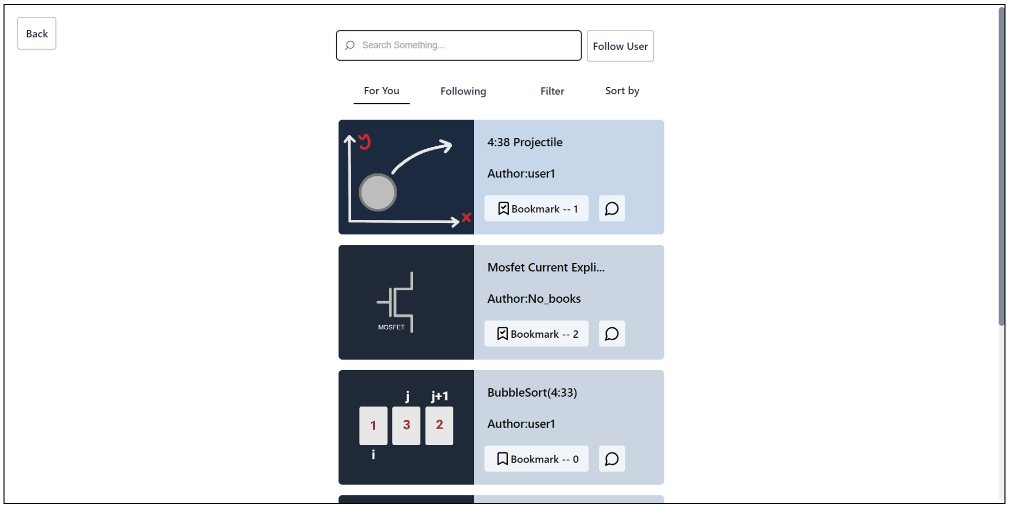
Additionally, Teachers can also write notes directly on the opened simulation, providing context or explanations, and later share these notes with students. This will make it easier to understanding complex concepts in physics, mathematics, and computer science.

Implementation

The way we used to integrate the simulation with whitebaord was with using dnd-kit library which allow us to wrap simulations with dnd draggable then drag it over dragrable area which is white board. this method worked perfectly with our needs with good performance.

4.4 Community

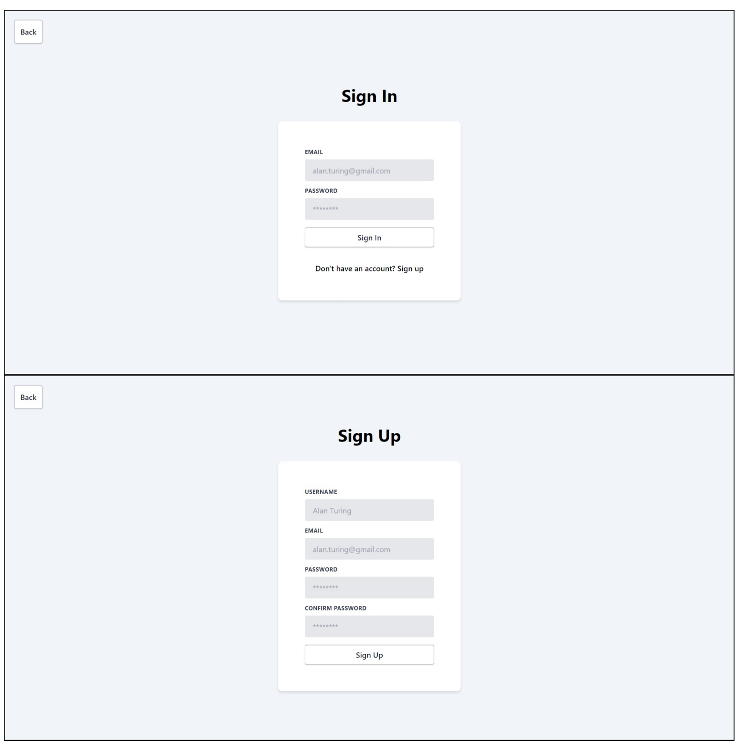
The community feature is a cornerstone of our application, designed to foster an engaging, interactive environment for users to share, explore, and collaborate on educational content. With its comprehensive set of tools and functionalities, this feature creates a dynamic and personalized experience, promoting meaningful interactions and easy access to valuable resources.



Features and Functionalities

4.4.1 User Authentication and Community Access

Account Creation and Management



To ensure a secure and personalized experience, we implemented a robust authentication system. This feature is fundamental to enabling users to participate actively in the community, as it provides the necessary framework for creating, sharing, and engaging with content.

Users must create an account to access the community's full suite of features. The registration process includes secure data handling practices, such as encrypted storage of passwords and email verification, to protect user information and prevent unauthorized access.

Authentication Features

- **Sign Up and Login:** Users can sign up with their email and password, enabling them to create an account and log in securely.
- **Session Management:** The system includes features like token-based authentication to ensure a smooth and secure user experience while maintaining session integrity.

Integration with Community Features

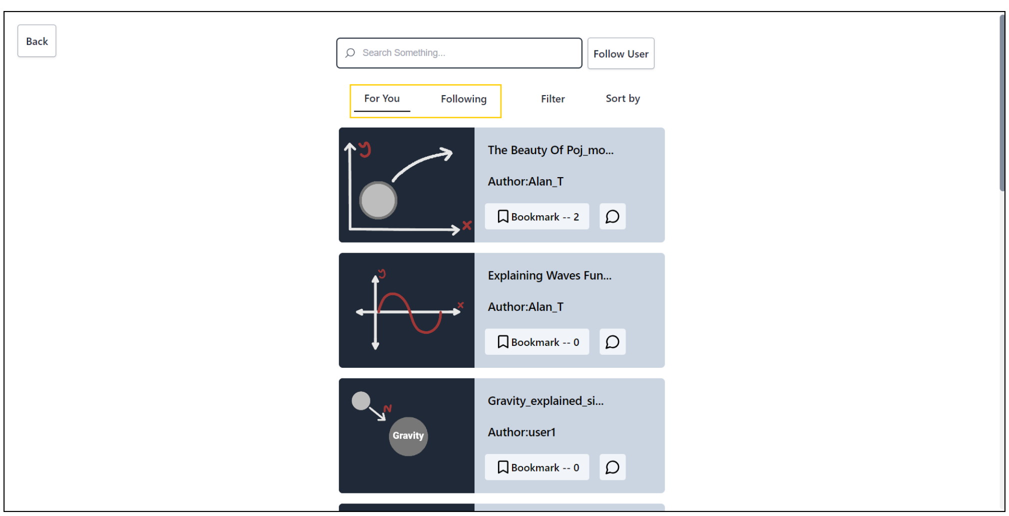
Once authenticated, users gain access to key community functionalities, including:

- Sharing posts.
- Following other users.
- Interacting with content through feedback, bookmarks, and sorting options.
- Managing personal profiles, including editing usernames, bios, and passwords.

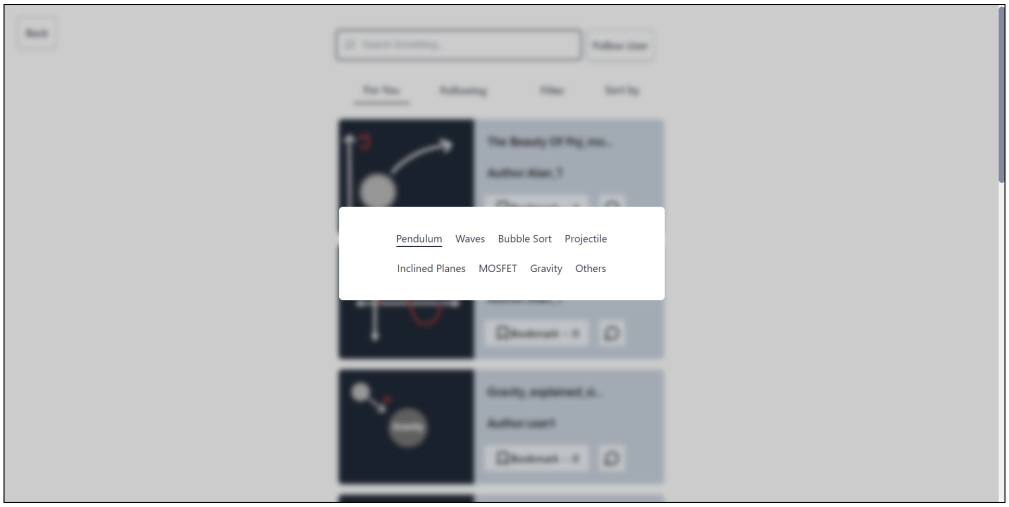
4.4.2 Post Display

Users can view posts through two primary sections:

- For You: Displays all posts available within the community.
- Following: Displays posts shared by users whom the viewer follows.



4.4.3 Filtering Options

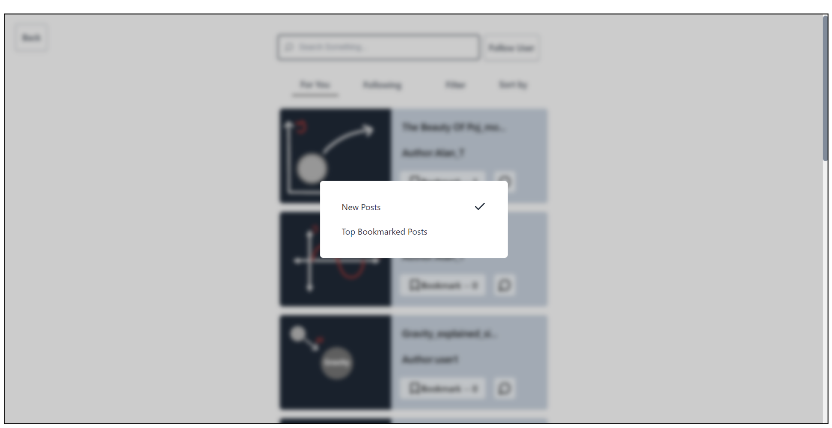


Posts can be filtered based on the type of simulation, allowing users to focus on specific topics of interest. Supported categories include:

- Pendulum
- Waves
- Inclined Planes
- Gravity
- Bubble Sort
- Projectile
- MOSFET
- Others

Filtering functionality is also available within individual user profiles, enabling targeted searches through their posts. This feature is particularly useful for users with extensive content.

4.4.4 Sorting Options

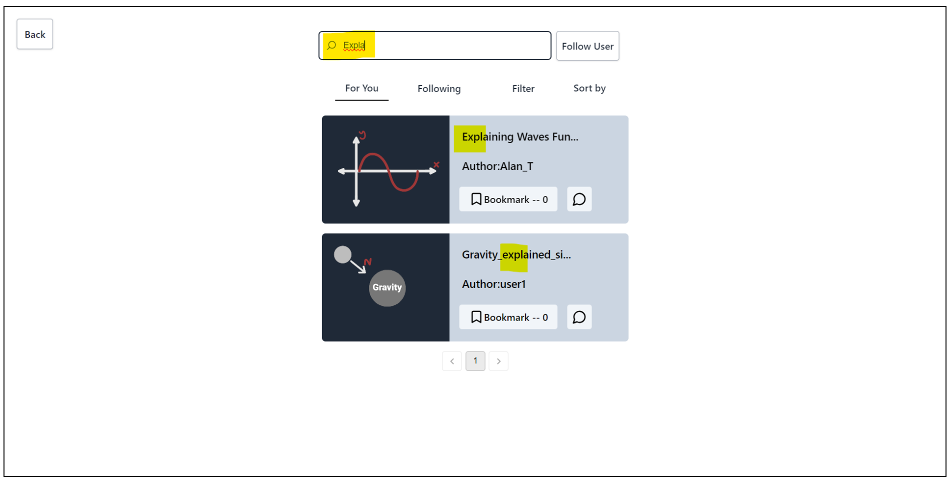


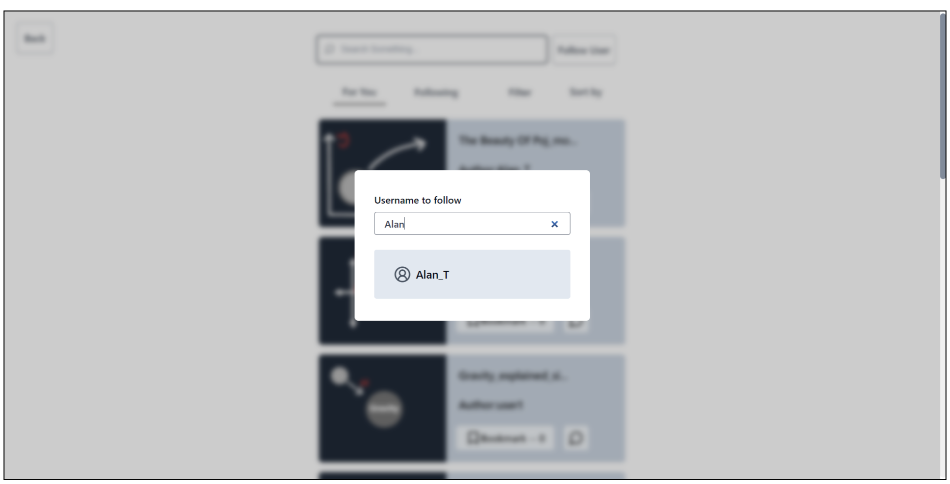
Users can sort posts using two criteria:

- New Posts: Displays the most recent posts first.
- Top Bookmarked Posts: Displays posts with the highest number of bookmarks, reflecting their popularity.

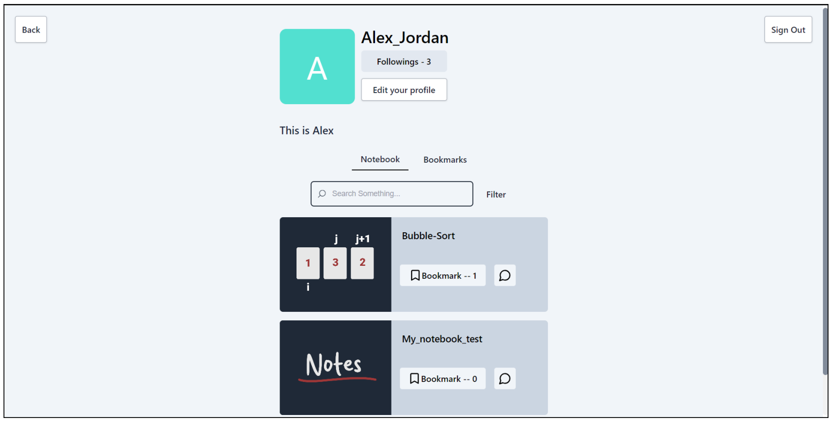
4.4.5 Search and Follow Features

- Post Search: Users can search for posts by name, facilitating quick access to specific content.
- Follow Users: Users can follow others by clicking a dedicated button. A text field allows searching for users to follow. Within individual profiles, users can follow or unfollow as desired. A follower list is available to track who follows the user.





4.4.6 Profile Management

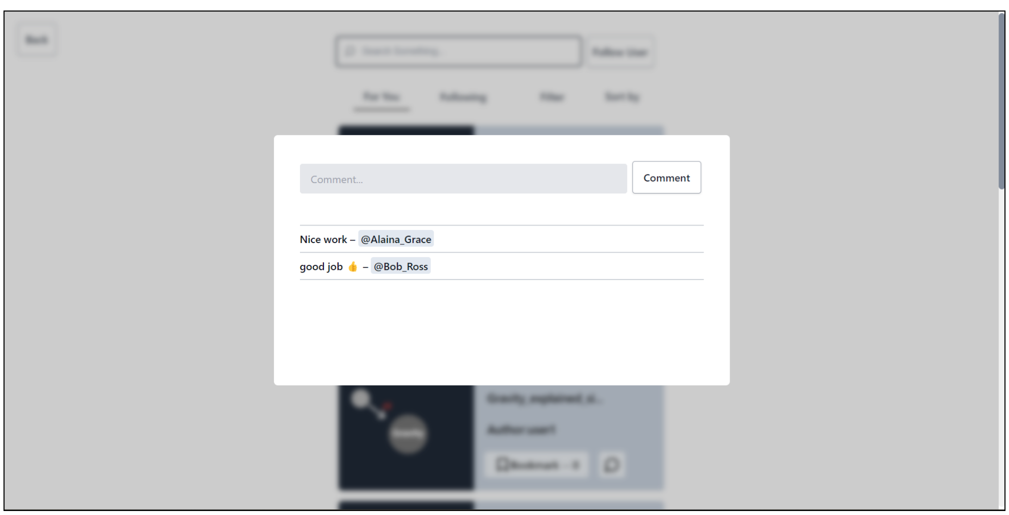


Users can manage their profiles with the following options:

- View their own posts and bookmarks for easy access and management.
- Edit username and bio.
- Update password for enhanced security.
- View the list of users they are follow.

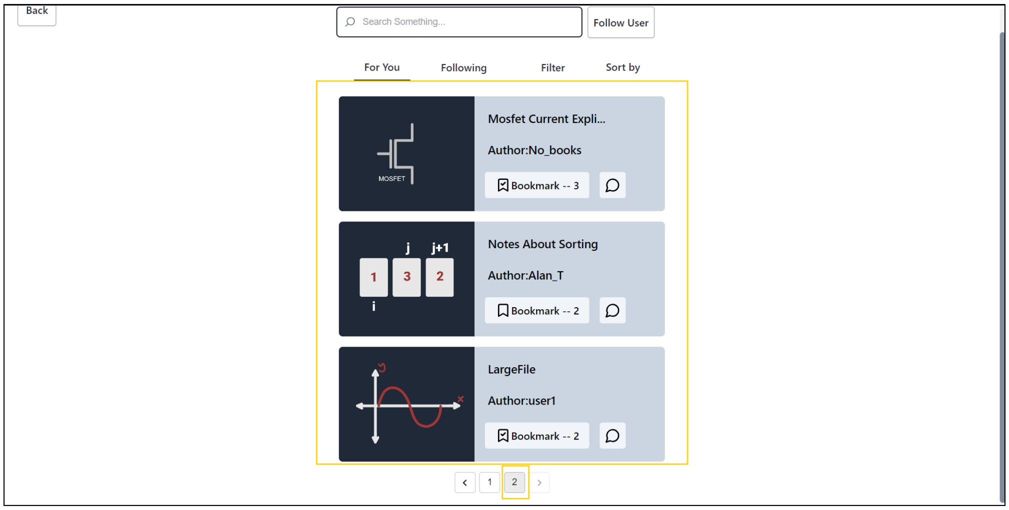
4.4.7 Post Interaction and Feedback

- Clicking on a post opens the associated whiteboard, displaying its content and any simulations included. Users can view simulations and their explanations directly.
- Users can bookmark posts for later reference. Bookmarked posts appear in the user's personal profile.
- Feedback can be provided on posts through a dedicated feedback feature, fostering engagement and improvement.



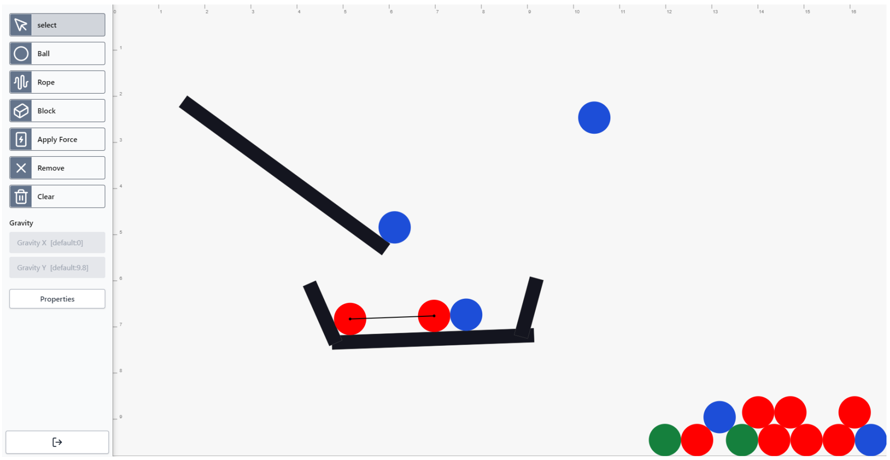
4.4.8 Pagination

To enhance performance and usability, pagination is implemented throughout the community, ensuring smooth navigation across large datasets. These community features create a dynamic and engaging environment for users, encouraging sharing, learning, and collaboration.



4.5 Sandbox: Physics Playground

The Sandbox is the fun part of white-simulation project which allows users to interact with a physics engine built using `matter.js`. This feature provides a space for experimenting with physics concepts through a variety of tools.



Features

The Sandbox includes the following interactive tools:

- **Select:** Allows users to drag and reposition objects within the sandbox using the mouse.
- **Ball:** Adds a ball as a physics object to the sandbox, which interacts with other objects based on the physics engine.
- **Rope:** Connects two objects, such as two balls, with a rope to demonstrate tension and connection in physics.
- **Block:** Creates static block barriers that act as obstacles or boundaries within the sandbox.
- **Apply Force:** Enables users to apply force on objects (balls) by clicking on a object will applies the specified force, demonstrating the effects of forces in motion.
- **Remove:** Deletes selected objects, such as balls or blocks, by clicking on them.
- **Clear:** Clears the entire sandbox, resetting the space for new experiments.
- **Gravity Fields:** Two input fields allow users to modify the gravity vector (x, y) within the sandbox, changing the direction and magnitude of gravitational force.
- **Properties Button:** Adds additional customization for specific tools:
  - *Ball:* Users can set the color and initial speed of the ball at the time of insertion.
  - *Apply Force:* Users can set the magnitude and direction of the force to be applied.

The sandbox space includes a Cartesian plane (resembling a ruler) to observe motion on objects and interactions. The physics engine simulates realistic behaviors such as collisions, gravity, and tension, allowing users to experiment freely and observe dynamic outcomes.

This flexible environment provides an engaging way to explore physics concepts and test ideas, making it a fun tool for experimentation.

Result

The interactive desktop application developed successfully integrates multiple simulations such as pendulum motion, additive wave functions, gravitational interactions, and algorithm visualizations like bubble sort. furthermore, the inclusion of the whiteboard right within the application offers users a means through which they can draw, write and explain as they wish. The simulations and the whiteboard are in one place, so users can investigate parameters, watch outcomes, and learn more about how it operates. The application also allows users to save and share notes so that the application promotes the sharing of notes between users.

Conclusion

In this project, it also showed that to maximize the learning capacity of the users, it is effective to include both the simulation in the form of games and note-taking area in the same application. The connection between the digital whiteboard and simulations provides more engaging and interactive way of teaching. It is for this reason that the ability to share annotated notes through the application supports collaborative learning ensuring that students and teachers benefit from it. Hire people with specific skill sets such as Electron.js, Node.js, Express.js and PostgreSQL are used to offer a reliable solution. Further development may cover enlarging the number of simulations, making their management and interaction with them more engaging and the user interface more friendly to achieve even better training.