**X. Technology Survey**

Given all the requirements that have been previously detailed, Moogle will undoubtedly need to be an incredibly robust web application. It will need to handle user input that will come in multiple forms, store large amounts of data relating to both its products and its customers, and frequently communicate with the databases that store all this data in order to perform expected web store tasks and display accurate information.

Building such an application will require the use of software tools that are both powerful and efficient. The proper selection of which tools to use is an integral decision in the development of Moogle; choosing incorrectly could result in multiple delays as well as a potentially less-than-sufficient final result. Fortunately, after performing research on many available options, we as the developers of Moogle, feel that we have found the proper tools to begin creating the web store of the future.

**X.1 Google App Engine**

The first, and perhaps the most important, of these tools is Google App Engine. Google App Engine is a platform as a service cloud computing platform for developing and hosting [web applications](https://en.wikipedia.org/wiki/Web_application) in Google-managed data centers. What this means is that the Moogle application will be built to run on top of Google’s world-class infrastructure and therefore we will not need to be concerned with provisioning and managing a data center ourselves because it is completely taken care of by Google. This allows us to focus entirely on building the best possible version of Moogle for the users by worrying solely about its interface and how the application should perform. Google App Engine allows its users to code their applications with a variety of programming languages including Python, Java, and PHP. We have chosen to use the Python programming language to create Moogle due to its intuitive syntax and its ability to perform very high level tasks in very few lines of code. This will allow us to create the final Moogle application in the shortest amount of time and with the greatest level of efficiency.

Perhaps the most important feature of App Engine from a business perspective is that it will be completely free to use. For very large scale applications, fees are charged for additional storage, bandwidth, or instance hours required by the application, but because Moogle will need time before it grows to such a large size, we will not need to concern ourselves with these fees at the current time.

Google App Engine also provides support for MySQL in the form of Google Cloud SQL. As Moogle will rely heavily on the use of databases, this is an incredibly useful feature of App Engine. A Cloud SQL instance is equivalent to a server and a single instance can contain multiple databases. Cloud SQL was designed to work well with Google App Engine and thus it is easily integrated into the code that will make Moogle work. Therefore we will be able to easily read, create, and update records within all of Moogle’s databases all within our Python code. These changes/updates will be easily automated using different Python procedures instead of having to be manually entered by the developers which can be an incredibly time-consuming task.

**X.2 Jinja**

The other software tool that we will use to build Moogle is Jinja. Jinja is a template engine for the Python programming language that allows us to use Python-like syntax within our HTML files. Jinja offers various benefits such as a powerful automatic HTML escaping system for cross site scripting prevention, template inheritance which makes it possible to use the same or a similar layout for all templates, and a debug system that can quickly detect errors in the Jinja code. All these benefits streamline the process of writing the HTML templates that will be used by Moogle to create a user-friendly interface.

One example of where Jinja will be useful in the creation of Moogle will be when it comes to our Herd Members. Since Herd Members receive a discount on all sale items in the Moogle store, we will need to display different web contents (such as sale price) to these users that aren’t normally displayed to regular users. Therefore we can use the Jinja “if” statement in the HTML template to first determine whether the user viewing the page is a Herd Member and if so, then we display the discounted price to this user. Another good example would be on the home page of the Moogle store. On this page we will want to display multiple products to the user in order to entice them to make an impulse buy. However, we would need to write the same HTML code repeatedly in order to display each of these products. However, if we instead use the Jinja “for loop”, we will be able to iterate through all the products we wish to display and pass each of these products to the same boiler plate HTML that displays these products on the web page. Because of this, we will be able to greatly reduce the amount of time spent writing HTML by reusing the same code with multiple parameters.

Through the combination of these software tools, we will be able to efficiently build Moogle into a powerful web application.