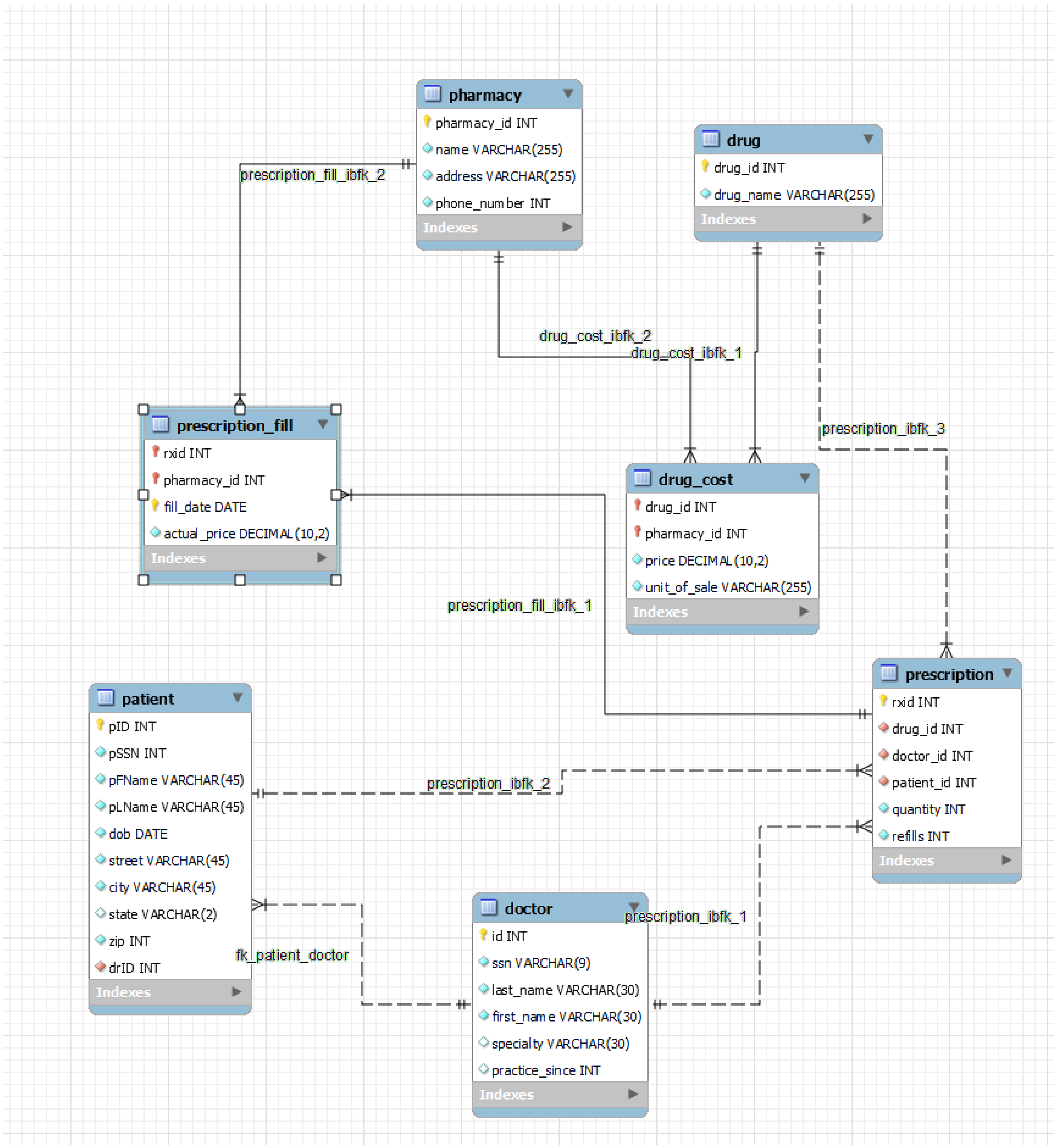
Lab 19: Web App SQL Group Project

Edward Torres

Rob Stanford

ER Diagram



This database design keeps track of prescriptions written by doctors and filled by pharmacies for their patients. It has seven tables: doctor, patient, prescription, drug, drug_cost, prescription fill, and pharmacy.

The doctor table creates an autogenerated primary key titled doctor ID and holds the doctor's information such as name, specialty, and years practiced. The patient table creates an autogenerated primary key patient ID and holds the patient information. This table has a foreign key of the doctor's ID to connect a doctor to the patient. The prescription table has an autogenerated primary key titled RXID and holds the information for a prescription such as quantity and refills. The foreign keys are the auto-generated IDs from the patient, doctor and drug table. The drug table has an autogenerated primary key and the drug name. The drug_cost table holds the price of each drug and unit of sale. It has foreign keys from the pharmacy table and the drug table to keep track of the cost of the drug at each pharmacy. The pharmacy table has an autogenerated ID to identify the pharmacy and hold the name, address, and phone number of the pharmacy. The prescription_fill table has the fill date as the primary key and holds the actual price of the prescription. It also has foreign keys that connect it to the prescription (rxid) and the pharmacy(pharmacy_id) where it is being refilled.

The doctor table has a non-identifying one-to-many relationship with both the patient and the prescription table since one doctor can have many patients and write many prescriptions. The patient table has a non-identifying one-to-many relationship with the prescription table since one patient can have many prescriptions. The prescription table has a non-identifying one-to-many relationship with the drug table since a drug can be in many prescriptions. It also has an identifying one-to-many relationship with the prescription_fill table since one prescription can be filled many times. The drug table has an identifying many-to-many relationship with the pharmacy table since many pharmacies can hold many drugs. This creates the table drug_cost which connects the cost of the drug at each pharmacy. The pharmacy table has an identifying one-to-one relationship with the prescription_fill table since one pharmacy will fill one prescription.

SQL Schema

```
create table doctor(
   id int primary key auto_increment,
   ssn varchar(9) not null unique,
   last_name varchar(30) not null,
   first_name varchar(30) not null,
   specialty varchar(30),
   practice_since int );

  CREATE TABLE IF NOT EXISTS `prescription`.`patient` (
  `pID` INT NOT NULL AUTO_INCREMENT,
  `pSSN` INT NOT NULL,
  `pFName` VARCHAR(45) NOT NULL,
  `pLName` VARCHAR(45) NOT NULL,
  `dob` DATE NOT NULL,
  `street` VARCHAR(45) NOT NULL,
  `city` VARCHAR(45) NOT NULL,
  `state` VARCHAR(2) NULL,
  `zip` INT NOT NULL,
  `drID` INT NOT NULL,
  PRIMARY KEY (`pID`),
  INDEX `fk_patient_doctor_idx` (`drID` ASC) VISIBLE,
  CONSTRAINT `fk_patient_doctor`
    FOREIGN KEY (`drID`)
    REFERENCES `prescription`.`doctor` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);

CREATE TABLE prescription.drug (
    drug_id INT AUTO_INCREMENT PRIMARY KEY,
    drug_name VARCHAR(255) NOT NULL
    );
```

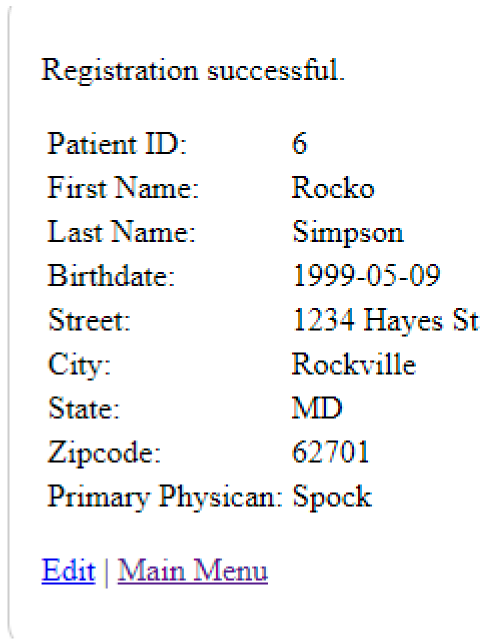
```
CREATE TABLE prescription.prescription (
    rxid INT AUTO_INCREMENT PRIMARY KEY,
    drug_id INT NOT NULL,
    doctor_id INT NOT NULL,
    patient_id INT NOT NULL,
    quantity INT NOT NULL,
    refills INT NOT NULL,
    FOREIGN KEY (doctor_id) REFERENCES doctor(id),
    FOREIGN KEY (patient_id) REFERENCES patient(pID),
    FOREIGN KEY (drug_id) REFERENCES drug(drug_id)
    );

drop table if exists prescription.pharmacy;
CREATE TABLE prescription.pharmacy (
    pharmacy_id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    address VARCHAR(255) NOT NULL,
    phone_number INT NOT NULL
);

drop table if exists prescription.drug_cost;
CREATE TABLE prescription.drug_cost (
    drug_id INT,
    pharmacy_id INT,
    price DECIMAL(10, 2) NOT NULL,
    unit_of_sale VARCHAR(255) NOT NULL,
    PRIMARY KEY (drug_id, pharmacy_id),
    FOREIGN KEY (drug_id) REFERENCES drug(drug_id),
    FOREIGN KEY (pharmacy_id) REFERENCES pharmacy(pharmacy_id)
);

drop table if exists prescription.prescription_fill;
CREATE TABLE prescription.prescription_fill (
    rxid INT NOT NULL,
    pharmacy_id INT NOT NULL,
    fill_date DATE NOT NULL,
    actual_price DECIMAL(10, 2) NOT NULL,
    PRIMARY KEY (rxid, pharmacy_id, fill_date),
    FOREIGN KEY (rxid) REFERENCES prescription(rxid),
    FOREIGN KEY (pharmacy_id) REFERENCES pharmacy(pharmacy_id)
);
```

Register as a new patient with last name “Simpson”, city “Rockville”, zip code 62701 and a doctor with name “Spock”.   Show a successful registration.

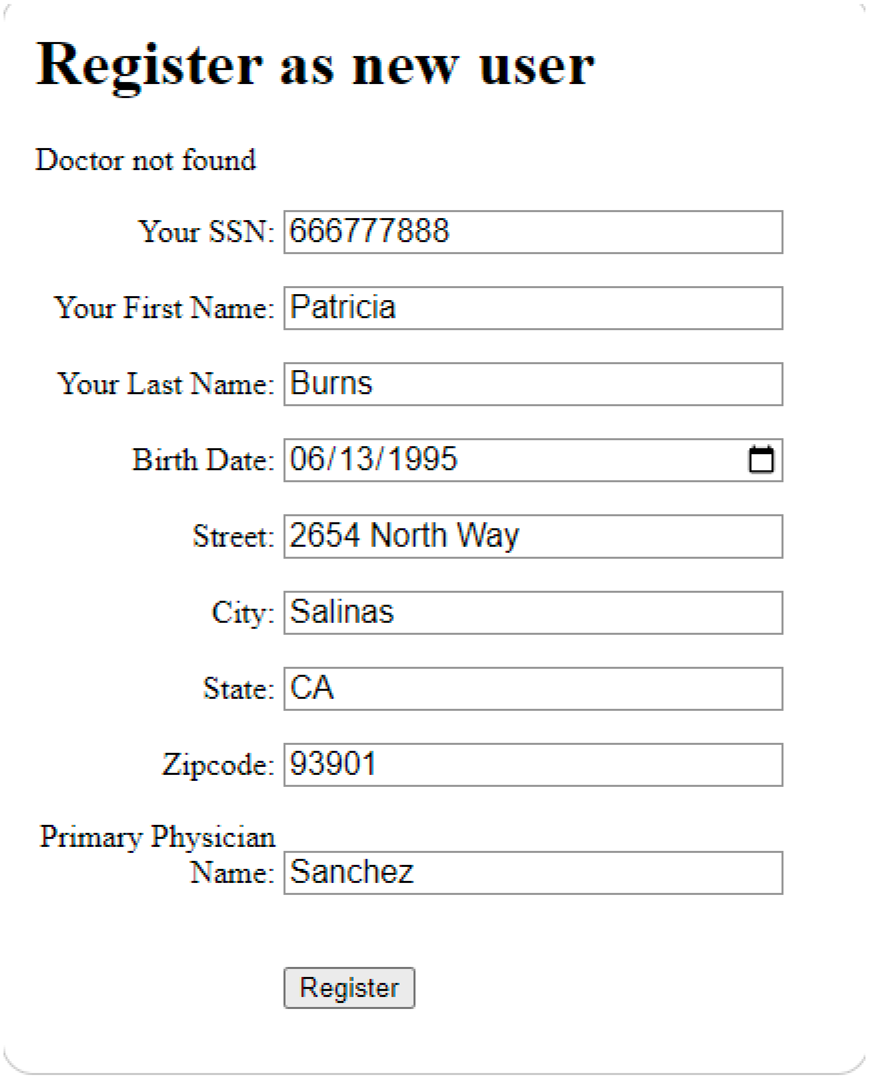


Registration successful.

Patient ID:	6
First Name:	Rocko
Last Name:	Simpson
Birthdate:	1999-05-09
Street:	1234 Hayes St
City:	Rockville
State:	MD
Zipcode:	62701
Primary Physican:	Spock

Edit | Main Menu

Attempt to register as a new patient with last name “Burns” but with a doctor name that does not exist. Show a screenshot of the patient register form with the error message.

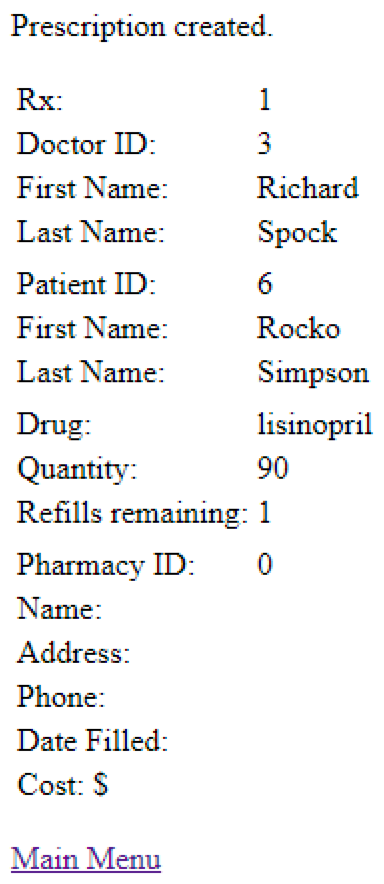


Register as new user

Doctor not found

Your SSN:	666777888
Your First Name:	Patricia
Your Last Name:	Burns
Birth Date:	06/13/1995
Street:	2654 North Way
City:	Salinas
State:	CA
Zipcode:	93901
Primary Physician Name:	Sanchez

Create a prescription for the patient “Simpson” and doctor “Spock” for a drug “lisinopril” and quantity 90. Show the screen with the success message and prescription display.

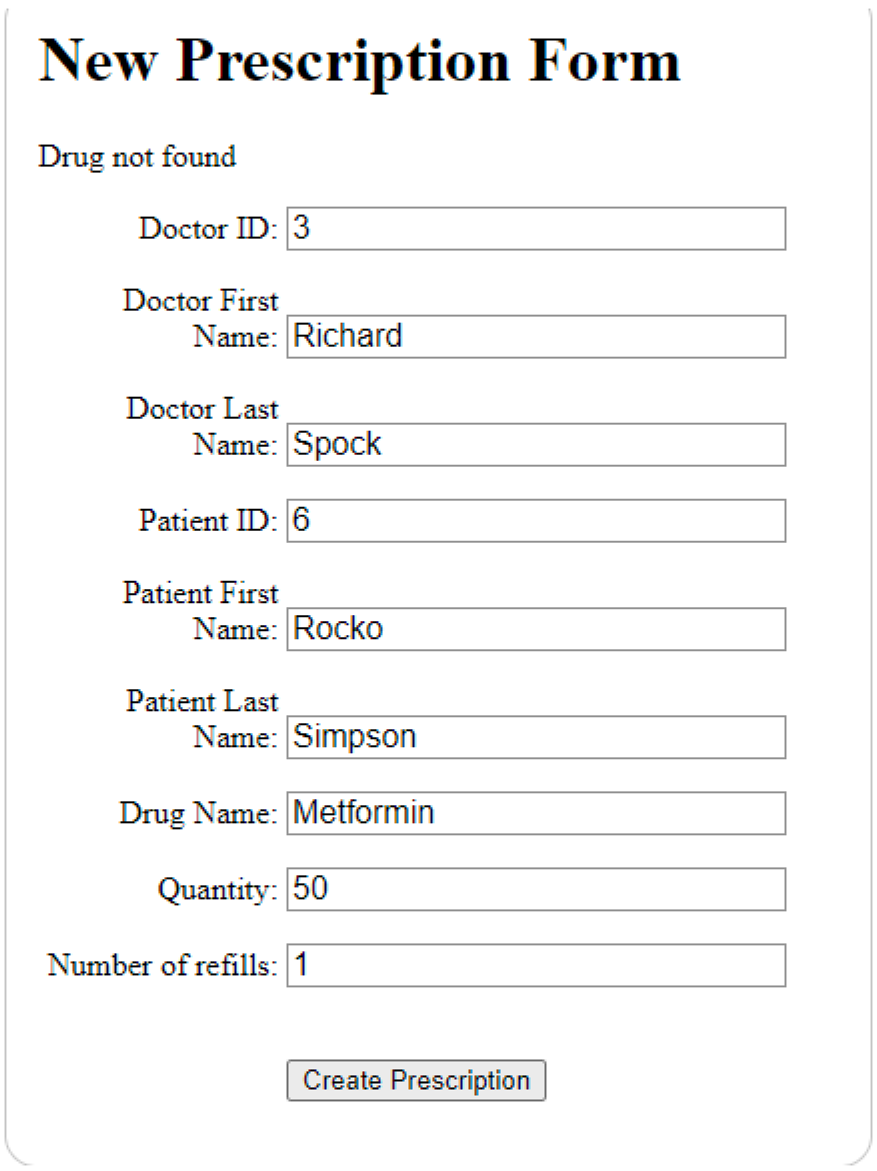


```
Prescription created.

Rx:                 1
Doctor ID:          3
First Name:         Richard
Last Name:          Spock
Patient ID:         6
First Name:         Rocko
Last Name:          Simpson
Drug:               lisinopril
Quantity:           90
Refills remaining:  1
Pharmacy ID:        0
Name:
Address:
Phone:
Date Filled:
Cost: $

Main Menu
```

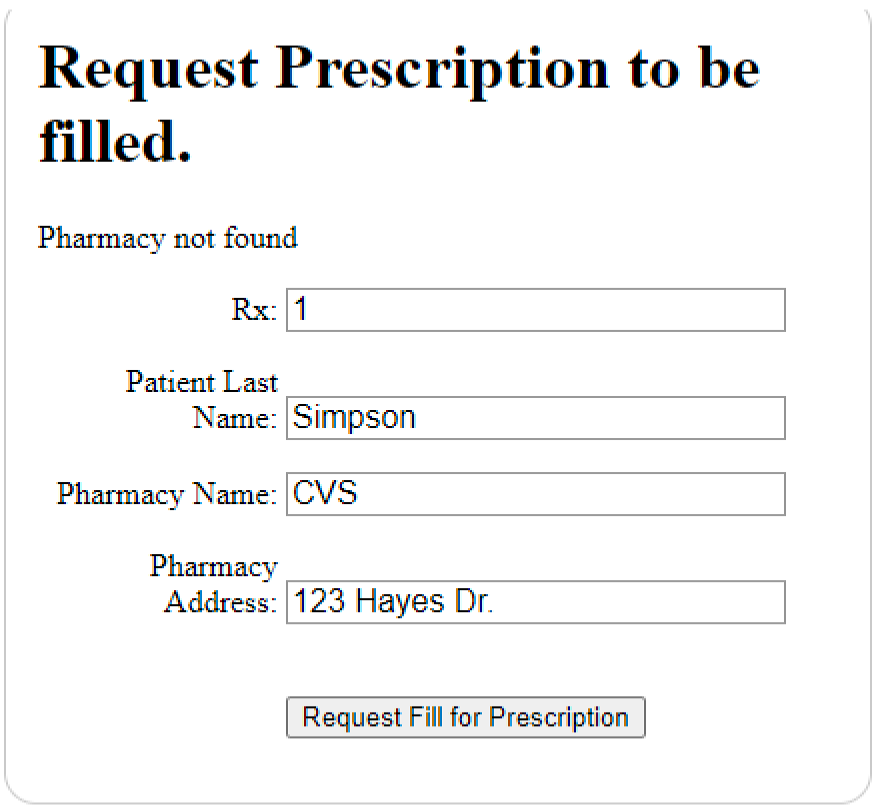
Attempt to create a prescription with an invalid drug name. Show a screen with the create prescription form and error message.


New Prescription Form

Drug not found

Doctor ID:	3
Doctor First Name:	Richard
Doctor Last Name:	Spock
Patient ID:	6
Patient First Name:	Rocko
Patient Last Name:	Simpson
Drug Name:	Metformin
Quantity:	50
Number of refills:	1

Attempt to fill a prescription with an invalid pharmacy name.

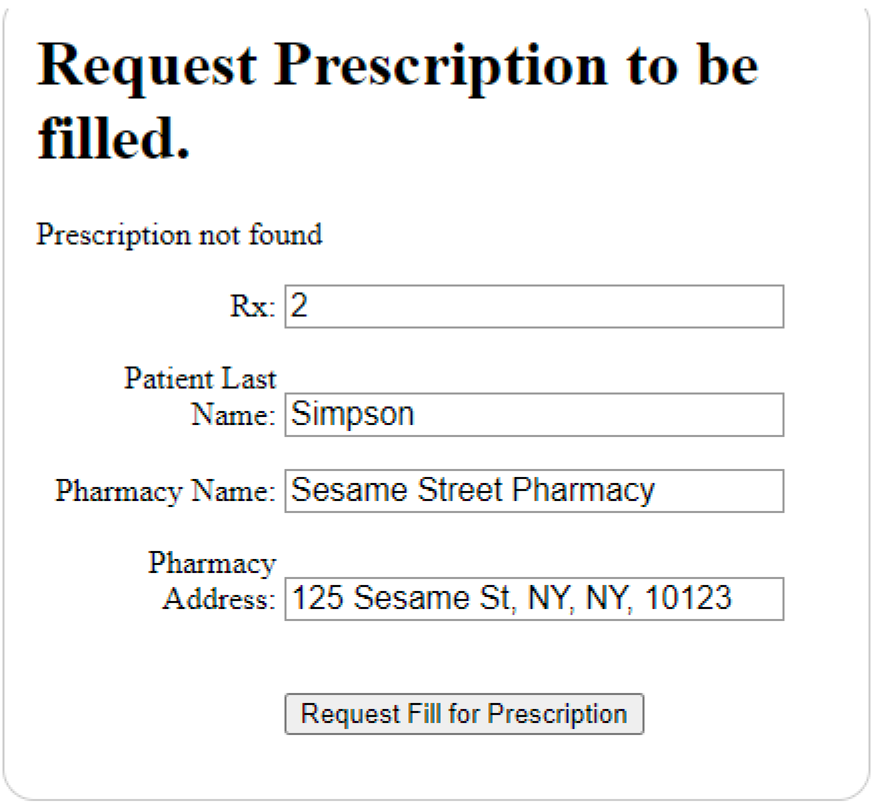


Request Prescription to be filled.

Pharmacy not found

Rx:	1
Patient Last Name:	Simpson
Pharmacy Name:	CVS
Pharmacy Address:	123 Hayes Dr.

Attempt to fill a prescription with an invalid rxid.

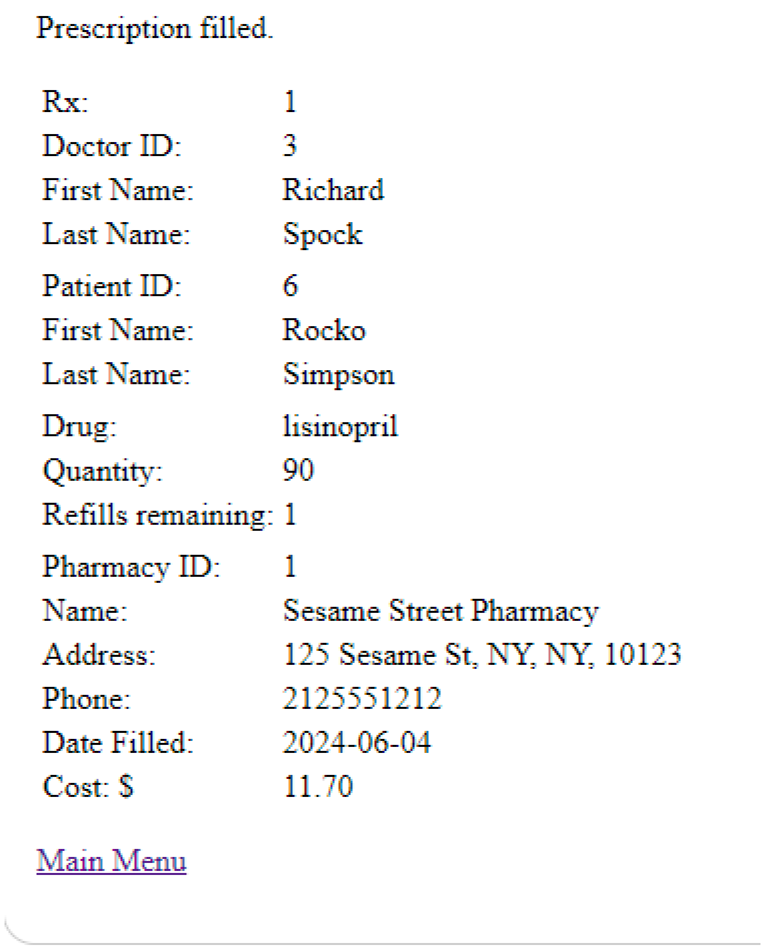


Request Prescription to be filled.

Prescription not found

Rx:	2
Patient Last Name:	Simpson
Pharmacy Name:	Sesame Street Pharmacy
Pharmacy Address:	125 Sesame St, NY, NY, 10123

Fill the prescription with success.

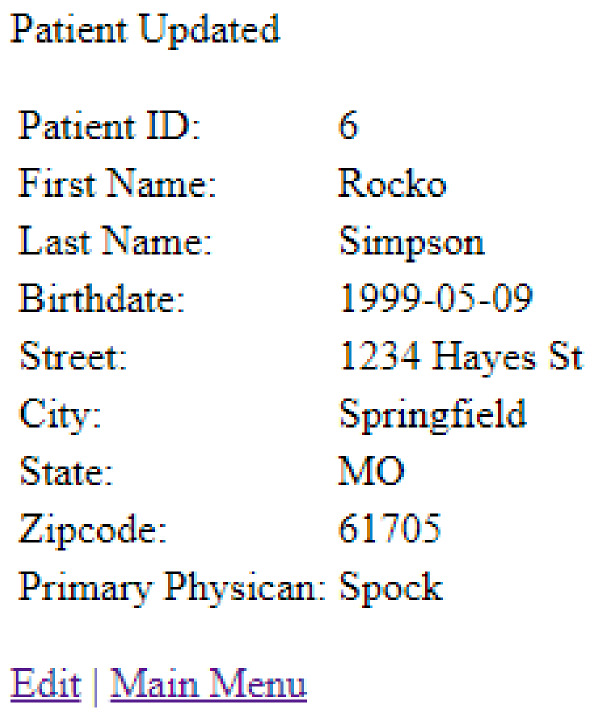


Prescription filled.

Rx:	1
Doctor ID:	3
First Name:	Richard
Last Name:	Spock
Patient ID:	6
First Name:	Rocko
Last Name:	Simpson
Drug:	lisinopril
Quantity:	90
Refills remaining:	1
Pharmacy ID:	1
Name:	Sesame Street Pharmacy
Address:	125 Sesame St, NY, NY, 10123
Phone:	2125551212
Date Filled:	2024-06-04
Cost: $	11.70

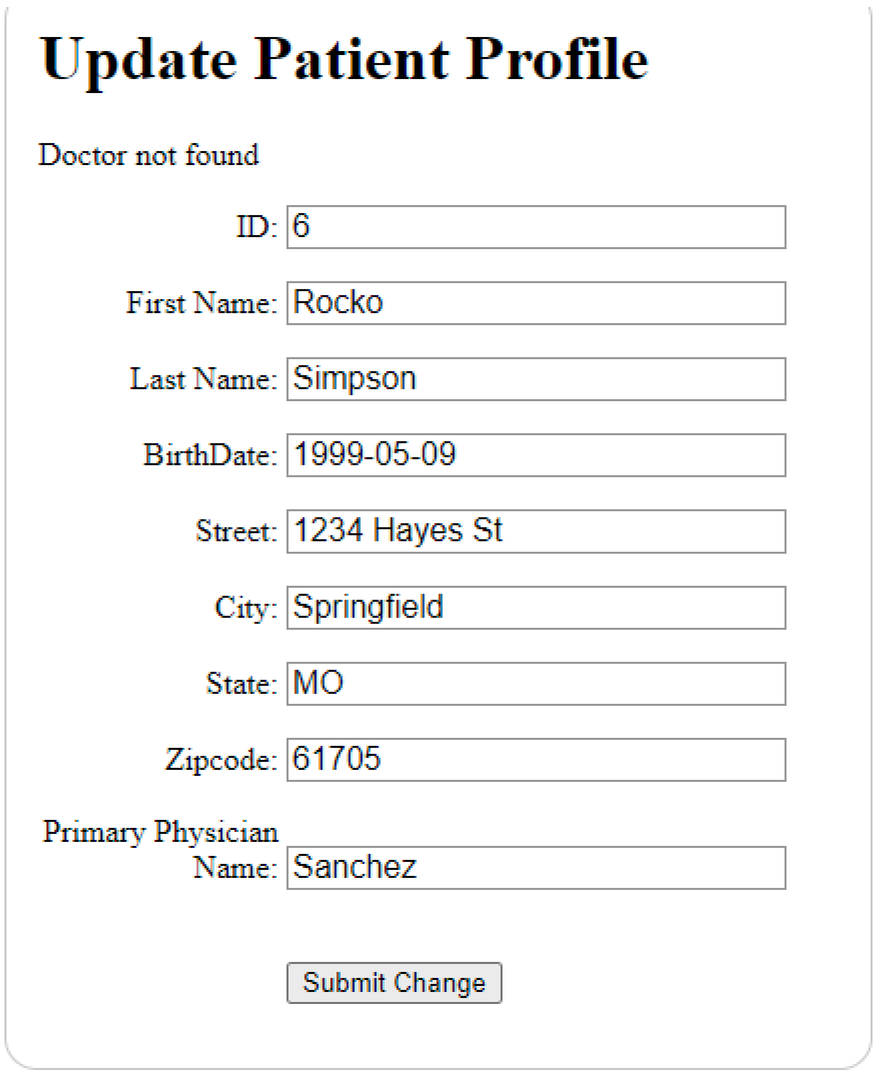
Main Menu

Get the profile for patient “Simpson” and edit the patient record for “Simpson” and change city to Springfield and zip code to 61705. Show the web page of the successful update.


Patient Updated

Patient ID:	6
First Name:	Rocko
Last Name:	Simpson
Birthdate:	1999-05-09
Street:	1234 Hayes St
City:	Springfield
State:	MO
Zipcode:	61705
Primary Physican:	Spock

Edit the patient record for “Simpson”. Attempt to change the doctor’s name to a doctor that does not exist. Show the error message and edit patient form.



Update Patient Profile

Doctor not found

ID:	6
First Name:	Rocko
Last Name:	Simpson
BirthDate:	1999-05-09
Street:	1234 Hayes St
City:	Springfield
State:	MO
Zipcode:	61705
Primary Physician Name:	Sanchez