Team C Design Document

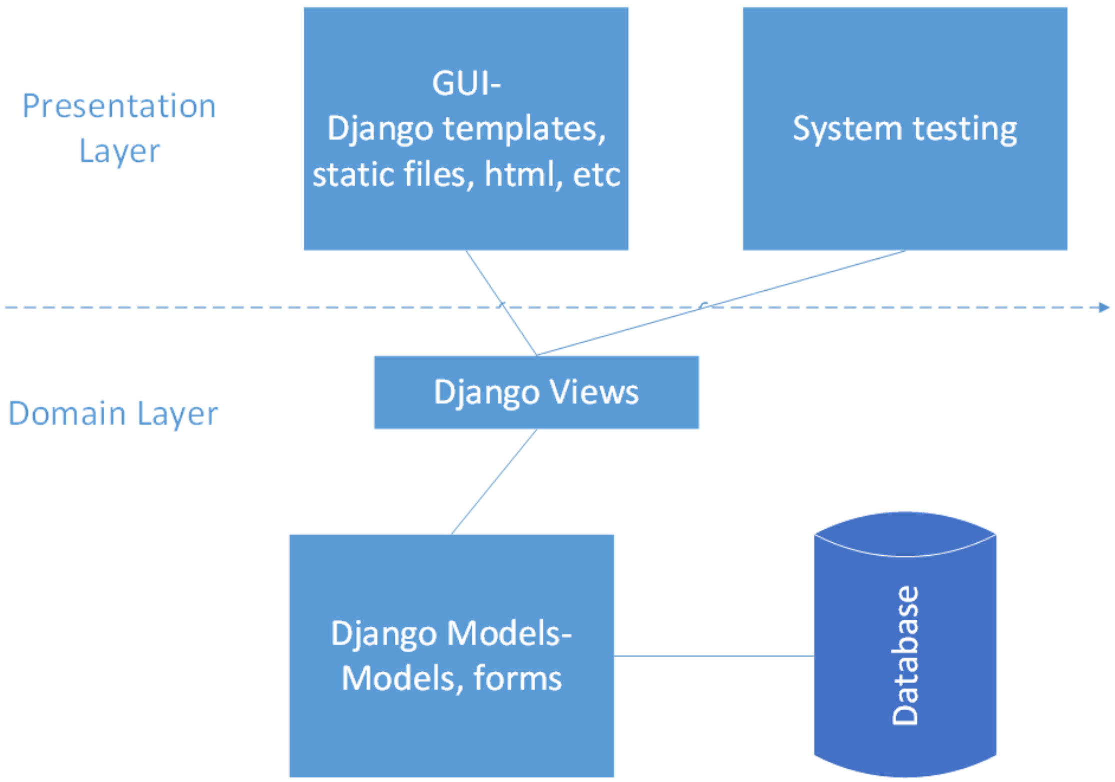
Including all design diagrams, models, and our design rationale.

Date: December 7, 2015

Version: 1.6

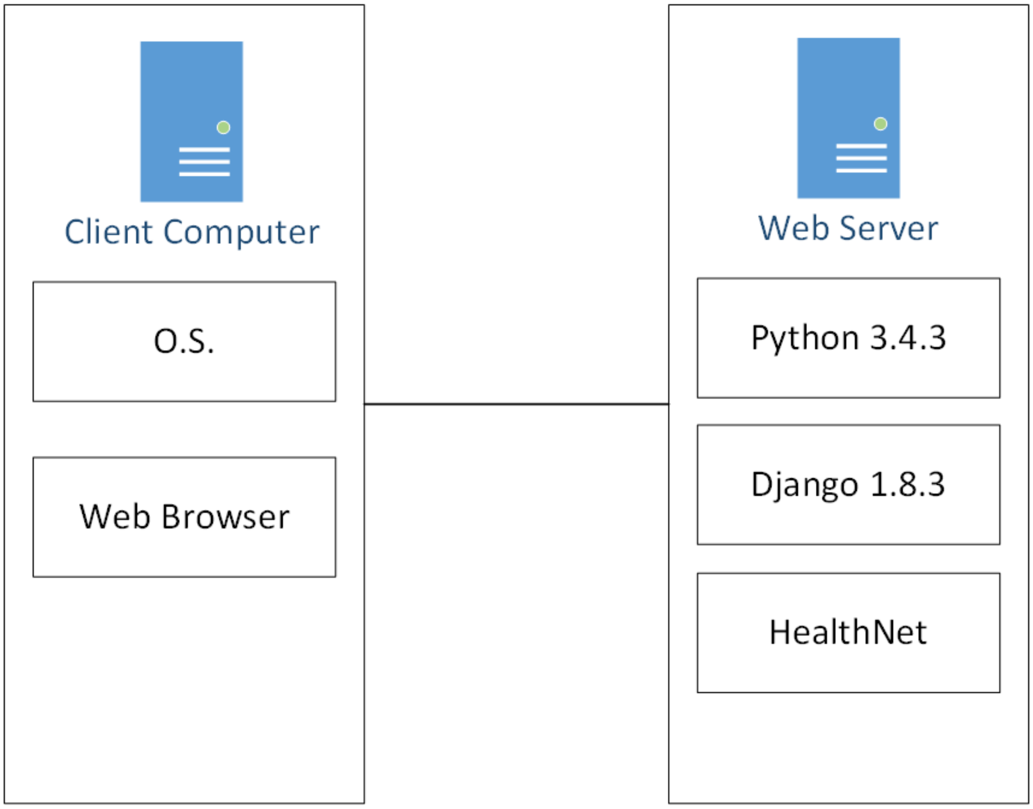
Team: f261-03C Team C (Team Squad)

1.0 Architectural Model


Presentation Layer
GUI- Django templates, static files, html, etc
System testing
Domain Layer
Django Views
Django Models- Models, forms
Database

Here is our architectural model. It consists of a presentation layer, for things that a user might see or interact with, and a domain layer, which includes most of the back end functionality.

2.0 Deployment Diagram



Our deployment is very simple. The web server running healthnet requires only Python 3.4.3 with Django 1.8.3, and of course health net. Any client computers need only an operating system and a web browser to use healthnet functionality.

3.0 Components and Classes

Component ID	Description
DES-001 **Patient App**	**Component state** ● **Keeps track of all patient data, including** o **personal data** o **contact info** o **medical info** **Component behavior** ● **Provides data to view in order to display a given user's user specific information**
DES-001 **Patient.views** **Patient.forms**	**Class state** ● **None** **Class behavior** ● **Give hompage on request** ● **Register patients** ● **Edit patients** ● **Make patient appointments** ● **View patient appointments** ● **Edit patient appointments** ● **Delete patient appointments** ● **View patient medical info**
DES-002 **Nurse app**	**Component state** ● **Keeps track of all nurse data, including** o **personal data** o **contact info** **Component behavior** ● **Provides nurse with capability to interact with the system**
DES-002 **Nurse.views** **Nurse.forms**	**Class state** ● **None** **Class behavior** ● **Give hompage on request** ● **Make patient appointments** ● **View patient appointments** ● **Delete patient appointments** ● **View patient medical info**
DES-003 Doctor app	**Component state** ● **Doctor basic info** ● **Contact info** ● **Accreditation** ● **Licenses** ● **Place of employment** ● **Patients**

	Component behavior ● **Allow doctor to view and interact with system** ● **Allow doctor to edit their patients information** ● **Allow doctor to make and view appointments**
DES-003 **Doctor.views** **Doctor.forms**	**Component State:** ● **None** **Component Behaviour** ● **Return doctor hompage** ● **View all patients** ● **View individual patient** ● **Edit patient** ● **Make appointments** ● **Make tests** ● **Edit tests** ● **View appointments** ● **Make prescriptions** ● **Edit profile** ● **Edit appointments** ● **Delete Prescriptions** ● **Delete tests** ● **Delete files** ● **Admit patient** ● **Transfer patient** ● **Upload files**
DES-004 **Administrator app**	**Component state** ● **Admin basic info** ● **Contact info** **Component behavior** ● **Add new staff** ● **Edit users** ● **Create users** ● **View system log and statistics** ● **Transfers**
DES-004 **Administrator.views** **Administrator.forms**	**Component State:** ● **None** **Component Behaviour:** ● **Return homepage** ● **Create staff** ● **Edit users** ● **View log** ● **View statistics** ● **Display info** ● **Do transfers**
DES-005	**Component state:** ● **Stores most of the state for the system** **Component behaviour:**

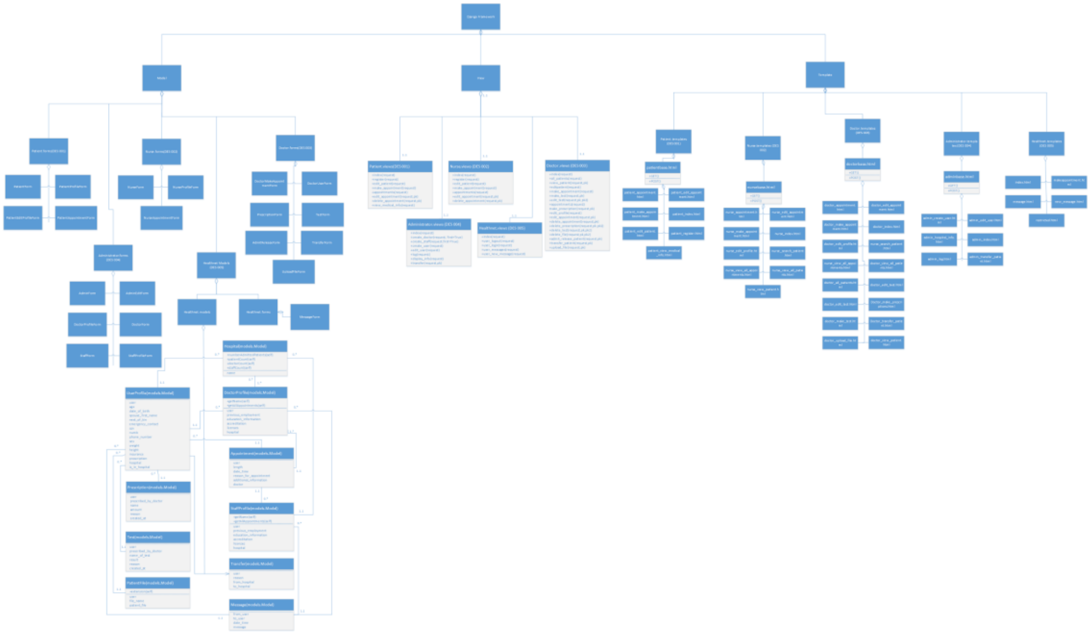
Healthnet App	● **Messages** ● **Log in** ● **Log out** ● **Main home page**
DES-005 **Healthnet.views** **Healthnet.forms**	**Component State:** ● **None** **Component Behaviour:** ● **Messages** ● **Log in** ● **Log out**
DES-005 **Healthnet.models** **Hospital**	**Component State:** ● **Name** **Component Behaviour:** ● **Return number of admitted patients** ● **Patient count** ● **Doctor count** ● **Staff count**
DES-005 **Healthnet.models** **User Profile**	**Component State:** ● **user** ● **age** ● **date of birth** ● **spouse name** ● **next of kin** ● **emergency contact** ● **ssn** ● **phone number** ● **sex** ● **weight** ● **height** ● **insurance** ● **prescription** ● **hospital** ● **Is in hospital boolean** **Component Behaviour:** ● **None**
DES-005 **Healthnet.models** **Prescription**	**Component State:** ● **user** ● **prescription** ● **name** ● **amount** ● **reason** ● **date created** **Component Behaviour:** ● **None**
DES-005	**Component State:** ● **user**

Healthnet.models **Test**	● **doctor prescribed by** ● **name** ● **result** ● **reason** ● **date created** **Component Behaviour:** ● **Messages** ● **Log in** ● **Log out**
DES-005 **Healthnet.models** **Patient File**	**Component State:** ● **user** ● **file name** ● **patient file** **Component Behaviour:** ● **extension**
DES-005 **Healthnet.models** **Doctor Profile**	**Component State:** ● **user** ● **previous employer** ● **education** ● **accreditation** ● **licenses** ● **hospital** **Component Behaviour:** ● **get name** ● **get all appointments**
DES-005 **Healthnet.models** **Appointment**	**Component State:** ● **user** ● **length** ● **date** ● **reason for appointment** ● **additional info** ● **doctor** **Component Behaviour:** ● **None**
DES-005 **Healthnet.models** **Staff Profile**	**Component State:** ● **user** ● **previous employment** ● **education** ● **accreditiation** ● **licenses** ● **hospital** **Component Behaviour:** ● **get name** ● **get all appointments**

DES-005 **Healthnet.models** **Transfer**	**Component State:** ● **user** ● **reason** ● **from_hospital** ● **to_hospital** **Component Behaviour:** ● **None**
DES-005 **Healthnet.models** **Message**	**Component State:** ● **from user** ● **to user** ● **date** ● **message** **Component Behaviour:** ● **None**

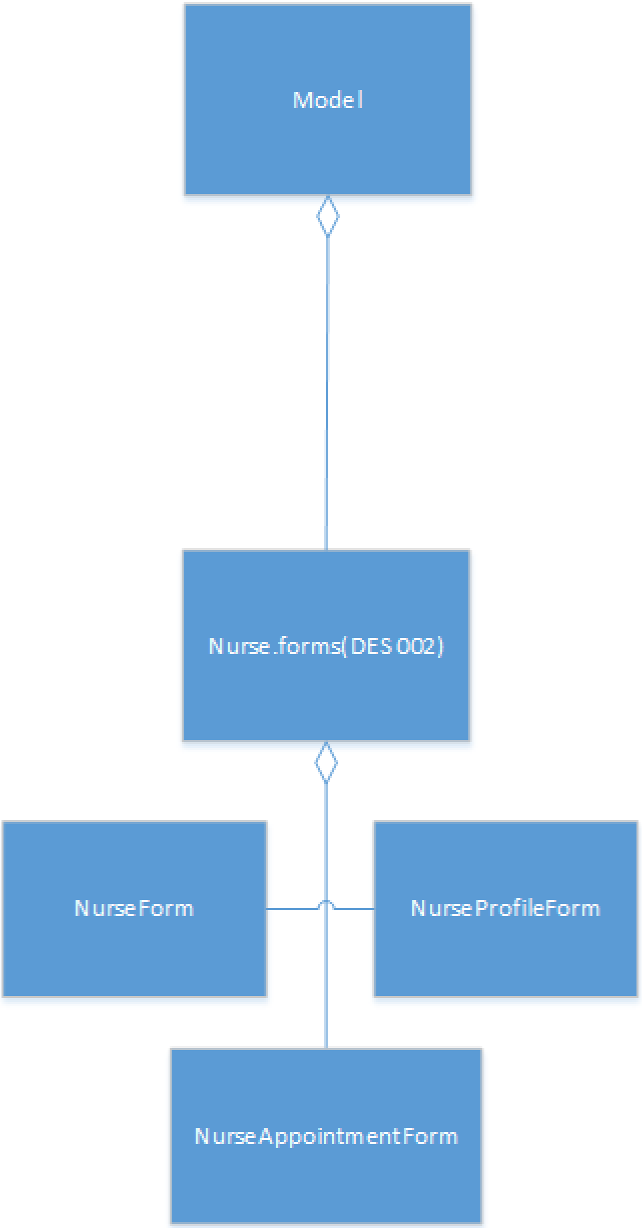
3.1 Class Diagram(s)

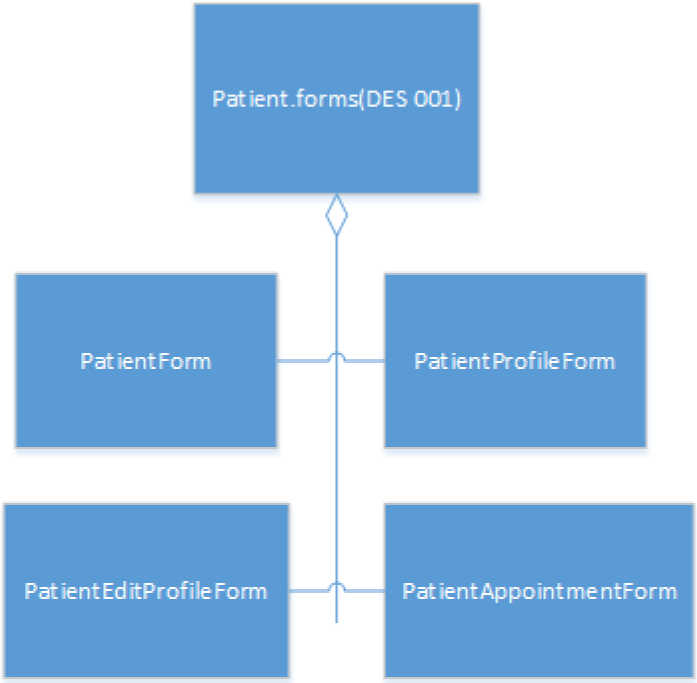
Overall: Shows the class diagram in a zoomed out position to see the entire hierarchy.

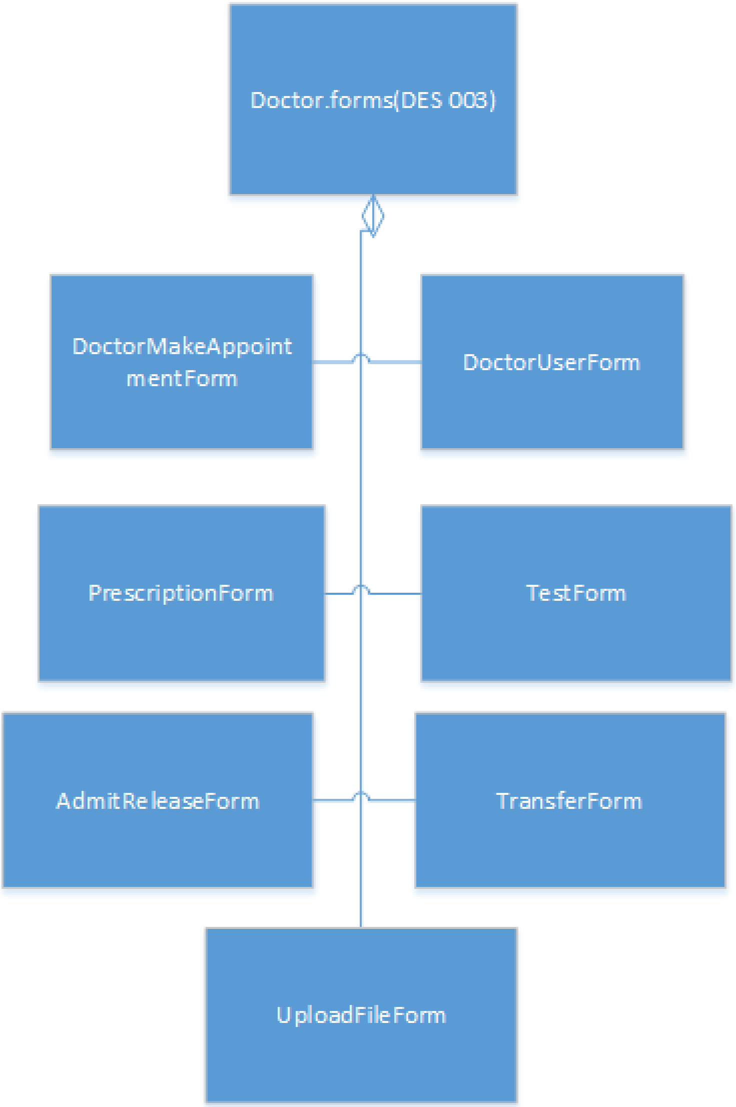


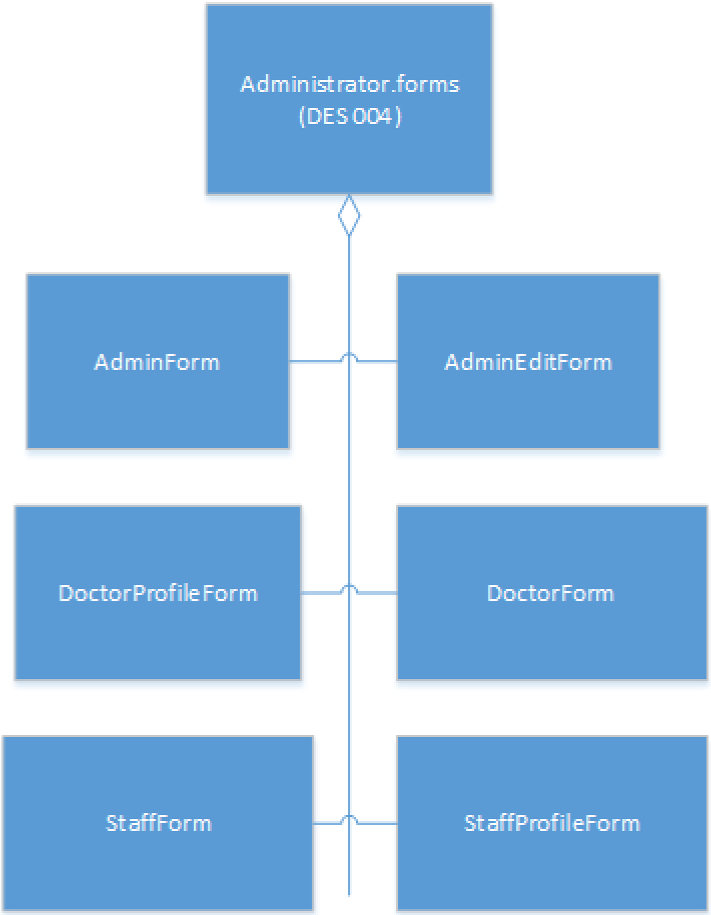
Models:

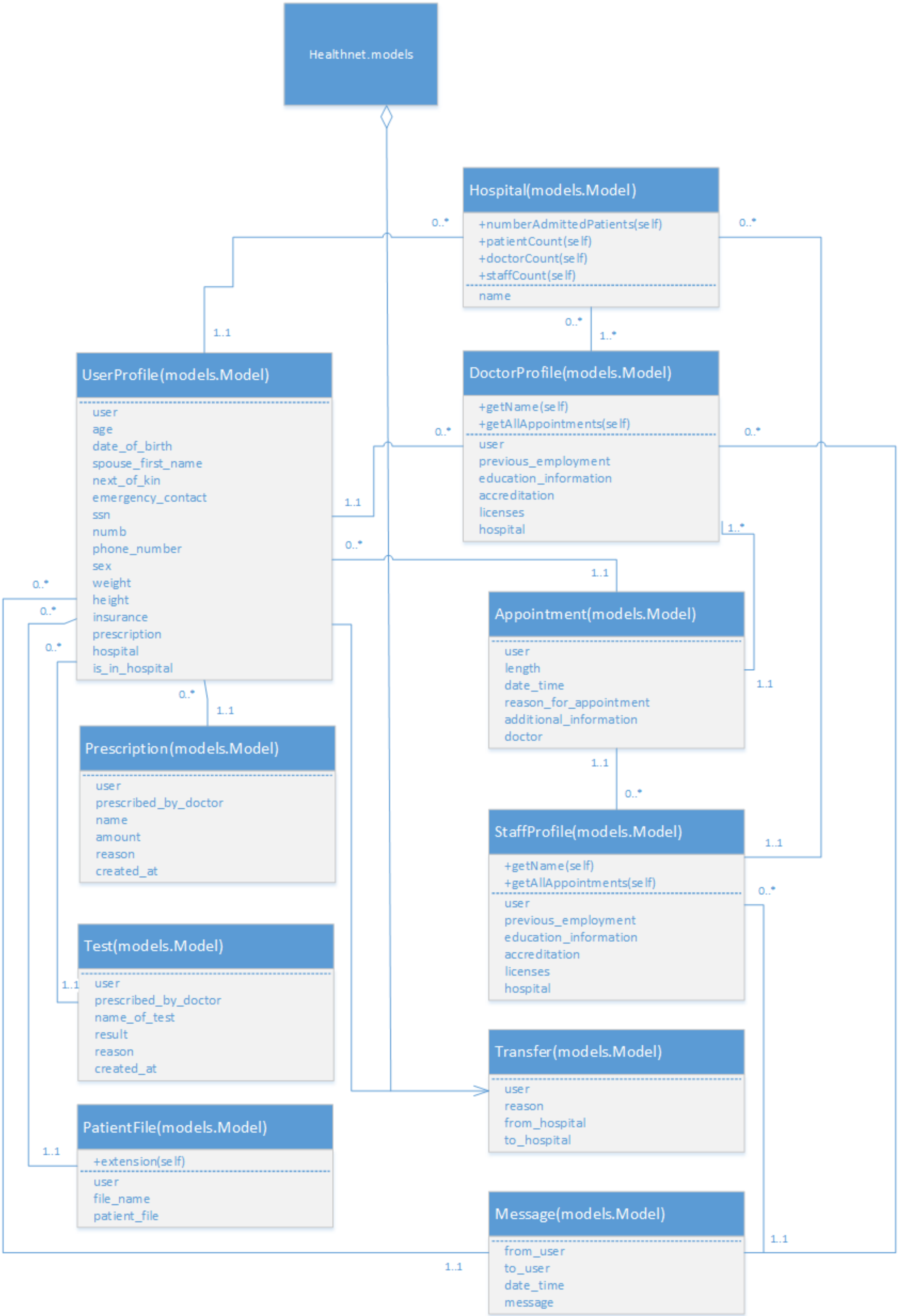
Split into five models by the components: Nurse, Patients, Doctor, Admin, and Healthnet.


Model
Nurse.forms(DES002)
NurseForm
NurseProfileForm
NurseAppointmentForm

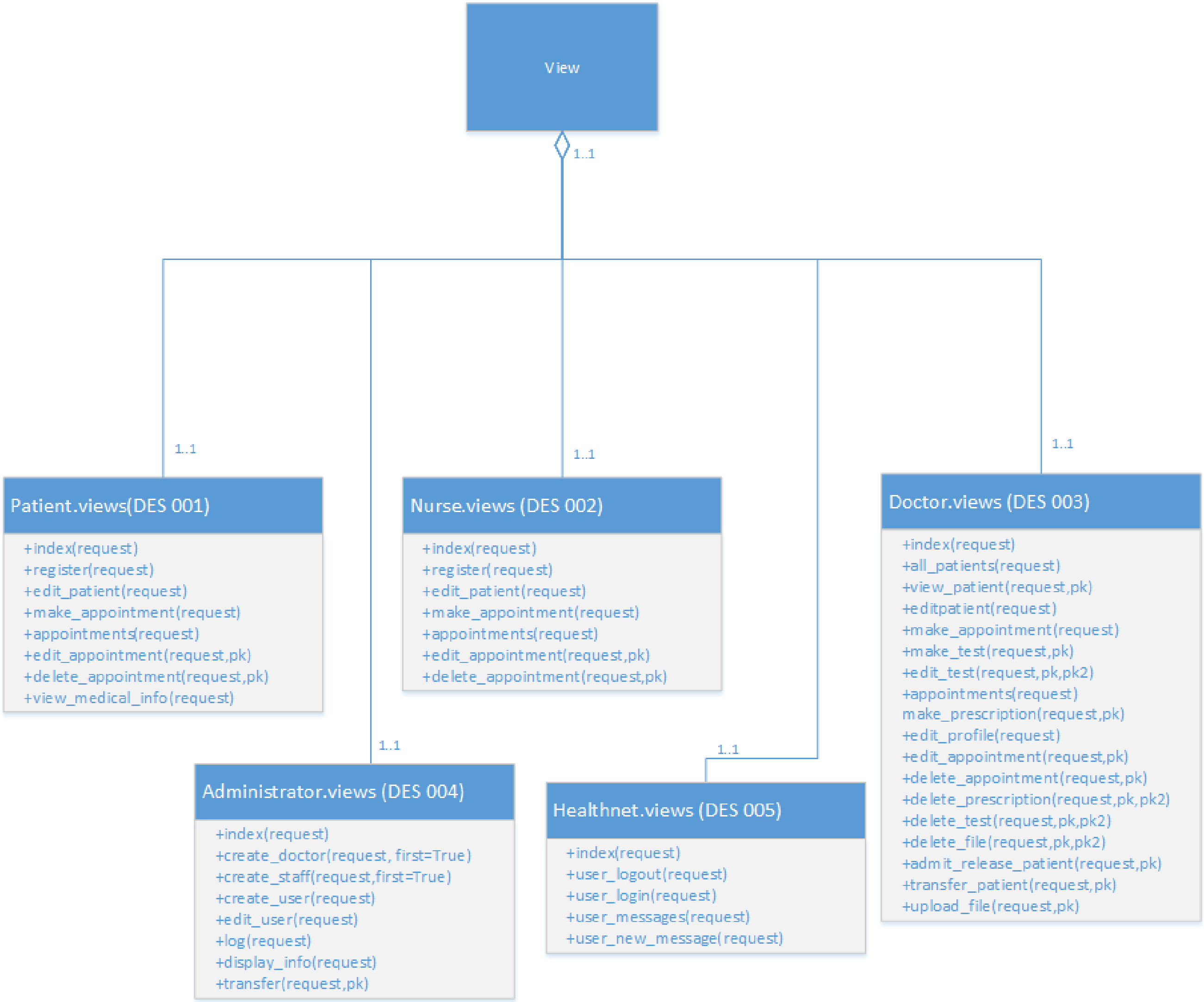

Patient.forms(DES 001)
PatientForm
PatientProfileForm
PatientEditProfileForm
PatientAppointmentForm


Doctor.forms(DES 003)
DoctorMakeAppointmentForm
DoctorUserForm
PrescriptionForm
TestForm
AdmitReleaseForm
TransferForm
UploadFileForm

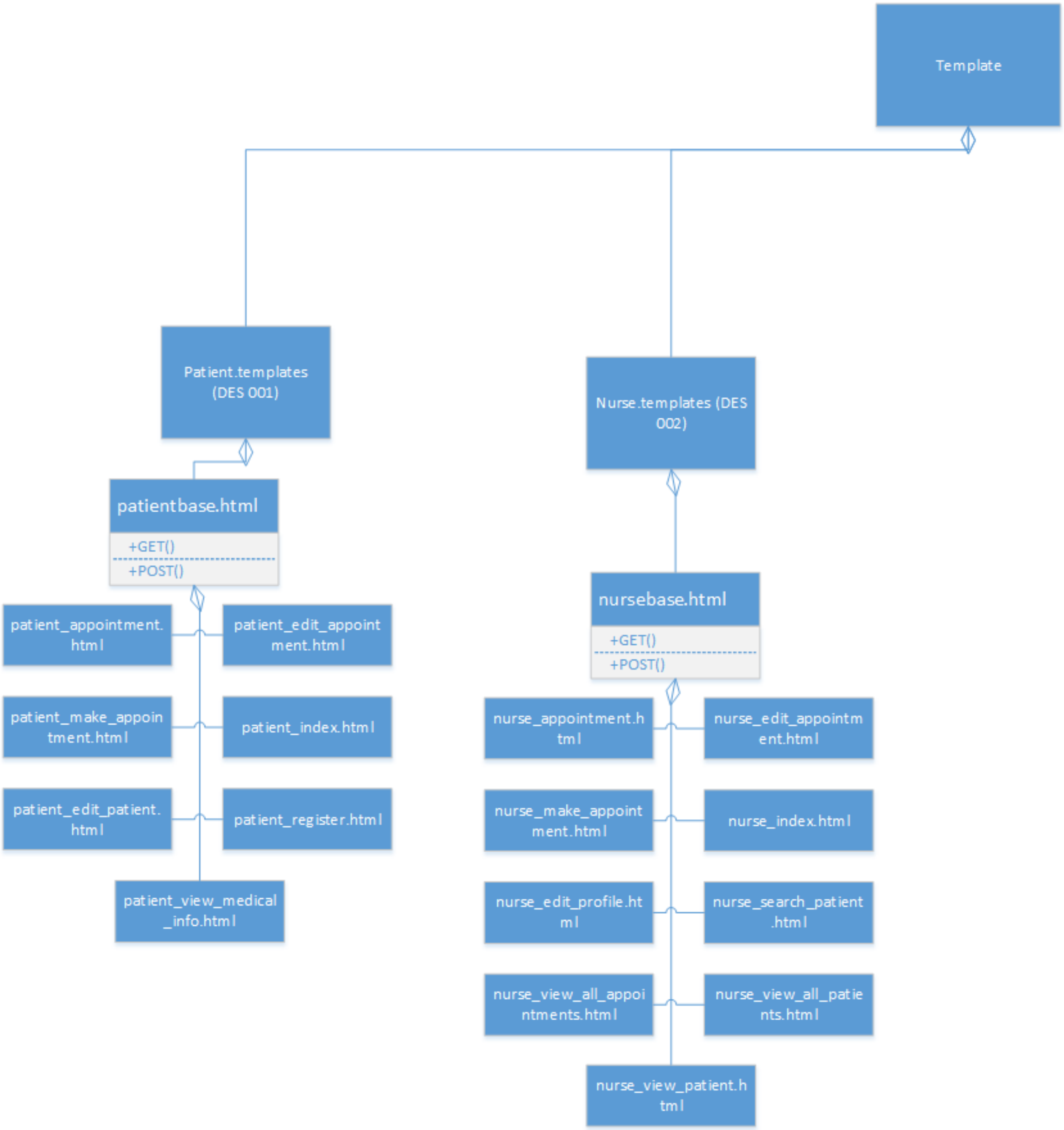

Administrator.forms
(DES 004)
AdminForm
AdminEditForm
DoctorProfileForm
DoctorForm
StaffForm
StaffProfileForm

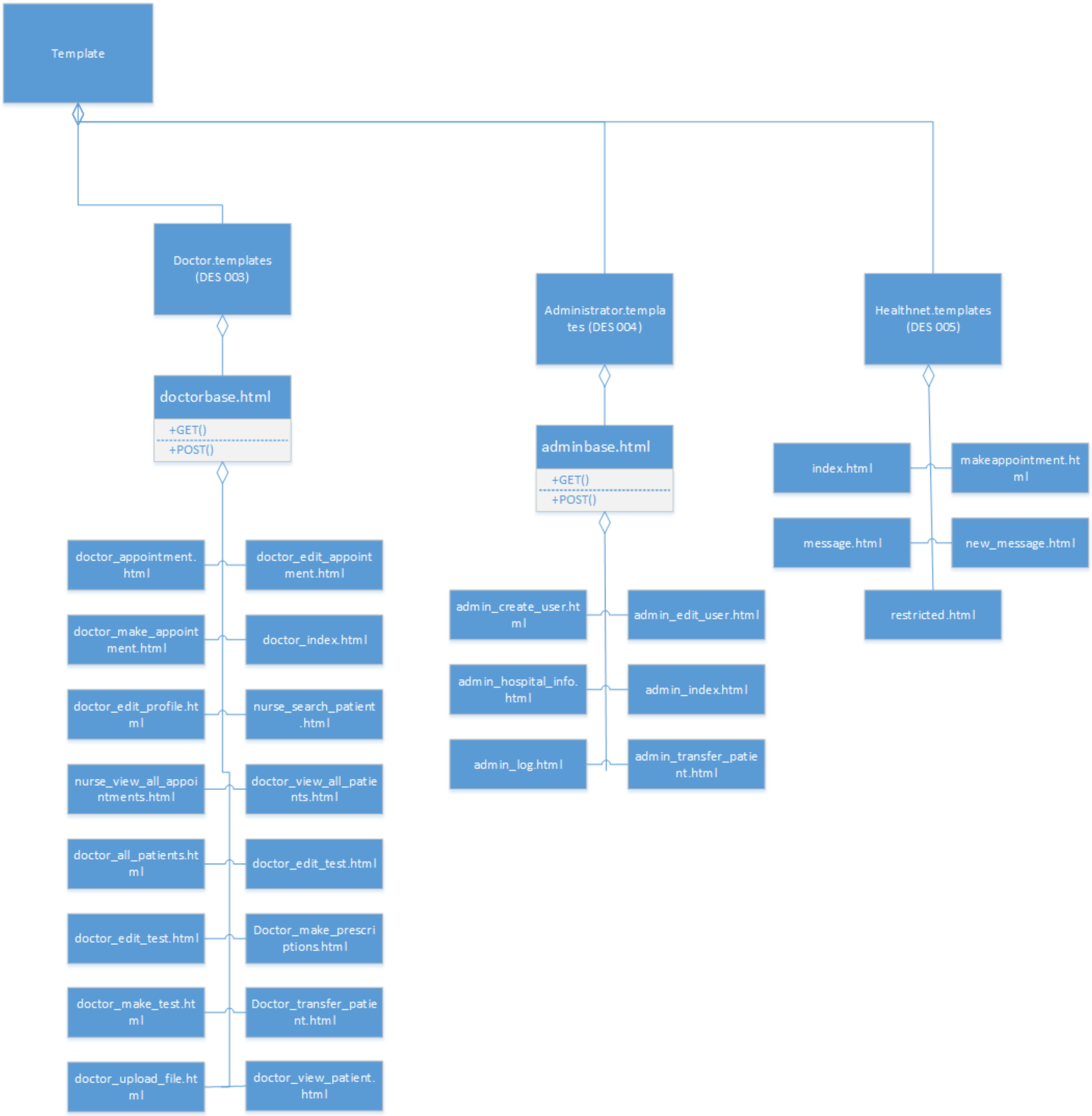

Healthnet.models
Hospital(models.Model)
+numberAdmittedPatients(self)
+patientCount(self)
+doctorCount(self)
+staffCount(self)
name
0..*
0..*
0..*
1..*
1..1
UserProfile(models.Model)
user
age
date_of_birth
spouse_first_name
next_of_kin
emergency_contact
ssn
numb
phone_number
sex
weight
height
insurance
prescription
hospital
is_in_hospital
DoctorProfile(models.Model)
+getName(self)
+getAllAppointments(self)
user
previous_employment
education_information
accreditation
licenses
hospital
0..*
0..*
1..1
0..*
1..*
1..1
Appointment(models.Model)
user
length
date_time
reason_for_appointment
additional_information
doctor
1..1
0..*
0..*
0..*
0..*
1..1
0..*
1..1
Prescription(models.Model)
user
prescribed_by_doctor
name
amount
reason
created_at
StaffProfile(models.Model)
+getName(self)
+getAllAppointments(self)
user
previous_employment
education_information
accreditation
licenses
hospital
1..1
0..*
Test(models.Model)
user
prescribed_by_doctor
name_of_test
result
reason
created_at
1..1
Transfer(models.Model)
user
reason
from_hospital
to_hospital
PatientFile(models.Model)
+extension(self)
user
file_name
patient_file
1..1
Message(models.Model)
from_user
to_user
date_time
message
1..1
1..1

Views Entire views for the project, shown under aggregate of the View component of Django’s Model View Template format.


View
1..1
1..1
1..1
1..1
1..1
1..1
Patient.views(DES 001)
+index(request)
+register(request)
+edit_patient(request)
+make_appointment(request)
+appointments(request)
+edit_appointment(request,pk)
+delete_appointment(request,pk)
+view_medical_info(request)
Nurse.views (DES 002)
+index(request)
+register(request)
+edit_patient(request)
+make_appointment(request)
+appointments(request)
+edit_appointment(request,pk)
+delete_appointment(request,pk)
Doctor.views (DES 003)
+index(request)
+all_patients(request)
+view_patient(request,pk)
+editpatient(request)
+make_appointment(request)
+make_test(request,pk)
+edit_test(request,pk,pk2)
+appointments(request)
make_prescription(request,pk)
+edit_profile(request)
+edit_appointment(request,pk)
+delete_appointment(request,pk)
+delete_prescription(request,pk,pk2)
+delete_test(request,pk,pk2)
+delete_file(request,pk,pk2)
+admit_release_patient(request,pk)
+transfer_patient(request,pk)
+upload_file(request,pk)
Administrator.views (DES 004)
+index(request)
+create_doctor(request, first=True)
+create_staff(request,first=True)
+create_user(request)
+edit_user(request)
+log(request)
+display_info(request)
+transfer(request,pk)
Healthnet.views (DES 005)
+index(request)
+user_logout(request)
+user_login(request)
+user_messages(request)
+user_new_message(request)

Templates Split half and half, with the first half showing patient and nurse templates, the second showing admin, doctor, and healthnet.

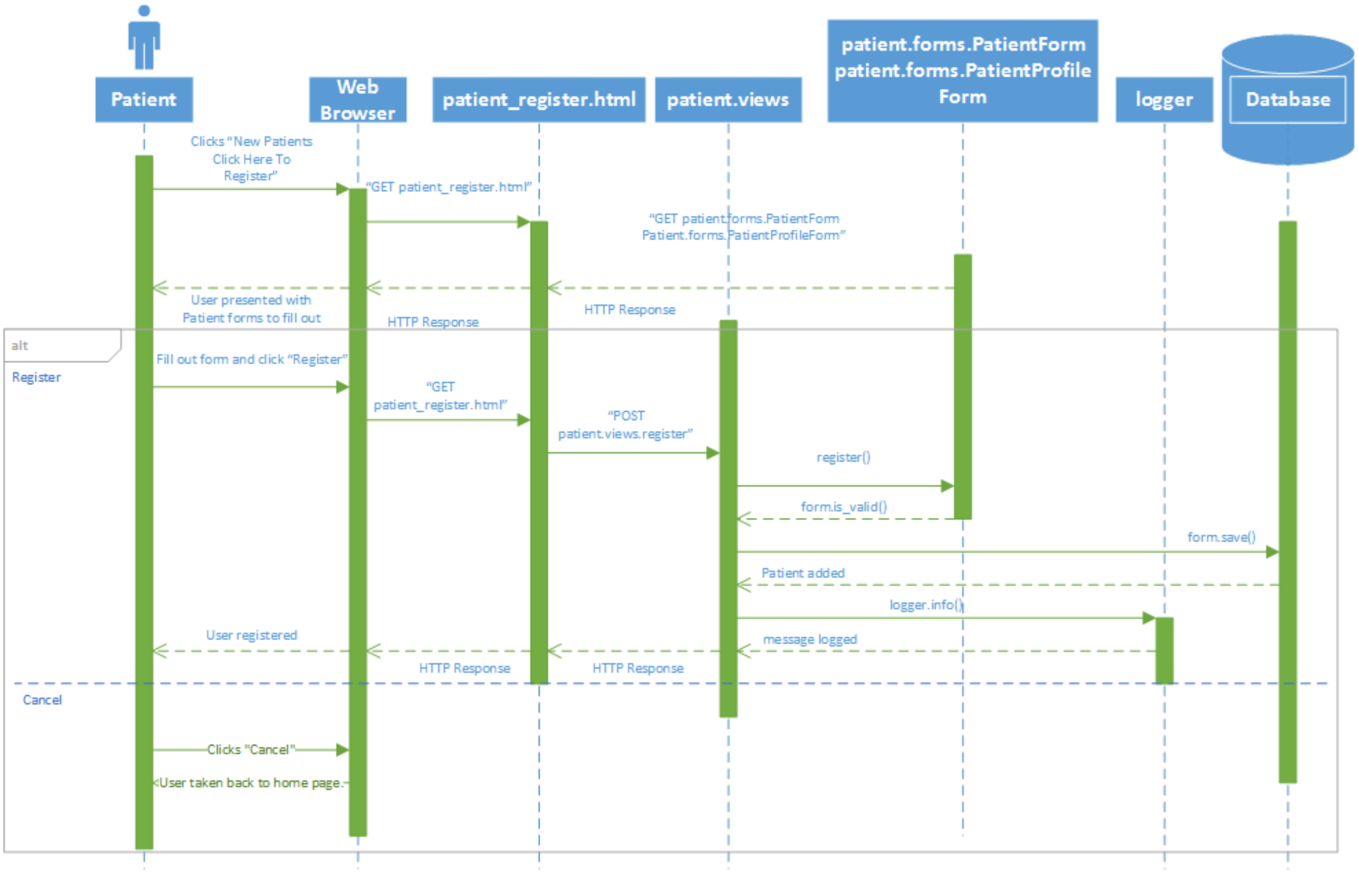

Template
Patient.templates (DES 001)
Nurse.templates (DES 002)
patientbase.html
+GET()
+POST()
nursebase.html
+GET()
+POST()
patient_appointment.html
patient_edit_appointment.html
patient_make_appointment.html
patient_index.html
patient_edit_patient.html
patient_register.html
patient_view_medical_info.html
nurse_appointment.html
nurse_edit_appointment.html
nurse_make_appointment.html
nurse_index.html
nurse_edit_profile.html
nurse_search_patient.html
nurse_view_all_appointments.html
nurse_view_all_patients.html
nurse_view_patient.html


Template
Doctor.templates (DES 003)
Administrator.templates (DES 004)
Healthnet.templates (DES 005)
doctorbase.html
+GET()
+POST()
adminbase.html
+GET()
+POST()
index.html
makeappointment.html
message.html
new_message.html
restricted.html
doctor_appointment.html
doctor_edit_appointment.html
doctor_make_appointment.html
doctor_index.html
doctor_edit_profile.html
nurse_search_patient.html
nurse_view_all_appointments.html
doctor_view_all_patients.html
doctor_all_patients.html
doctor_edit_test.html
doctor_edit_test.html
Doctor_make_prescriptions.html
doctor_make_test.html
Doctor_transfer_patient.html
doctor_upload_file.html
doctor_view_patient.html
admin_create_user.html
admin_edit_user.html
admin_hospital_info.html
admin_index.html
admin_log.html
admin_transfer_patient.html

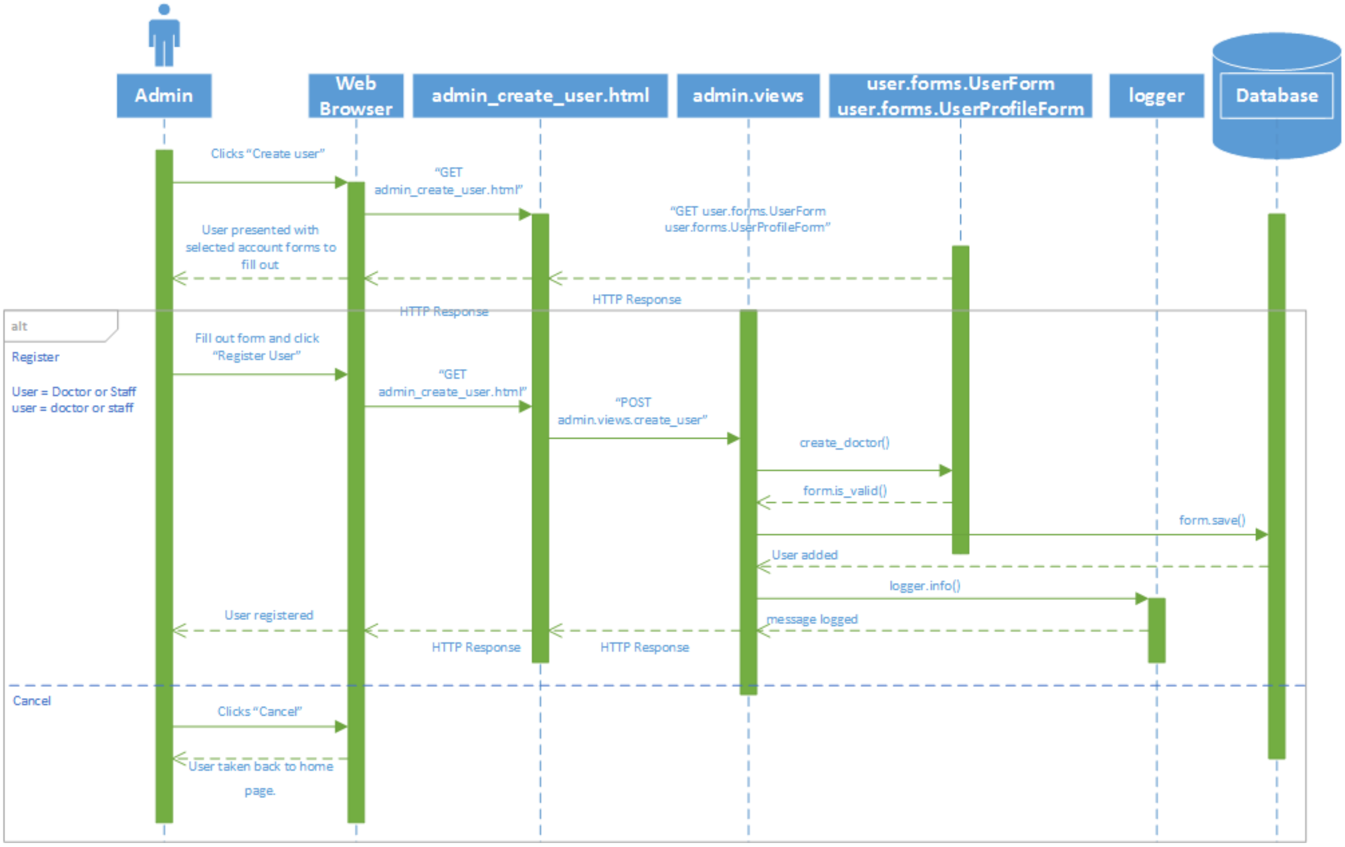
3.2 Sequence Diagram(s)

Our sequence diagrams are each traceable to a use case in the requirements which they are based off of, and to the class diagram from which they derive their functionality. Created using Visio 2013

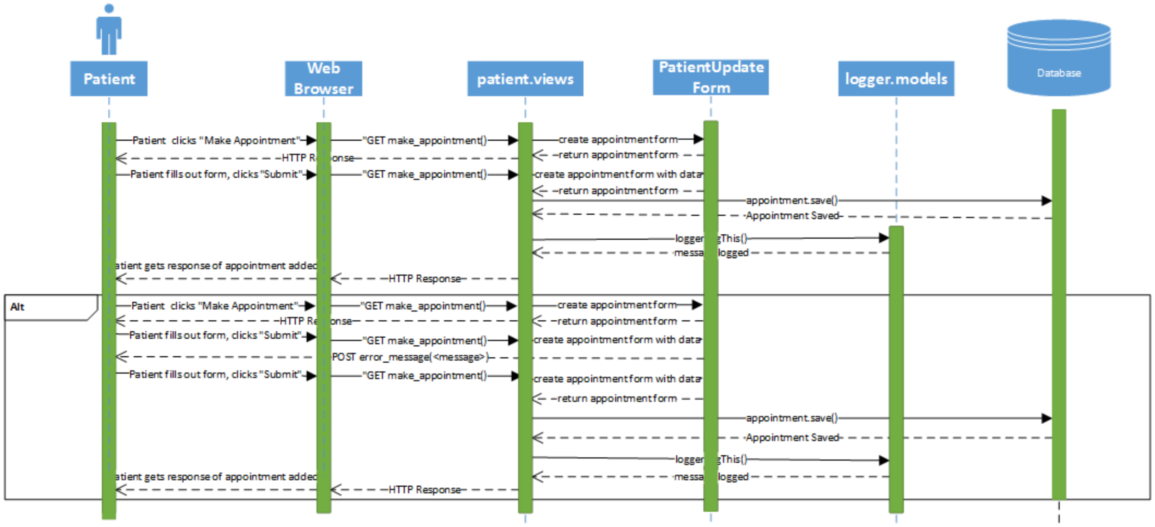
Use Case 01: "Patient Registration"


Patient
Web Browser
patient_register.html
patient.views
patient.forms.PatientForm
patient.forms.PatientProfile Form
logger
Database
Clicks "New Patients Click Here To Register"
"GET patient_register.html"
"GET patient.forms.PatientForm Patient.forms.PatientProfileForm"
User presented with Patient forms to fill out
HTTP Response
HTTP Response
alt
Register
Fill out form and click "Register"
"GET patient_register.html"
"POST patient.views.register"
register()
form.is_valid()
form.save()
Patient added
logger.info()
message logged
User registered
HTTP Response
HTTP Response
Cancel
Clicks "Cancel"
User taken back to home page.

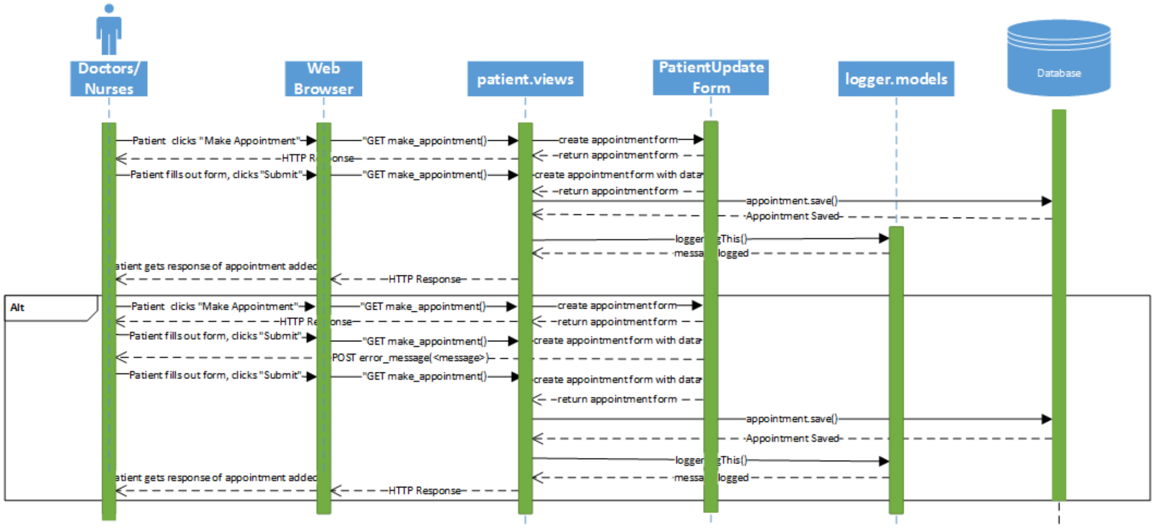
Use Case 02: "Administrator Registration"



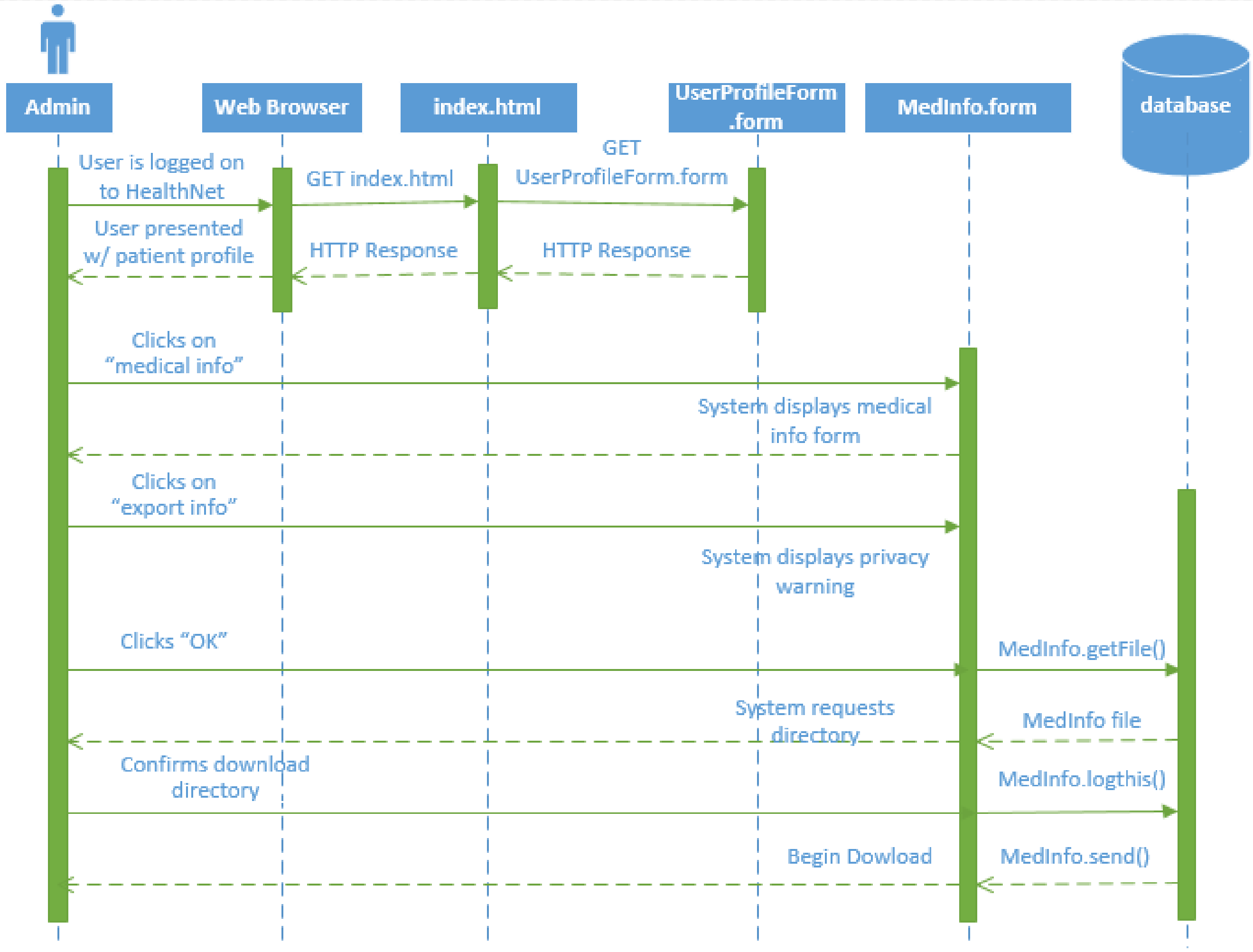
Use Case 03: "Update Patient Profile Information"


Patient
Web Browser
patient.views
PatientUpdate Form
logger.models
Database
Patient clicks "Make Appointment"
"GET make_appointment()
create appointment form
return appointment form
HTTP Response
Patient fills out form, clicks "Submit"
"GET make_appointment()
create appointment form with data
return appointment form
appointment.save()
Appointment Saved
logger gThis()
messa logged
atient gets response of appointment added
HTTP Response
Alt
Patient clicks "Make Appointment"
"GET make_appointment()
create appointment form
return appointment form
HTTP Response
Patient fills out form, clicks "Submit"
"GET make_appointment()
create appointment form with data
POST error_message(<message>)
Patient fills out form, clicks "Submit"
"GET make_appointment()
create appointment form with data
return appointment form
appointment.save()
Appointment Saved
logger gThis()
messa logged
atient gets response of appointment added
HTTP Response

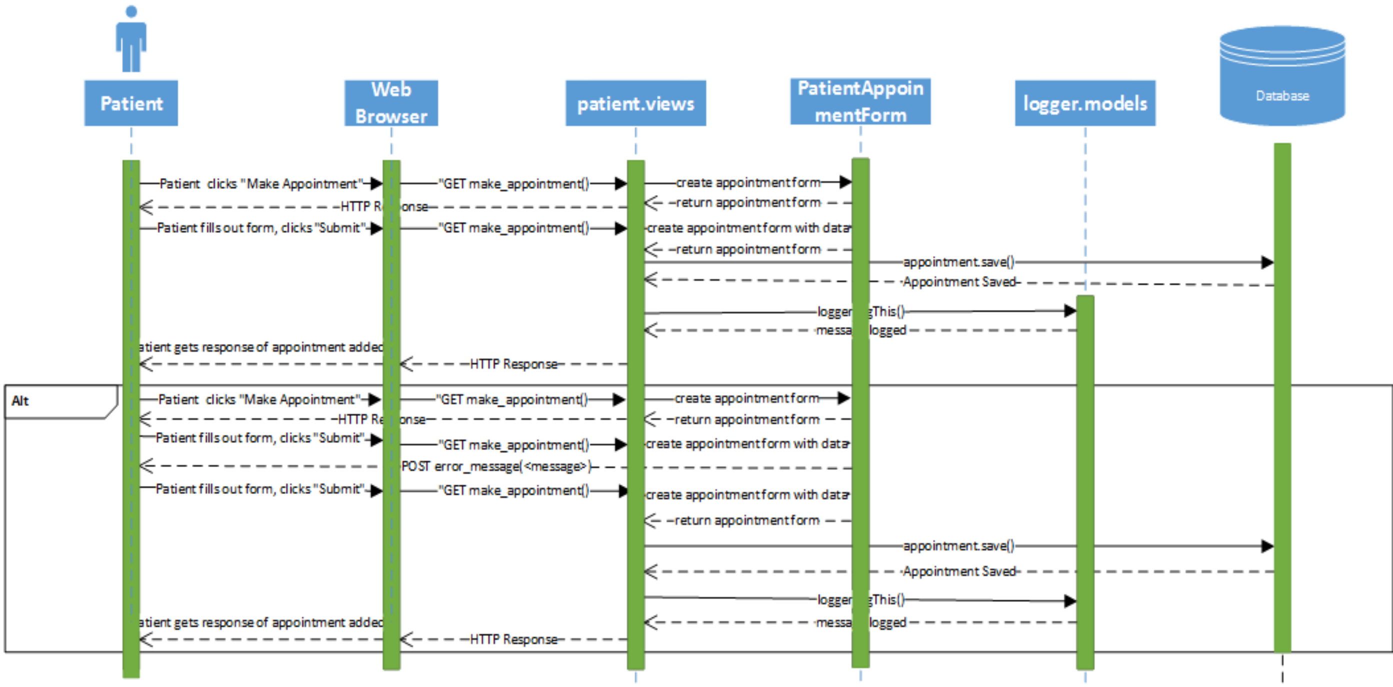
Use Case 04: “Updating Patient Medical Information”



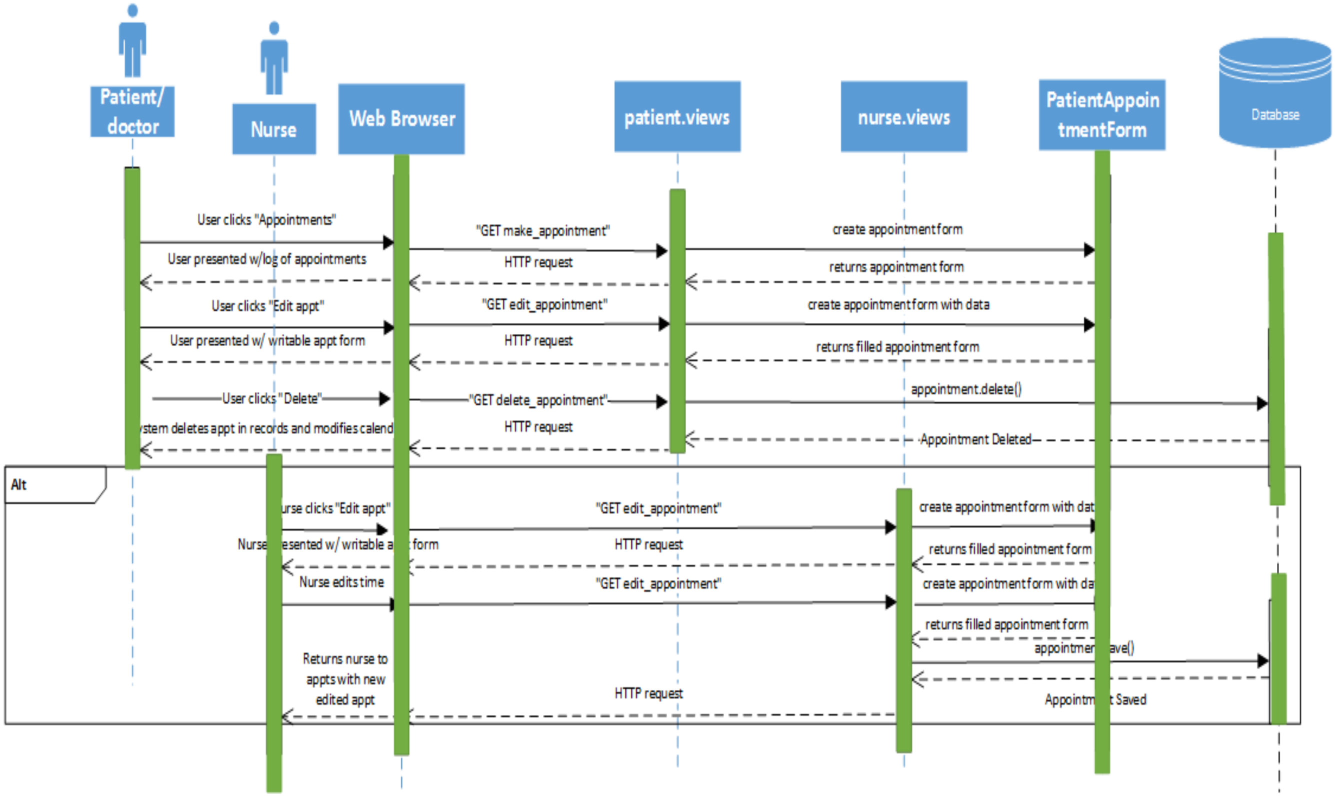
Use Case 05: "Export Information"



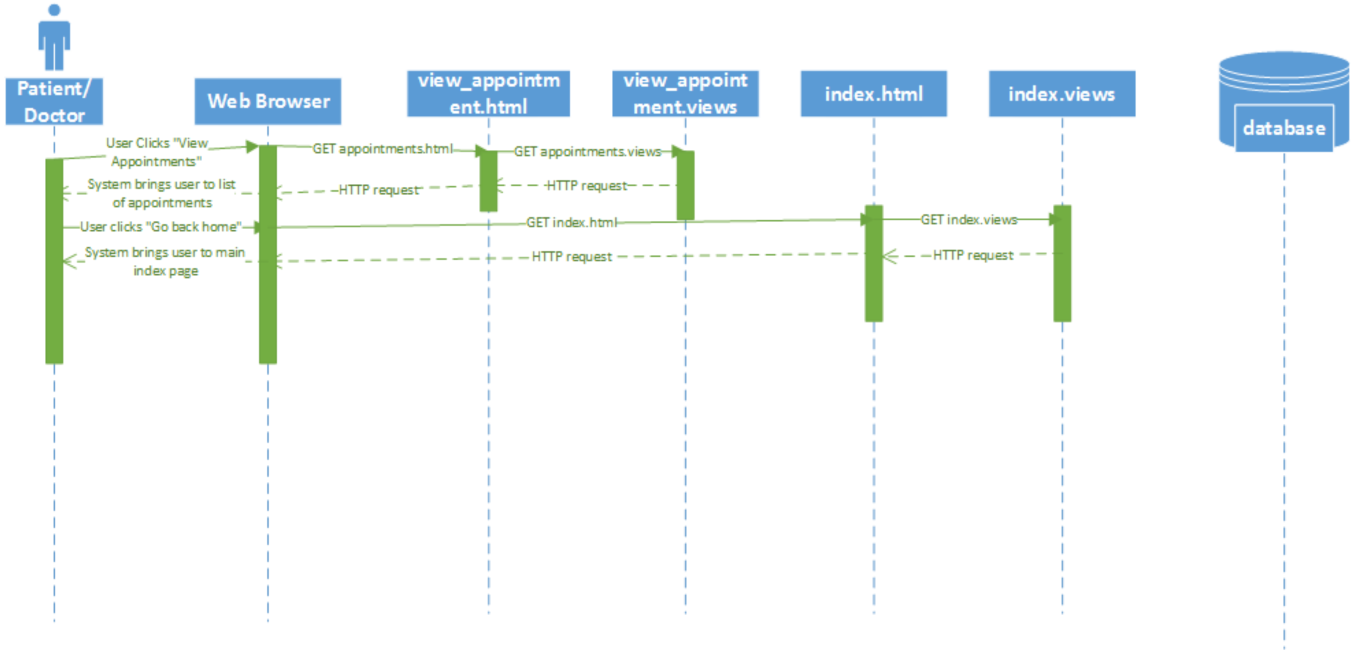
Use Case 06: “Create or Update Patent Appointment”



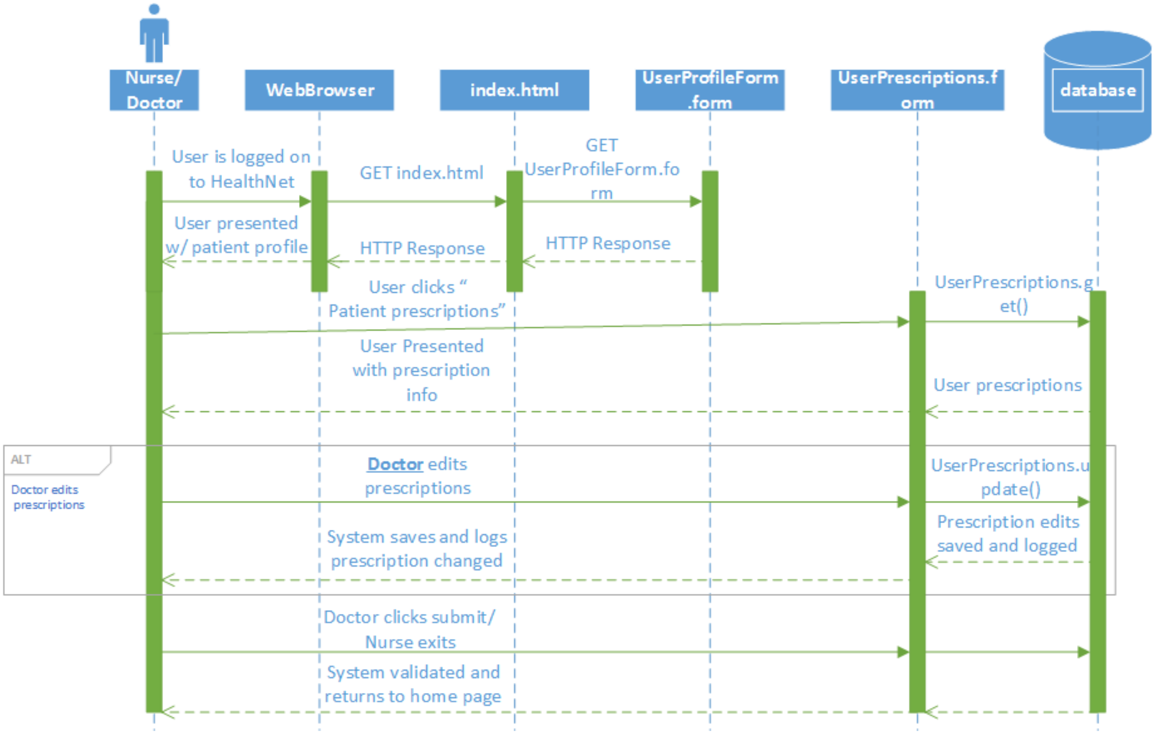
Use Case 07: “Cancel Patient Appointment”



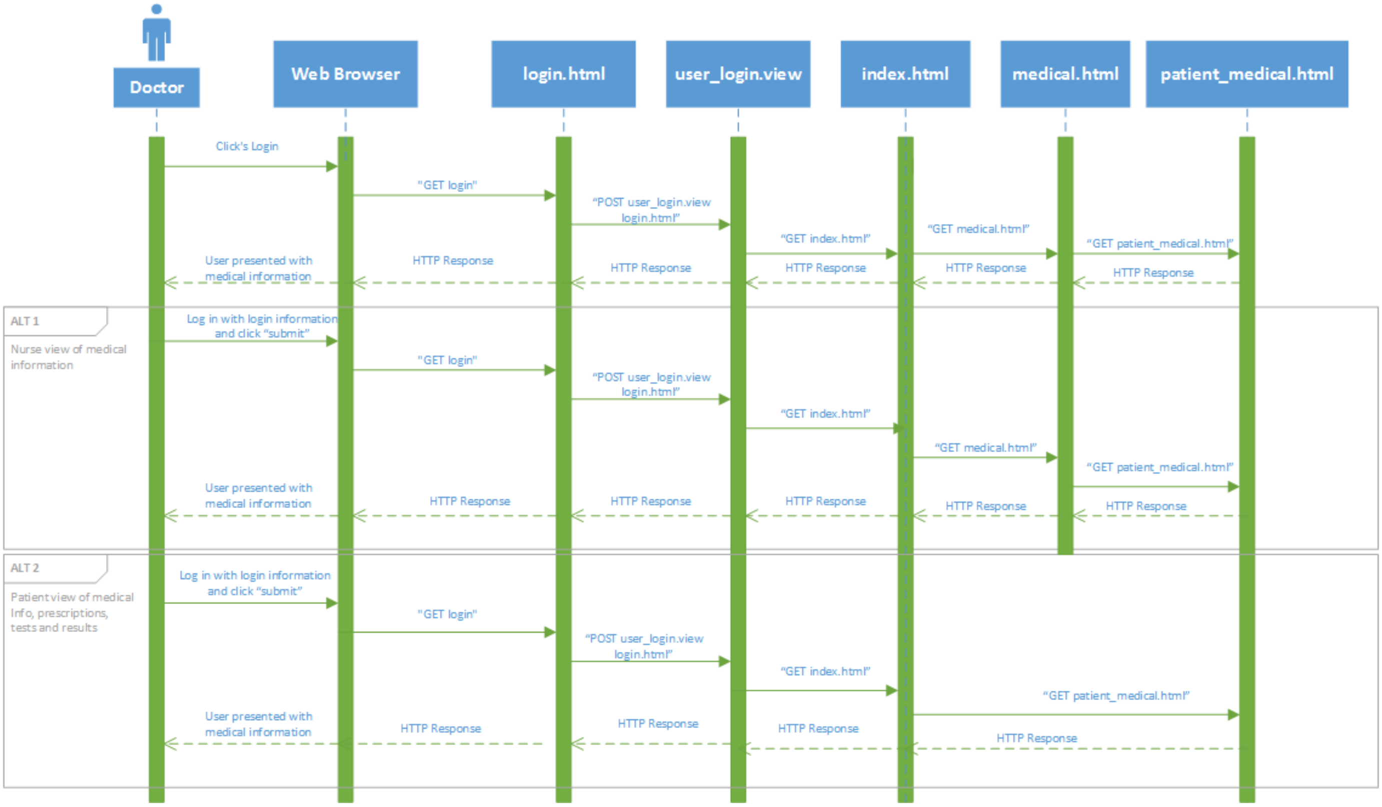
Use Case 08: “Appointment Calendar”



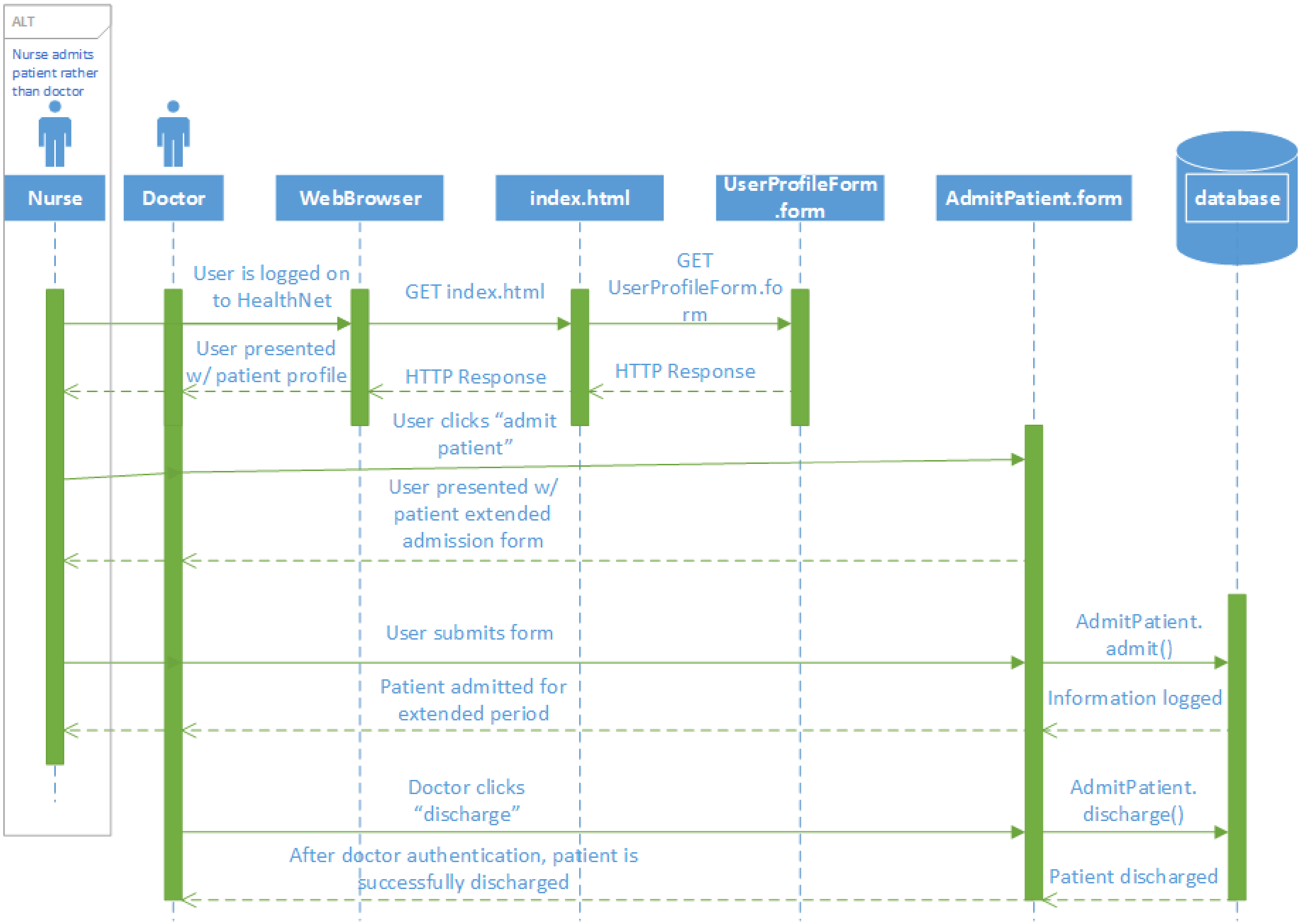
Use Case 09: “Viewing/Updating Patient Prescriptions”


Nurse/ Doctor
WebBrowser
index.html
UserProfileForm .form
UserPrescriptions.f orm
database
User is logged on to HealthNet
GET index.html
GET UserProfileForm.fo rm
User presented w/ patient profile
HTTP Response
HTTP Response
User clicks “ Patient prescriptions”
UserPrescriptions.g et()
User Presented with prescription info
User prescriptions
ALT
Doctor edits prescriptions
Doctor edits prescriptions
UserPrescriptions.u pdate()
System saves and logs prescription changed
Prescription edits saved and logged
Doctor clicks submit/ Nurse exits
System validated and returns to home page

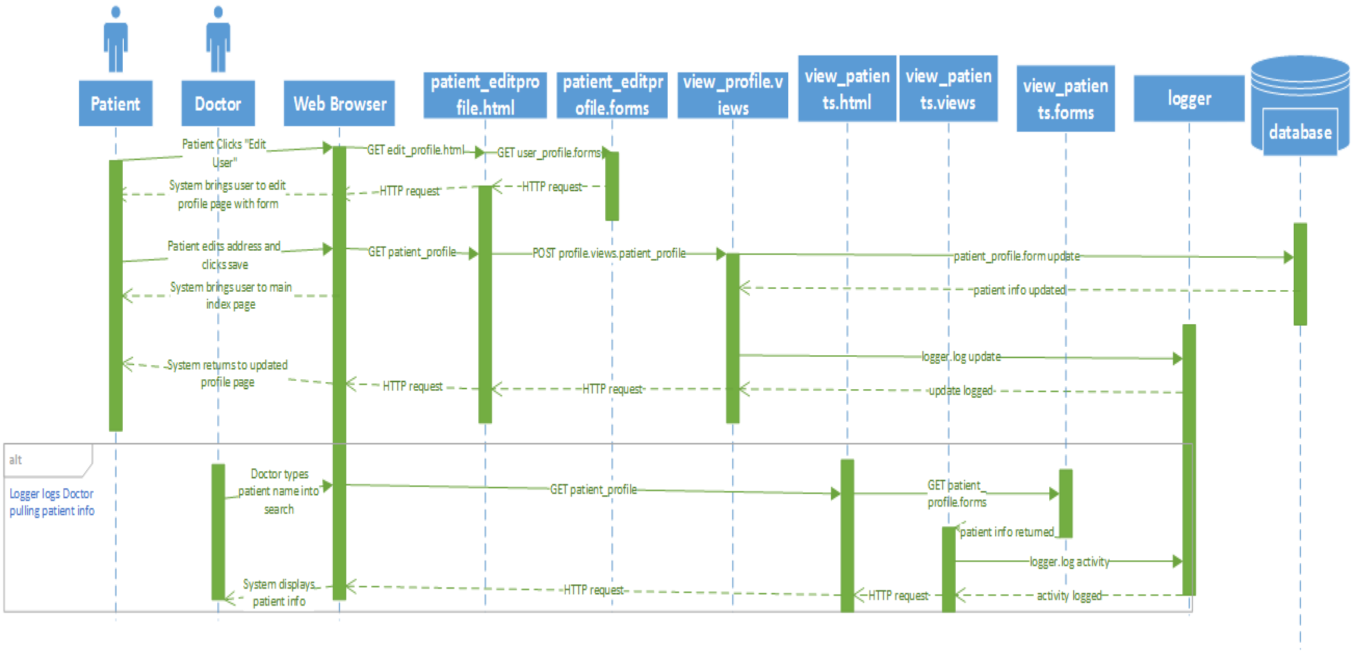
Use Case 10: “Viewing Patient Medical Information, Prescriptions and Tests and Results”


Doctor
Web Browser
login.html
user_login.view
index.html
medical.html
patient_medical.html
Click's Login
"GET login"
"POST user_login.view login.html"
"GET index.html"
"GET medical.html"
"GET patient_medical.html"
User presented with medical information
HTTP Response
HTTP Response
HTTP Response
HTTP Response
HTTP Response
ALT 1
Nurse view of medical information
Log in with login information and click "submit"
"GET login"
"POST user_login.view login.html"
"GET index.html"
"GET medical.html"
"GET patient_medical.html"
User presented with medical information
HTTP Response
HTTP Response
HTTP Response
HTTP Response
HTTP Response
ALT 2
Patient view of medical Info, prescriptions, tests and results
Log in with login information and click "submit"
"GET login"
"POST user_login.view login.html"
"GET index.html"
"GET patient_medical.html"
User presented with medical information
HTTP Response
HTTP Response
HTTP Response
HTTP Response

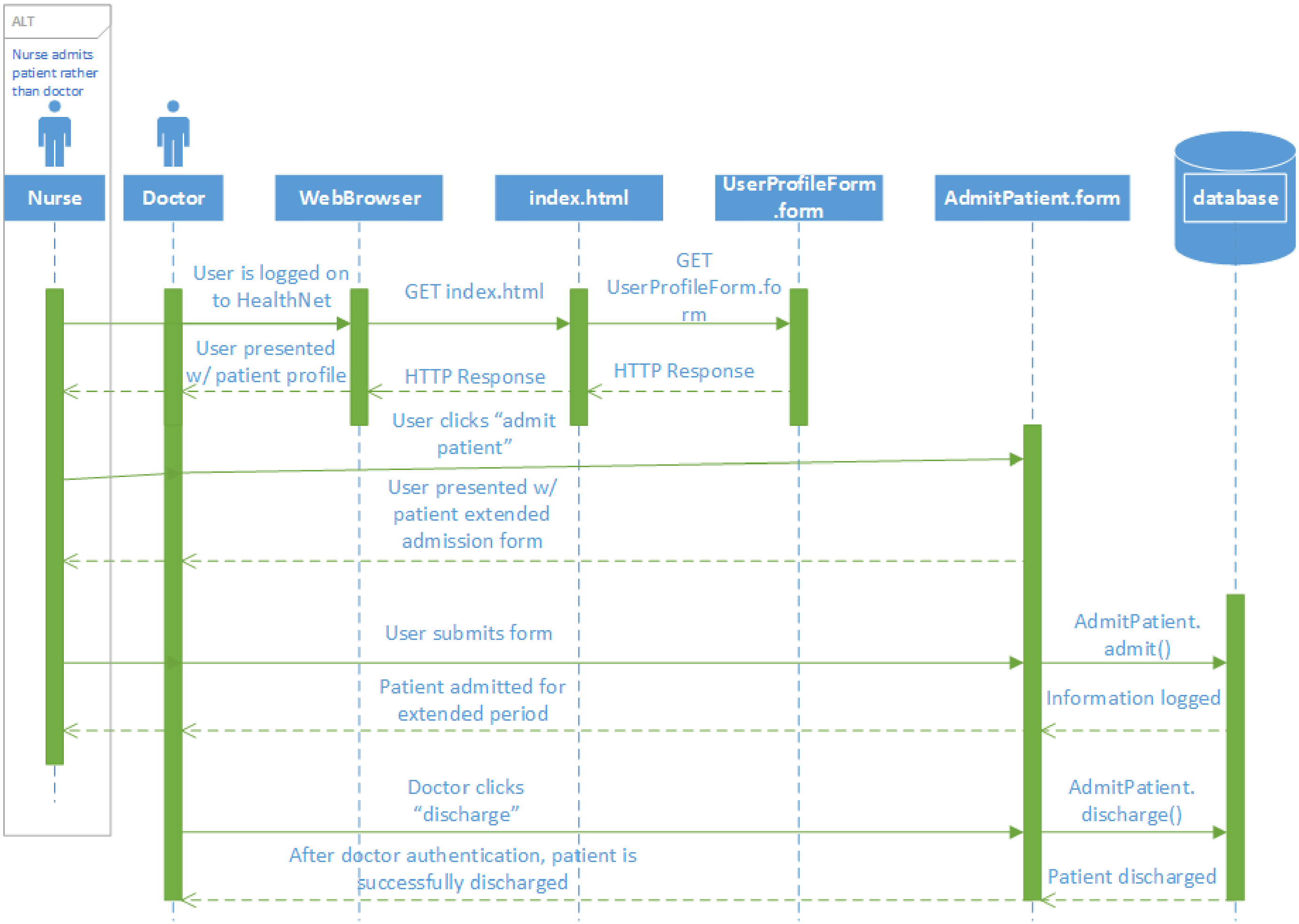
Use Case 11: "Release Patient Test Results"


ALT
Nurse admits patient rather than doctor
Nurse
Doctor
WebBrowser
index.html
UserProfileForm .form
AdmitPatient.form
database
User is logged on to HealthNet
GET index.html
GET UserProfileForm.form
User presented w/ patient profile
HTTP Response
HTTP Response
User clicks "admit patient"
User presented w/ patient extended admission form
User submits form
AdmitPatient. admit()
Patient admitted for extended period
Information logged
Doctor clicks "discharge"
AdmitPatient. discharge()
After doctor authentication, patient is successfully discharged
Patient discharged

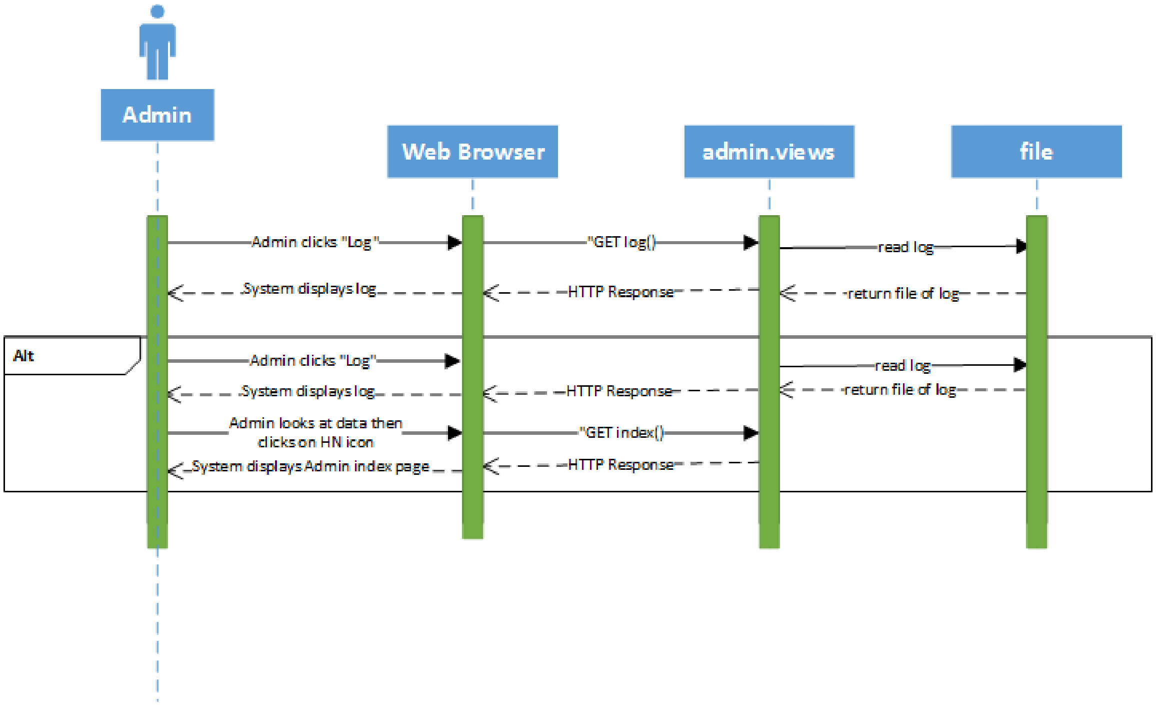
Use Case 12: “Logging System Activity”


Patient
Doctor
Web Browser
patient_editprofile.html
patient_editprofile.forms
view_profile.views
view_patients.html
view_patients.views
view_patients.forms
logger
database
Patient Clicks "Edit User"
GET edit_profile.html
GET user_profile.forms
System brings user to edit profile page with form
HTTP request
HTTP request
Patient edits address and clicks save
GET patient_profile
POST profile.views.patient_profile
patient_profile.form update
patient info updated
System brings user to main index page
logger.log update
System returns to updated profile page
HTTP request
HTTP request
update logged
alt
Logger logs Doctor pulling patient info
Doctor types patient name into search
GET patient_profile
GET patient_profile.forms
patient info returned
logger.log activity
System displays patient info
HTTP request
HTTP request
activity logged

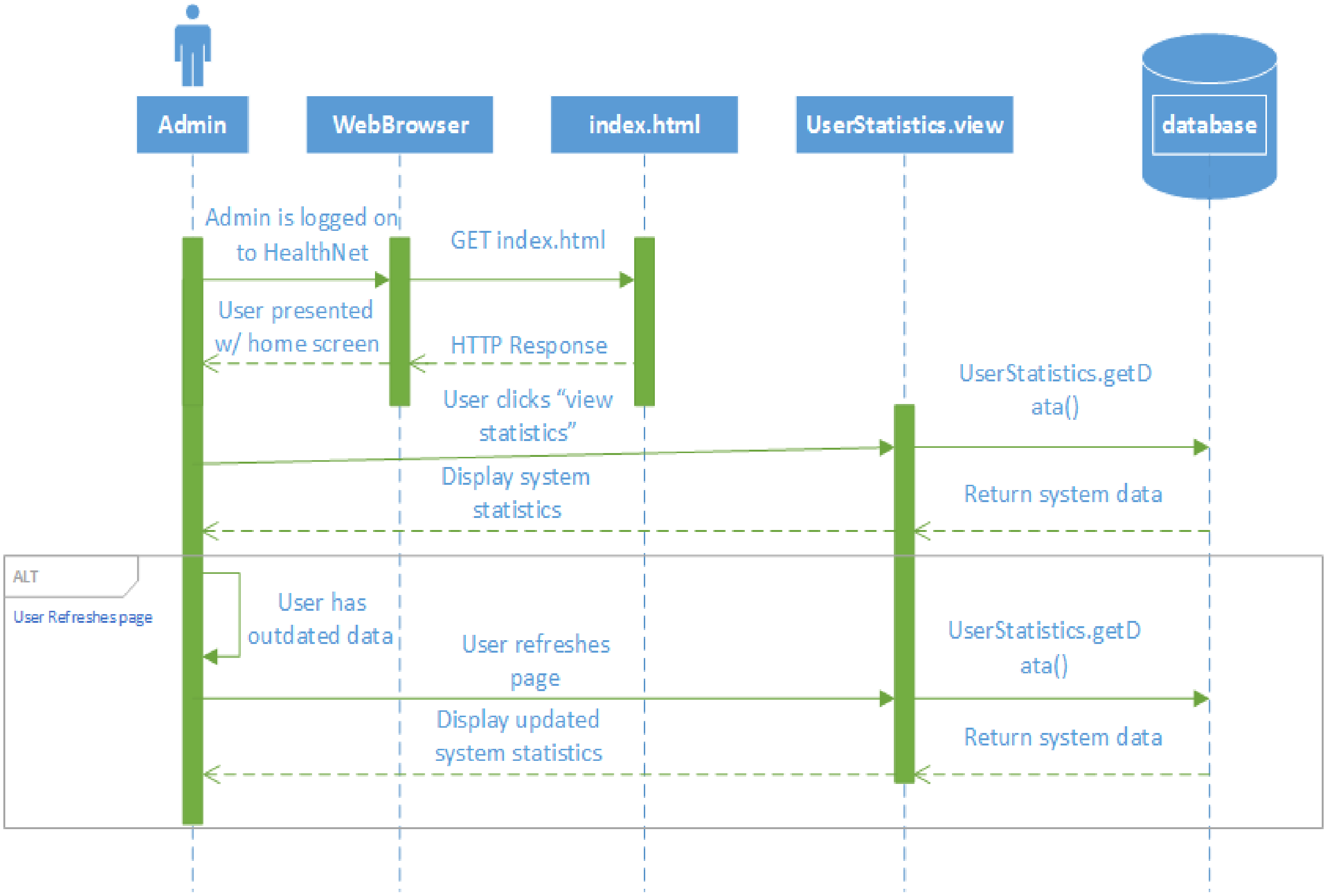
Use Case 13: "Admission and Discharge to/from Hospital"


ALT
Nurse admits patient rather than doctor
Nurse
Doctor
WebBrowser
index.html
UserProfileForm.form
AdmitPatient.form
database
User is logged on to HealthNet
GET index.html
GET UserProfileForm.form
User presented w/ patient profile
HTTP Response
HTTP Response
User clicks "admit patient"
User presented w/ patient extended admission form
User submits form
AdmitPatient.admit()
Patient admitted for extended period
Information logged
Doctor clicks "discharge"
AdmitPatient.discharge()
After doctor authentication, patient is successfully discharged
Patient discharged

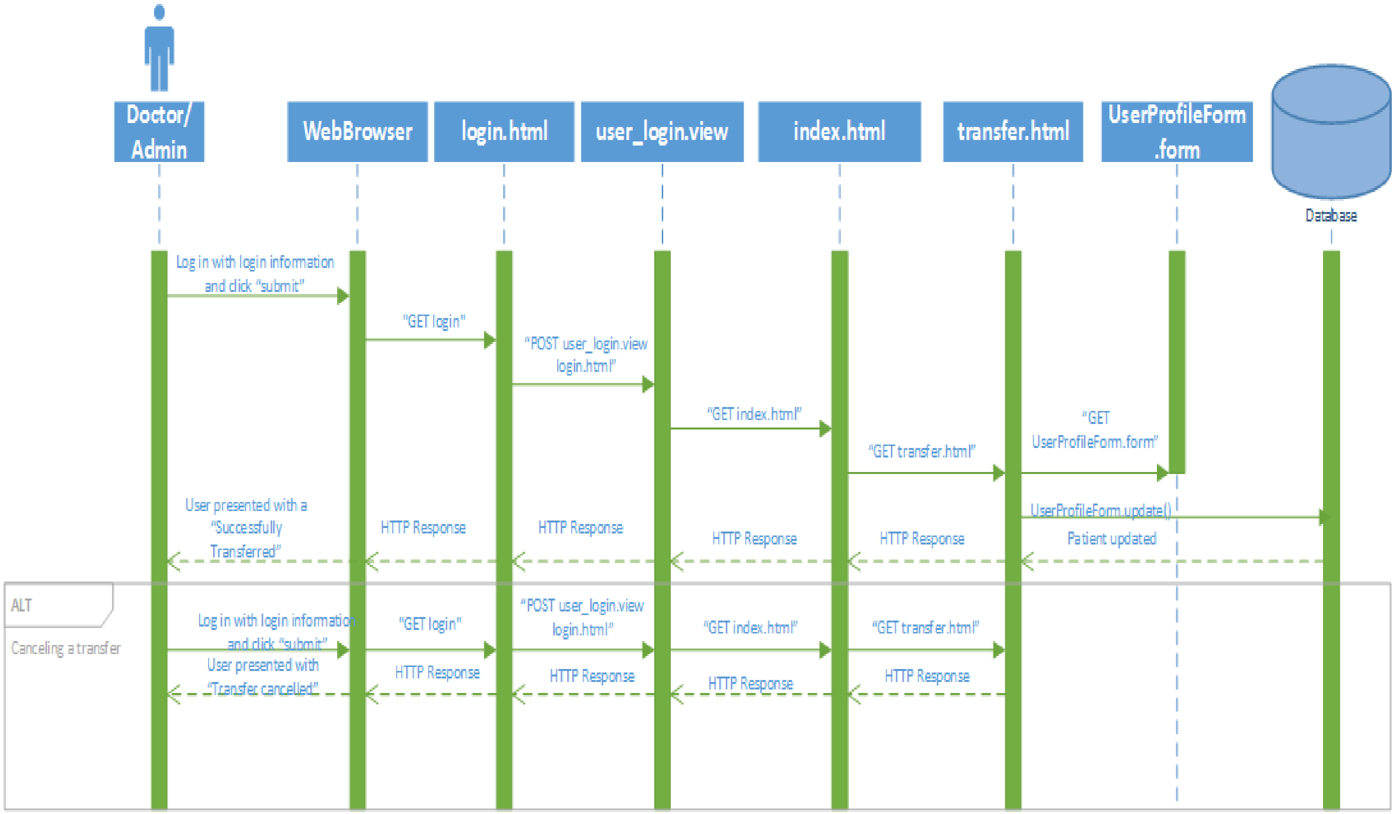
Use Case 14: “Viewing Activity Log”



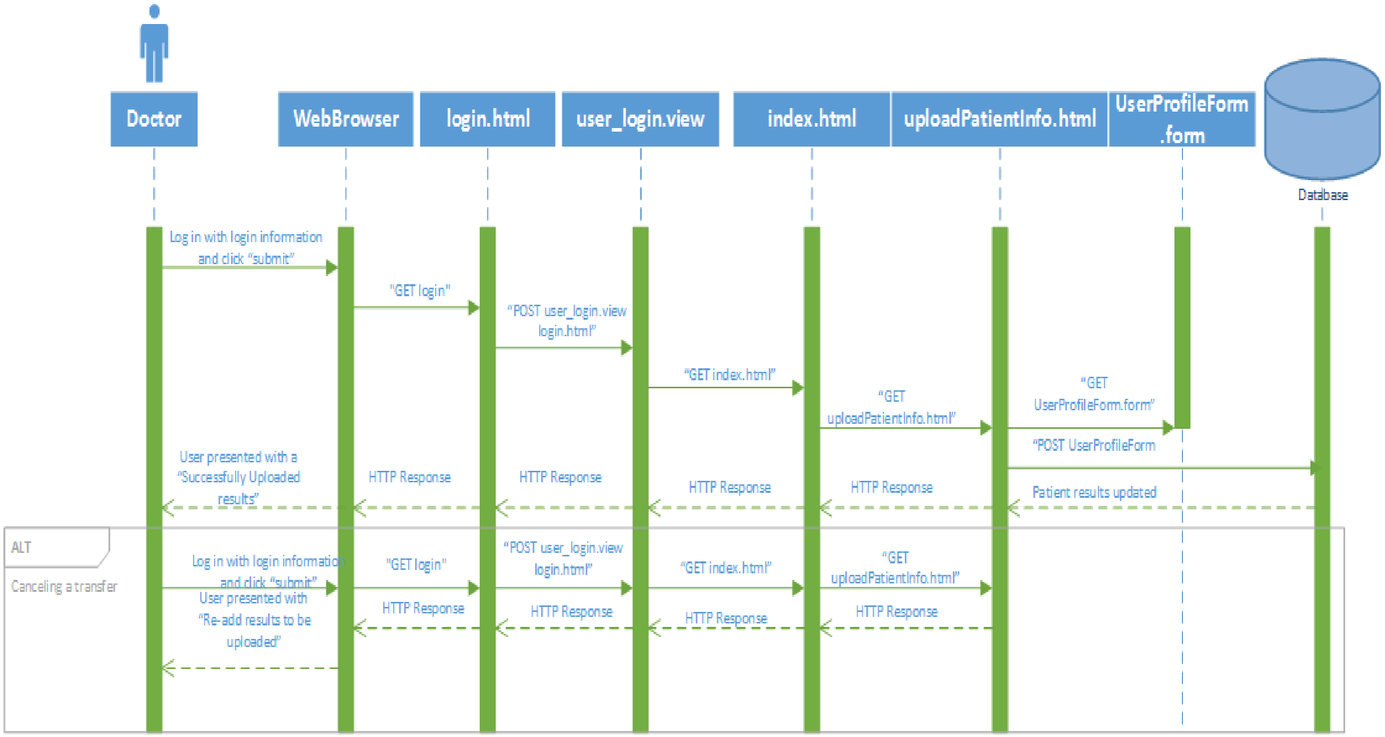
Use Case 15: "Viewing System Statistics"



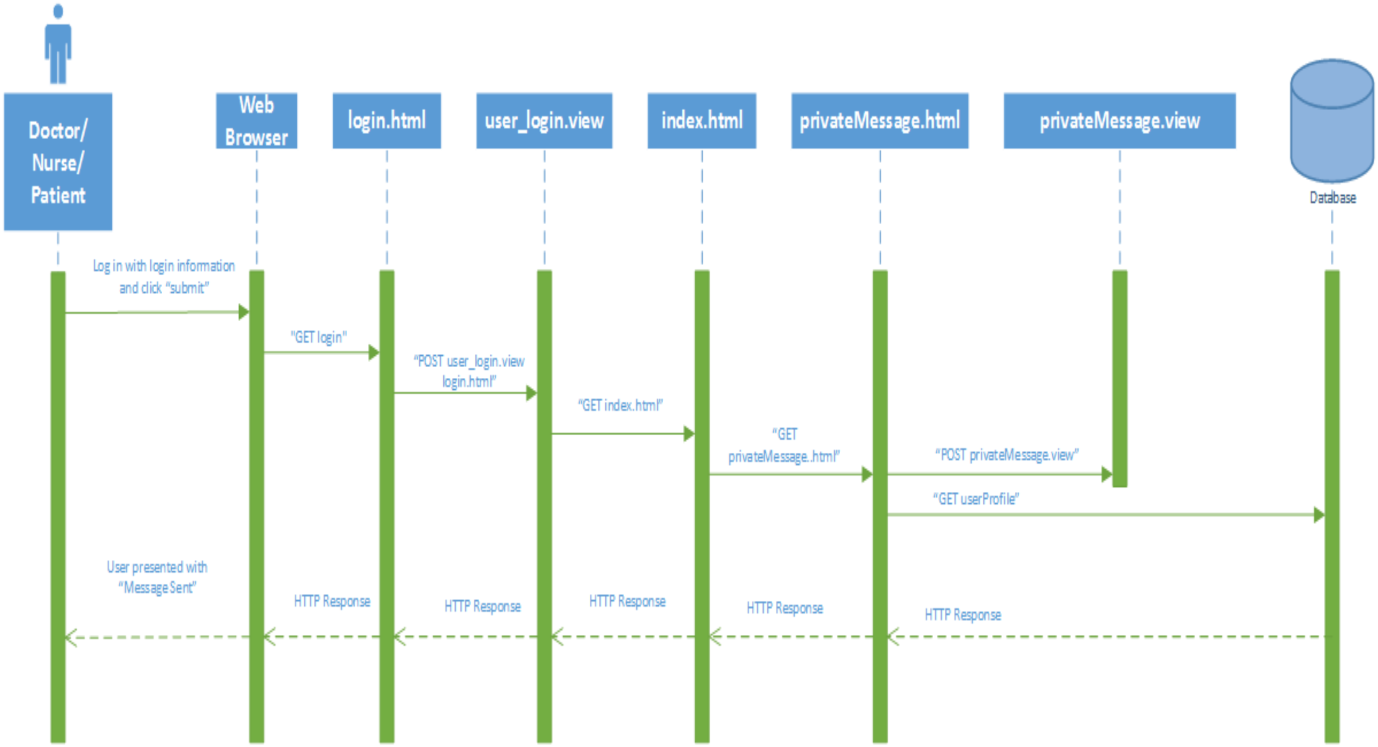
Use Case 16: “Patient transfer”



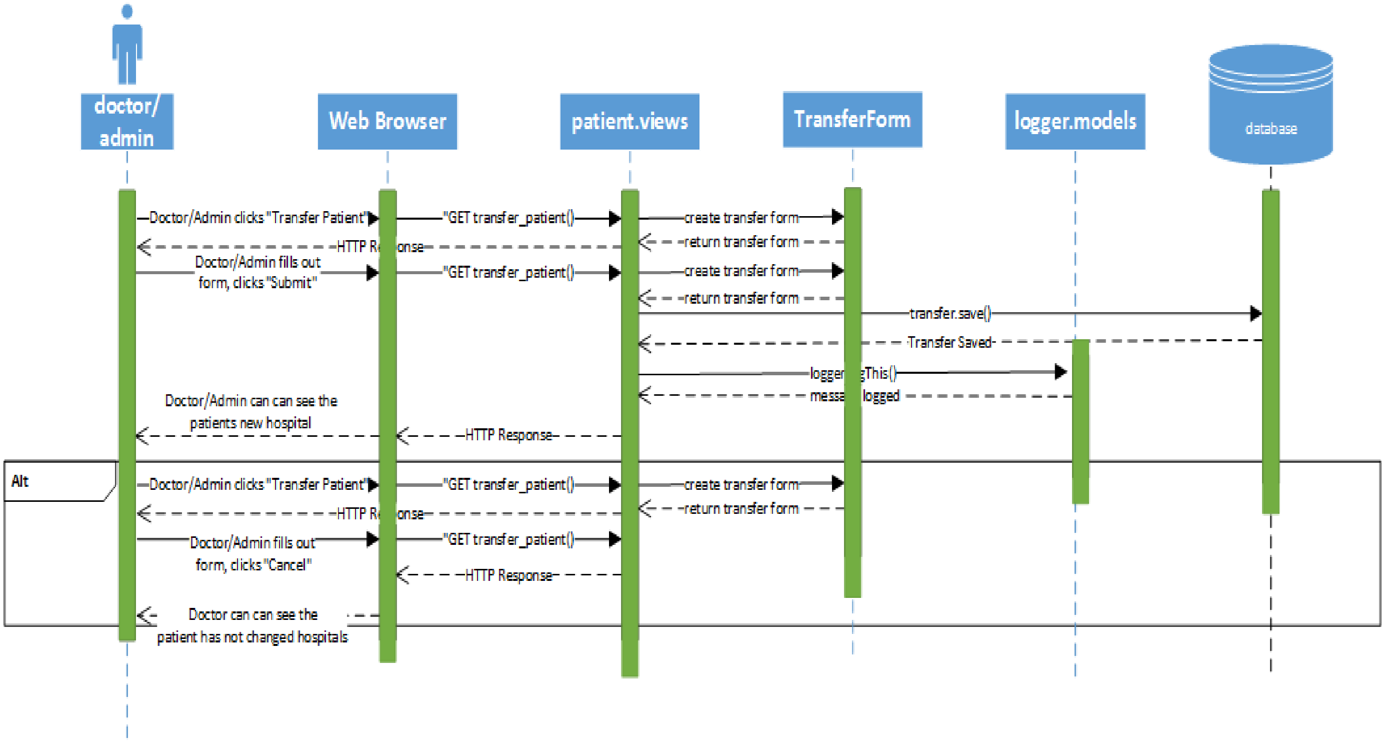
Use Case 17: “Upload Patient Information”


Doctor
WebBrowser
login.html
user_login.view
index.html
uploadPatientInfo.html
UserProfileForm .form
Database
Log in with login information and click “submit”
"GET login"
“POST user_login.view login.html”
“GET index.html”
“GET uploadPatientInfo.html”
“GET UserProfileForm.form”
“POST UserProfileForm
User presented with a “Successfully Uploaded results”
HTTP Response
HTTP Response
HTTP Response
HTTP Response
Patient results updated
ALT
Canceling a transfer
Log in with login information and click “submit”
"GET login"
“POST user_login.view login.html”
“GET index.html”
“GET uploadPatientInfo.html”
User presented with “Re-add results to be uploaded”
HTTP Response
HTTP Response
HTTP Response
HTTP Response

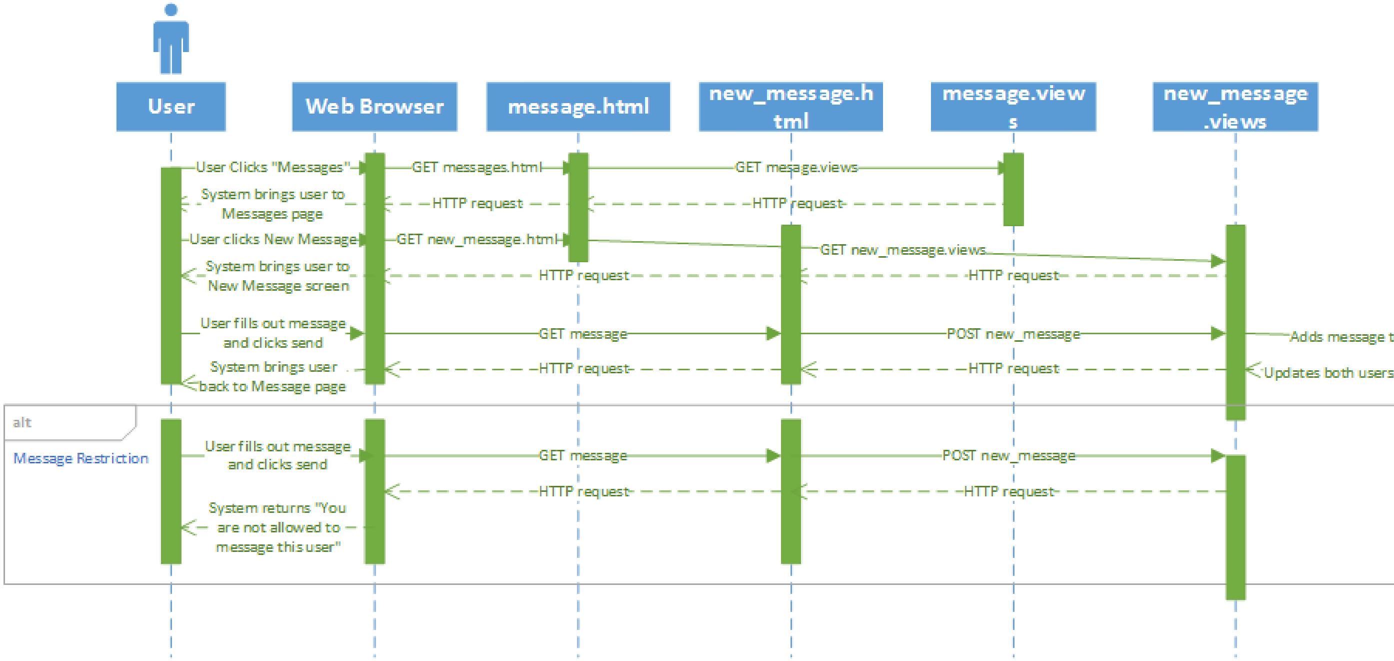
Use Case 18: “Send Private Message”



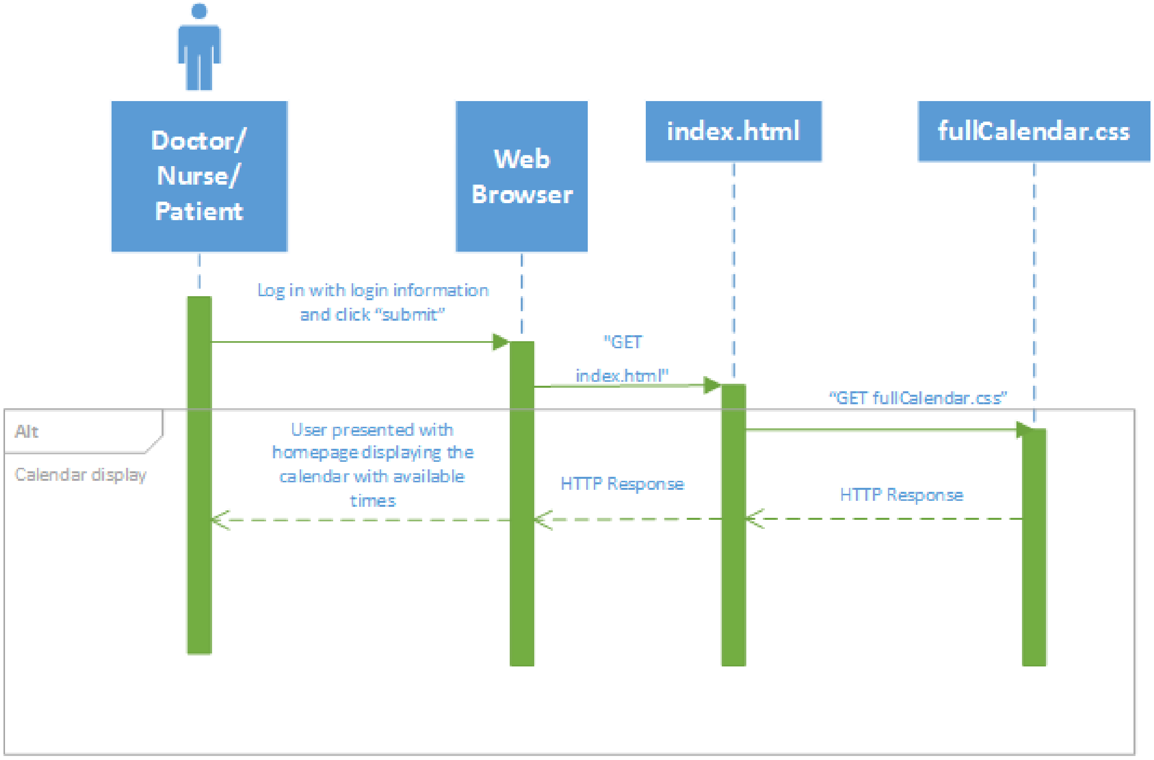
Use Case 21: “Patient Being Transferred with Reason”



Use Case 22: “Message Traffic Restriction”

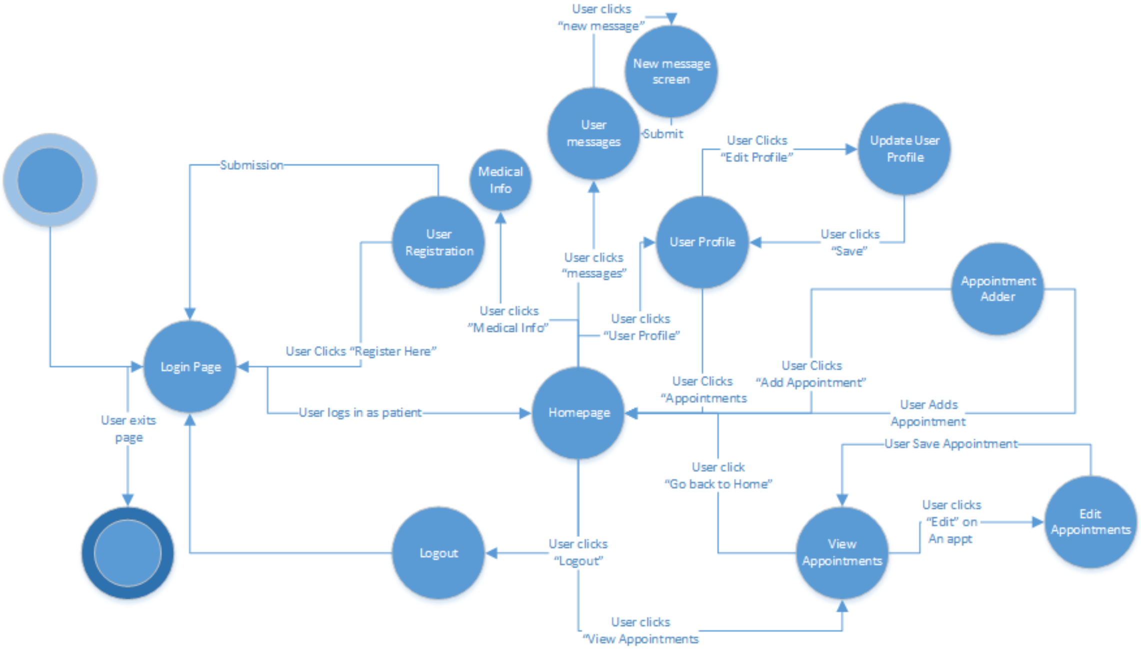


Use Case 23: “Available Times on Calendar”


Doctor/ Nurse/ Patient
Web Browser
index.html
fullCalendar.css
Log in with login information and click "submit"
"GET index.html"
"GET fullCalendar.css"
Alt
Calendar display
User presented with homepage displaying the calendar with available times
HTTP Response
HTTP Response

.3 State Diagram(s) - Screen (Web Page) Flow Diagram

Our Screen Flow diagram represents what we consider to be our “major” flow - A patient. Each node represents a point in which the patient transitions from one screen to another while navigating through healthnet.



4.0 Design Rationale

Our design has many goals, but at the forefront are to be simple and user friendly. Many of our design decisions were based off this fact, including:

- Layout: Each type of user has a custom made navigation bar for easy traversal through healthnet. This also helps to declutter everything, and make what the user sees only what they will need.
- While the customers specified many additional use cases, we made sure to put their original use cases at the fore front of our design. This is why things like user appointments, the calendar, etc. are at the fore front of our design.
- Insulation. We want to make sure that a patient only sees what the patient needs, and an admin only sees what an admin needs.

Internally, we had several design decisions made to complement this design. This includes:

- Use of separate django apps, as opposed to an abstract user. This adds to our insular design, helps reduce clutter, etc. We also hope this may provide for high cohesion and low coupling by having several discrete, testable applications that each work as their own seperate entity.

- Only functionalities that are almost identical among each application are shared using the “healthnet” application. These include messages, log in, and log out.
- While each navigation bar is unique depending on who is using it, we still made the system feel similar and intuitive enough that any experienced user of healthnet could easily pick up use of another user profile should the need arise.