Pandemic Processing

Software Testing Plan

(Version 1.2)

April 4th, 2019

Project: Epidemiological Modeling Portal

Team Members:

Anthony Schroeder

Joseph Eppinger

Tanner Massahos

Sponsor:
Dr. Joseph Mihaljevic

Mentor:
Dr. Eck Doerry

Table of Contents

1 INTRODUCTION

Infectious disease has been a problem throughout all human history. For example: The Plague of Justinian in 541 lasted over 200 years and resulted in an estimated 25-50 million deaths; the Black Death during 1347-1350 killed ~60% of the European population. Compare this to a modern epidemic: The West African Ebola Outbreak resulted in the deaths of 11,310 individuals. It is clear that there has been a dramatic change in the number of casualties due to infectious disease when compared to plagues of the past, and one reason for this is the development of the science of epidemiology.

Simply put, epidemiology is the study of infectious disease and how it spreads within a defined population. An important tool used by epidemiologists to manage infectious disease outbreaks is modeling. Models are essentially predictive networks of mathematical formulas developed by epidemiologists that show how disease can spread through a community. Given a set of initial conditions, a model can generate predictions of key outcome variables, such as how the fraction of the population that is infected changes overtime, speed of disease spread, and mortality rate. These models ultimately are an attempt at predicting how a disease will affect a population given what epidemiologists know about it. When a model closely represents the real world public health officials can plan properly and respond accordingly.

These models can differ in style such as deterministic mathematical models or complex spatially-explicit stochastic models. Models are programmatically created to guarantee reliability and maintainability of data. Epidemiological modelling allows:

1. Scientists to create models of infectious disease given a certain set of assumptions and data.
2. Based on these models, scientists can predict important parameters dealing with the infectious disease such as speed of spread and potential population impacts.

Epidemiologists also apply various methodologies and base their models on varying assumptions, meaning there could be a handful of differing models for one infectious disease all of which could generate different optimal vaccination times and predictive preventative measures.

Many of these scientists are creating these models in parallel, and must discuss them in order to determine which is the most accurate. This deliberation component

is key in that it ensures the optimal model is chosen for a given situation. The best model could be any one of the models proposed, or even a new hybrid created from the discussion.

This is where the major problem arises, according to the project client Dr. Joseph Mihaljevic, an epidemiologist and assistant professor at NAU SICCS. There is a lack of an efficient system where epidemiologist from all around the world can share, review, test drive, then critique and discuss one another's models in order to receive an optimized model that generates the best predictions as quickly as possible. Streamlining this collaborative discussion and refinement process is key to improve rapid and efficient responses to developing disease outbreaks.

It is not that this current workflow is at its core broken; it is simply that it could be further optimized. In the world of epidemiology there is a lack of an efficient system where epidemiologists from all around the world can critique and discuss one another's models to arrive at an optimized model that generates the best predictive measures. Some specific problems with this workflow include:

- Epidemiologists must wait until full publication to view models of other epidemiologists they are not personally acquainted with.
- Deliberation is difficult when no public space for it exists. Currently, epidemiologists can only communicate through emails and GitHub comments.
- Communities of epidemiologists are isolated clusters, as in they only tend to communicate with those that they already know well, which reduces the extent to which deliberation can produce new ideas.
- Current discussion and models are not extremely public or easily accessible. An epidemiologist must publish or actively send their data to authorities before action can be taken.
- Difficult to communicate results with managers and public health officials.

The modelling of infectious disease is a rather well-defined process. Each time an epidemiologist wants to model the outbreak of an infectious disease, they step through the following process:

1. **Collect data via data sets or researched data:**
   The epidemiologist gathers data about the population and how the infection spreads from a mathematical standpoint.
2. **Model data using C, R, or Python:**
   Using this data, the epidemiologist crafts a model using a series of mathematical equations which models dynamics of interest such as the rate of mortality and speed of spread. This model is then implemented as a

modular computer program so that it can be rerun quickly with varying parameters.

3. **Develop the best model through discussion:**
   Next, they discuss with other epidemiologists, critiquing and modifying the model eliminating of any potential incorrect assumptions. Ideally, this discussion leads to refinements to produce a model that should most closely approximate the real-world spread of the disease.
4. **Build predictions based on the model:**
   The epidemiologist then predicts the best possible course of action for minimizing the damage done by the disease, and proposes preventative measures to the authorities who can act accordingly.

Epidemic Observation Network (EON) will be a secure, fast, and user friendly web application where epidemiologists from all around the globe can share and discuss their models. EON will allow epidemiologists to:

- **Share models with the community**
  - *Method:* EON will be a place where epidemiologists can post their models' code for others to view and discuss.
  - *Purpose:* This will allow for quick critiques and harbor discussion. These critiques and discussions will lead to an optimized model which will generate the best predictive measures.
- **Decide how their models appear to viewers**
  - *Method:* EON will allow epidemiologists to decide how public their models are. For example, a scientist might want to keep their code private, but the final graphical output is fine for viewing.
  - *Purpose:* Some scientists do not want to share their models or data due to publication purposes, so the user will have the option to make a post public, private, or limited sharing.
- **Interact with and provide feedback on other user's models**
  - *Method:* EON will provide a discussion forum for epidemiological models. Additionally, users will be able to modify and re-upload each other's code.
  - *Purpose:* This builds a sense of community and allows users to modify assumptions and visualize how it changes the output.
- **Discuss future models without a fully developed model existing**
  - *Method:* EON's forum will not just be for models which already exist. Users can create discussions without any working prototype developed.
  - *Purpose:* This allows users to simply post an in-progress model for discussion, meaning no code is required upon upload.

The above bulleted points are essential to thoroughly test as they are key aspects that satisfy the requirements of the project. By streamlining these key steps in the workflow EON will provide an invaluable missing resource for epidemic management. Specifically, it will streamline comparing and discussing models to come to a faster consensus. EON will allow epidemiologists to quickly and efficiently generate models, compare and discuss models, and arrive at an optimal model so they can create preventative measures to combat disease outbreaks.

Currently in the developmental process a majority of the required functionality has been implemented though there is a need for verification and possible refinement to ensure a smooth end user experience.

The alpha testers for EON are going to be graduate students within Dr. Mihaljevic's epidemiology class. As we receive feedback via surveys and reviews from the alpha testers rapid modification will be made to the application ideally increasing the user experience when it comes to accomplishing essential tasks within EON.

This type of rapid testing and modification will allow for better pinpointing of some of the smaller or more major problems epidemiologists have with our application as it pertains to the end user experience. A proper first release of this application is essential in EON's success and this is an ideal testing strategy to assure EON is developed in a way that meshes best with epidemiologists and satisfies the needs of the end user.

Testing for this application is split into three subsections: unit testing, integration testing, and usability testing, all of which will be discussed in further detail in the following sections. Specifically, there is a focus when it comes to usability testing as we have verified a majority of functionality and it is important that the niche group of users understand how to utilize and navigate the web application.

2 UNIT TESTING

The proper implementation of testing is critical to ensure the effectiveness of any software application. Tests have been broken up into these different scopes: Unit, Integration, and Usability testing. Performing tests within this specified order will save time and energy, by verifying that smaller components are working correctly, more extensive tests that use several parts become easier to fix.

The first set of these tests, Unit Testing, is responsible for testing the individual components of a system to verify that they are performing correctly. Distinguishing the different parts of a system is critical to properly identify what section a test might fall under. EON's main components are:

- User
- Models
- Groups
- Forums

These various components are independent and don't directly rely on each other. Django allows these components to be placed within individual Django applications, which help to maintain separation of concerns. Django already has the testing library *unittest*, providing a location to incorporate tests for of these applications. This is highly helpful as most of the testing will be centered around these different applications/components. But because Django is based on the view paradigm, and each of these applications provide functions to manage the view, model, and HTML response some testing is extended to operational tests by developers to verify proper information being displayed.

2.1 User

The Django User application is responsible for managing user profiles, and friend functionalities.

2.1.1 Account Management

As users perform different operations, they expect to be able to join the platform, customize their profile, change their password, add friends, and more. Here are some proposed user tests to manage personal accounts.

Unit Test	Description	Boundary Values	Example Parameters	Expected Response
Validate Username	Validating user operations and verifying unique usernames is important to maintain a bug free platform.	Usernames need to: - Be Alphanumeric - Be unique - Have no spaces - Be between	Good Example: "Model_Smasher"	Reject submissions with invalid or already taken Usernames.
Validate Email	To prevent the creation and use of malicious accounts, new user accounts will have emails verified before allowing access to limit the creation of fake accounts.	- No Spaces - All characters are Alphanumeric	Good Example: "as3379@nau.edu" Bad Example: "NotAnEmailAddress"	Reject submissions with invalid Emails.
Validate Password	Passwords need to be complex enough for user security.	- Longer than 8 characters long	Good Example: "Dnsh56" Bad Example: "Password"	Reject submissions with invalid Passwords

2.1.2 Ability to edit User values

While being a part of the platform, users are likely to change user information, Usernames, and passwords. These tests verify the ability for users to personalize and keep an up to date profile.

Unit Test	Description	Boundary Values	Example Parameters	Expected Response
Edit Profile information and settings	Users information is subject to change and should be manageable by the end user.	- Affiliation - Contact information - Bio	Example: Contact information: as3379@nau.edu or at (123)-456-7890	The appropriate data is changed, and the user is informed of the changes.
Edit Username and passwords	Validate new usernames and passwords before changing them.	Username, Password	N/A	User's information is changed and stored.

2.2 Models

The Model Django application is likewise responsible for storing and managing models that have been uploaded to the platform.

2.2.1 Model Management

Once a user has joined the platform, they are able to upload code and create graphs for other users to see. Additional users should have the ability to modify the model's parameters, title, and description as different aspects of their project change.

Unit Test	Description	Boundary Values	Expected Response
Upload Model	The main attraction to this web application is the ability to upload Epidemiological models to discuss	N/A	User code is saved to a flat file location, and a new model entry is entered in the PostgreSQL database.

Delete Model	Over time, models might be unneeded, and users might wish to remove old and irrelevant models.	N/A	The model is removed from the PostgreSQL database.
Edit Model	Only users with given privileges can edit and modify the model's description.	N/A	Models should have the ability for their description and default parameters to be changed.
Attempt to view private model	Users should be able to view Public Models	Settings are private	Only selected users can view private models.
Attempt to view public model	Users should be able to view Public Models	N/A	Any user can view the public models, and manipulate parameters to rerun code.
Models support multiple coding languages.	Users are able to upload and run different types of code,	Acceptable Languages: - R - C	After specifying the language being used, the code is ran and produces an appropriate model.

2.2.2 FireJail

Although most of our platform is designed on Django some operations need to be carried to external components such as:

- FireJail
- JSON

Here is an associated test verifying the security of the system from the uploaded code.

Unit Test	Description	Example Parameters	Expected Response
Code for different languages is correctly run through FireJail	Code with malicious intent should not be able to access any sensitive information on the platform.	Bad Example: -Upload code that attempts to access files outside of	Fire jail will prevent the program from accessing any critical information.

2.3 Group

The Django Group application is responsible for managing groups, group members, group invites, and member privileges within the group. These functions including table management as models, and HTML responses through the view.

2.3.1 Group Management

Epidemiologists might want to track their collaborations better and decide to create a group to manage the different models they wish to discuss. Here are some tests verifying this operation.

Unit Test	Description	Boundary Values	Example Parameters	Expected Response
Create a group with a valid group name	To ensure inaccurate operation of the platform, some simple boundaries need to be met.	Group names need to: - Be alphanumeric - Be unique - Have no paces - Be between (1-100) characters	Example: "Cool Model Group"	The group is created, and the original creator is made an admin for that group.
Create a group with an Invalid group name	To ensure inaccurate operation of the platform, some simple boundaries need to be met.	Group names need to: - Be alphanumeric - Be unique - Have no paces	Example: (An already taken name)	A message will be displayed with the proper error, describing the issue so the user can appropriately respond.

2.3.2 Group Membership

Groups are also likely to change as new users join and leave the platform, and the need to maintain relevant members and roles is essential for groups and collaborations to remain lean and agile to work on different projects.

Unit Test	Description	Boundary Values	Expected Response
Test Group's Invite	Group members should be able to invite new members to groups,	- Only non-group members can receive the invite.	The recipient receives an invite detailing the group and requesting user.
Test accept Group Invite	Invites from other users to groups need to be accepted, before adding the recipient to the group	- Recipients can accept or reject these requests.	The user is added to the group with default privileges.
Apply to group	Users might wish to work within a specified group	Users can only apply to groups they are not a part of.	An application is made to the Group's Admins and Moderators.
Accept application	Already existing group members with administrator privileges can Accept applicant join requests.	- Admins and Moderators can accept or reject group applicants.	The user is added to the group with default privileges.
Remove member	Members might become inactive and don't require access to sensitive information	- Admins and Moderators can remove group members.	The user is removed from the group.

2.3.3 Edit Group settings

Groups are likely to change their group members, information, privacy, and edit Privileges. These privacy settings include "Private/Public" which influences what general users can see about models and forums associated with the group, and "edit privileges" which allows general users of EON contribute to a given project. These tests verify the ability of administrators and moderators to manage groups and change different settings.

Unit Test	Description	Boundary Values	Expected Response
Edit Groups Name and About	As projects change groups need the ability	Group names need to: - Be Alphanumeric - Be unique	If fields are correctly filled out, the Groups

	to adapt and modify their information.	- Have no paces - Be between (1-100) characters <u>In addition, about descriptions need to:</u> - Be less than 500 characters	information will be changed.
Edit Members Role in group	Groups are likely to promote/demote member privileges based on acquaintance.	<u>Members can be changed to:</u> - Admin - Moderator - contributor - Member	The member's role is changed, and different privileges are also modified according to the new role.

<u>2.3.4 Group privacy settings</u>

Unit Test	Description	Boundary Values	Expected Response
Test Group's Private View Setting	If a group wishes to be private and hide their research efforts hidden until publishing their findings.	Private / Public	A group with privacy settings doesn't show their affiliated models, and forums.
Test Group's public contribute Setting	If a group wishes to provide access to their work and allow everyone to help assist.	Editable / Not Editable	A group with public contributes settings allows anonymous users to view, and edit models

<u>2.4 Forums</u>

Users should have the ability to contribute to general discussions, with some limitations to general users.

Unit Test	Description	Boundary Values	Example Parameters	Expected Response
Make a comment on a forum	Create Table entry in the form associated with the model.	- Generic text - Comments can be at a max of 800 characters.	<u>Example:</u> "Wow, this is a cool model !"	This comment will be made to the forum and be listed along

				with other comments.
Reply to a comment	Create Table appropriately describing the comment as a reply.	- Generic text - Comments can be at a max of 800 characters.	Example: "Wow, this is a cool model!" → "No, it's not."	This comment will be formatted as a reply to the previous comment.
Only Admins can create Forums	To regulate user privilege, only some users can create forums.	-Forum Name is: -Alphanumeric and < 30 characters	N/A	N/A

These user tests will help ensure proper operation of the platform and will validate the process of each component of the EON web application. The next section, Integration Testing will discuss how these different components will be tested together to verify correct operations.

3 INTEGRATION TESTING

Integration testing is the second iterative step to verifying proper operation and is responsible for testing the functionality and interactions between components. These tests are responsible for checking the proper functionality when using several Django applications in conjunction. As Unit testing evaluated the operations of each individual component, these tests verify that these different components work together into a coherent system.

3.1 Model discussion boards

As users browse other models, they might have comments suggesting improvements and can post them below a given model. Additionally, users might have further constructive comments and wish to reply to them.

Unit Test	Description	Expected Response
Public Models have associated discussion boards	Unless specified, default models will have associated discussion boards.	Every model generated has an associated place to post comments.

Users can make comments to uploaded models	To allow constructive feedback, users should be able to leave feedback so other users can interact and discuss different modeling techniques, and solutions to various problems.	If a group has enabled public viewing, all users are able to add comments and replies to discussions.

3.2 Group management and settings

As group admins you can manage different privacy settings of all models associated with the group as explored in unit testing. Changes made to a group's privacy settings apply to all models of the group.

Unit Test	Description	Expected Response
Models associated with groups share privacy settings.	To provide secure private communication and sharing, groups can easily create, and view models built by other group members.	Only members of a group can view its associated models.
Group models with privacy settings	Creating models for a private group makes it easy to have secure communication and sharing between members.	Only members of a group with proper privileges can view associated models.
Group models with private editable settings	Creating models for a group with non-editable settings allows for contributions to be controlled and manageable.	Only members of a group with proper privileges can edit associated models.
Group models with no privacy, editable by public setting, are easily contribute-able 'places.'	Creating models for an open source group makes it easy to have community members communicate and share with other open source contributors.	Any user can view and edit models associated with the group.
Group models with no privacy, editable by public setting, are easily contribute-able 'places.'	Creating models for an open source group makes it easy to have community members communicate and share with other open source contributors.	Any user can view and edit models associated with the group.

These integration tests will help ensure proper operation of the platform but is not enough to determine adequate functionality and limitations. Even with extensive unit and integration testing, bugs and issues are still likely to arise. The most reliable way to verify the stability, functionality, and ease of use, is to incorporate user testing alongside these previous tests. The next section will discuss usability testing, and the different challenges faced.

4 USABILITY TESTING

And finally, usability testing, takes real inexperienced users, and tracks their personal experience with the platform, recording their frustrations, and criticism. This is the final step to ensure proper functionality and operation free from bugs and issues that might be present after unit and integration testing. Usability testing is also integral to verifying that the platform is easy to use and comprehend.

User Testing and Survey:

Pandemic Processing is currently working with Dr. Joseph Mihaljevic to have students from his epidemiological class to test the EON platform.

Students will be given a lab manual describing a set of general tasks to accomplish. These tasks will have nonspecific directions, allowing the user to naively navigate the web site, allowing for real-world experience. Ideally, the user should be able to follow these tasks (with limited directions), as the interface should provide clear navigation, and formatting to help direct them to perform basic tasks.

These tasks including:

- Create a user account
- Create a group
- Upload a model they have created in class
- Create a post for model
- and reply to another user

By keeping directions vague, we can get better feedback about any difficult to understand operations of the platform.

The developers will guide participants through this proposed lab manual and record any notable interactions of the participants. This will help construct a better insight with how users interact with the system, as users might be inclined to hold criticism back in the survey. For example, if users find it difficult to find a link, or find it challenging to perform a task, they might not mention any issues because it might make them feel incompetent about performing tasks.

After completing the lab manual, participants will also be asked to complete a survey detailing their experience, thoughts, and suggestions. Questions on the survey might include:

- What was the most challenging task to achieve and why?
- Was user sign up easy and straight forward?
- Were there any occurrences of strange behavior? Such as a bad link, or unexpected result?

By performing this session, Pandemic Processing should get a good idea of what works, and what needs to be improved on for the EON platform.

Producing and implementing all of these tests is essential in ensuring robustness, and rigidity for release. As each set of tests are designed to catch and fix any potential issues that might hinder the next, they are all critical in the development cycle and can't be ignored. For example, fully completing unit and integration testing is essential before conducting usability testing, because small bugs might hinder the experience, and prevent first-round participants from accessing functionality.

5 CONCLUSION

Infectious diseases are a major health challenge and affect modern day society. Hundreds of thousands die every year from preventable diseases. The use of data-driven disease modeling in modern epidemiology can dramatically reduce the number of lives lost. A model must be created of a particular disease in order to properly plan, and epidemiologists construct these models so that they can determine the optimal preventative measures to be taken. The problem is that in creating these models, these scientists have a difficult time exchanging information before publication. Thus, EON will bridge this gap in the model development process by giving epidemiologists a public place for them to present and review their ideas. EON will become a public repository for epidemiological models, and a forum through which those models can be critiqued and refined.

EON is integral in an age where information spreads quickly and diseases move even faster. EON will be the solution that can revolutionize epidemiology and could be applied on an immense scale and generate a large future impact. It has the potential to save millions of lives around the world.

In order to accomplish these goals and provide this bridge for epidemiologists, proper testing must be conducted followed by rapid iterative development. Using the detailed plan discussed above, testing will be conducted in the following weeks. We are confident in our testing plan and that it will lead to positive refinements that will enhance and verify the capabilities of EON.