CS340-Fall2020

Group 98

Team Members: Sanjay Ramanathan and Abhash Sharma

Project Step 4 Draft

Link to Working Site: <http://flip2.engr.oregonstate.edu:8042>

Feedback on Step 4 Draft

Text, letter

Description automatically generatedText, letter

Description automatically generated

Text

Description automatically generatedText, letter

Description automatically generated

Text, letter

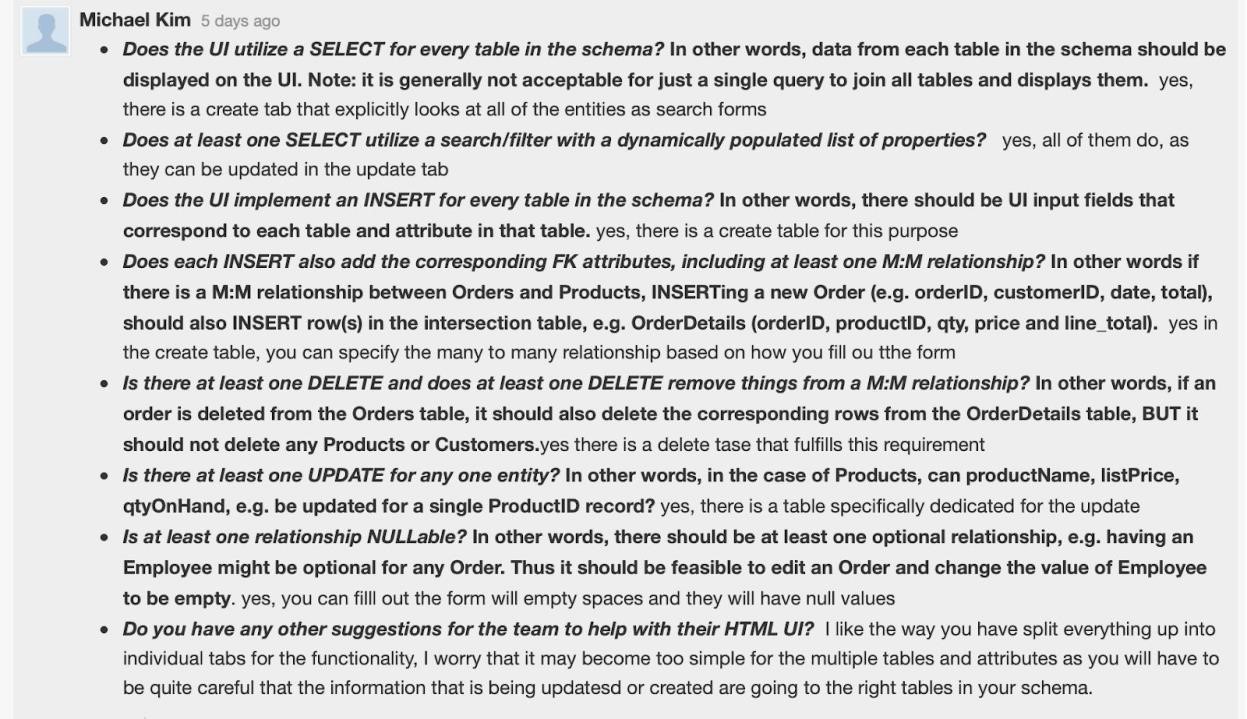
Description automatically generated

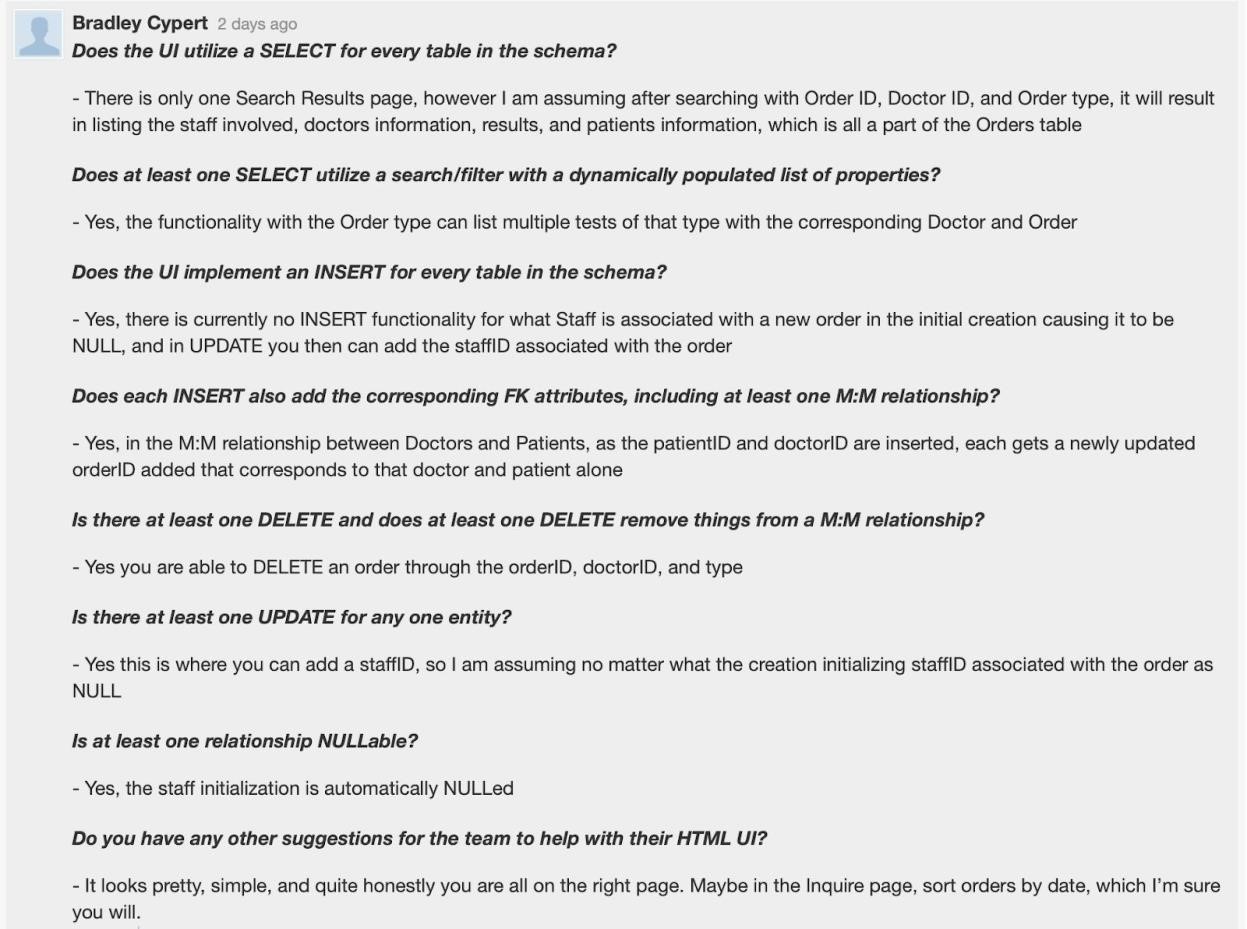
Changes Based on Feedback

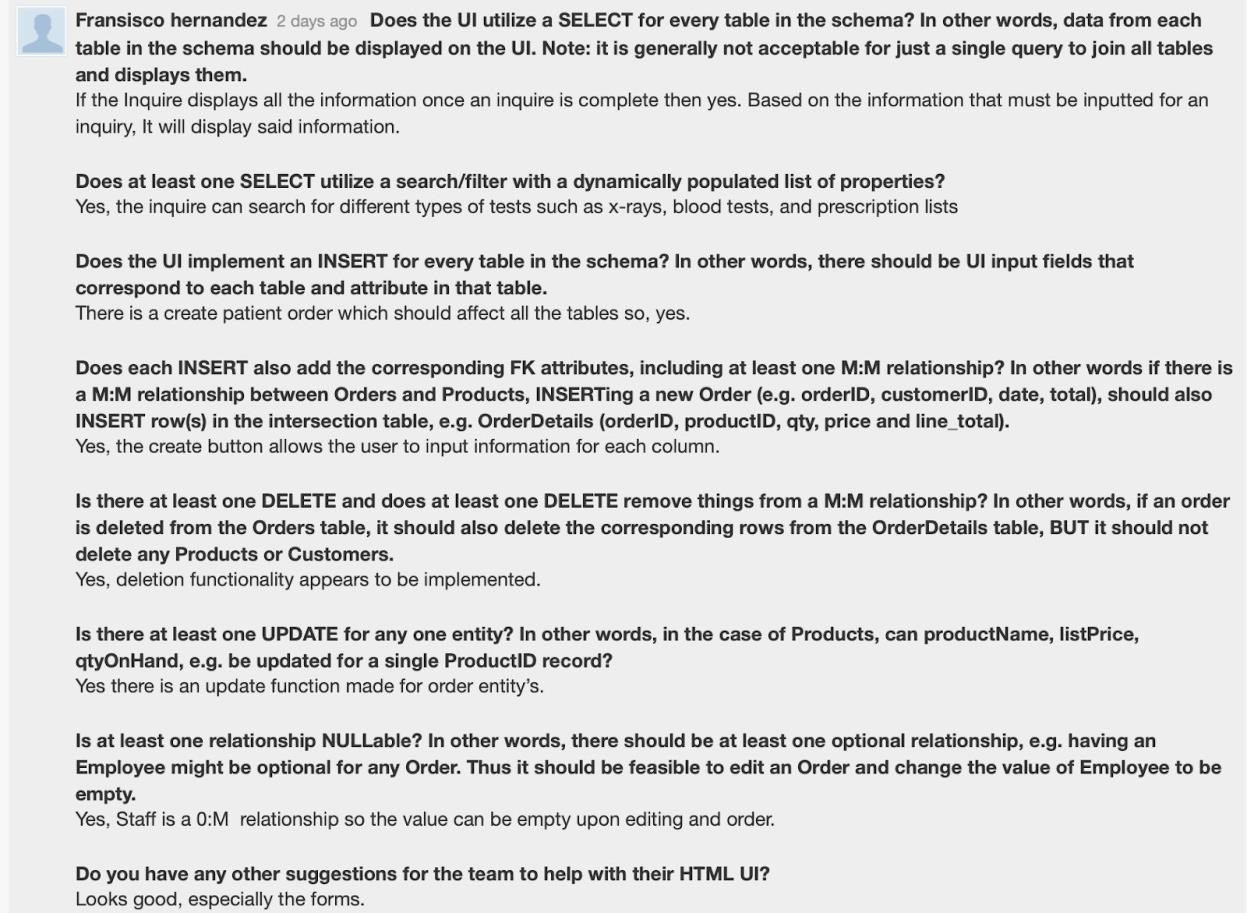
* Read queries on Doctors, Patients, and Staff Displays more than just their id value.
* Orders and Results tables correctly source into MariaDB, the issue with reviewer Roger Hill was an anomaly.

Work From Previous Steps:

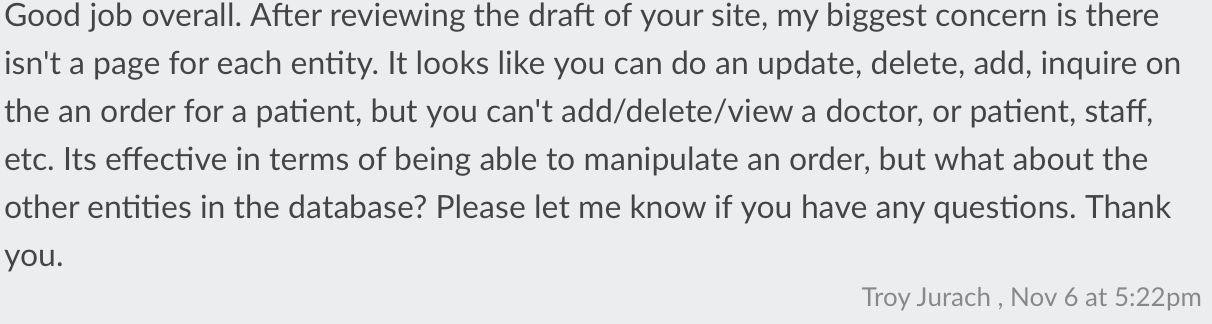
Feedback by Peer Reviewer







Comments from grader(Troy Jurach):



Actions based on Feedback

* We overhauled our website design to allow “select every table in the schema”. Now, clicking search allows users to search for doctors, orders, patients, staff, and result of orders.
* Also, to allow ‘insert for every table in the schema”, clicking add allows users to choose what they want to add to the database. There is a separate add page for each table.
* We added a sample result for every search page. It shows that the website utilizes a dynamically populated list to display results.

Upgrades to the Draft Version - The Draft version is left as is.

* Only, the name of the website is changed to reflect the overhaul in design. We changed it from DoctorBase to HospitalBase.

**CS 340 Team Evaluation Form**

**November 8, 2020**

RATE YOUR TEAMS PERFORMANCE USING THE SCALE BELOW.

**1 = Strongly Disagree 2 = Disagree 3 = Agree 4 = Strongly Agree**

|  |  |  |
| --- | --- | --- |
| GROUP NUMBER |  |  |
| NAME OF GROUP TEAM MEMBERS: |  |  |
| SCALE AND COMMENTS | RATING | ADDITIONAL COMMENTS |
| **HOW PREPARED WAS YOUR TEAM?**  Research, reading, and assignment complete | 4 |  |
| **HOW RESPONSIVE &**  **COMMUNICATIVE WERE YOU BOTH AS A TEAM?**  Responded to requests and assignment modifications needed. Initiated and responded appropriately via email, Slack etc. | 4 |  |
| **DID BOTH GROUP MEMBERS**  **PARTICIPATE EQUALLY**  Contributed best academic ability | 4 |  |
| **DID YOU BOTH FOLLOW THE INITIAL TEAM CONTRACT?**  Were both team members both positive and productive? | 4 |  |

Are there any suggestions for improvement for your team and what are your goals moving forward?

* No suggestions for improvement.
* Goals: try to shift the website into a python/flask version, if time permits.

**All Material From Previous Steps**

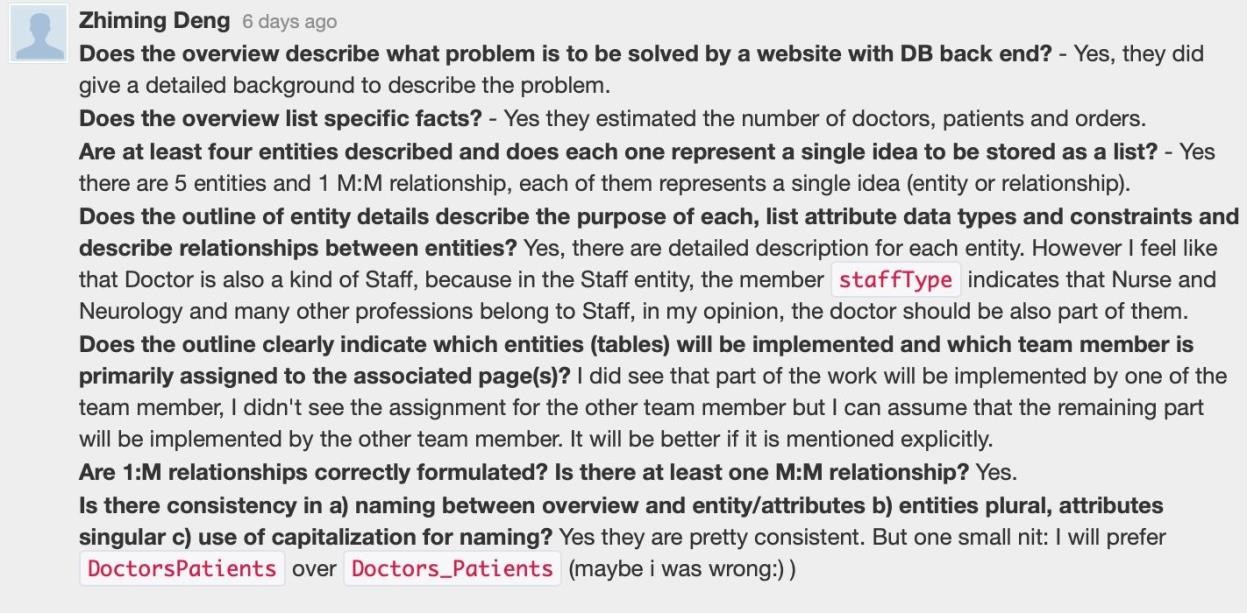
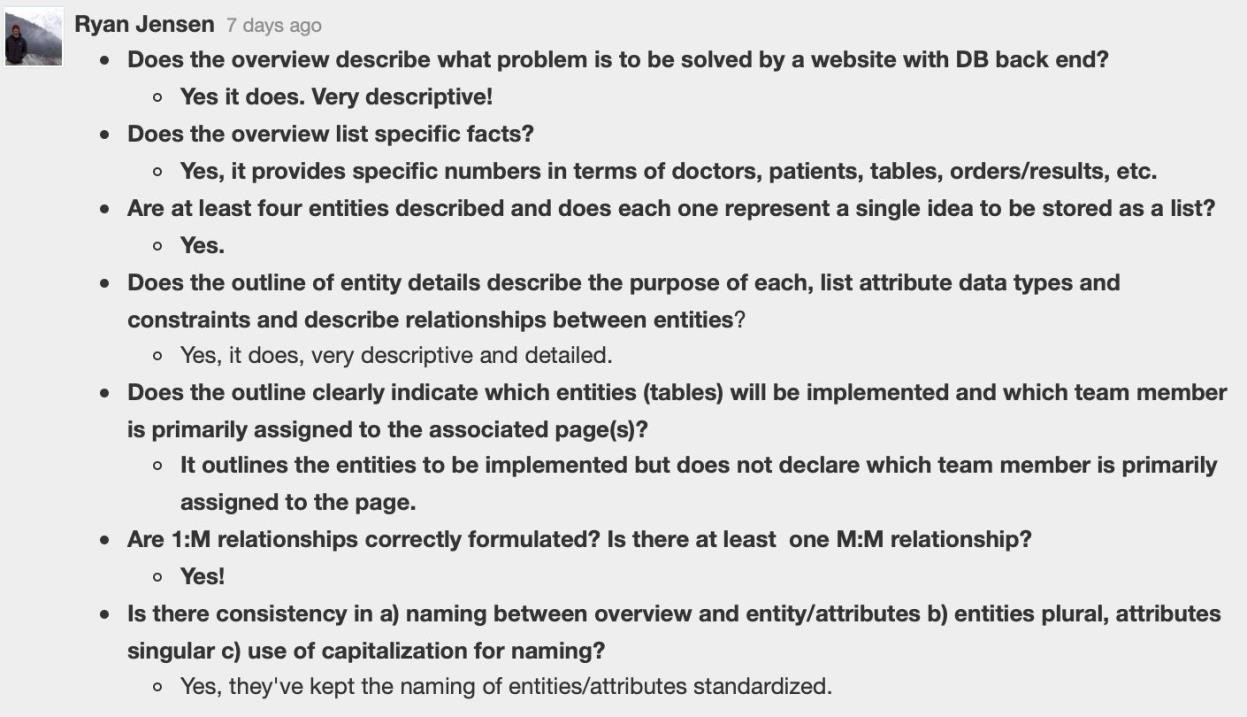
Project Step 3 Draft Version: Design HTML Interface

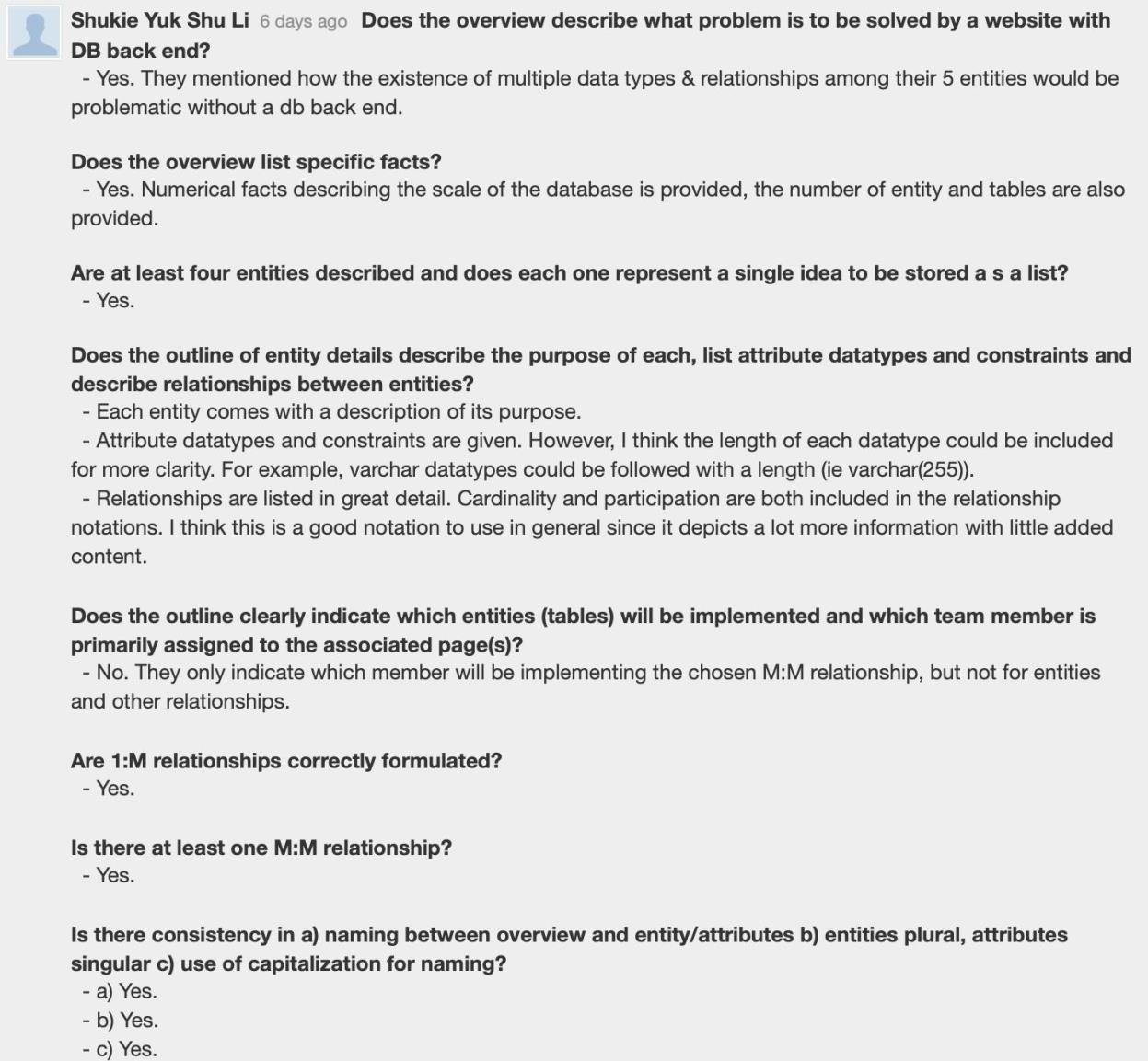
**Link:** [**http://flip2.engr.oregonstate.edu:7020**](http://flip2.engr.oregonstate.edu:7020/)

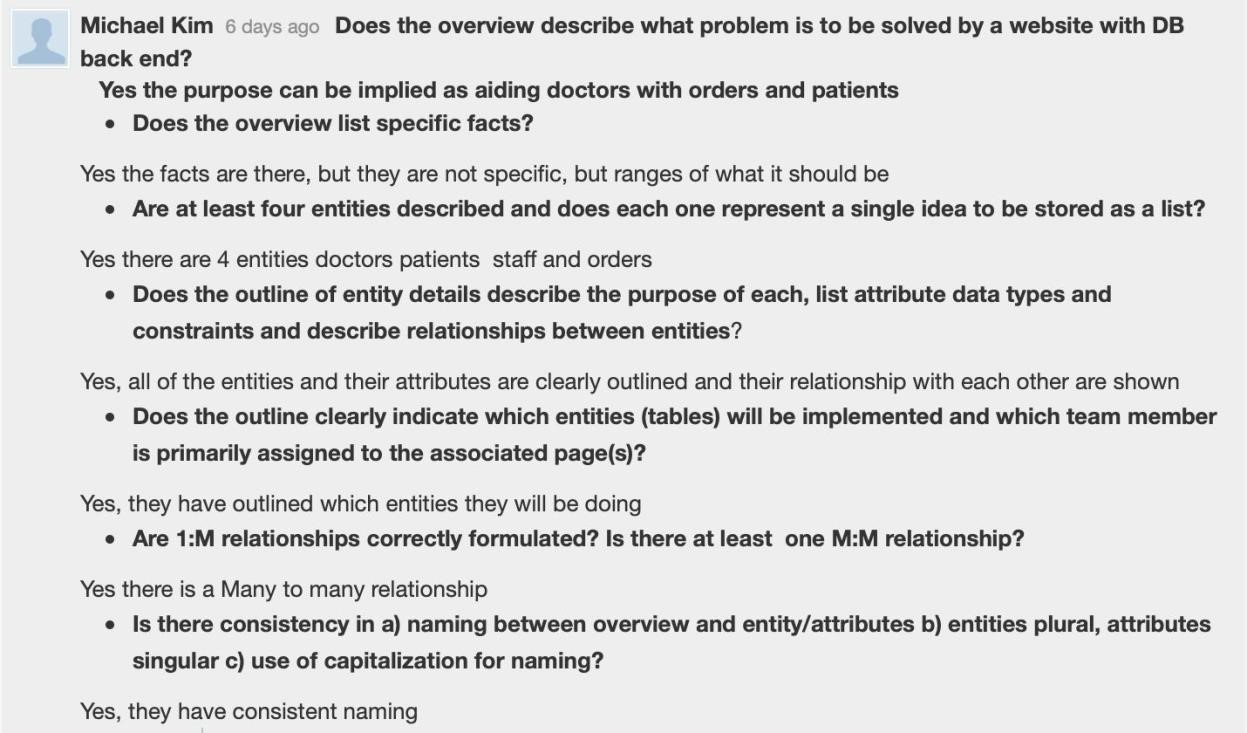
**Changes Based on Feedback on Step 2 Final Draft:**

Our TA informed us that everything seemed right with our Final Draft for Step 2. Hence, we decided to not make any changes for Step 3 Draft.

**Feedback by the peer reviewer**







**Actions based on the feedback**

* Actions not taken:
* Kept Doctors\_Patients naming convention as is because we are already comfortable with Doctors\_Patients over the suggested DoctorPatient.
* It was suggested that we must include Doctors in the Staff entity. While this is a great suggestion, we have decided to separate Doctors from other members of the hospital workforce. We will not be handling the case when a Doctor refers a patient to another Doctor.
* Actions taken:
* Now we have a member responsible for each entity table.
* Where appropriate we have added varchar(255). This would be beneficial when coding so we have decided to change it.

**Upgrades to the Draft version**

* No self-inforced design changes on the database.
* Changed the rough hand-drawn ERD and Schema with clean computer generated diagrams.

**Updated: Project Step 2 Draft Version: ERD & Schema**

1. **Fixes based on Feedback from Step 1:**
   * The grader advised us to clearly point out the entities of our database in our proposal section. Also, we were told to explain how a client could benefit from our database.
   * We have decided to fix our proposal by adding two new sections to the Overview of our proposal. First we will clearly list out our entities. Then, we will explain how our database could help potential clients.
   * Out of the two M:M relationships, only the Doctors:Patients relationship will be implemented in the final project.

1. **Project Outline and Database Outline - Updated Version:** 
   1. Overview:

A hospital consisting of 50+ doctors and staff members, 1000+ patients, and 5000+ orders and results, allows for quite a bit of complexity. There are five entities in the database: 1) Doctors, 2) Patients, 3) Staff, 4) Orders, and 5) Results. Among these 5 entities, there can be different types, multiple relationships, and numerous ways for information to get lost/mixed up.

A database with around 5-10 tables (for entities and relationships) in combination with a neat front-end display with about 3-5 pages, allows for a much easier and efficient way of viewing and managing this data.

Doctors would be able to put in Orders(additional visits, blood tests, x-ray, etc) for Patients, and assign it to a pertinent Staff. At the other end, Staff can check the Orders that need to be fulfilled, fulfill it accordingly and load up the Results of the Order.

Eventually, Doctors, in their own time, or in Patient’s next visit would be able to check on the Results of their Order.

Our Database will allow a small hospital to securely initiate, track, and fulfill orders for all Patients that decide to get treatment there. Besides that, more functionalities, where only certain types of doctors or staff can issue certain orders or change the status of the order, can be added.

* 1. Database outline, in words:

A database for a small clinic/hospital.

1. **Doctors**: Records the details of Doctors that work in Hospital. - doctorID: int, auto\_increment, unique, not NULL, PK

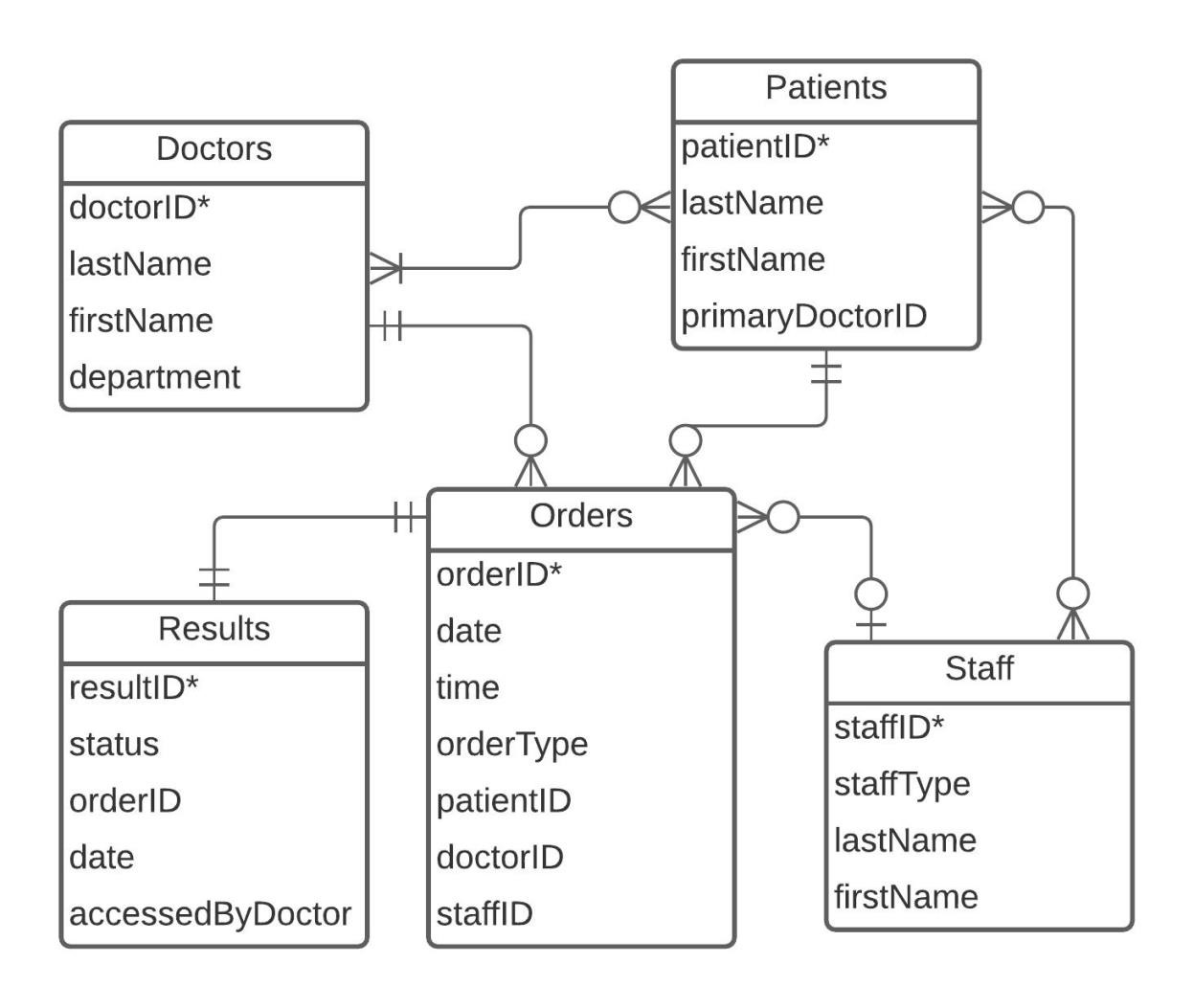
* + - lastName: varchar(255), not NULL
    - firstName: varchar(255), not NULL
    - department: varchar(255), not NULL
    - Relationship (Doctor-Patient)(0M:1M): Doctors can have zero or many Patients and a Patient can have one or many Doctors. Defined in Doctor\_Patient table.
    - Relationship (Doctors-Orders)(11:0M): An Order can come from one and only one Doctor, while a Doctor can send zero to multiple Orders, doctorID is FK in Order
    - Relationship (Doctor-Patient)(11:0M): All Patients must have one and only one primary Doctor, while a Doctor can be a primary Doctor for zero to multiple Patients, primaryDoctorID is a FK in Patient
    - Implemented by: Abhash Sharma

1. **Patients**: All the Patients who are being/have been treated by the hospital
   * patientID: int, auto\_increment, unique, not NULL, PK
   * lastName: varchar(255), not NULL
   * firstName: varchar(255), not NULL
   * primaryDoctorID: assigns Doctor to a Patient, not NULL FK
   * Relationship (Patient-Order)(11:0M): an Order can be connected to just one Patient, but a Patient can have zero to multiple Orders. patientID is FK in Order.
   * Relationship (Patient-Staff)(0M:0M): A Patient can be handled by zero or multiple Staff and vice versa. Defined in Staff\_Patient table.
   * Relationship (Doctors-Patients)(0M:1M): Doctors can have zero or many Patients and a Patient can have one or many Doctors. Defined in Doctor\_Patient table.
   * Implemented by: Abhash Sharma
2. **Staff**: All types of Staff members currently active in the hospital
   * staffID: int, auto\_increment, unique, not NULL, PK
   * staffType: (Nurse, Pharmacy, Lab Technician, Radiology, Neurology etc.), varchar(255), not NULL
   * lastName: varchar(255), not NULL
   * firstName: varchar(255), not NULL
   * Relationship (Staff-Order)(01:0M): A Staff can have zero or multiple Orders to handle. An Order can only be handled by one and only one Staff. staffID is FK in Order.
   * Relationship (Patient-Staff)(0M:0M): A Patient can be handled by zero or multiple Staff and vice versa. Defined in Staff\_Patient table. - Implemented by: Sanjay Ramanathan
3. **Orders**: All the Orders made to Patients
   * + - orderID: int, auto\_increment, unique, not NULL, PK
       - date: date, not NULL, date the Order was generated
       - time: time, not NULL, time the Order was generated
       - orderType: (prescription, x-ray, blood test, specialist appointment, etc.), varchar(255), not NULL
       - patientID: used to connect Patient to Order, not NULL, FK
       - doctorID: used to connect an Order to Doctor, not NULL, FK
       - staffID: NULL (until assignment), used to connect an Order to Staff, FK
       - Relationship (Order-Result)(11:11): An Order can have one and only one Result, and a Result can have one and only one Order. orderID is FK in Results
       - Implemented by: Sanjay Ramanathan
4. **Results**: The Results of all the Orders made to Patients
   * + - resultID: int, auto\_increment, unique, not NULL, PK
       - status (Received/in progress/Completed/Sent): varchar(255), not NULL
       - orderID: used to connect an Order to a Result, not NULL, FK
       - date: NULL, date on which the Result was generated (on which the Order was fulfilled)
       - accessedByDoctor: bool, FALSE, changed to TRUE only after doctorID in Order or patientID’s primaryDoctorID accesses the Results.
       - Implemented by: Sanjay Ramanathan
5. **Doctors\_Patients**: Assigns a Doctor to a Patient(M:M relationship table)
   * + doctorID: the Doctor being assigned to a Patient, not NULL, FK
     + patientID: the Patient being assigned to a Doctor, not NULL, FK
     + Implemented by: Sanjay Ramanathan

**Identify entities and M:M relationship to implement and assign to team members.**

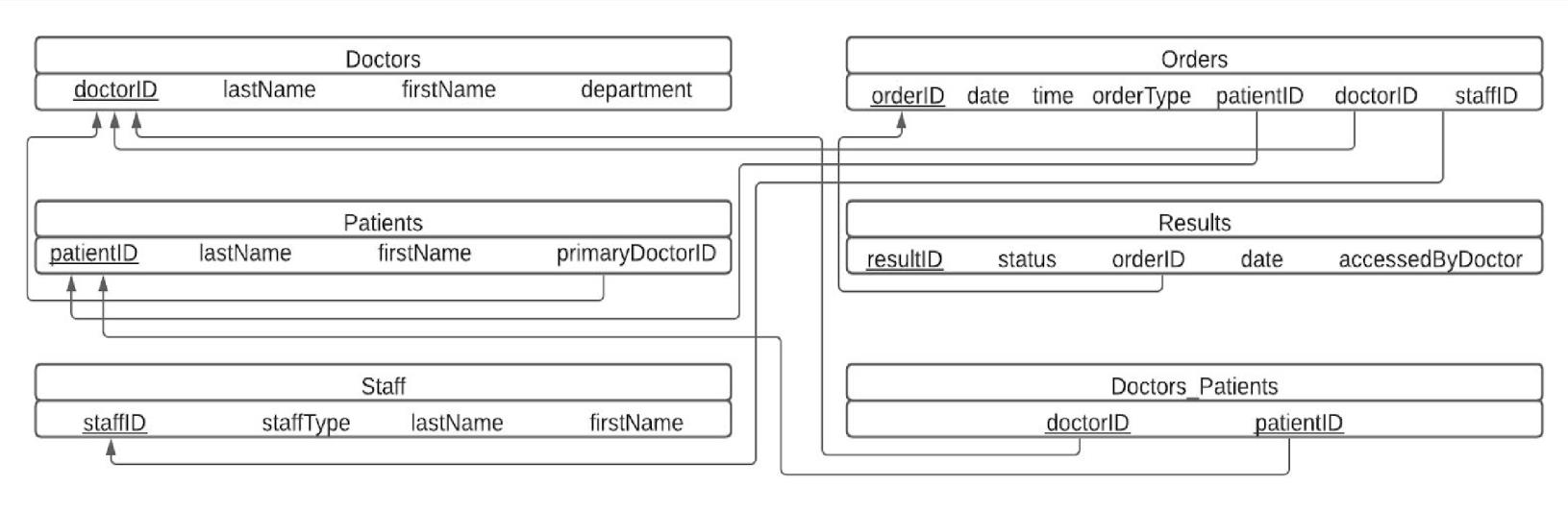
* + - The final tables we will apply in our project are Doctors, Patients, Staff, Orders, Results, and Doctors\_Patients.
    - The two many-to-many relationships from Step 1 were: 1) Between Doctors and Patients and 2) Between Staff and Patients.
    - We will implement the 1) Doctor-Patient M:M relationship.
    - Other site features such as the home page and error pages will be implemented by Sanjay Ramanathan

1. **Entity-Relationship Diagram:**



The primary key in the 5 entities here is an auto incrementing ID, named doctorID, patientID, orderID, resultID, and staffID, correspondingly. For the M:M relationship table, the primary key is the combination of the two foreign key references, named (doctorID, patientID).

1. **Schema:**



**CS340 Team Evaluation Form**

**October 28, 2020**

RATE YOUR TEAMS PERFORMANCE USING THE SCALE BELOW.

**1 = Strongly Disagree 2 = Disagree 3 = Agree 4 = Strongly Agree**

|  |  |  |
| --- | --- | --- |
| GROUP NUMBER | 98 | |
| NAME OF GROUP TEAM MEMBERS: | Abhash Sharma  Sanjay Ramanathan | |
| SCALE AND COMMENTS | RATING | ADDITIONAL COMMENTS |
| HOW PREPARED WAS YOUR TEAM?  Research, reading, and assignment complete | 4 |  |
| HOW RESPONSIVE &  COMMUNICATIVE WERE YOU BOTH AS A TEAM?  Responded to requests and assignment modifications needed. Initiated and responded appropriately via email, Slack etc. | 4 |  |
| DID BOTH GROUP MEMBERS  PARTICIPATE EQUALLY  Contributed best academic ability | 4 |  |
| DID YOU BOTH FOLLOW THE INITIAL TEAM CONTRACT?  Were both team members both positive and productive? | 4 |  |

Are there any suggestions for improvement for your team and what are your goals moving forward?

(Better communication, follow the contract better, modify the initial team contract, more contribution, etc?)?

* Going forward we will keep working as we have been till this point.
* Our goal is that this will yield the same level of success that we have been at.