|  |
| --- |
| vietnam national university – ho chi minh city international university |
| WEB PROJECT REPORT |
| Web application development |
|  |
| **Chung Minh Nhat – ITITIU170?? Nguyen Tran Chi Hieu – ITITIU17110 Nguyen Xuan Tung – ITITIU17008** |
| **16-Jun-20** |

|  |
| --- |
| A report submitted to the instructor of the Web Application Development course in fulfillment of the course project, Spring 2020 semester. |

# Introduction

## Motivation

It is a concern for every family and every person about their own health. Nowadays, it is not such an easy thing to get access to doctors and medical needs. Often it is the case that hospitals are situated far from one’s home, and they are often full, and queuing to meet with the doctor often takes a long time.

## Problems Statement

The purpose of our project is to design an interactive webpage for doctors and patients alike. We aim to provide a platform for doctors and patients to interact, make appointments, etc., and to be able purchase medicines and medical equipments. Such a platform can greatly increase communication efficiency of patients and doctors, reducing time consumed on medical needs, making medical care more accessible to people.

## Scope

Our system is intended to be a hybrid of the medical E-commerce system, and the doctors booking system. Our system allows patients to select doctors, and make appointments. We do not implement a feature for 1-on-1 chat, or doctor-patient communication from our system (but it can be done with third-party applications). Doctors can manage patients, and appointment schedules, but are not allowed to make purchases (unless they are also the patient). They are allowed to make and manage prescriptions, but the choice of purchasing is left for the patient.

# Literature Review

## Similar Applications/Systems

There are various similar systems and applications:

* UMC medpro: An online doctor booking platform by the university medical center HCMC.
* Medicare: An E-commerce platform focusing on medical products.
* Hello Doctor: A chain of clinics, which also offer a platform for doctor appointments.

There are also various other platforms for these purposes, most of them are for booking of doctors. Medical e-commerce systems are relatively uncommon.

## Platform and Tools Review

The UMC medpro system by the University Medical Center HCMC allows for easy appointment booking of doctors. However it is limited to appointment and scheduling only.

The medicare system is an e-commerce application focusing on medical products. However it lacks the interactive features of the umedpro system. It also does not allow access to experts for guidance on the medical matter.

There are various other online doctor appointment booking platforms or medical e-commerce platforms, but they all have these weaknesses.

# System Design

## System Requirement Specification

**Functional requirements**

1 – For products:

* + Medicines and medical tools are according to database models.
  + System products are priced and listed.
  + System products can be purchased.

2 – For users:

* + Users can buy stuff (medicine, medical tools)
  + Users can register as patient or professional (doctor/ pharmacist/ researcher).
  + Users are according to the database models.

a. As Patients

* + Patients can edit their profile: name, password, symptom reports
  + Patients can file admission forms.
  + Patients can see their prescription with doctors after each session and decide whether to purchase what are specified in the prescription.

b. As Doctors

* + Doctors can accept or reject patients’ requests. An appointment when accepted will be assigned by an empty slot in the appointment list of a doctor.
  + Doctors can read patients’ symptom report.
  + Doctors can meet their patients on Zoom when it’s time.
  + Doctors can write prescriptions after each session.

3 – Interactions between doctor and patient

* + Report of symptoms.
  + Medical prescription: prescriptions after defined will be transferred to the patients’ carts.
  + Chat.

**Requirements analysis**

For chat: low priority, difficult, database + interactive web technology

For doctors and patients, they share same attributes like username, password, usertype → use inheritance

To find suitable shifts for appointment requests, an array that reflects the states of being empty or occupied should be used. Any new accepted request will be filled in based on the request submit time and empty slots left in the array. If requests are made before 7 am or after 5:40 pm, appointments will be rejected (or set to the next day).

**Non – functional requirements**

Performance Requirements

Any activities requiring interactions with database has to last in no more than 3 seconds.

Transition from one window to another should be smooth (inertia movement is preferable) and last within 1 second.

The system shall be available to all users all the time. Downtime shall not exceed 5 seconds in any day.

Security Requirements

C.I.A triad and A.A.A must be satisfied.

Specific measures should be defined to avoid any Sensitive Data leakage.

Software Quality Attributes

*Usability requirements:*

The system should be easy to use by all staff and should be organized in such a way that user errors are minimized.

Staff should be able to use all the system functions after two hours of training. After this training, the average number of errors made by experienced users shall not exceed two per hour of system use.

*Maintainability requirements:*

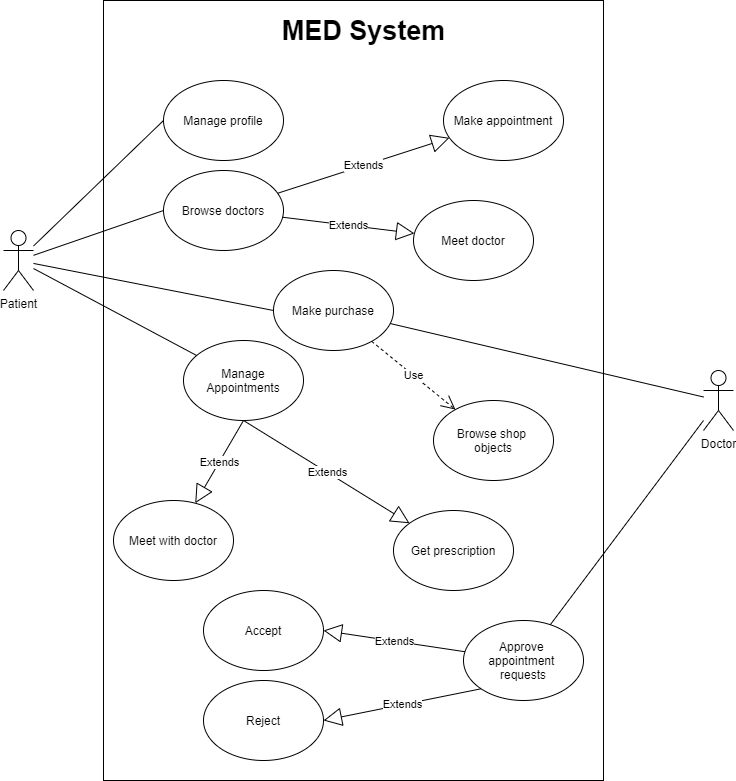
System updates and maintainence must be informed to all users at least 3 days in advance, and should last for at most 1 day to avoid implementation failure or information loss, etc.

*Reliability requirements:*

Calculations and comparison from extracted information must be guaranteed to be exact.

## System Design Specification

Use case diagram:



There are two actors: doctors and patients. A brief description of each use case is as follow:

- Manage patients (summary goal UC): Doctors can manage their patients’ information, such as medicine prescribed, time and date, appointments, etc.

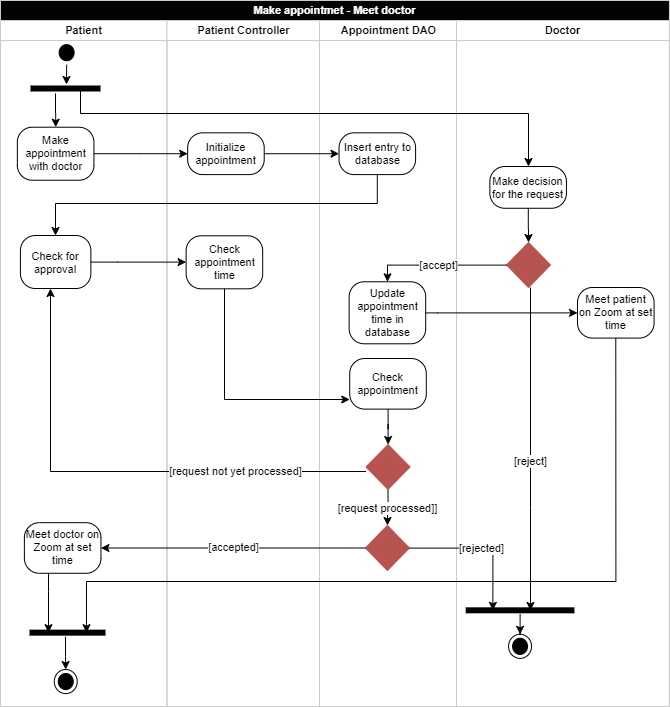
- Make purchase: Doctors and patients are allowed to purchase medicine and medical equipments on the website.

- 1-on-1 chat: Doctors and patients can chat with each other through message forms. They can also send pending messages when either party is offline.

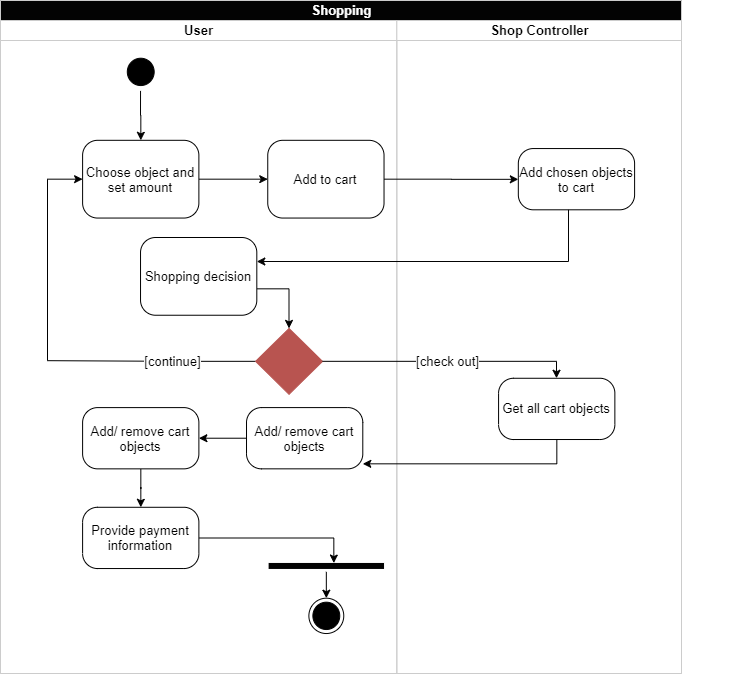
- Manage appointments: Patients can make appointments to doctors. Doctors can manage their own schedule, manage patient appointment timetable, etc.

Activity diagram:

For the use case: make appointment and meet doctor

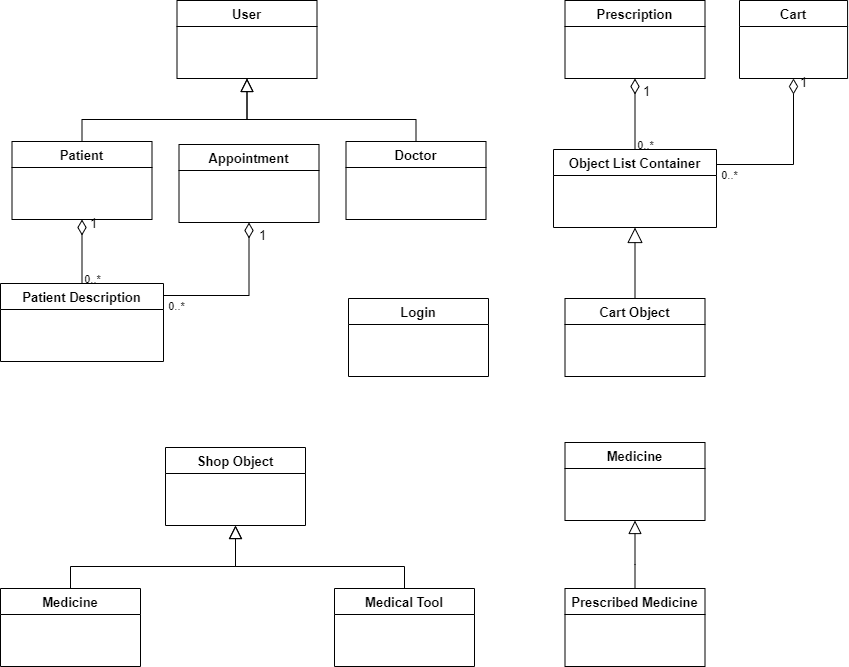


For the use case: shopping



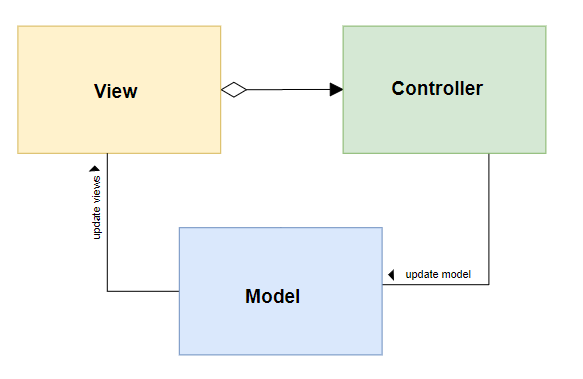
Class diagram:

This diagram shows all models in the system and their relationship

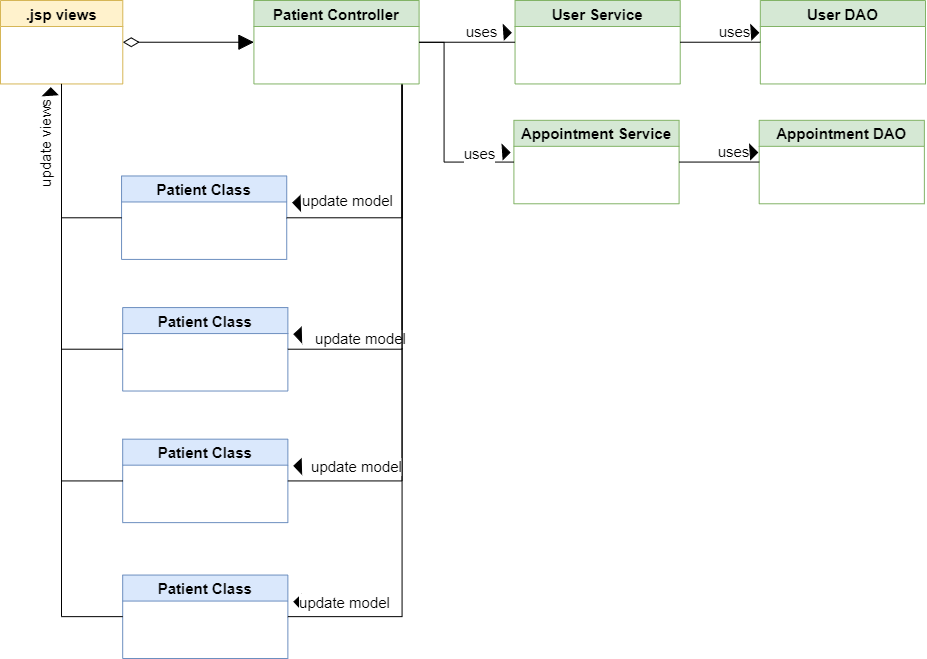


MVC architecture: the following class diagram containing models, controllers and views classes will depict MVC architecture. These classes are drawn based on 2 basic functions: make appointment – meet doctor and shopping.

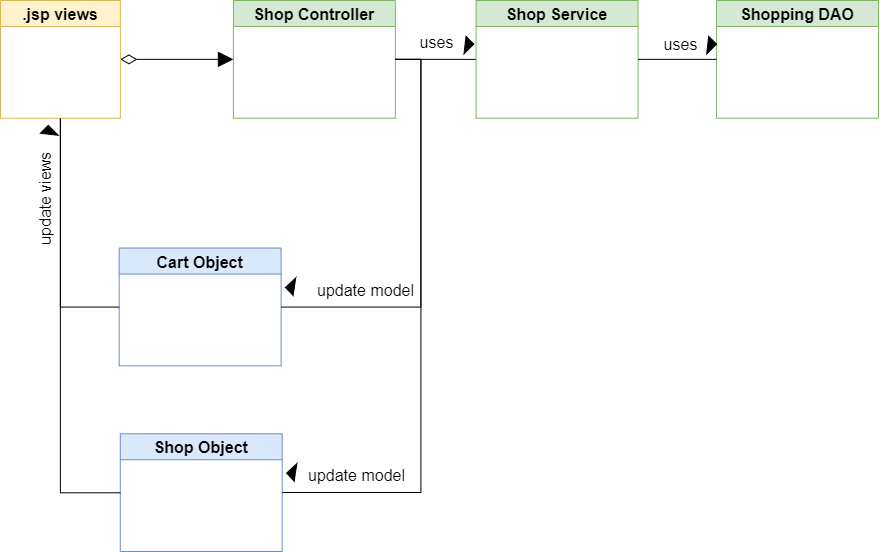
All diagrams below follow a standard flow like this:



Patient – doctor interactions:

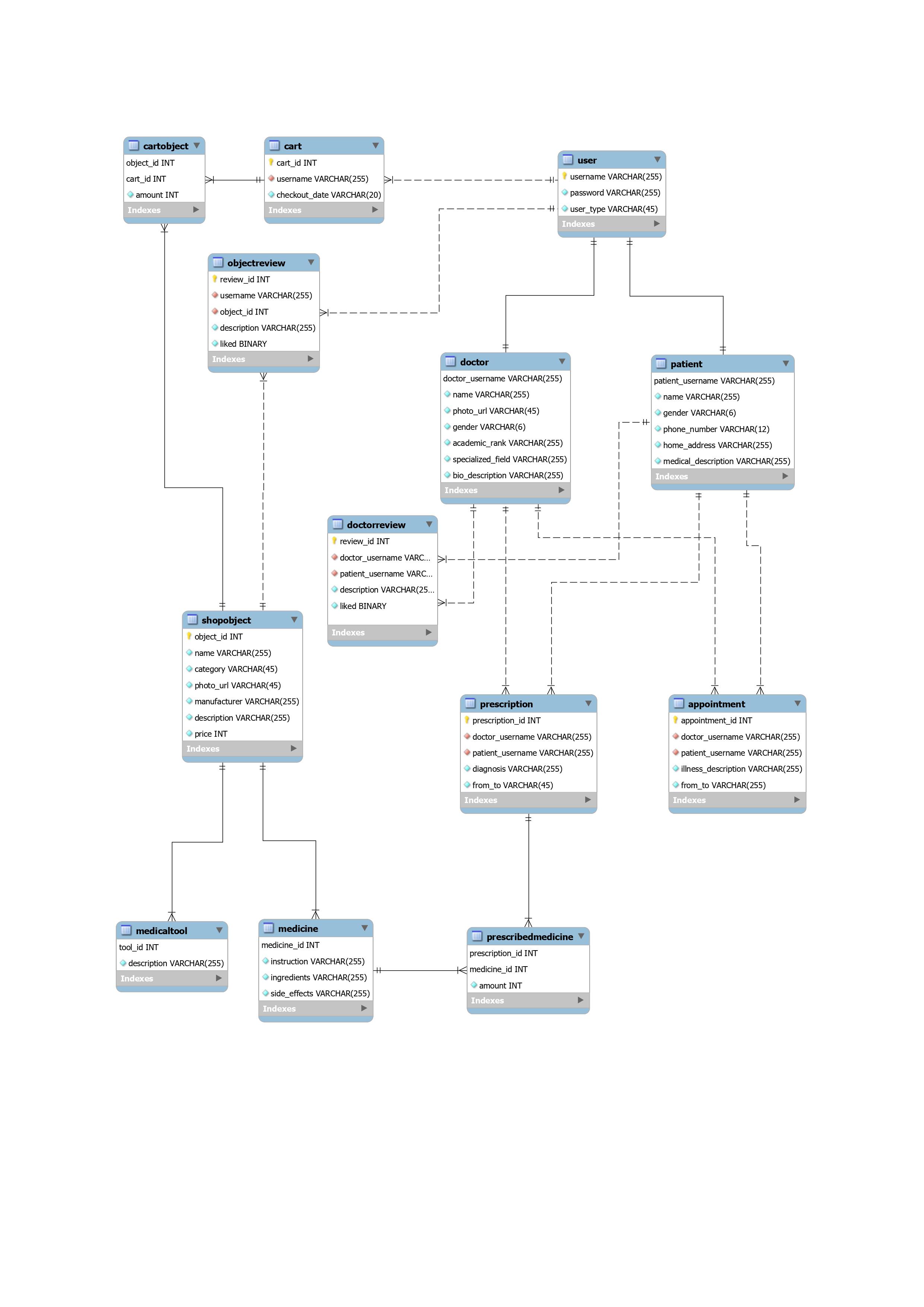


Shopping:



# System Implementation

Our database consists of several tables. The design is shown below



The database model was designed on Microsoft MySQL Workbench on a MySQL database server. The database was generated using MySQL Workbench’s forward engineering.

# Conclusion and Discussion

## List of accomplished work

We managed to implement a complete system with the intended features, and a user-friendly interface. It is not without faults, however, but it is functional, and is easy to maintain and fix, should any issues arise.

## Strength and Weakness

Our strength:

* We had clear and concise planning before implementation.
* We applied the MVC framework to a complete project.
* We utilized many different technologies, including (but not limited to): Apache Maven, Javascript, Bootstrap, MySQL database system.
* We were able to effectively utilize Git for easy collaboration.

Our weaknesses:

* Our time management was not very good, so we were more fast-paced towards the end.
* Our project was not completely without errors. It was functional, but may require additional work towards the long run.
* We needed to simplify several functions from our initial plan, due to not having enough time, and not having the knowledge and experience to do all of them.

## Future Work

There are various room for improvement for our project:

* Our project is not error-proof.
* There are room for more features.
* We can improve our interfaces for better user experience.

# References