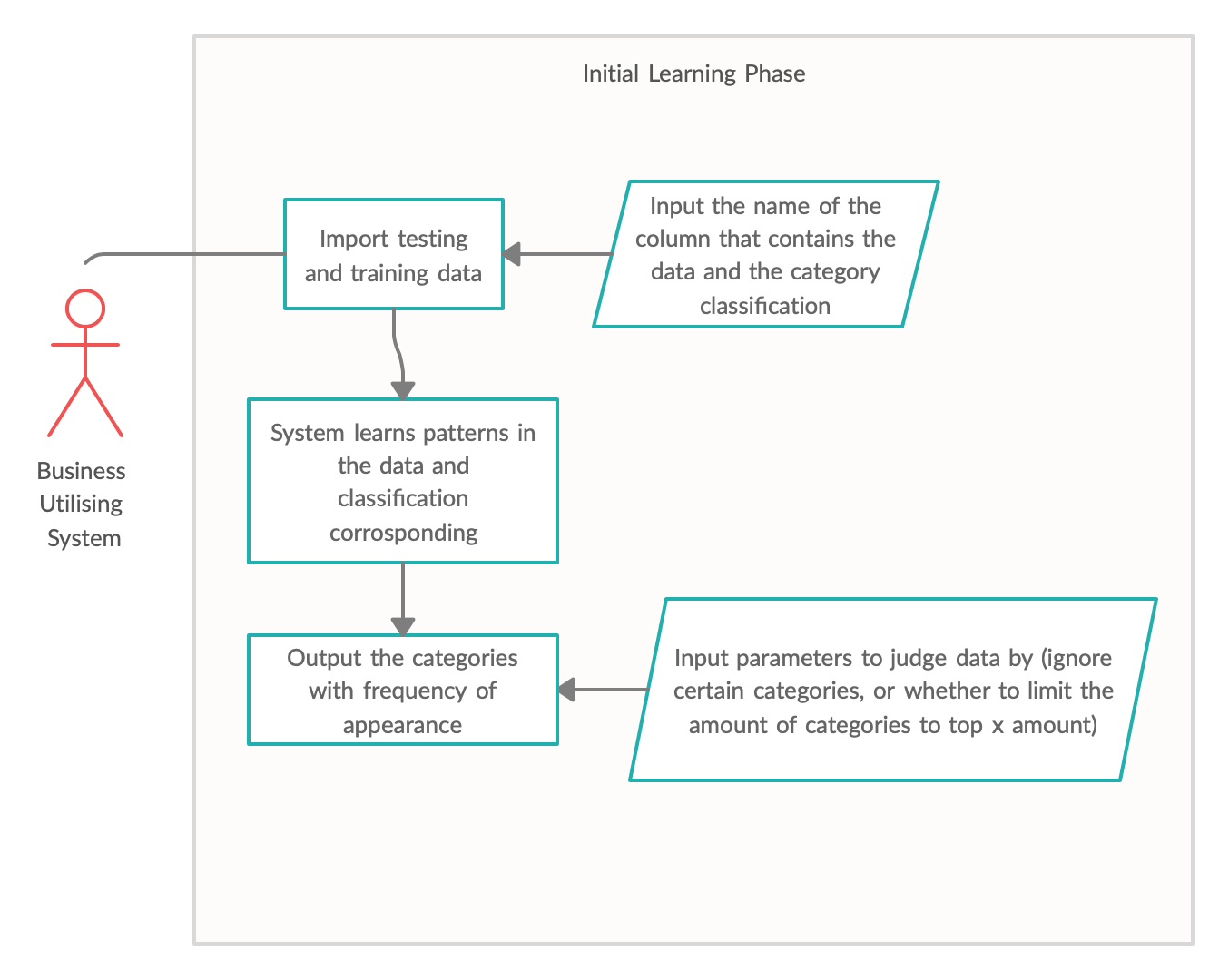
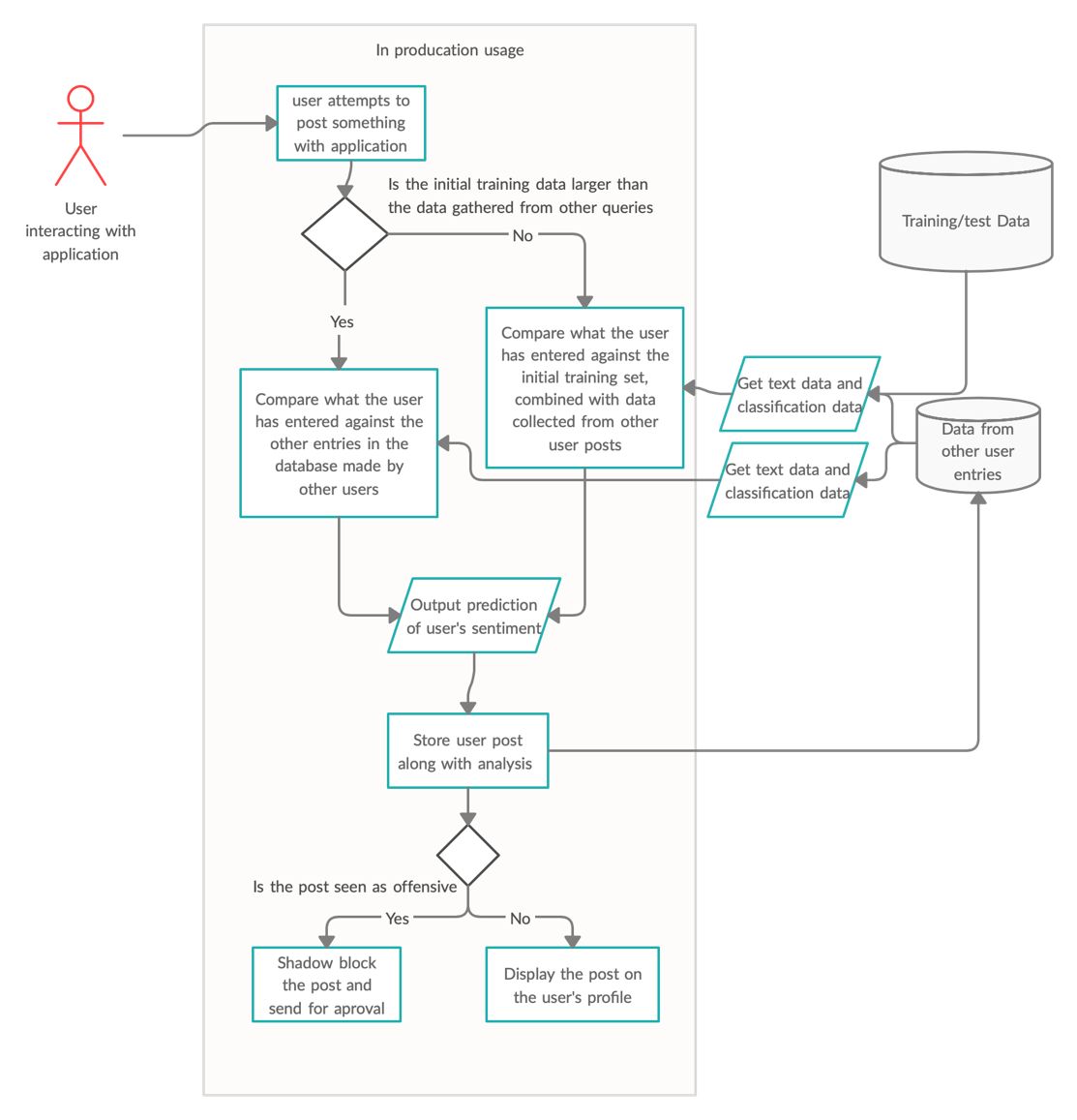
Solution Design

# Solution Diagram

In order for the system to be used in a production environment, there is a small “learning” phase that is planned to happen. During this phase, the business who would be using our potential system, would first have to import their own training and testing data. During this import, the user will then be asked which column contains the text data (the data that will be analysed) and the classification/label data. Once this has been extracted, and our algorithms have analysed the data, the user will then be shown a list of the categories as well as how often they appear in the data and asked if they would like to remove any of them from the training (for example, if reviewing movie genres, and there were only 2 of one specific category); the user will also be asked if they want to limit the categories to the top X amount, which would then only return the top X most frequent categories for the system to be trained on (for example, this may help the system be more accurate on things that are currently trending, possibly improving efficiency)



In this example, a system that could be used for social media is shown. Initially, to start the system, a user will attempt to post something with the application; we only plan for this system to analyse the text of a post so it would be limited to text based posts. Once the user has submitted their post, the system will take a tally of the training data and the collected user data so that it can decide which dataset should be used for predicting the response. If there is more test data than user data, the system will default to using the test data to analyse the text, however, we also plan on adding actual user data as well so that it has more examples. If there is more user data than test data, we plan on moving completely to the collected user data. Once the data has been pulled from the database and a model is created, we then preform a prediction based on what the user has input. This point would be standard for any program; however, the next portion is more specific to a social media website; the prediction would return the category that shows the most similarities to the text submitted by the user. For this example, we are just showing how the system would operate if it were attempting to detect something that may be seen as offensive. If the system determines that the content posted is offensive, the application may place a shadow ban on the post (not technically a ban, but not showing the content either) and sends it for review; reviewing content that has flagged allows the system to remove false positives, while allowing users to flag content that has gone through the filter helps the system learn false negatives too. If the content posted does not trigger the filter, it will be posted to the application.

# Data Description

During the development, testing, and “demo” period, the demo period being how the program would ship to the end user and would be the state between the program being finished and being altered by the end user, the data being used by the program will be a list of movies taken from IMDB, complete with description and category. We will be using movies and categories initially; this is because the data for movies is readily available and allows us to test a wide variety of different scenarios. While the overarching idea for the program is to allow many different types of data to be classified, we decided on limiting it to movies, as doing so allows us to accurately train and test the algorithm developed, without having to step through societal issues. We may have faced ethical issues if we were to categorise political posts on social media, for example, as our own political biases may have altered our perception of results.

In order for users to input their desired data, we plan to allow the user to import their own CSV file as well as tell the program which column contains the data and which column contains the classifications. The CSV file should just contain the raw data, as the program will process and remove things that may negatively affect the model’s ability to learn.

In the production of the application, we plan for the test data to be used initially, however, as more and more examples are added to the system, we plan on adding them to separate file and using real-world data that is being used within the system to help make the end result more accurate and make it more relevant to what is being processed by the system. In the end, when data is outnumbered by the actual user data, we plan on phasing it out completely in order to keep the data relevant to what actual users are using the system for.

# Solution Motivation

With the growth of flash-trends, and the popularity of things growing and then disappearing within days, a solution that enables an artificial intelligence to scale is more important today than ever before. Allowing content filters, such as the one we are creating, to automatically discover out what is growing in popularity or fading out of relevance helps keep the decision making quick but allows it to keep large amounts of data to stay accurate. The solution we are developing also allows small-medium scale operations use AI to improve their business without having to worry about updating their datasets to keep on top of constantly shifting trends.

Allowing users to set their own parameters in terms of analysing the data also allows operations to tailor their analysis to what is required. For example, if you are entering an emerging market and want to focus on improving the experience you offer your clients or customers, you could collect their feedback and format your data in order to show what is a “suggestion” and what is not and prioritise things that contain suggestions for your user experience staff to focus on addressing. This helps keep your staff working on what you want to focus on without having to go through feedback that is not relevant to your current goals.