Milestone 1

Team Name: JPENJ-322

For this milestone, we will provide the details about key tools, technologies, and the process model used for this course project. Below are different divided sections that are used to provide details about the tools

Key Tools:

- Microsoft Visual Studio Code: We used Microsoft Visual Studio Code as an integrated development environment (IDE) to write all the sources code required for our librarian assistant project. We are using this IDE because it supports hundreds of languages as well as many different extensions that allow for our development process to go smoothly.

- Github: We plan to use Github as a repository for all our code, used for pushing and deploying different versions of our project. We are using Github because it allows for groups of developers to collaboratively work on the same project more efficiently. With the version control that Github provides, we are able to go back and view different versions of project in order to maintain the integrity of the original project.

- Microsoft Word: We will use Microsoft Word for the documenting our work throughout this project. Doing this will allow us to create the documentation needed to properly express and explain our topic which is a library assistance webapp. It will also help us with for explaining our ideas in more clear detail to the user. We will use screenshots and written words to document our work.

- Discord: For team coordination and communication. Allows for groups of people to be connected through a “server” which allows for members of said server to type and use voice chat. It also supports features such as share screen and turn on webcam which are helpful for collaborative work. Servers allow for the organizations of information into different text “channels.” For example, we can have channels for customer requirements, documentation, etc. This organization of data will allow for our server to be well organized and effective in sharing information about the project.

Technologies:

- Angular/Angularjs (Javascript, HTML, CSS): We are using Angular to develop our front-end for our library web application. Angular, while complex, is feature-rich and enables us to build a stylish and in-depth user interface. Additionally, it’s very easy to build the project to be cross-platform as well. We are likely going to be using Angularjs instead of regular Angular, due to the team not being as familiar with Typescript. CSS is used for styling.

- Django (Python): We are using Django for its REST API. We want to use Django to allow the browser to communicate with the back-end database. We will also use Django to host a local server for the project. We are using Django because it is easy to learn and apply to a project, meaning we can spend most of our work time on the front-end and database. It is also very secure compared to other APIs. Lastly, Django is built in python, which is an easier language to pick up.

- PostgreSQL: We will use PostgreSQL as the database for our project. Postgres is a very in-depth and scalable relational database management system. It is very capable of handling whatever library data we need to manage. Additionally, as some of our members are in CPTS 451, we will have some people with experience in Postgres.

- Node.js (Javascript): We are using Node to handle and manage Angular's packages. It's the easiest way to get Angular on a system. Additionally, we have decided that we may try to host the server publicly using Node.js.

Process Model: Prototype-Scrum/Sprint

We plan on starting with Prototyping as an assistive process. The prototyping model follows the traditional Communication->Planning->Modeling->Construction->Deployment steps, the difference is though that once we finish designing, we release a prototype. We can choose to keep the prototype and expand upon its features, or we can throw it away if the customer does not like it. This model is useful in order to gauge the exact requirements that the customer is looking for. Our goal of using this model is to have a working project as soon as possible. Then, we will add any changes we want to the backlog for our scrum process. A sprint/scrum process takes 3-4 weeks, which consist of planning, making a backlog of tasks, assessing the tasks, selecting one, developing it and reviewing the code. We repeat this cycle starting from assessing until the backlog is empty. The Prototype-Sprint process is the most desirable option for our group because it requires minimal client interaction, is flexible, and minimizes risk for our smaller project. It also does not require a huge amount of documentation, which can take away from time spent coding.