**Log In**

The user starts the application and provides his or her username and password. The application takes the user credentials and opens a connection to the database. The application verifies the user’s credentials through the database and returns the available actions to the user.

**Access Patient Chart**

The user, who must be a Doctor, Nurse Practitioner, Nurse, Administrator or Patient, logs in to the application. The user then accesses the chart of the patient they wish to update.

The user will chose to either search for a patient or select the patient from a list. The application presents the appropriate interface to the user. Based on the user input, the application retrieves the list of matching patients (or all patients) from the database and displays this list to the user.

The user then clicks on the patient chart they wish to access. The application requests all the data for that patient from the database, and then displays the patient chart to the user.

**Edit Patient Chart**

The user, who must be a Doctor, Nurse Practitioner or Nurse, logs in to the application. The user then accesses the chart of the patient they wish to update.

The user then chooses the piece of data on the chart to edit. The application sets the field as editable and allows the user to modify it. The user provides the updated data to the application.

The application then locks the user record, and requests the current database information. If any field differs from the local copy of the patient chart, the application informs the user and updates the local copy. If no fields differ, then the application will push the updated data to the database.

The application will then request the current data from the database and verify that it is the same as the data in the local copy. The database lock is released. The data retrieved from the database is then compared to the local copy, and the user is informed of the result of the comparison.

**Order Lab**

The user, who must be a Doctor or Nurse Practitioner, logs in to the application. The user then accesses the chart of the patient they wish to update.

The user then selects the option to order a diagnostic lab for the patient. The application retrieves a list of possible diagnostic labs from the database and displays them to the user.

The user selects the lab to order. The application creates a new lab entry and provides it to the user. The user fills out the required data for the lab. The application takes the lab data and adds a lab entry to the database. The user is then informed of the result of the database add.

**View Lab**

The user, who must be a Doctor, Nurse Practitioner, or Nurse, logs in to the application. The user then accesses the chart of the patient they wish to update.

The user then selects the option to view the current diagnostic labs for the patient. The application retrieves the list of labs that apply to the patient from the database and displays that list to the user.

The user then selects the lab to view. The application retrieves the details for the selected lab from the database and displays those details to the user.

**Order Prescription**

The user, who must be a Doctor or Nurse Practitioner, logs in to the application. The user then accesses the chart of the patient they wish to update.

The user then selects the option to order a prescription for the patient. The application retrieves the list of medications from the database and displays that list to the user. The user chooses the medication and the application creates a new prescription entry for that medication.

The user then fills out the prescription entry. The application verifies that the prescription entry is correctly filled out for the specific medication request. If the prescription is not filled out correctly, the user is prompted to correct then entry. If the prescription is correctly filled out, the entry is added to the database.

The application then retrieves the prescription entry from the database and compares it to the local copy. The user is then informed of the result of the comparison.

**Contact Patient**

The user, who must be a Doctor, Nurse Practitioner, Nurse or Administrator, logs in. The user then accesses the chart of the patient they wish to update.

The user then selects the patient to contact. The application retrieves a list of contact information from the database and displays the list to the user. The user then selects a contact method. The application takes the contact information and passes it to the appropriate external application.

**Update Patient Vitals**

The user, who must be a Doctor, Nurse Practitioner or Nurse, logs in. The user then accesses the chart of the patient they wish to update.

The user then selects the option to add an entry to the patient’s vital sign history. The vital signs entry form comes up, and the user enters the patient’s current pulse, respiration rate, blood pressure and temperature.

The application will validate that the vital signs are within a valid range, and then ask the user for confirmation. If the user confirms the entry, it is added to the database. If the user does not confirm the entry, then the entry is discarded, and the user is returned to the vital sign entry form.

**Use Case:** *Update Patient Vitals***Actors:** UI, ECSInterface (ECS)  
**Initiating Actor:** UI  
**Events:**

* UI req. ser. *Create New ECS* from ECS
* ECS req. ser. Accept ECS Object from UI
* UI req. ser. *Log in* from ECS
* ECS req. ser. Accept Available Actions from UI
* UI req. ser. *Get Patient List*  from ECS
* ECS req. ser. Accept Patient List from UI
* UI req. ser. *Get Patient Chart* from ECS
* ECS req. ser. Accept Patient Chart from UI
* UI req. ser. *Get New Vitals Entry* from ECS
* ECS req. ser. Fill Out Vitals Entry from UI
* UI req. ser. *Commit Vitals Entry* from ECS
* ECS req. ser. Accept Commit Result from UI

**Use Case:** *Log In***Actors:** ECS, AppInterface (AppInt), DBInterface (DBInt)  
**Initiating Actor:** ECS  
**Events:**

* ECS req. ser. *Create AppInterface* from AppIntF
* AppIntF req. ser. *Store AppInterface* from ECS
* ECS req. ser. *Create DBInterface* from DBIntF
* DBIntF req. ser. *Store DBInterface*  from ECS
* ECS req. ser. *Verify Login*  from DBInt
* DBInt req. ser. *Get Available Actions*  from AppInt
* AppInt req. ser. *Return Available Actions* from ECS

**Use Case:** *Get Patient List***Actors:** ECS, AppInterface (AppInt), DBInterface (DBInt)  
**Initiating Actor:** ECS  
**Events:**

* ECS req. ser. *Retrieve All Patient Names* from AppInt
* AppInt req. ser. *Retrieve All Patients*  from DBInt
* DBInt req. ser. *Create Patient Name List* from AppInt
* AppInt req. ser. *Return Patient List* from ECS

**Use Case:** *Get Patient Chart***Actors:** ECS, AppInterface (AppInt), DBInterface (DBInt)  
**Initiating Actor:** ECS  
**Events:**

* ECS req. ser. *Retrieve Patient Chart* from AppInt
* AppInt req. ser. *Retrieve Patient Data* from DBInt
* DBInt req. ser. *Create Patient Chart*  from AppInt
* AppInt req. ser. *Return Patient Chart* from ECS

**Use Case:** *Commit Vitals Entry***Actors:** ECS, AppInterface (AppInt), DBInterface (DBInt)  
**Initiating Actor:** ECS  
**Events:**

* ECS req. ser. *Commit Vitals Entry*  from AppInt
* AppInt req. ser. *Commit Vitals Entry* from DBInt
* DBInt req. ser. *Accept Commit Result*  from AppInt
* AppInt req. ser. *Return Commit Result* from ECS