Digital Twin for Pandemic Simulation and Health Prediction

**Team Members**

**K.S.Abhihkeshav (21CSB0F03)**

**Department of Computer Science and Engineering**

**National Institute of Technology Warangal**

**Telangana, India – 506004**

**April 2025**

**Abstract**

# In recent years, the convergence of digital technologies and healthcare has unlocked innovative methods to model and respond to pandemics. This report presents the development and implementation of a Digital Twin system specifically designed for pandemic simulation and health prediction. Leveraging multi-agent systems, data visualization, and predictive/probabilistic analytics, the digital twin aims to replicate real-world population dynamics and disease transmission in a simple manner. The model enables simulation of multiple pandemic scenarios and evaluates the impact of interventions such as social distancing and vaccination. This system empowers decision-makers with a data-driven platform to evaluate strategies, mitigate risks, and improve public health responses. The system’s additional functionality to predict the health given the history of readings also helps in anticipating critical conditions early, enabling timely intervention and more accurate simulation of real-world disease progression. The results demonstrate that digital twins are not only effective in simulating pandemic spread but also in anticipating healthcare demand and guiding preventive measures.

**Introduction**

Pandemics like COVID-19 have underscored the urgent need for agile, data-driven decision-making tools in public health. Traditional epidemiological models, while informative, often fall short in adaptability, personalization, and real-time responsiveness. This is where digital twin technology offers a compelling solution. Originally developed for industrial applications, a digital twin is a dynamic virtual replica of a physical system that evolves in real time through continuous synchronization with live data. In the context of healthcare and pandemics, a digital twin can represent individual patients, populations, or entire healthcare infrastructures simulating their behaviours and responses under varying conditions.

The core idea behind a digital twin involves a continuous data exchange between the physical and virtual worlds, enabling simulation, prediction, and optimization. The physical entity, such as a person or a hospital, is mirrored by a virtual model that is constantly updated with data from IOT sources like wearable sensors, electronic health records, mobility data, and epidemiological surveillance systems. This real-time feedback loop allows the twin to adapt and improve its predictions, providing deep insights into emerging health trends and system behaviours.

Some common terminologies associated with digital twin systems include the **Agent**, which represents an individual unit within the system such as a person, device, or hospital bed whose behavior is being tracked and simulated. The **Model** refers to the mathematical or computational representation that captures the agent's dynamics, state transitions, and interactions. The **Digital Twin** itself is the integrated system that continuously mirrors the state of its real-world counterpart, using live or periodically updated data. **State Estimation** involves inferring or predicting the current and future state of an agent based on available data, while the **Digital Thread** represents the flow of information that connects the physical and virtual entities throughout their lifecycle. **Synchronization mechanisms** ensure that real-time updates are reflected in the twin, enabling accurate prediction, simulation, and optimization. In healthcare, these concepts come together to support applications like personalized treatment simulations, predictive diagnostics, and strategic resource allocation during health crises such as pandemics.

This project aims to harness the potential of digital twins by building a scalable, agent-based simulation tailored to pandemic scenarios. Each agent represents an individual with a health profile that evolves over time based on interactions, health metrics, and probabilistic models. The integration of run-time visualization and dynamic state updates allows the system to reflect the complexities of disease spread and healthcare demand with high granularity. In doing so, the digital twin not only supports better forecasting and planning but also bridges the gap between static epidemiological models and responsive, data-driven public health systems. Furthermore, the additional feature of health prediction plays a pivotal role in understanding a person's current and potential future health status. By analyzing a history of vital sign readings and other relevant health data, the system can forecast possible health trajectories, flag early signs of deterioration, and suggest preventive measures. This predictive insight empowers individuals to take proactive steps toward managing their health, such as seeking timely medical attention or adjusting lifestyle habits. In the context of pandemics, such forecasting can be life-saving, enabling early isolation, treatment, or vaccination strategies. This mitigates risks to both the individual and the broader community.

The specific objectives of this project include:

* Designing a digital twin framework for simulating pandemic dynamics.
* Implementing multi-agent systems to represent individual behaviour and interaction.
* Visualizing health data and simulation outcomes during run-time.
* Health Prediction framework for a history of health data.

**Methodologies**

Existing models for pandemic simulation have utilized various digital twin platforms such as Microsoft Azure Digital Twins, Siemens MindSphere, and AnyLogic. These frameworks provide a powerful base for constructing virtual representations of real-world systems enabling dynamic simulations based on live or historical data. Some implementations have taken a highly realistic approach, designing visually rich environments that reflect detailed agent behaviour and human mobility. In contrast, other models prioritize computational efficiency and abstraction, depicting people as simple blobs and homes as static geometric shapes. Despite the visual simplicity, these models are often optimized for analysing infection dynamics without overloading computational resources.

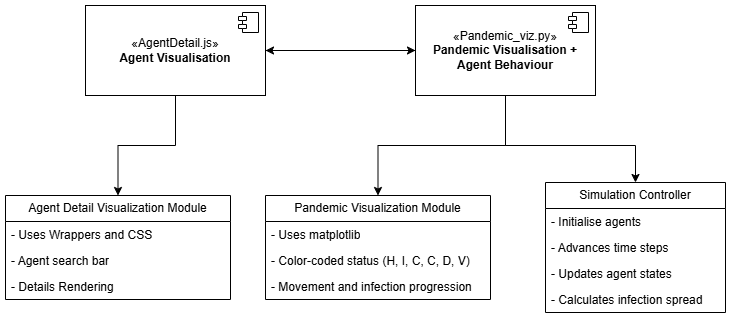
In the healthcare sector, digital twins have gained traction for their ability to represent and monitor patient states and forecast disease progression. These systems often rely on continuous streams of data sourced from IoT-enabled medical devices such as heart rate monitors and wearable activity sensors. The virtual model evolves in sync with the real-world patient, offering a platform for personalized medicine. By simulating various treatment scenarios or predicting potential deterioration, healthcare providers can proactively mitigate health risks. However, a significant challenge in these systems lies in the limited availability of high-resolution and time-dependent medical data. Much of this data is considered sensitive, governed by strict privacy regulations, and often inaccessible for model development or public research.

Health prediction models, particularly those that aim to capture time-variant trends, depend heavily on such longitudinal data. Without access to authentic and granular datasets, many digital twin systems rely on generalized assumptions or synthetic datasets, which may not always capture the diversity or complexity of real-world patient trajectories. This limitation becomes particularly critical when the goal is to forecast outcomes or prescribe interventions based on subtle variations in health patterns.

The COVID-19 pandemic served as a turning point for the application of digital twins in public health. Several research institutions and companies worked on virtual population models to track and predict disease transmission. Some focused on macro-level simulations for policy-making, incorporating data on mobility, demographics, and social behaviour. Academic models using tools like Mesa in Python or Repast explored more open-source, lightweight implementations of digital twins that simulate urban-scale pandemic scenarios.

Still, many digital twin projects in healthcare and pandemic management remain limited in interactivity or scope. Some offer only static environments where data is visualized but not meaningfully simulated while others lack real-time data integration. The most impactful digital twin systems are those that combine detailed modelling, data-driven updates, and user interactivity to provide not just a view of what is happening but actionable insights into what could happen next. As technology matures and data governance evolves, digital twins have the potential to become indispensable tools in managing both individual health and public health emergencies.

**Implementation and Results**

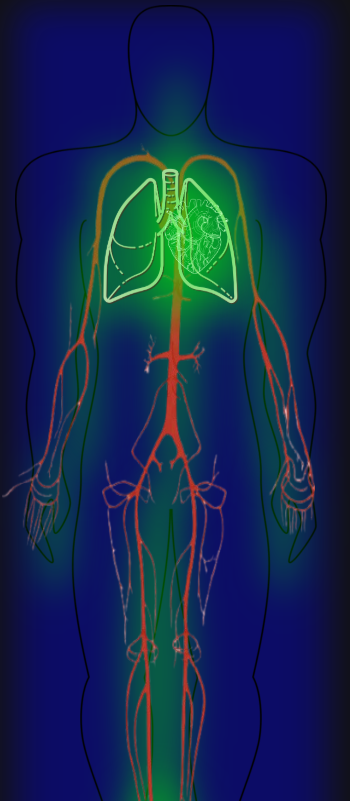
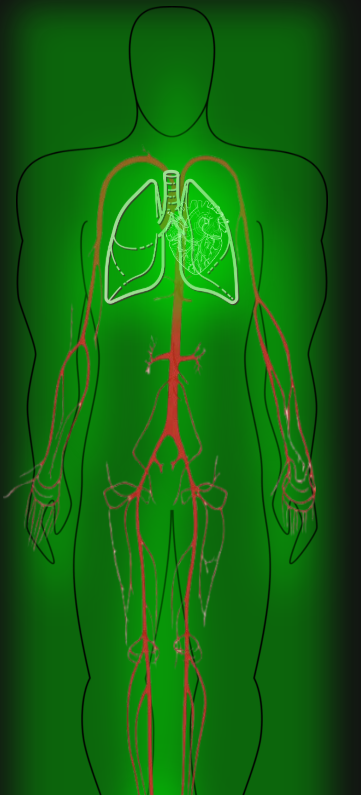
****

A backend Python script (pandemic\_viz.py) and a frontend JavaScript file (AgentDetailElement.js) make up the implementation. In order to facilitate more complex AI-driven visualizations, they collectively expand the Mesa agent-based modeling framework. Their main goal is to combine deep learning, simulation, and interactive user interface design to allow for a thorough examination of each agent's health status and predictions.

Mesa's web interface is directly integrated with a custom visualization element defined by the JavaScript component. When a user looks for and chooses an agent in the simulation, this component displays structured health information. It greatly improves the default Mesa experience by dynamically displaying current health conditions, anticipated outcomes, and visual representations like anatomical overlays. In contrast to the typical chart and grid modules, the panel provides a more user-friendly and information-rich view.

On the backend, pandemic\_viz.py implements the Python-side logic required for the custom visualization. It defines a subclass of Mesa's VisualizationElement, named AgentDetailElement, which uses render() and render\_model() methods to serialize agent-level data into a JSON-compatible format for the frontend. This file also initializes two pre-trained PyTorch models: one for predicting current health parameters (parameter\_model.keras) and another for analysing temporal health evolution (health\_time\_series\_model.keras). These models bring clinical relevance to the simulation by providing medically meaningful predictions that reflect each agent’s health progression.

To streamline deployment, organ/body images used in the visualization are base64-encoded in the backend and embedded directly into the frontend interface. This approach avoids external asset dependencies and ensures the visualization is portable and robust across different runtime environments. During initialization, all required models and images are loaded so the system can serve complete data during Mesa’s rendering loop. Further, each of the visualised organs glow green, red or blue indicating healthy, cautionary and critical conditions.

**** **** 

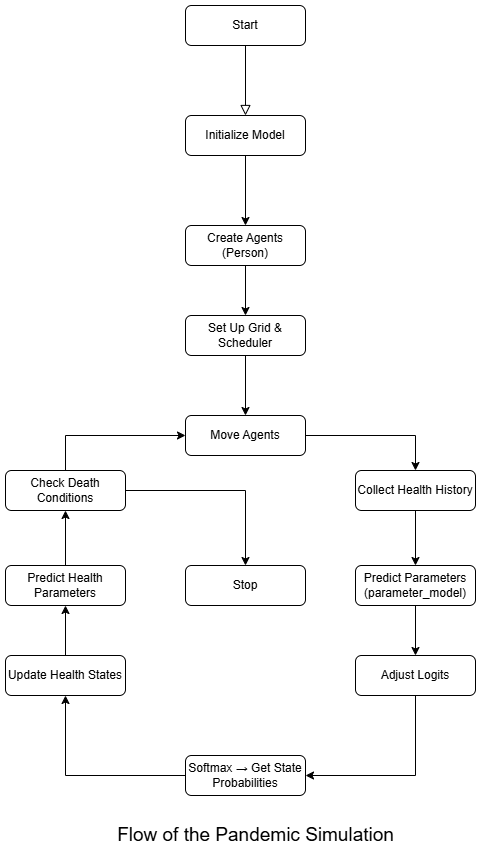
**Red -> Cautionary Health** **Blue -> Critical Health** **Green -> Good Health**

Integration is completed by adding the AgentDetailElement instance to the list of visualization modules during Mesa server initialization. This allows the detail panel to work alongside standard modules like CanvasGrid and ChartModule, creating a comprehensive interface that supports both spatial overviews and granular, agent-level inspection.

The agent simulation logic, centered around the Person class, coordinates batch-processed deep learning predictions. When multiple agents are present, predictions are made in batches for efficiency. The simulation first invokes the parameter model, which outputs a probability distribution over possible future health states for each agent. From this distribution, a target health state is sampled to reflect the natural variability of disease progression. This probabilistic step injects realistic randomness into the simulation, mimicking how individuals might respond differently under similar conditions.

Next, the health time series model smooths the transition between the agent’s current state and the chosen future state. Rather than jumping abruptly jumping from one condition to another, this model tries to smoothly transition between states closely resembling real-world health changes. This helps maintain biological plausibility and improves the interpretability of the simulation.

Simple spatial rules are used to control agent movement. Unless they are confined by walls or boundaries, agents move randomly up, down, left, or right within the grid. However, agents are programmed to migrate toward hospital nodes if their health deteriorates below a predetermined threshold, such as dangerously low oxygen levels or a high fever. This behavior gives the simulation more behavioral depth by simulating real-world instincts to seek medical attention when ill.

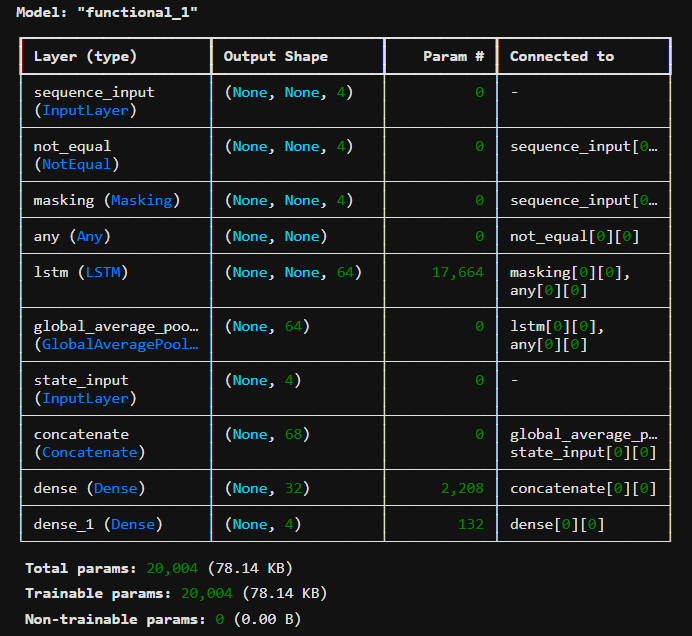
****When the simulation runs with only one agent, the system shifts from stochastic multi-agent dynamics to a deterministic health forecasting model. In this mode, probabilities are replaced with concrete model outputs. The agent’s health history alone drives the predictions, which are processed using the same AI models but without random sampling. This mode allows the application to behave as a personalized health predictor rather than a population-level simulation.

Because the models were trained on data with time intervals spanning weeks to months, the predictions reflect longer-term changes. Even though the simulation runs frame-by-frame, each frame corresponds to a clinically meaningful time step. The result is a visualization of the agent's future health that feels both smooth and contextually grounded.

For patient-specific applications such as health monitoring, forecasting, or instructional demonstrations, this deterministic mode is perfect. Without being hampered by the random actions of other agents, it allows users to investigate how specific conditions change over time based on actual data.

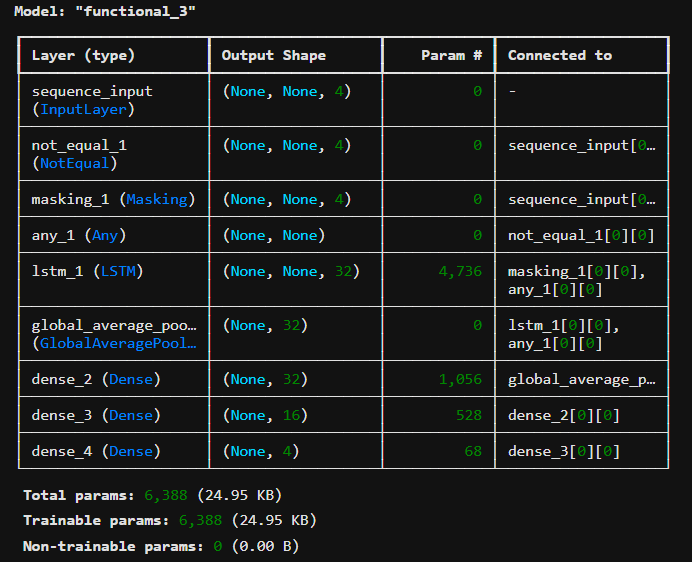
Altogether, this system skilfully blends interactive visualization, agent-based simulation, and machine learning. It offers a versatile platform for researching disease progression or investigating upcoming medical technologies by enabling both in-depth individual perspectives and extensive population dynamics. **ML models architecture summary:**

1. **Health\_time\_series\_model.keras**

****

The loss after training was 5%.

1. **Parameter\_model.keras**

****

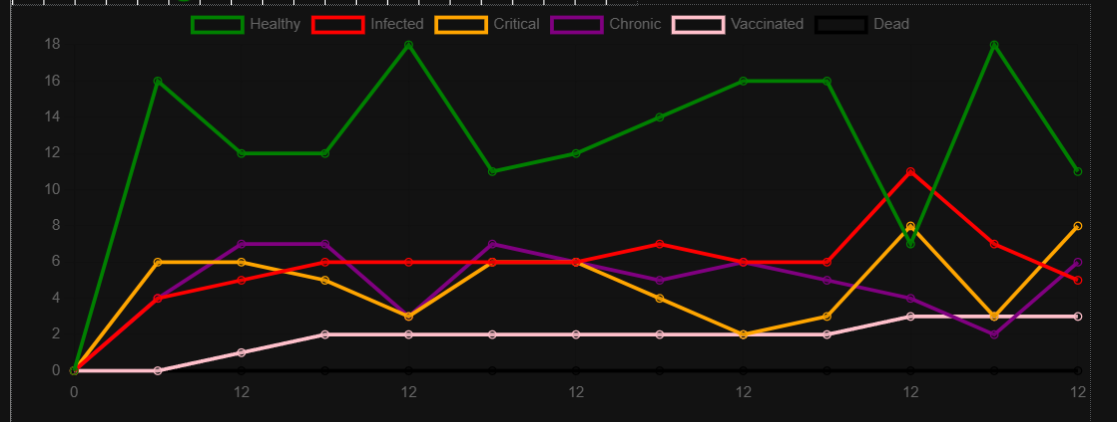
The loss after training was 0.005%.

**Visualisation**

**Grid Environment:**

****

**Environment Statistics:**



**Discussion**

While the implementation is relatively simple and straightforward, the use of Mesa alongside JavaScript for visualization is somewhat unconventional. Unlike more established and user-friendly UI frameworks such as Microsoft Azure or Siemens’ platforms, this approach can feel cumbersome and difficult to customize due to inherent limitations in Mesa’s design. However, as a free and open-source framework, Mesa remains a valid option for developing digital twins especially when cost is a constraint.

Compared to other methodologies, our framework operates on a small, time-varying dataset which contributes to higher loss values but still produces functional results. Notably, the overall complexity of our approach is significantly lower as it requires only a single backend file to operate, whereas other platforms typically demand a complete API and more extensive infrastructure to run a digital twin.

**Limitations**

* Small time variant dataset, leading to ML models with high loss.
* Inflexible Framework of Mesa.
* The Model needs to run a few number of steps before the simulation and health predictions are accurate.

**Future Work**

* Training the models with proper time-variant health datasets.
* Deploying the application onto the web.
* **Add functionality to include effects of vaccines rather than just reducing chances to get infected.**
* **Add more different types of blocks such as walls, homes or parks in the grid.**

**References**

1. <https://ieeexplore.ieee.org/document/10883917>
2. <https://mesa.readthedocs.io/latest/getting_started.html>
3. <https://www.kaggle.com/datasets/chidozieuzoegwu/cvd-vital-signs>

**Code**

[**https://github.com/K-S-Abhihkeshav/Digital-Twin-Pandemic**](https://github.com/K-S-Abhihkeshav/Digital-Twin-Pandemic)