**Software Implementation and Testing Document**

**For**

**Group Germ Theory**

Version 2.0

**Authors**:

Hector Rizo

Adam Pelah

# Programming Languages (5 points)

*List the programming languages use in your project, where you use them (what components of your project) and your reason for choosing them (whatever that may be).*

* Python -> this language has support for data collection and visualization. It is intuitive and easy to read so we chose it as our primary language of implementation.

# Platforms, APIs, Databases, and other technologies used (5 points)

*List all the platforms, APIs, Databases, and any other technologies you use in your project and where you use them (in what components of your project).*

* Numpy -> for data manipulation and array functions
* Matplotlib -> for data visualization
* Random -> for random number generation
* Possible API’s include: BeautifulSoup for webscraping, WHO api for gathering data
* Basic java script to show the results of our implementation

# Execution-based Functional Testing (10 points)

*Describe how/if you performed functional testing for your project (i.e., tested for the* ***functional requirements*** *listed in your RD).*

* A grid was created for the cells. The grid was tested visually using matplotlib as a visualizer. It was possible to pass in a dictionary of cells to the gameboard object and to change cells within the gameboard class.
* There were issues controlling the boundaries of the board.

# Execution-based Non-Functional Testing (10 points)

*Describe how/if you performed non-functional testing for your project (i.e., tested for the* ***non-functional requirements*** *listed in your RD).*

* The grid was expanded and the numbers increased till performance lagged.

# Non-Execution-based Testing (10 points)

*Describe how/if you performed non-execution-based testing (such as code reviews/inspections/walkthroughs).*

* We had a few code walkthrough sessions.