SE4347 Section 003

Database Systems

Management System for

Makerspace

Partner: Jacob Pinksa

Partner: Jared Pinksa

Partner: Anthony Ngo

Partner: Neil Barot

Instructor: Dr. Jalal Omer

12/04/22

**Table of Contents**

| Introduction……………………………………………………………………………………. | 3 |
| --- | --- |
| System Requirements……………………………………………………………………….. | 3 |
| Conceptual Design of the Database……………………………………………………….. | 6 |
| Logical Database Schema…………………………………………………………………... | 8 |
| Functional Dependencies and Database Normalization…………………………………. | 12 |
| The Database System……………………………………………………………………….. | 13 |
| Additional Queries and Views………………………………………………………………. | 16 |
| User Application Interface…………………………………………………………………… | 19 |
| Conclusions and Future Work………………………………………………………………. | 30 |
|  |  |

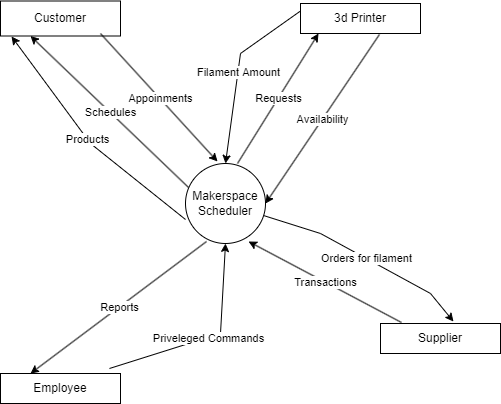
**Introduction**

WIth the emerging popularity of makerspaces as a way for anyone to be able to use professional grade equipment without investing thousands of dollars into equipment, it becomes more and more necessary to create a management tool that is able to accurately and efficiently allocate the resources of a makerspace. One of the most popular machines in makerspaces tends to be the 3D printer due to the sheer versatility of the things that one can create with it. With a bit of experience and the right filament, someone can create anything from simple desk toys all the way to complex, multi-faceted machines. Due to the popularity of 3D printers and the growing need of a management tool for Makerspaces, our team has decided to create a system that allows customers of makerspaces to easily schedule appointments and gives employees a method of checking on the status of 3D printers and order history.

Information about this system will be outlined in this document. The paper will begin by talking about the planning portion of the project. These sections will include the system requirements, the conceptual design of the database, the logical schema of the database, as well as the functional dependencies and database normalization. Afterwards, the paper will speak about the creation of the database in the database system, and the additional queries and views section. FInally, we will cover the development of the front end in the user application interface section.

**System Requirements**

Before designing the physical database for the makerspace management system, it is important to have information as to how the application will be used and who it will be used by. This step was done using a context diagram which consisted of four different entities: a customer, an employee, a supplier, and a 3D printer. This diagram will be shown below:



Each entity is shown to interact with the application in a different way. The customer is able to schedule appointments, check on previously scheduled appointments, and take a look at the various products the makerspace offers. These products primarily take the form of appointments on machines such as 3D printers. Another entity includes the employee which interacts with the application by making and viewing reports as well as by using privileged commands. The supplier also interacts with the scheduler by making up the transactions and by taking orders for filaments. Finally, the 3D printer interacts with the application by showing off its availability and filament amount, as well as by handling requests for its usage.

Knowing which entities will interact with the system also allows the developers to have a better idea of how to develop an interface that will be the most appealing and useful to the user. With the insights from the context diagram in mind, it was determined that the interface must be simple and easily navigable due to the fact that the users most likely won’t need nor want an interface that has any unnecessary fluff. They most likely just want to make or check on a reservation. Furthermore, for employees the application must have a dashboard that gives them information about things such as availability, filament type, and filament amount for each 3D printer. Since the user experience will be slightly different between employees and customers, it is important to have a section that allows one to sign in as either an employee or as a customer.

The system has multiple requirements which must be fulfilled before it can be

considered to be fully operational. These requirements can be split up into two separate categories: functional and non functional requirements. Functional requirements include any requirements that define the functionality of a product whereas non functional requirements include any considerations that must be made that do not directly affect how the product functions.

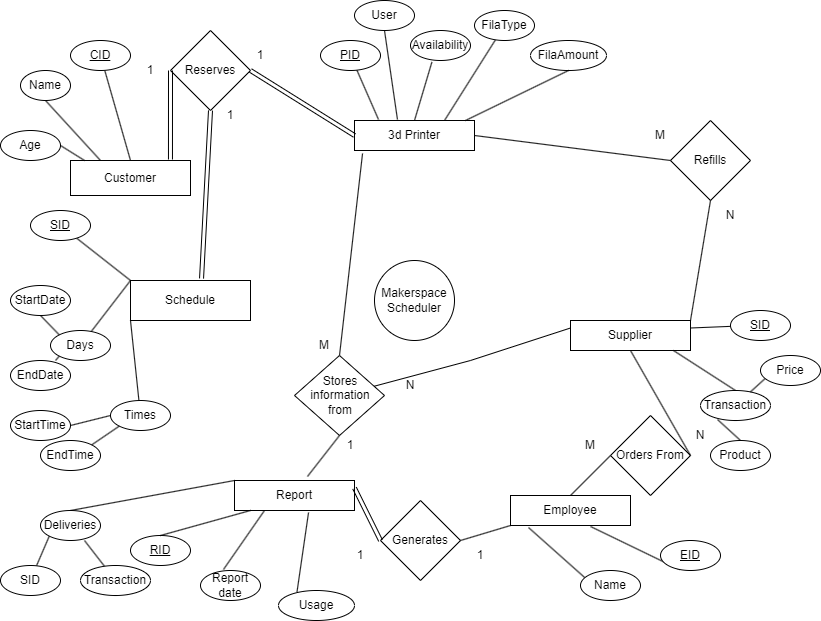
There are a few main functional requirements that were considered before developing the makerspace management tool. The first of these requirements includes the primary idea that this project was built upon, which is the ability to schedule an appointment for a 3D printing machine in a maker space. The second functional requirement includes designing the application so that it can warn the user which machines require a refill for the filament. The third requirement reports when a specific machine is available next. The fourth builds off the fourth and reports how busy each machine is. The fifth functional requirement is the ability of our program to report the supplier with the cheapest cost.

Alongside the various functional requirements, there are also a handful of non functional requirements that are integral to the development of our application. The first of these requirements is to build the machine so that it can work as expected even when multiple users are accessing the database. Alongside this, the second consideration is designing our tool so that there are no security issues. Users must not be able to compromise our database through attacks such as SQL injections. Alongside this the application must also be easily maintainable so that it can remain functional for longer periods of time, as well as scalable so that the tool can be expanded as the makerspace grows or changes. Finally, the design of the application must be intuitive and easily navigable by a user even if it’s their first time using the application.

While there may be other requirements, these are the ones that have had the largest impacts when designing our application. Over time, as our understanding of what exactly we needed our product to do changed we modified or removed some requirements. Despite this, the requirements stated above have remained constant.

**Conceptual Design of Database**

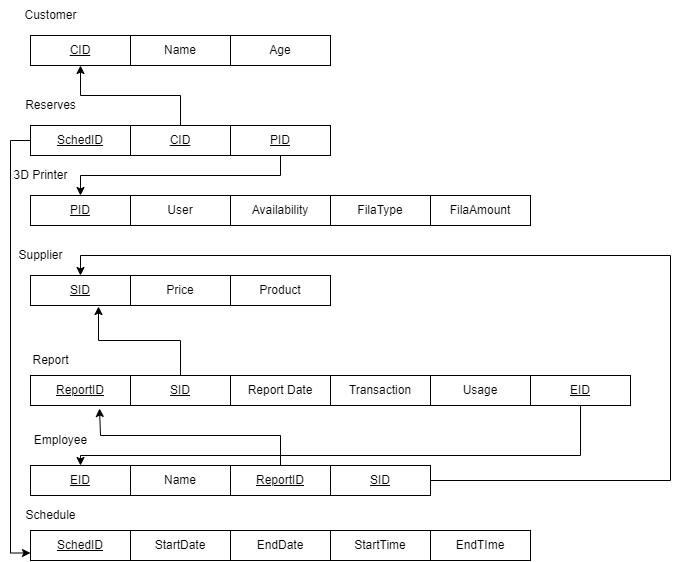
Before considering the various logistical aspects of creating a database it is important to create a conceptual design. There is no use worrying about things such as which programming language to use or how to optimize queries when things such as the elements that create the schema of the database are unknown. This is why it is important to consider things such as what entities will make up the database, how entities relate to each other, what constraints each entity has, or whether each entity is a primary or secondary key before creating the database. This was done with the help of the entity relationship model shown below.



With the ER diagram as a tool, the next step in designing the database was to understand and create a set of constraints that the data in the database must follow. The constraints implemented in the database were derived from a series of business rules that may affect the database. Examples of these business rules include things such as the fact that only people above the age of eighteen may use the makerspace, the fact that a reservation may not end before it starts, or that every 3D printer must be set to use at least one type of filament. The existence of these business rules prevent any instances of events that may cause issues for the business.

**Logical Database Schema**

The next step in the process of creating the database is to transcribe the entity relationship diagram into a more coherent form using the diagram below.

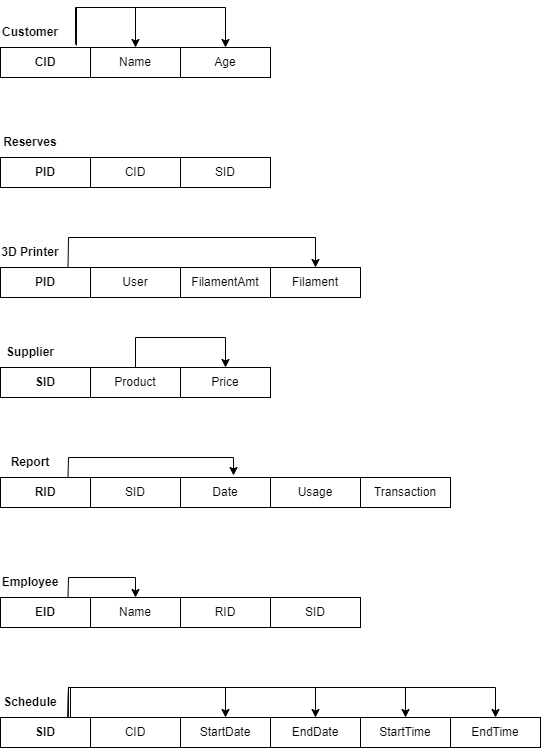


Organizing the data in this manner allows for the developer to easily convert the diagram into SQL commands in order to create the database. Alongside this, it also very clearly outlines important information such as the primary key for each table as well as the dependencies between each table. Once the table was finalized, the next step in the procedure was to develop a series of SQL commands. These are shown in the table below.

| **Customer:** |
| --- |
| CREATE TABLE Customer (  CID INT NOT NULL,  Name Varchar(30) NOT NULL,  Age INT NOT NULL,  Primary Key(CID),  CHECK (Age >= 18)  ); |
| **Reserves:** |
| CREATE TABLE Reserves (  PID INT NOT NULL,  CID INT NOT NULL,  SID INT NOT NULL,  Foreign Key(PID) References Printer(PID)  Foreign Key(CID) References Customer(CID)  Foreign Key(SID) References Schedule(SID)  ); |
| **3D Printer:** |
| CREATE TABLE Printer (  PID INT NOT NULL,  User BOOL,  FilamentAmt DOUBLE NOT NULL Default(0),  Filament varchar(30) NOT NULL Default(“PLA”),  Primary Key(PID)  Foreign Key(FilamentType) References Supplier(Product)  ON DELETE set Default ON UPDATE cascade,  ); |
| **Supplier:** |
| CREATE TABLE Supplier (  SID INT NOT NULL,  Product varchar(30) NOT NULL,  Price DOUBLE NOT NULL,  Primary Key(SID),  ); |
| **Report:** |
| CREATE TABLE Report (  RID INT NOT NULL,  SID INT NOT NULL,  Date Varchar(10) NOT NULL,  Usage DOUBLE NOT NULL,  Transaction INT  Primary Key(RID),  Foreign Key(SID) References Supplier(SID),  ON DELETE cascade ON UPDATE cascade  ); |
| **Employee:** |
| CREATE TABLE Employee (  EID INT NOT NULL,  Name varchar(30) NOT NULL,  RID INT,  SID INT,  Primary Key(EID),  Foreign Key(SID) References Supplier(SID),  ON DELETE set NULL ON UPDATE cascade  Foreign Key(RID) References Report(RID),  ON DELETE set NULL ON UPDATE cascade  ); |
| **Schedule:** |
| CREATE TABLE Schedule (  SID INT NOT NULL,  CID INT,  StartDate Varchar(10) NOT NULL,  EndDate Varchar(10) NOT NULL,  StartTime Varchar(6) NOT NULL,  EndTime Varchar(6) NOT NULL,  Primary Key(RID),  Foreign Key(CID) References Customer(CID),  ON DELETE cascade ON UPDATE cascade  CHECK(StartDate < EndDate OR (StartDate == EndDate AND StartTime <= EndTime))  ); |

The commands above will help create the framework of the database, however once this framework is created it becomes necessary to understand how much data it will contain. The largest section of the database will undoubtedly be the section responsible for storing data on customers. This section will be as long as the number of memberships that the makerspace has. Following this, the rest of the sections will be relatively small apart from the report. While the report may start small, it will grow significantly over time as the business continues to operate. The other sections such as employee and 3D printers should remain fairly static unless an employee joins or quits the team or unless a printer breaks or is added. The schedule table is fairly variable and will depend on the demand of each 3d printer. However, considering the amount of time that it takes to print out even a simple device, there likely won’t be more than 4 reservations per day per machine.

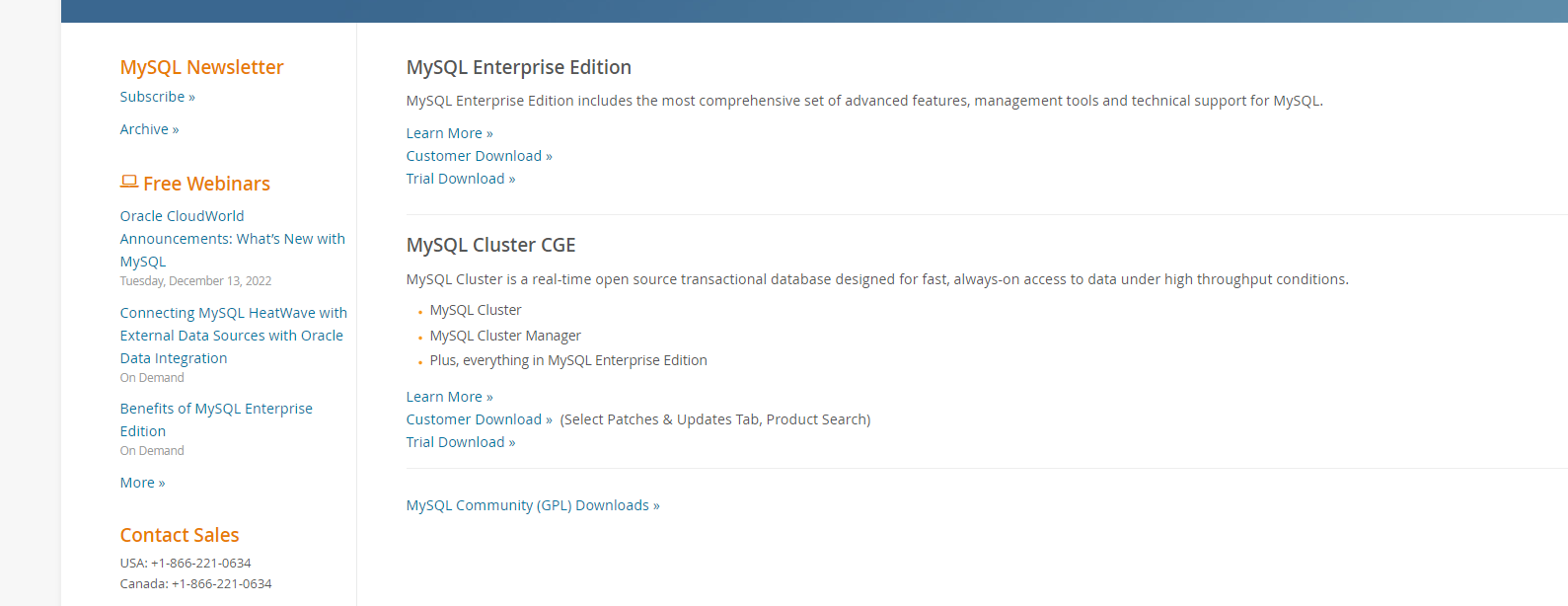
**Functional Dependencies and Database Normalization**

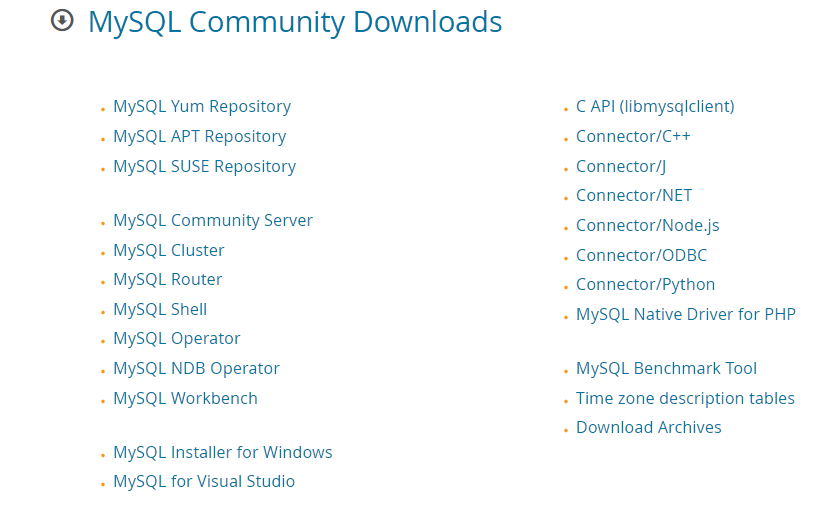


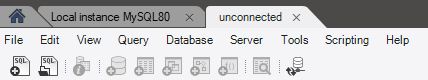
The diagram above illustrates the functional dependencies of the database. Normalizing the database leads to no changes. The database is already in 3NF, so there is no normalization process or normalized SQL statements. The tables are identified using a unique ID, which keeps the dependencies straightforward.

**The Database System**

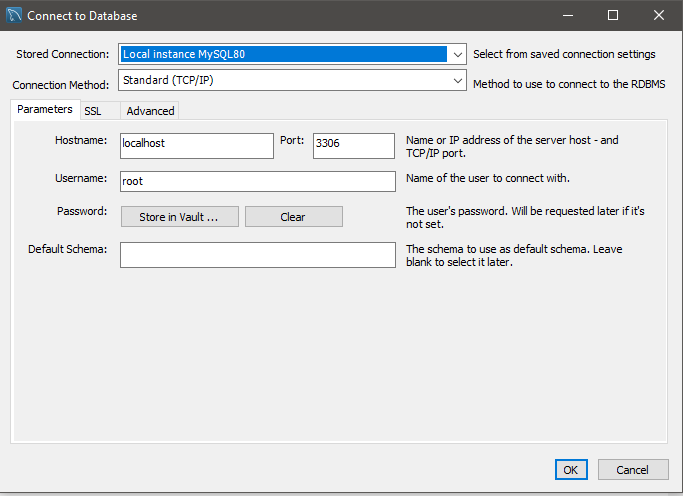
To install our system, first you must install mySQL, navigate to the download page of mysql.com and click the mySQL community (GPL) Downloads link at the bottom center of the screen capture below.



Next, click and download the mySQL Workbench from this page:

and connect to your database system by opening the database tab as you see here.

Which should open a window like this:

where you can select which database you want to connect to.Next you should use the create commands from the logical database schema section above to make the tables that hold all the relevant information you need.

To fill the tables with information you use the insert command like the following code:

INSERT INTO Customer (CID, Name, Age)

VALUES (1, "James R. Hill", 21);

INSERT INTO Customer (CID, Name, Age)

VALUES (2, "Enrique M. Brown", 35);

INSERT INTO Customer (CID, Name, Age)

VALUES (3, "Lonnie K. Timmons", 19);

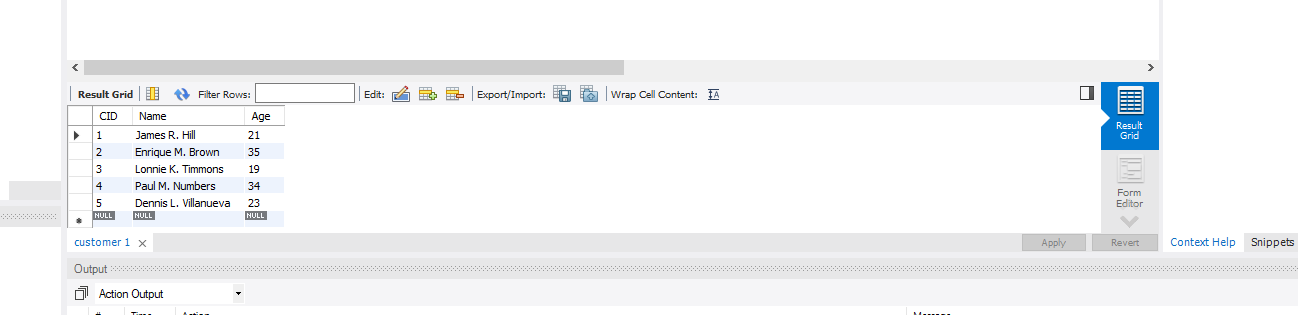
INSERT INTO Customer (CID, Name, Age)

VALUES (4, "Paul M. Numbers", 34);

INSERT INTO Customer (CID, Name, Age)

VALUES (5, "Dennis L. Villanueva", 23);

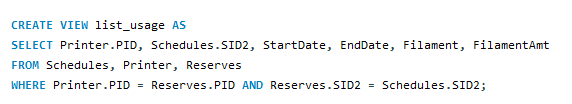
This is an example of how to insert into the Customer table which will yield the following results:



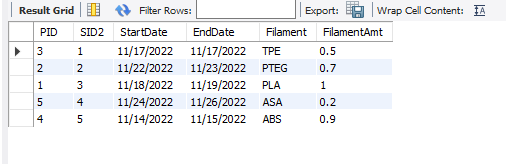
To fill the rest of the tables you can use the same command except with the proper data types.

**Additional Queries and Views**

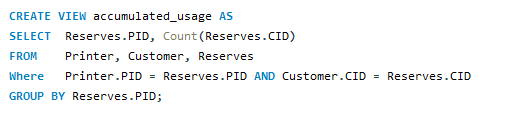
**View:**



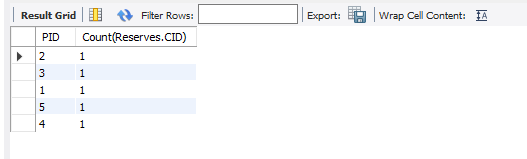
**Result:**



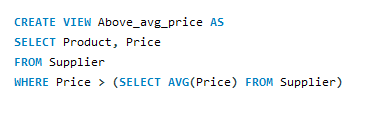
**View:**



**Result:**



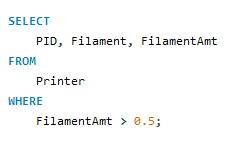
**View:**



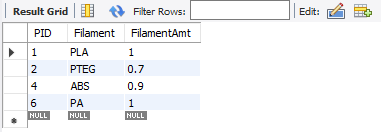
**Result:**



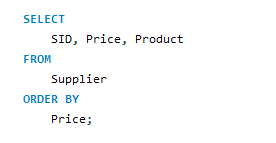
**Query:**



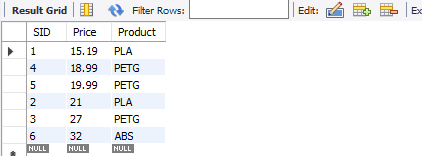
**Result:**



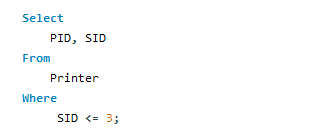
**Query:**



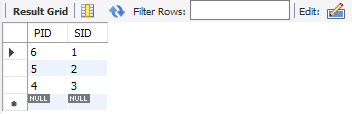
**Result:**



**Query:**



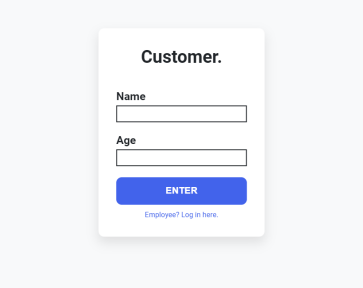
**Result:**

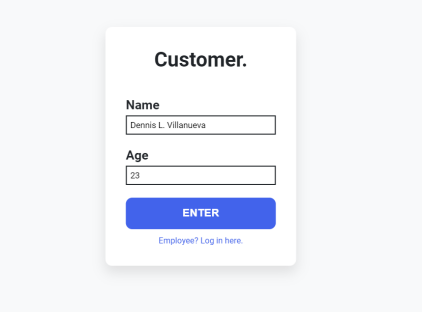


**User Application Interface**

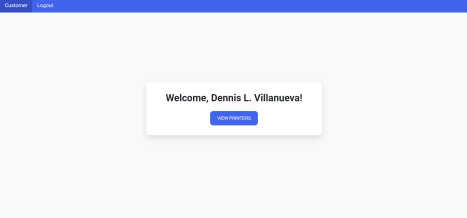
The user application interface was built using Java Server Pages (JSP) to receive input from the user and display the data retrieved from the database to them. When the user enters data into a form, that data is stored into the request body and query parameters and sent to the Java dynamic web application. The request will reach the appropriate servlet that will connect to a MySQL database. Once the servlet retrieves the data from the request body and query parameters, it will use that data to perform the appropriate query based on the user's request. For example, when a user logs in to the user application interface, they will enter their name and age. Their name and age will be sent to the dynamic web application, where it will perform a select query to the MySQL database with such input. Once the web application receives the data, if any, from the database, it will send a response back to the user with a JSP file. So, once the user logs in to the application with the correct credentials, they will be redirected to the home page.

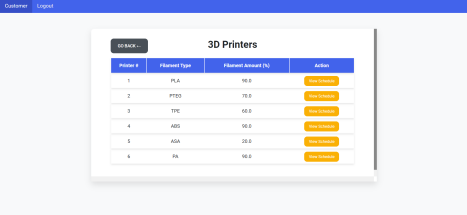
**1. Customer login page (1)**

**2. Customer login page (2)**

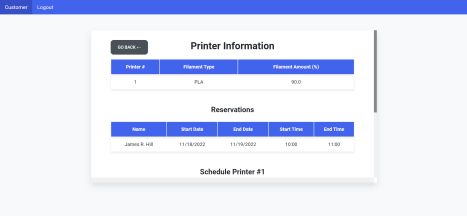
****

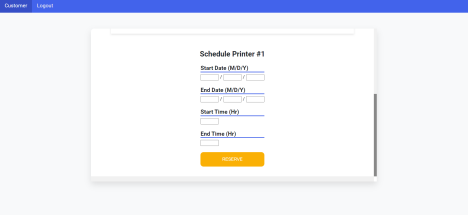
**3. Customer login success**

**4. Customer view printers**

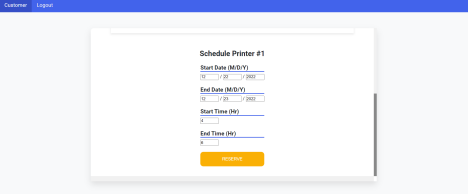
****

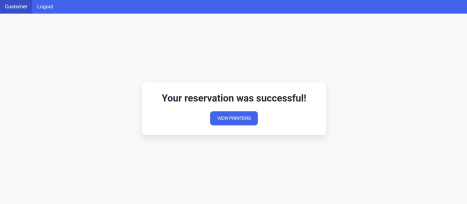
**5. View Printer #1 (1)**

**6. View printer #1 (2)**

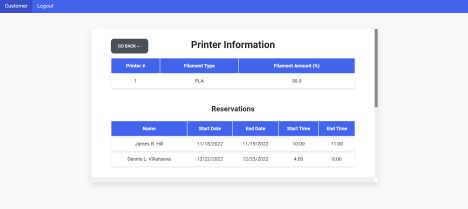
****

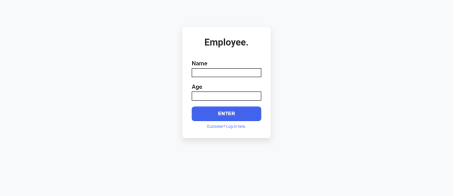
**7. Schedule reservation for printer #1**

**8. Reservation success**

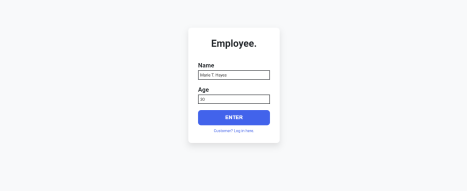
****

**9. Updated printer #1 schedule (can see that Dennis schedule appears)**

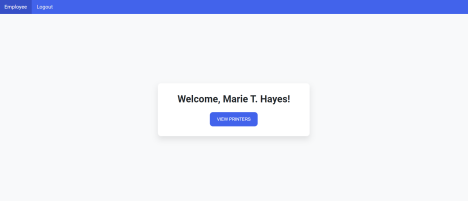
**10. Employee login page (1)**

****

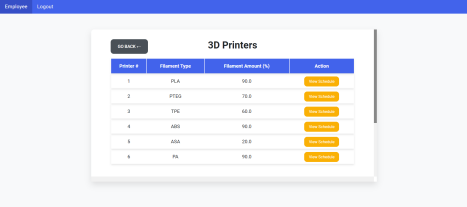
**11. Employee login page (2)**

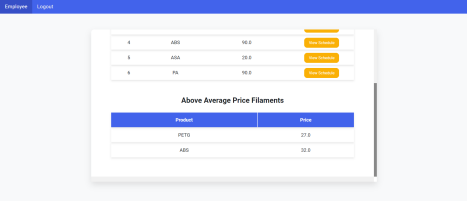
****

**12. Employee login success**

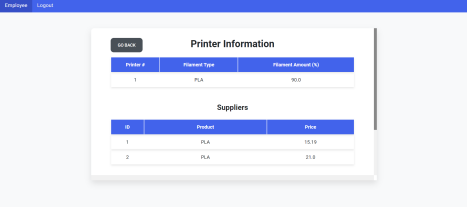
****

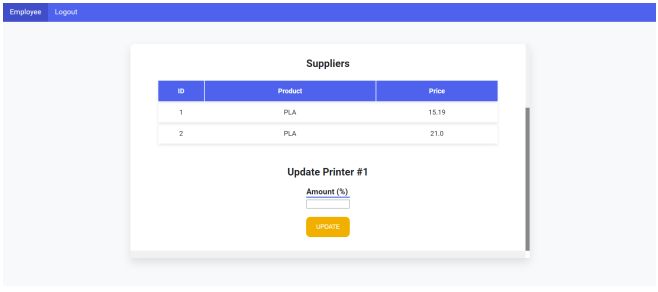
**13. Employee view printers (1)**

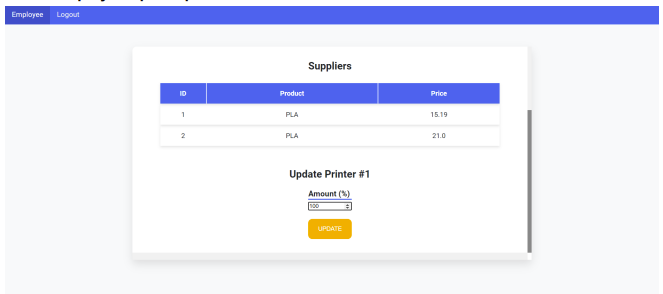
**14. Employee view above average price filaments**

****

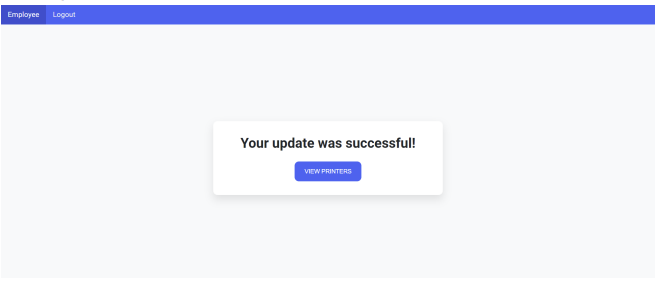
**15. Employee view printer #1 (1)**

**16. Employee view printer #1 (2)**

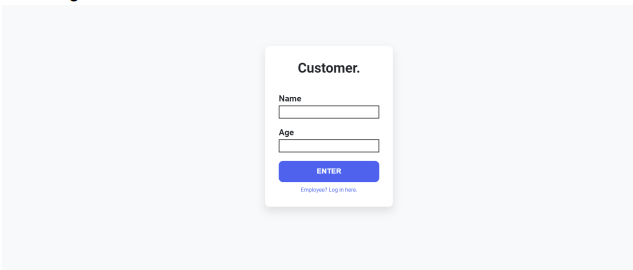
****

**17. Employee update printer filament amount **

**18. Update success**

****

**19. Logout**

****

**Conclusions and Future Work**

At the end of the project all sections of the project worked as intended. An efficient database was created which allowed users to track, schedule, and register for the makerspace, and employees to maintain the printers and records for the makerspace. Furthermore, a user interface was created which allows for users to have an intuitive experience when working with the application.

Despite all of the functionality that was included, there is always room for growth. One of the main drawbacks of our application is the lack of support for machines such as CNC, Lathe, and other machines that may take a long time to use. The reasoning for not including these for this iteration of the project is due to the fact that most of these machines will only require an hour or so of use time, but a section to schedule for these machines would be incredibly useful nonetheless.

Alongside adding support for other types of machines, the application can be further expanded by giving makerspaces a method of tracking subscriptions for users. Adding a column for subscription time and adding a constraint in the schedule which prevents users from scheduling after their subscription ends would help prevent makerspaces from losing revenue through untracked subscriptions.