International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

A comprehensive approach to Virus Mutation Study by Simulation

Arpita Santra[1], *Anunay Das Bairagya*[2], *Soumen Goswami*[3], *Apurbo Mallick*[4], *Divya Ghoshal*[5], *Soumik Saha*[6]

[1]Assistant Professor, [1]Department Of Electronics and Communication Engineering, Narula Institute of Technology, Agarpara, Kolkata 700109, India
[2-6]B.Tech Student, Department Of Electronics and Communication Engineering, Narula Institute of Technology, Agarpara, Kolkata 700109, India
Corresponding author: arpita.santra@nit.ac.in

ABSTRACT—

The Virus Mutation Simulation project is an interactive tool that models the evolution of viruses over time using real genomic data. Developed using Python and Streamlit, this tool enables users to customize settings such as mutation rate and population size to observe genome alterations and track evolutionary dynamics visually. Designed for both education and research, it helps students grasp virology concepts and aids professionals in analysing viral evolution and planning responses.

Keywords: mutation, vaccine, Covid-19, public health.

Introduction

Viruses are microscopic entities that depend on host organisms to reproduce, and one of their defining traits is the ability to mutate. These genetic changes, though often minor, can significantly influence a virus's behaviour, impacting how easily it spreads, how severe it is, and how well it responds to treatments or vaccines. The COVID-19 pandemic highlighted the real-world implications of viral mutations, with variants like Alpha, Delta, and Omicron altering public health responses globally. Understanding how viruses mutate is vital in fields such as virology and epidemiology, but studying these mutations in real time can be challenging and resource-intensive.

To address this, the "Virus Mutation Simulation" project [1] presents an interactive, user-friendly platform that models viral evolution over multiple generations. Developed using Python for the backend and Streamlit for visualization, the tool simulates how mutations (insertions, deletions, or substitutions) accumulate in a virus's genome. Users can customize variables such as mutation rate, generation count, and environmental pressures to observe the resulting genetic drift and adaptation. Visual outputs, such as mutation graphs and flagged changes in significant genome regions (e.g., the spike protein), help make abstract biological [2] concepts more concrete. Reports can also be downloaded for deeper offline analysis.

This simulation serves multiple purposes. For students, it simplifies the learning of complex topics, such as viral evolution and genetic variation, through hands-on experimentation [3]. Researchers can use it to model potential viral behaviours and test scientific hypotheses. Educators gain dynamic teaching aids for explaining molecular genetics, while policymakers can better understand the urgency of genomic surveillance [4]. By bridging theory and practical application, this simulation not only enhances biological education but also supports strategic planning in public health and vaccine development [5].

literature review

To ensure scientific credibility and real-world relevance, the Virus Mutation Simulation project is grounded in data and insights from reputable genomic databases [6], international health organizations, and peer-reviewed scientific research. These sources collectively provide a robust framework for accurately modelling virus mutation behaviour [7].

The NCBI Virus Variation Database serves as a foundational resource [8], offering curated viral genome sequences from pathogens like Influenza, Dengue, Zika, and SARS-CoV-2. These sequences, annotated with functional genomic features, help the simulator replicate biologically accurate mutation patterns based on real virus evolution data.

GISAIDS, a global data-sharing platform, contributes up-to-date genomic sequences of influenza viruses and SARS-CoV-2 variants. It was especially critical during the COVID-19 pandemic in tracking rapidly evolving strains like Delta and Omicron. Incorporating this dynamic data allows the simulation to model real-time mutation trends and test hypothetical scenarios under evolving conditions.

The World Health Organization (WHO) provides reports on variant classification, transmission impact, and vaccine interactions, emphasizing the importance of mutation surveillance. This aligns with the simulator's goal of promoting mutation awareness and preparedness.

Similarly, the Centers for Disease Control and Prevention (CDC) offers frameworks for genomic surveillance and public health response, which inform the simulator's precautionary evaluation logic and its potential as a training tool.

Lastly, peer-reviewed journals like Nature and databases like PubMed Central (PMC*)* offer validated models and biological interpretations of viral mutations. These references support the scientific accuracy of the simulator's algorithms, including mutation rates and genome repair models, ensuring alignment with current research in virology and computational biology.

TECHNOLOGY STACK AND TOOLS

The Virus Mutation Simulator was developed using a well-integrated technology stack that balances computational efficiency with user interactivity. The core technologies were selected for their simplicity, flexibility, and effectiveness in building scientific simulations accessible through the web.

Python is the foundation of the project, chosen for its readability, vast library ecosystem, and ease of use in data-heavy applications. It handles key operations like genome generation, mutation simulation, and tracking changes across generations through modular functions such as generate_genome(), mutate_genome(), and simulate_generations().

Streamlit is used to create the web interface, allowing users to set parameters like genome length, mutation rate, and number of generations. With interactive sliders and buttons, Streamlit makes it easy to run the simulation and view outputs in real time, all without needing front-end development skills.

Matplotlib is employed for visualizing the mutation growth across generations. It generates informative line plots showing how mutations accumulate and how precautions impact mutation trends. These visuals enhance the user's understanding of simulation outcomes.

The random module from Python's standard library simulates biological randomness in genome generation and mutation decisions, ensuring that each run produces unique and realistic results.

Additionally, the code's logical structure is well-organized into reusable functions, and the integration of logic, interface, visualization, and decision layers provides a smooth and scalable workflow. This structure makes the system intuitive, modular, and ready for future enhancements.

METHODOLOGY

As we know, our project is about a *Virus Mutation Simulation System*, which combines *Biological Modeling* with an *Interactive Visualization Interface* to simulate how viral genomes mutate over time and how these changes impact viral behaviour. Let us give a detailed breakdown of this exciting project.

A. Modules Used in the Project and Their Roles

1. Genome Initialization Module:

This module generates the initial DNA sequence (Generation 0), which serves as the baseline for the entire simulation. It randomly assembles a sequence of nucleotides—A, T, C, and G—based on the genome length selected by the user. Python's random.choices() function is used for this purpose. The original genome is displayed on the Streamlit interface for easy reference before any mutation takes place.

2. Mutation Simulation Module:

This is the heart of the simulation. It goes through each base in the genome and generates a random number to determine whether a mutation should occur, based on the user-defined mutation rate. If a mutation is triggered, the base is replaced by a different nucleotide. This function is executed repeatedly over a number of generations, storing each resulting genome in a list for further analysis.

3. Mutation Tracking Module:

To measure how the genome evolves, this module compares each new generation to the one before it, using Python's zip() function to align sequences base by base. Any difference is counted as a mutation. A cumulative total is maintained to reflect the buildup of mutations over time, which serves as the basis for graph plotting and trend analysis.

4. Visualization Module:

This module uses the *Matplotlib* library to convert numerical data into a clear line graph. The X-axis represents the generation number, while the Y-axis displays the total number of accumulated mutations. The plot is rendered within the Streamlit interface and updates in real time, offering users a visual insight into mutation dynamics.

5. Risk Evaluation & Precaution Module:

Once simulation data is complete, this module evaluates the risk based on mutation accumulation. It categorizes risk into *Low (green)***,** *Moderate (orange)*, or *High (red)***.** Corresponding precautions—like reducing the mutation rate or generations—are then recommended, mimicking real-world public health strategies.

6. Interactive Streamlit UI Module:

The Streamlit interface makes the simulation user-friendly and engaging. Sliders allow users to select genome length, mutation rate, and number of generations. A "Run Simulation" button launches the process, and results are displayed live, including generation-wise genomes, graphs, and risk assessments. Expandable sections provide deeper insights, making it ideal for both learners and researchers.

B. Step-by-Step Working of the Code

Step 1: Problem Definition and Objectives

The simulator aims to help users understand how viral genomes mutate and evolve. By adjusting variables like mutation rate and generations, users explore how different conditions affect viral behaviour.

Step 2: Initial Genome Generation

The system begins by generating a random DNA sequence using nucleotides (A, T, C, G) through a random process. The choices() function. This sequence represents the original virus and is saved for future comparisons.

Step 3: Apply Mutations Across Generations

Each generation introduces possible mutations. For every base in the genome, a random number is compared to the mutation rate. If it's lower, a new base replaces the original one. This continues for all generations, simulating natural mutation patterns.

Step 4: Mutation Comparison and Tracking

Each mutated genome is compared to the previous generation. Using zip(), base mismatches are identified and counted. The mutation totals are stored in a list to track how quickly or slowly mutations are accumulating.

Step 5: Visualization of Mutation Trends

The stored mutation data is plotted using *Matplotlib*. As the generations progress, users see how mutations grow or stabilize, giving a clear trend line. This live chart is embedded in Streamlit for instant interpretation.

Step 6: Risk Evaluation and Suggestions

At the end of the run, mutation totals are analyzed. Based on predefined thresholds, the risk is classified and displayed along with safety suggestions like decreasing the mutation rate or limiting generations.

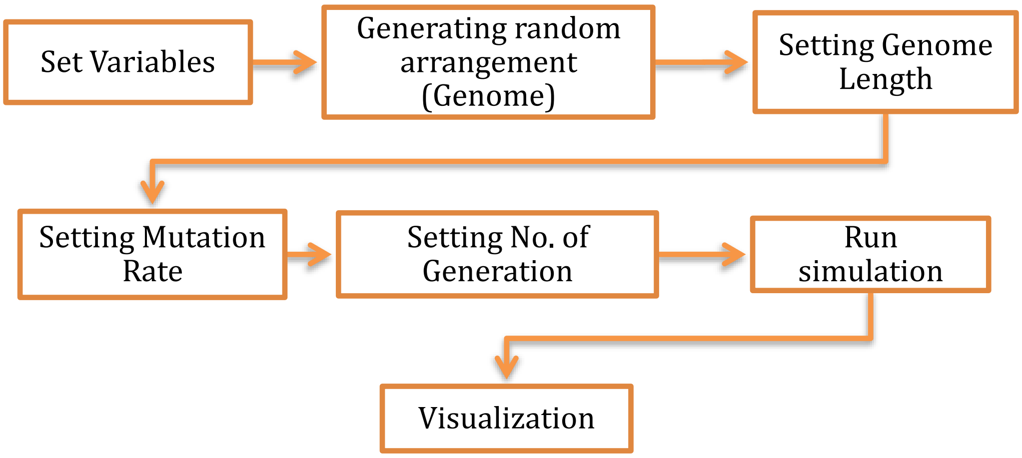
Step 7: Re-simulation With Corrective Measures

If the simulation result shows high risk, users are encouraged to modify parameters and re-run the simulation. For example, halving the mutation rate mid-way mimics recovery or external intervention. A new graph is generated to compare the original vs. corrected simulation.

Step 8: Display Genome Changes Step-by-Step

Finally, users can explore generation-by-generation genome changes. For instance, from Gen 00 to Gen 05, each genome sequence is displayed to show how mutations have accumulated, helping users understand base-level evolution.

C. System Design: Block Diagram



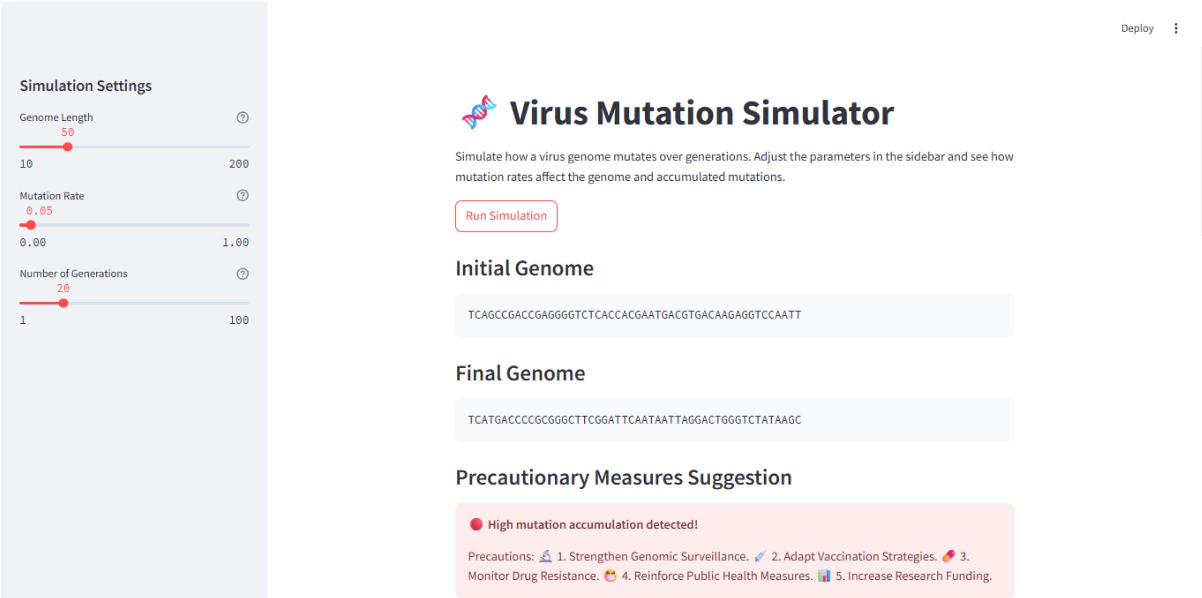
Result

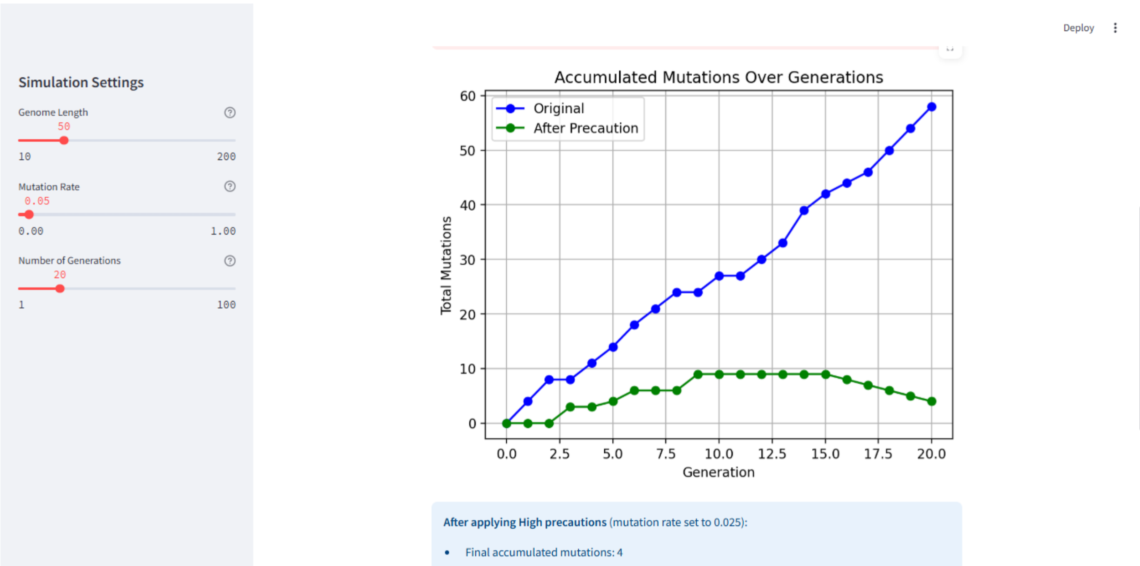
The Virus Mutation Simulation model effectively demonstrates how viruses mutate over generations by applying probabilistic mutation rules to synthetic genome sequences. The simulation produces key outcomes that offer both scientific and educational insights.

1. **Mutation Frequency Distribution:** Line graphs reveal that higher mutation rates cause exponential genome divergence, while lower rates preserve sequence stability.
2. **Genome Variation Visualization:** Color-coded genome displays highlight mutation hotspots and changes at specific positions across generations.
3. **Scenario-Based Observations:**
    - *Low Mutation Rate:* Stable genomes with minimal change.
    - *Moderate Rate:* Gradual, observable evolution with lineage splits.
    - *High Rate:* Rapid, excessive mutation leading to genome instability.
4. **User-Driven Controls:** Real-time adjustment of mutation rate, genome length, and generation count allows dynamic exploration of mutation trends, enhancing understanding through immediate visual feedback.

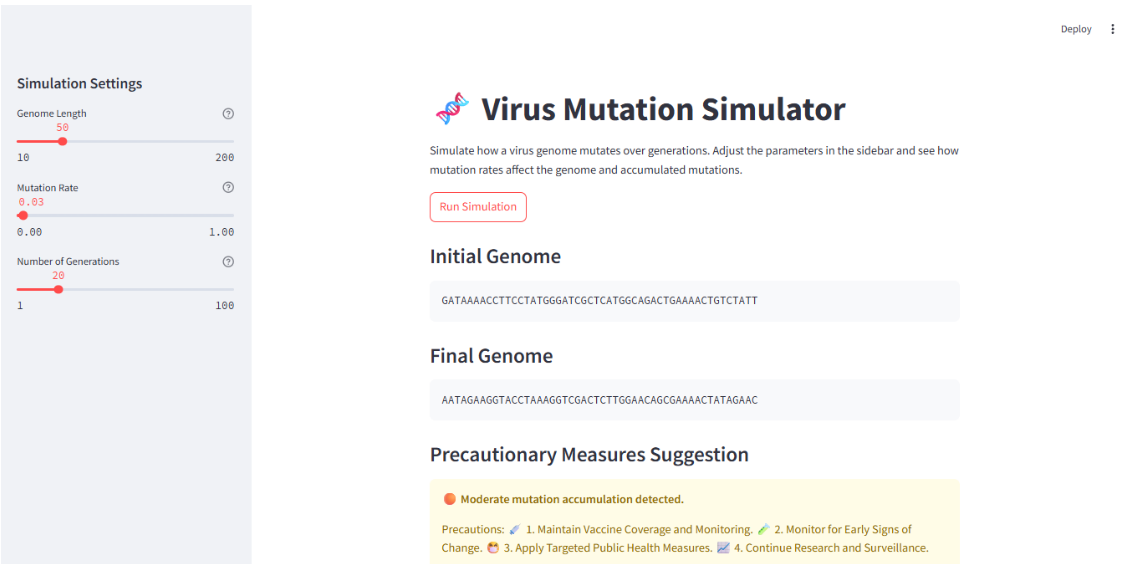
A. Website simulation:

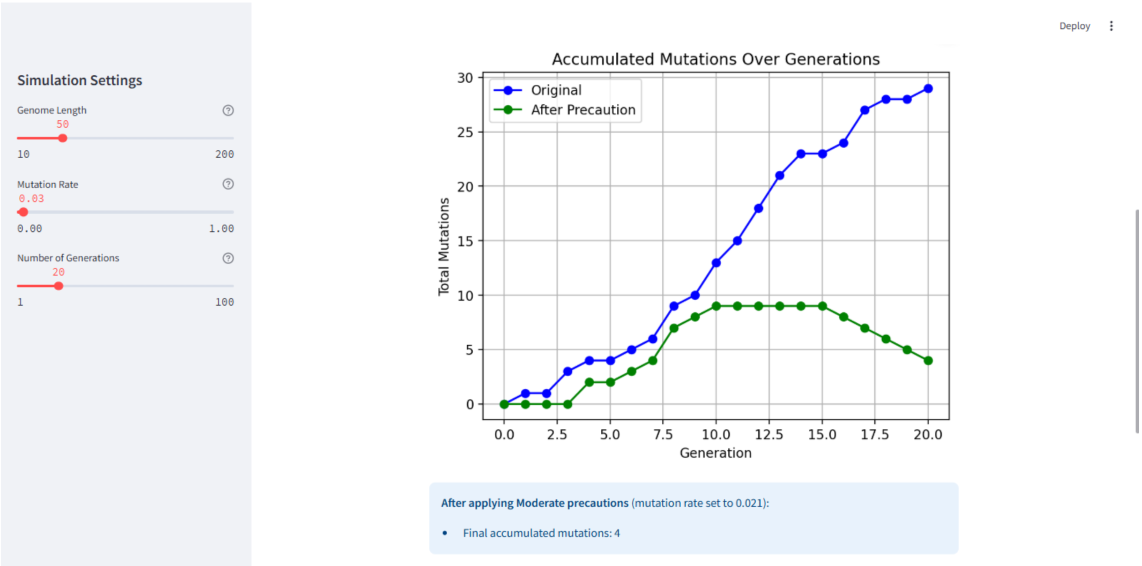
1. For high mutation



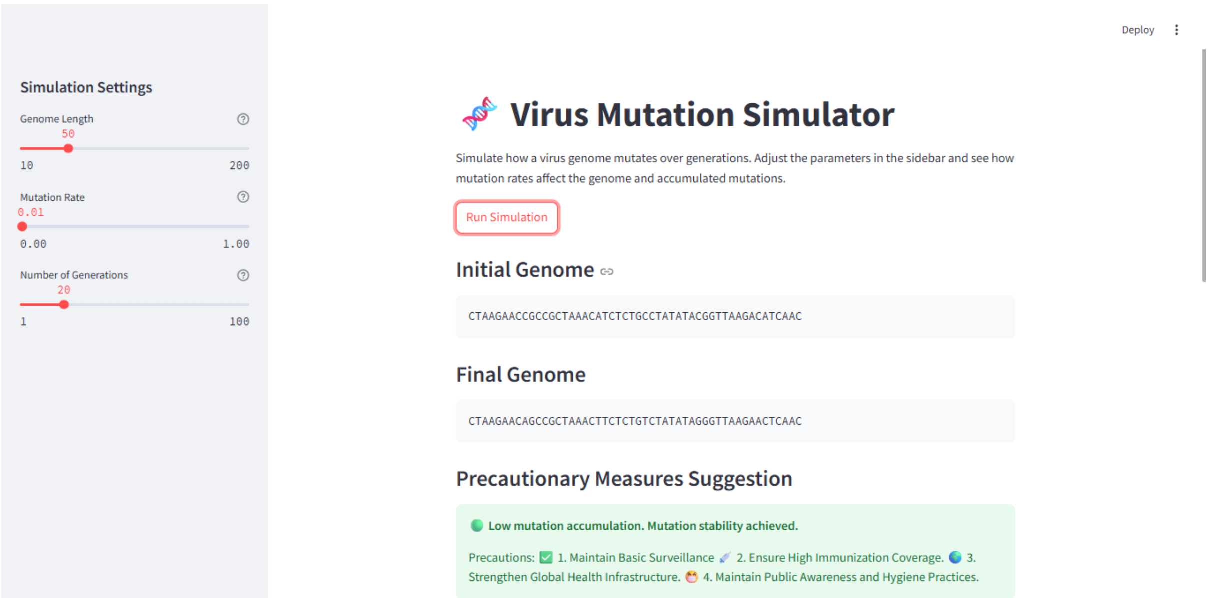


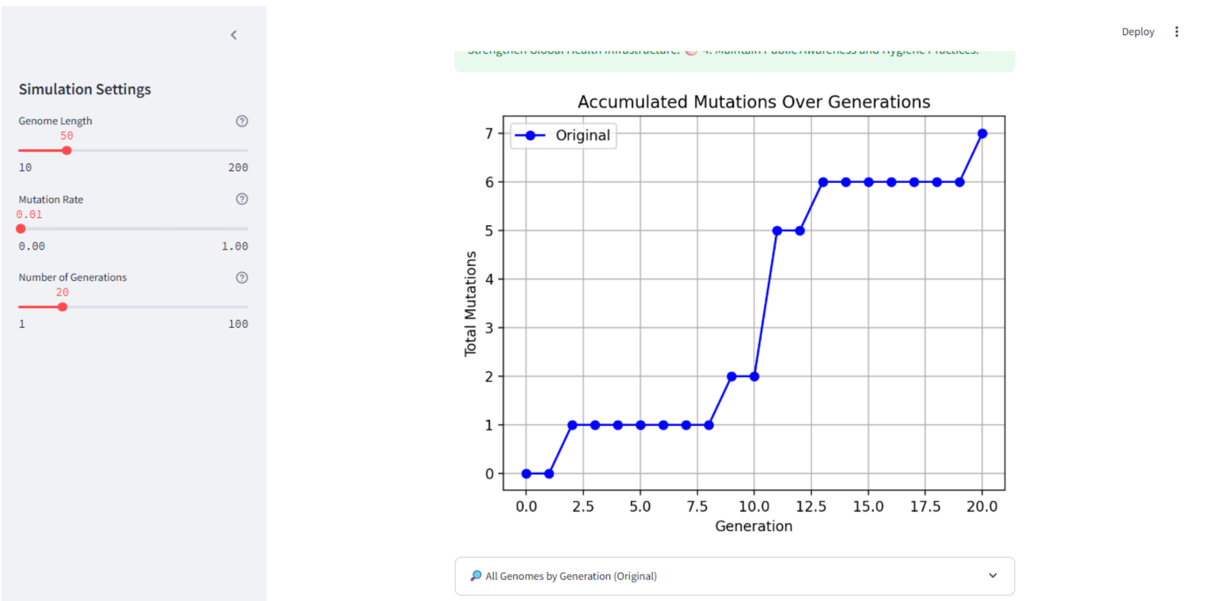
2. For moderate mutation





3. For low mutation





DISCUSSION

The Virus Mutation Simulation demonstrates strong *scientific relevance* by replicating mutation patterns aligned with real-world data such as SARS-CoV-2 and influenza. Its *educational value* lies in simplifying complex concepts like genome drift and variant emergence through interactive visualizations. However, it has *limitations*, as it does not model epigenetics, recombination, or immune escape. *Computationally*, it performs well for mid-scale tasks, though larger runs require optimization. Despite simplifications, the simulation shows real-world potential, offering a *pre-analytical tool* for research and policy-making to explore hypothetical viral mutation scenarios and evaluate associated risk levels in a safe, controlled environment.

Challenges and resolution:

1. Complexity in Modeling Mutation Patterns

- *Issue:* Viral mutations follow highly complex and often unpredictable biological rules.
- *Why it's critical:* Accurate simulation of real-world mutation behaviour is the foundation of the project.
- *Resolution:* Use probabilistic algorithms like Monte Carlo simulations and integrate real-world genetic data to enhance realism and biological relevance.

2. High Computational Requirements

- *Issue*: Simulating thousands of genomes or long sequences can lead to performance bottlenecks.
- *Why it's critical*: Limits scalability and responsiveness, especially during real-time or large-scale simulations.
- *Resolution:* Use efficient algorithms, multithreading, and consider cloud-based solutions like AWS, Google Colab, or even GPU acceleration to handle large workloads.

3. Visualization of Mutations

- *Issue*: Mutation data can be difficult to interpret, especially for non-experts.
- *Why it's critical:* If users cannot understand the output, the educational and analytical value is lost.
- *Resolution*: Implement color-coded genome maps, phylogenetic trees, and interactive visualizations to enhance clarity and usability.

Conclusions

The Virus Mutation Simulation Project exemplifies the role of computational modeling in understanding viral mutation dynamics. Integrating algorithms with real genomic data offers insights into mutation likelihoods and their influencing factors. Despite inherent challenges such as data limitations, complex biology, and computational overhead, the project addresses them through strategic resolutions, ranging from efficient code optimization to user-friendly visualization tools.

The platform not only advances scientific comprehension but also serves as a valuable asset for research, education, and public health. Its design ensures ethical use while supporting informed decision-making, paving the way for predictive modeling and preparedness planning in virology and epidemiology.

future work

1. Integration with Live Genomic Databases

- *Why important*: Enables real-time simulation of current viral mutation trends using live data from sources like GISAID and Nextstrain.
- *Impact*: Keeps the simulation up-to-date and relevant for public health, education, and research.

2. AI and Machine Learning Implementation

- *Why important:* Adds predictive power by identifying hidden mutation patterns and forecasting future evolutionary paths.
- *Impact:* Transforms the simulation from reactive to intelligent and proactive.

3. Host-Virus Interaction Modeling

- *Why important*: Captures how different hosts, immunity levels, and species interactions influence mutation behavior.
- *Impact:* Increases biological realism and supports zoonotic disease understanding.

4. Epidemiological Model Integration

- *Why important*: Merges mutation data with disease spread models (SIR/SEIR).
- *Impact:* Simulates variant-driven outbreaks, essential for pandemic preparedness and policy-making.

5. Modeling Drug/Vaccine Resistance

- *Why important*: Helps simulate how viruses evolve under drug or vaccine pressure.
- *Impact:* Guides vaccine design, therapeutic development, and resistance forecasting.

Acknowledgment

We would like to express our sincere gratitude to Prof. Arpita Santra for her invaluable contributions to the success of the Virus Mutation Simulation project. Her dedication, expertise, and collaborative spirit were instrumental in bringing this innovative project to fruition. Her insightful inputs and tireless efforts greatly enhanced the functionality and efficiency of the clap-controlled automation system. Her commitment to excellence and passion for technology have left an indelible mark on the project. We are truly thankful for her unwavering support, technical expertise, and collaborative approach throughout the development of this project. Her contributions have played a pivotal role in making this venture a reality.

REFERENCES

1. Domingo, E., & Holland, J. J. (1997). RNA virus mutations and fitness for survival. *Annual Review of Microbiology*, 51(1), 151–178. Form: https://doi.org/10.1146/annurev.micro.51.1.151
2. Sanjuán, R., & Domingo-Calap, P. (2016). Mechanisms of viral mutation. *Cellular and Molecular Life Sciences*, 73, 4433–4448 form https://doi.org/10.1007/s00018-016-2299-6
3. Python Software Foundation. (2024). *random — Ge nerate pseudo-random numbers*. Retrieved from https://docs.python.org/3/library/random.html
4. Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3), 90-95.
5. Streamlit Inc. (2024). *Streamlit documentation*. Retrieved from https://docs.streamlit.io/
6. Korber, B., et al. (2020). Tracking changes in SARS-CoV-2 Spike: evidence that D614G increases infectivity of the COVID-19 virus. *Cell*, 182(4), 812-827.e19.
7. World Health Organization. (2024). *Tracking SARS-CoV-2 variants*. Retrieved from https://www.who.int/en/activities/tracking-SARS-CoV-2-variants/
8. Nature Education. (2024). *Mutation and Genetic Variation*. Retrieved from https://www.nature.com/scitable/topicpage/genetic-mutation-441/