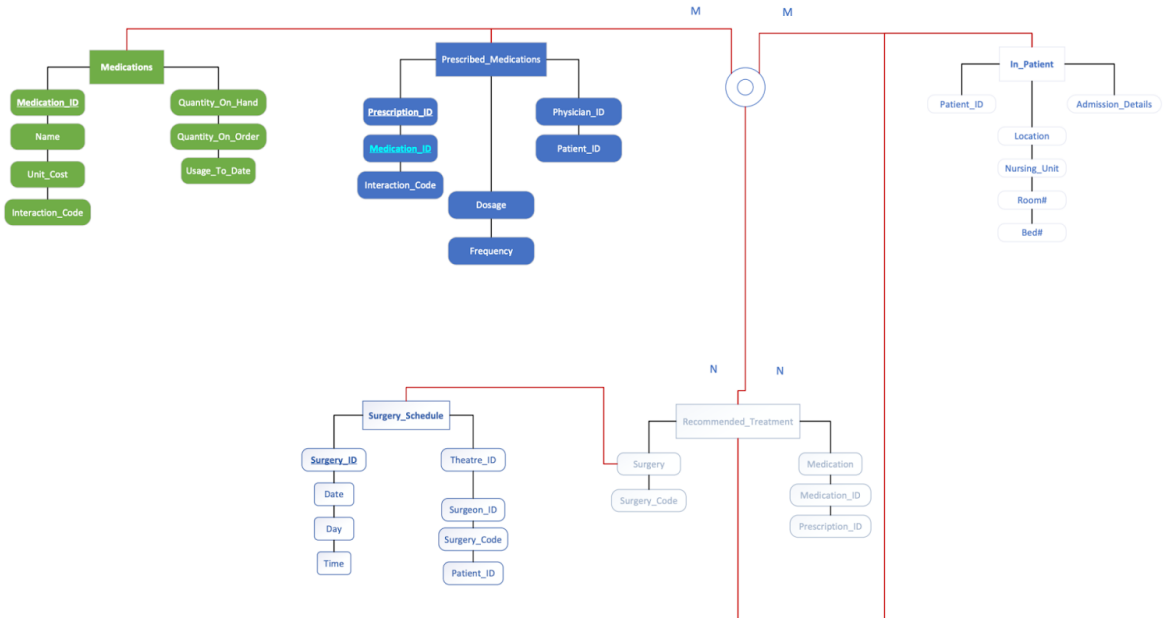
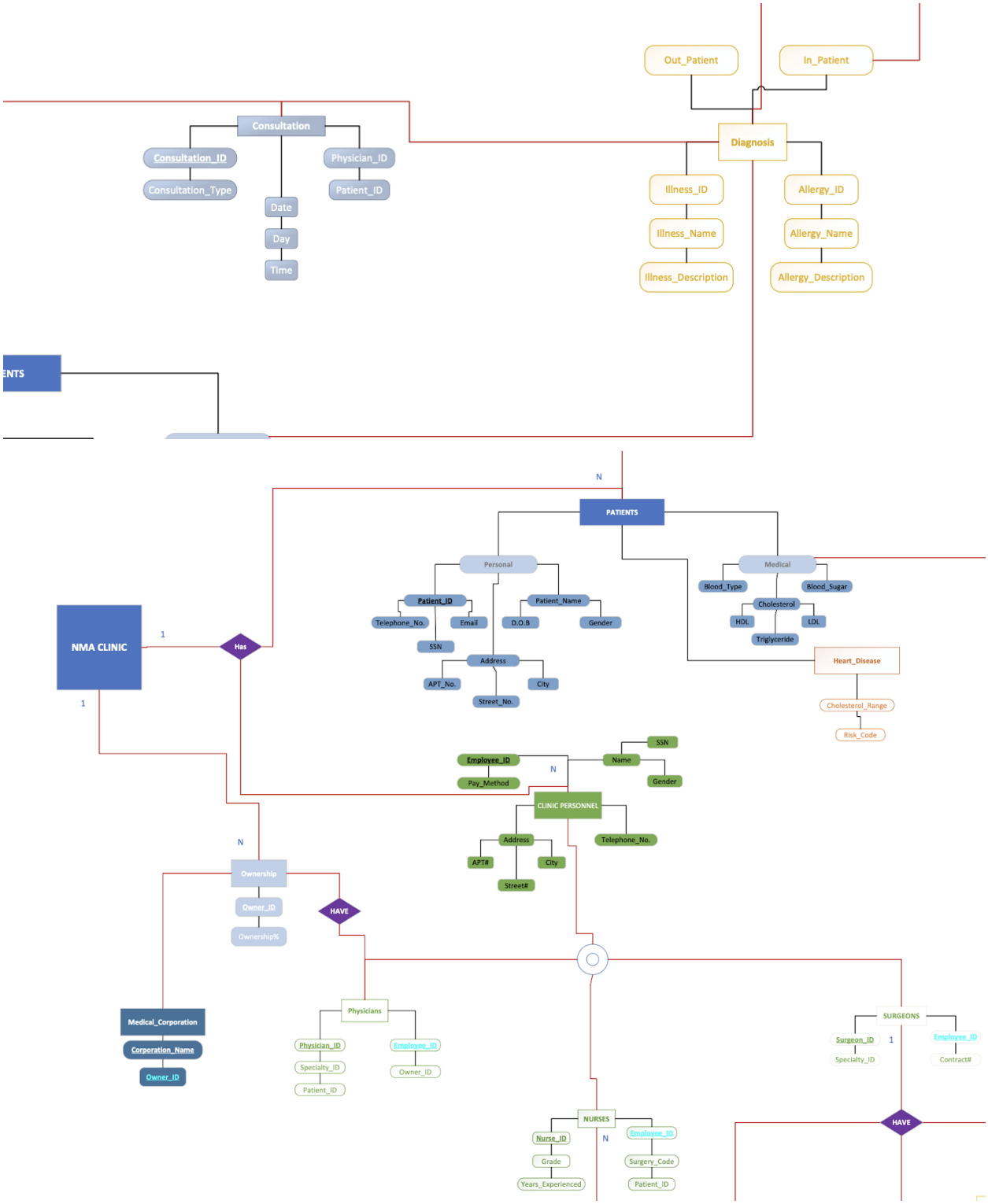
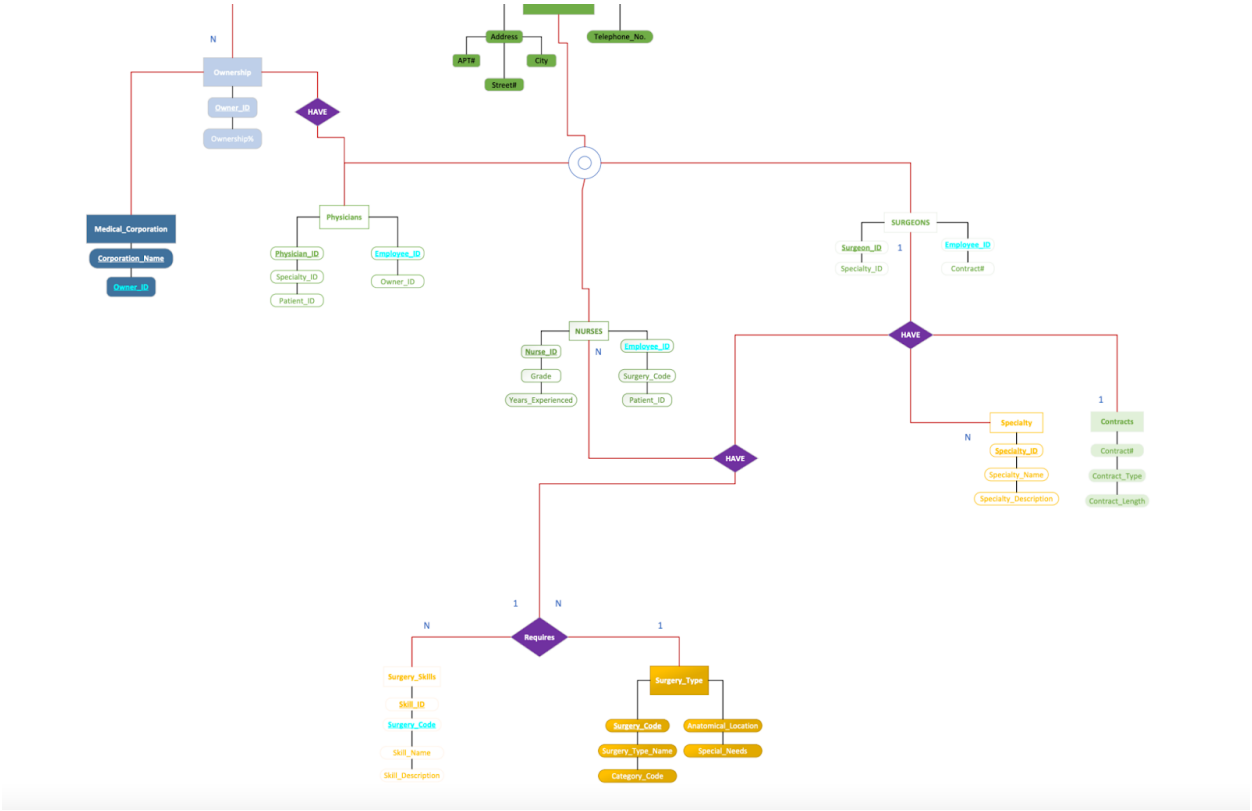
VARUN SHAH (WORKED ALONE)
CS631 Final Project Report

ER DIAGRAM:
The ER diagram is the only part of this project that was worked on by my partner The rest was all done by myself. My partner started the project 3 days before it was due (prior to the extensions that were given). I had already done the whole thing by this point so there was no further collaboration. I did not plan to do a whole final project in the last 3 days and I did not think it was acceptable.
I have attached the full pdf version separately.




Out_Patient
In_Patient
Consultation
Diagnosis
Consultation_ID
Physician_ID
Consultation_Type
Patient_ID
Date
Day
Time
Illness_ID
Allergy_ID
Illness_Name
Allergy_Name
Illness_Description
Allergy_Description
ENTS
N
PATIENTS
Personal
Medical
Patient_ID
Patient_Name
Telephone_No.
Email
D.O.B
Gender
SSN
Address
APT_No.
City
Street_No.
Blood_Type
Blood_Sugar
Cholesterol
HDL
LDL
Triglyceride
Heart_Disease
Cholesterol_Range
Risk_Code
NMA CLINIC
1
Has
1
SSN
Employee_ID
Name
N
Gender
Pay_Method
CLINIC PERSONNEL
Address
Telephone_No.
APT#
City
Street#
N
Ownership
Owner_ID
Ownership%
HAVE
Medical_Corporation
Corporation_Name
Owner_ID
Physicians
Physician_ID
Employee_ID
Specialty_ID
Owner_ID
Patient_ID
SURGEONS
Surgeon_ID
1
Employee_ID
Specialty_ID
Contract#
HAVE
NURSES
Nurse_ID
N
Employee_ID
Grade
Surgery_Code
Years_Experienced
Patient_ID



Relational Schema:

Step 1: Handling Entities

Clinic_Personnel(employee_ID, name, gender, telephone_number, SSN, Address)
Physician(employee_ID, physician_ID, specialty, owner_ID, number_of_patients)
Corporation(owner_ID, name)
Patient(patient_ID, name, gender, date_of_birth, address, telephone_number, blood_type, cholesterol, blood_sugar, surgery_required, stay_in_clinic, allergy, illness)
Inpatient(patient_ID, name, surgery_required, room, bed, nurse)
heart_disease(patient_ID, cholesterol_ratio, risk)
Nurse(employee_ID, nurse_ID, surgery_type, skill_ID, patient_ID, grade, years_of_experience)
surgery_type(surgery_ID, skill_ID, name)
support_staff(employee_ID)
Surgeon(employee_ID, surgeon_ID, contract_ID, specialty_ID)
specialty(specialty_ID)
Surgery(surgery_ID, surgeon_ID, nurseID)
contract(contract_ID, type_of_contract, contract_length)
Medication(Medication_code, name, on_hand, on_order, unit_cost, usage)
prescribed_medication(Medication_code, physician_ID, patient_ID, dosage, frequency)
Surgery_Schedule(Surgery_ID, date, surgeon_ID, patient_ID)
Surgery_skills(skill_ID)

Step 2: Weak Entities

Inpatient(patient_ID, name, surgery_required, room, bed, nurse)
heart_disease(patient_ID, cholesterol_ratio, risk)

Step 3: One to One relationships
A surgeon can only have 1 contract and 1 contract can only belong to 1 surgeon.
- - -Surgeon(employee_ID, surgeon_ID, contract_ID, specialty)
contract(contract_ID, type_of_contract, contract_length) - - -

Step 4: One to N relationships
Surgeon(employee_ID, surgeon_ID, contract_ID, specialty_ID)
specialty(specialty_ID)

Nurse(employee_ID, surgery_type, skill_ID, patient_ID, grade, years_of_experience)
Surgery_skills(skill_ID)

Step 5: M:N relationships
Nurse(employee_ID, surgery_type, skill_ID, patient_ID, grade, years_of_experience)
surgery_type(surgery_ID, name)

Patient(patient_ID, name, gender, date_of_birth, address, telephone_number, blood_type, cholesterol, blood_sugar, surgery_required, stay_in_clinic, allergy, illness)
Medication(Medication_code, name, on_hand, on_order, unit_cost, usage)
Step 6: Multivalued Attributes
Surgery(surgery_ID, surgeon_ID, nurseID) must have at least 2 nurses
Nurse(employee_ID, nurse_ID, surgery_type, skill_ID, patient_ID, grade, years_of_experience) multiple skills. Multiple patients.
surgery_type(surgery_ID, skill_ID, name) can be multiple skills

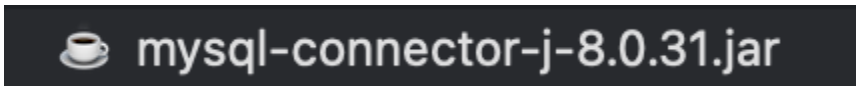
Step 7: Higher Order relationships
none

Step 8: Specialization
Clinic personnel is split into: physicians, surgeons, nurses and support staff.

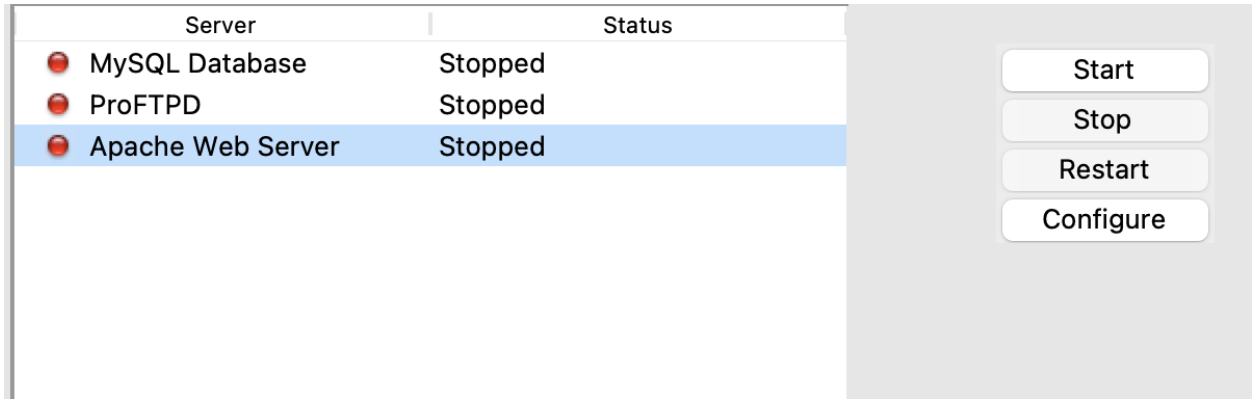
Step 9: Aggregation

There is no aggregation.

System Requirements:
I used Apache Netbeans 14 IDE for my Java programming. These were JFrame forms as they have the Java GUI component. I added a mysql-connector library to my IDE so that I could connect to a database using mysql.



I used XAMPP to connect to my localhost phpMyAdmin.



MySQL database and apache web server will both be running

That is all I used to create my application.

PROGRAM:
I will add all my java files so you can see the code if you would like to.
I had a main page like a welcome page to the hospital. You can manage staff, patients and room/bed. I used java and java gui for this project. I used XAMPP to connect the project to my localhost with phpMyAdmin to have a database. It was a local database.
All the application requirements outlined were accomplished.
I programmed this on netbeans. Also I added a java library to be able to connect with mysql. So netbeans, the java library, and xampp are needed to run the program.

My program was fine with design. I centered all of the pages so it looked nice when a new button was pressed so it was easy to find and no searching was needed.
I liked how I added a table on every page so you could always see what was being added and it made it very easy to delete and update entries.

These are the sql commands I used to populate the tables:

```
Statement s = con.createStatement();
rs = s.executeQuery("select MAX(patientnumber) from Patient");
rs.next();
rs.getString("MAX(patientnumber)");

if(rs.getString("MAX(patientnumber)") == null){
    patientnum.setText("1");
}
else{
    int id = Integer.parseInt(rs.getString("MAX(patientnumber)"));
    id++;
    patientnum.setText(Integer.toString(id));
}
```

```
nt("select * from Patient");
```

update Patient set name = ?, gender = ?, dateofbirth = ?, address = ?, phone = ?, bloodtype = ?, cholesterol = ?, bloodsugar = ?, surgeryrequired = ?, stayinclinic = ?, allergy = ?, illness = ?, doctor = ?, nurse = ? where patientnumber = ?

```
("insert into Previous(name,allergy,illness)values(?,?,?)");
```

```
:("delete from Patient where patientnumber = ?");
```

```
Statement s = con.createStatement();
rs = s.executeQuery("select MAX(staffnumber) from Staff");
rs.next();
rs.getString("MAX(staffnumber)");

if(rs.getString("MAX(staffnumber)") == null){
    staffnum.setText("1");
}
else{
    int id = Integer.parseInt(rs.getString("MAX(staffnumber)"));
    id++;
    staffnum.setText(Integer.toString(id));
}
```

```
t("select * from Staff");
```

```
:("insert into Staff(staffnumber,name,gender,phone,ssn,address, type, shift)values(?,?,?,?,?,?,?,?)");
```

```
ent("update Staff set name = ?, gender = ?, phone = ?, ssn = ?, address = ?, type = ?, shift = ? where staffnumber = ?");
;
```

```
t("delete from Staff where staffnumber = ?");
```

```
rs = s.executeQuery("select MAX(roombednumber) from RoomBed");
rs.next();
rs.getString("MAX(roombednumber)");

if(rs.getString("MAX(roombednumber)") == null){
    roombednum.setText("1");
}
else{
    int id = Integer.parseInt(rs.getString("MAX(roombednumber)"));
    id++;
    roombednum.setText(Integer.toString(id));
}
```

```
("select * from RoomBed");
```

```
ıent("update RoomBed set roomnumber = ?, bednumber = ?, patient = ?, date = ?, surgeon = ? where roombednumber = ?");
```

```
ıt("insert into RoomBed(roombednumber,roomnumber,bednumber,patient,date,surgeon)values(?,?,?,?,?,?)");
```

```
nt("delete from RoomBed where roombednumber = ?");
```

```
("select * from RoomBed where roomnumber = ?");
```

Those are all of my sql commands used throughout the program.
The question marks get filled in by a java command.

```
pst.setString(1, pr);
rs = pst.executeQuery();
```

This is an example.

```
PreparedStatement pst;
ResultSet rs;
```

To establish a connection between the program and my local database I used a function called connection.

```
public void Connect(){
    try {
        Class.forName("com.mysql.jdbc.Driver");
        con = DriverManager.getConnection("jdbc:mysql://localhost:3307/Clinic", "root", "");
    } catch (ClassNotFoundException ex) {
        Logger.getLogger(Patient.class.getName()).log(Level.SEVERE, null, ex);
    } catch (SQLException ex) {
        Logger.getLogger(Patient.class.getName()).log(Level.SEVERE, null, ex);
    }
}
```

SAMPLE OUTPUT/DATABASE:

The sample output is shown in the presentation. Not very easy to show it in this format without video walkthrough. I have met all the requirements for this project listed.

patientnumber	name	gender	dateofbirth	address	phone	bloodtype	cholesterol	bloodsugar	surgeryrequired	stayinclinic	allergy	illness	doctor	nurse
1	d	Male	1999-03-21	727	2014557283	A-	100	100	Yes	Yes	none	none		
2	bob	Male	1992-03-19	998	2018938283	O-	100	100	No	Yes	none	non		
3	bobb	Female	1994-04-12	909	2015556738	AB+	103	105	No	No	non	non		
4	pop	Male	1993-03-19	019	2018923482	A-	100	100	No	No	non	non	john	
5	efe	Male	1997-02-02	565	2013432323	A+	115	115	No	No	no	no		
6	gtgtgt	Male	1994-03-04	928	2013458392	A+	234	345	Yes	Yes	no	no		bo
7	kokko	Male	1900-01-23	828	828	A+	123	123	Yes	Yes	nep	noop		
8	Robert	Male	1989-04-05	83 5th avenue	2017263849	O+	100	100	No	No	peanut	none	jospeh	

This is an example of what my patient table looked like after a few insertions and deletions shown during the video presentation.

name	allergy	illness
d	none	none
bob	none	non
bobb	non	non
pop	non	non
efe	no	no
gtgtgt	no	no
kokko	nep	noop
Robert	none	none
Robert	none	none
Robert	pollen	none
Robert	peanut	none

This is an example of my previous table which records all previous diagnoses of patients.

roombednumber	roomnumber	bednumber	patient	date	surgeon
1	1	1		1900-01-02	
2	1	2	bob	1900-01-01	
3	2	1		1900-01-03	
4	2	2			john

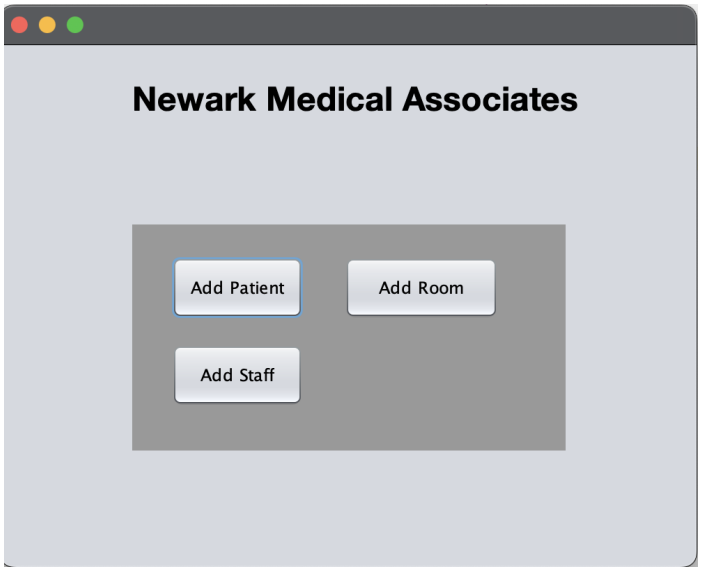
This is an example of the table that shows available or non-available rooms and beds.

staffnumber	name	gender	phone	ssn	address	type	shift
1	bobbo	Male	345	231	456	Physician	
2	kokok	Female	222	222	222	SupportStaff	
3	lopop	Male	333	333	333	Surgeon	
4	ssdds	Male	444	444	444	Nurse	
5	joj	Male	20143	345	345	Surgeon	0

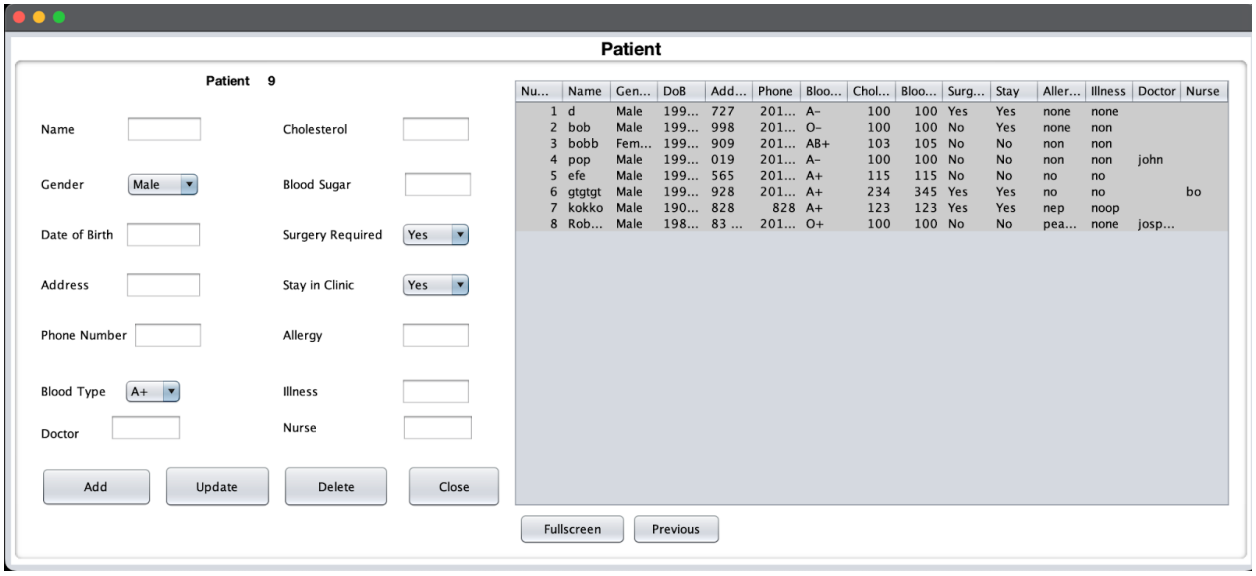
This is an example of the table for Staff insertions and deletions.

DESIGN:

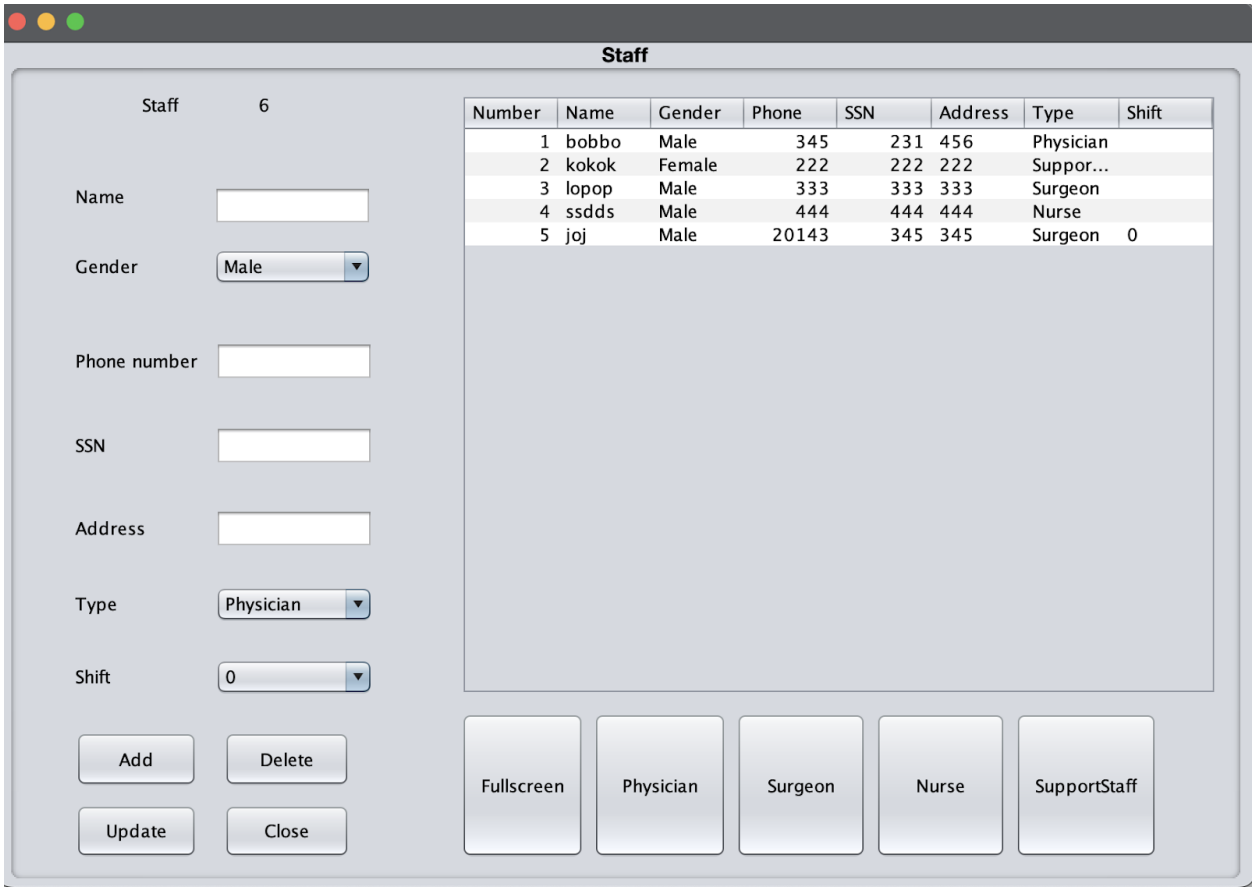
I kept most of my program in grey, black and white so it was simple. It is easy to enter information and see it get updated. It is very user friendly. It looks nice and simple and honestly I like it. The new pages open and close properly and anyone can understand how to use it immediately. All of the pages are centered for easy use. I did not face any problems creating my design. It was pretty easy to do it all with JFrame forms on Java. I just chose the basic, grey,black and white since it's easy to see and anyone will be able to use it without issues. I chose to have a table on every page so it is easy to see everything at all times. I personally like this more than having to go to a whole different page just to see what is entered. It is easy to update the tables since you just have to click on the table already available on the page. So it is easy to see all the information. I used combo boxes when the input could only be certain inputs. I used buttons since that is how I like it. I put bold letters on top of each page for easy knowledge of what page is being looked at.



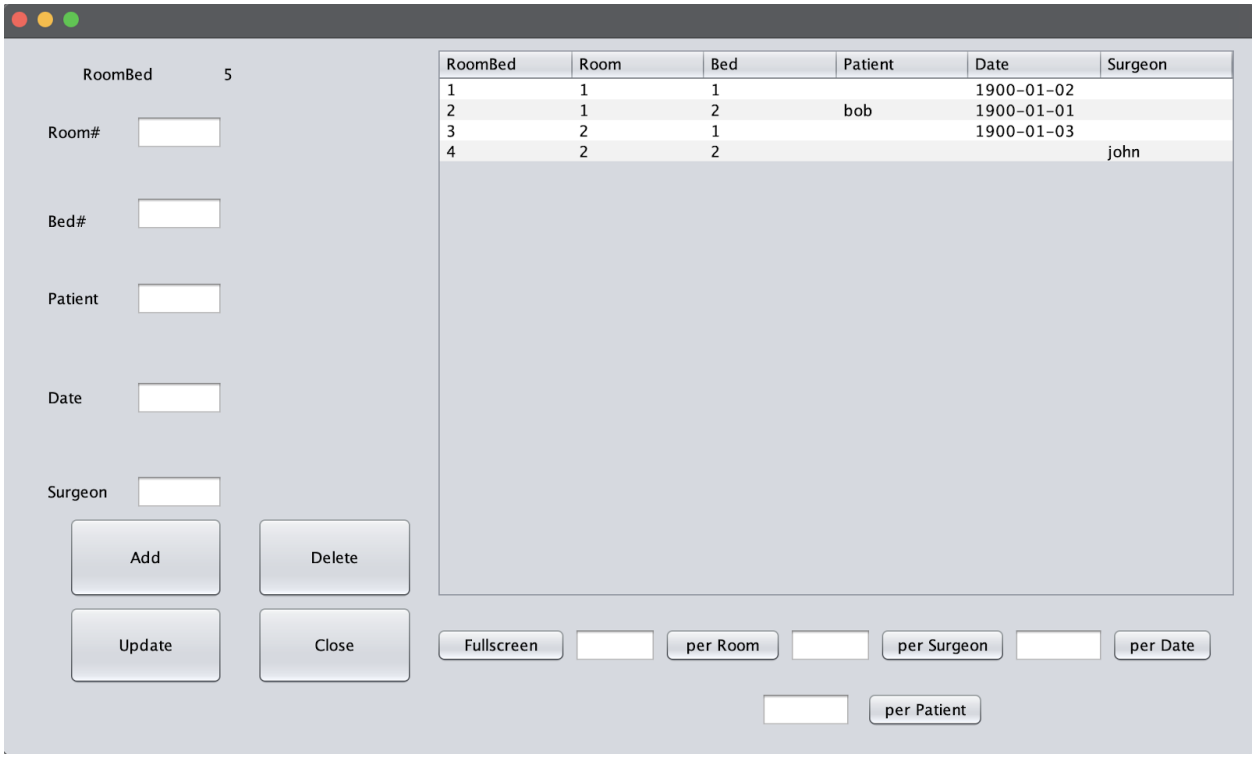
This is the welcome page when you start the application.



This is the patient page. You can see all the information easily without too much navigation.



This is the staff page. You can see all the information easily without too much navigation.



This is the roombed page. You can see all the information easily without too much navigation.

All of these pages are shown more in-depth on the video presentation so all the aspects are covered.

Application Requirements:

Patient management

Insert a new patient
My program successfully adds new patients

View patient information
My program allows you to view the patient information almost at all times. You can also fullscreen it if you please.

Schedule an appointment with a Doctor
You are able to assign a doctor to a patient and schedule with them.

Check previous diagnoses and illnesses
There is a button labeled “previous” that allows you to see all previous diagnoses and illnesses.

View scheduled per doctor and per day
You can check by doctor and by day.

In-patient management

Check for available room/bed
You can see all available rooms and bed easily in the table.

Assign/remove a patent to a room/bed
This is able to be completed

Assign/remove a doctor to a patient
This is able to be completed

Assign/remove a nurse to a patient
This is able to be completed

View scheduled surgery per room and per day
This is able to be completed

View scheduled surgery per surgeon and per day
This is able to be completed

Book a surgery
This is able to be done. Just add a time to the room with surgeon name,

View scheduled surgery per patient
Yes, just filter by patient which is easy to do.

Medical staff management

Add/remove a staff member
It is easy to add a staff member

View stuff member per job type
You can view a staff member at all times on the staff page. You can also full screen. There are buttons for each job type.

Schedule job shift
You can schedule a job shit by selecting which shift you want to assign to a staff member.

Appendix:
These are the sql commands I used to populate the tables:

```
Statement s = con.createStatement();
rs = s.executeQuery("select MAX(patientnumber) from Patient");
rs.next();
rs.getString("MAX(patientnumber)");

if(rs.getString("MAX(patientnumber)") == null){
    patientnum.setText("1");
}
else{
    int id = Integer.parseInt(rs.getString("MAX(patientnumber)"));
    id++;
    patientnum.setText(Integer.toString(id));
}
```

```
nt("select * from Patient");
```

update Patient set name = ?, gender = ?, dateofbirth = ?, address = ?, phone = ?, bloodtype = ?, cholesterol = ?, bloodsugar = ?, surgeryrequired = ?, stayinclinic = ?, allergy = ?, illness = ?, doctor = ?, nurse = ? where patientnumber = ?

```
("insert into Previous(name,allergy,illness)values(?,?,?)");
```

```
:("delete from Patient where patientnumber = ?");
```

```
Statement s = con.createStatement();
rs = s.executeQuery("select MAX(staffnumber) from Staff");
rs.next();
rs.getString("MAX(staffnumber)");

if(rs.getString("MAX(staffnumber)") == null){
    staffnum.setText("1");
}
else{
    int id = Integer.parseInt(rs.getString("MAX(staffnumber)"));
    id++;
    staffnum.setText(Integer.toString(id));
}
```

```
t("select * from Staff");

:("insert into Staff(staffnumber,name,gender,phone,ssn,address, type, shift)values(?,?,?,?,?,?,?,?)");

ent("update Staff set name = ?, gender = ?, phone = ?, ssn = ?, address = ?, type = ?, shift = ? where staffnumber = ?");
;

t("delete from Staff where staffnumber = ?");

  rs = s.executeQuery("select MAX(roombednumber) from RoomBed");
  rs.next();
  rs.getString("MAX(roombednumber)");

  if(rs.getString("MAX(roombednumber)") == null){
      roombednum.setText("1");
  }
  else{
      int id = Integer.parseInt(rs.getString("MAX(roombednumber)"));
      id++;
      roombednum.setText(Integer.toString(id));
  }

("select * from RoomBed");

ent("update RoomBed set roomnumber = ?, bednumber = ?, patient = ?, date = ?, surgeon = ? where roombednumber = ?");

t("insert into RoomBed(roombednumber,roomnumber,bednumber,patient,date,surgeon)values(?,?,?,?,?,?)");

nt("delete from RoomBed where roombednumber = ?");
```

```
("select * from RoomBed where roomnumber = ?");
```