T808 Final Project Documentation

Aditya Bhargava
Jackson DeDobbelaere
Nishil Shah
Sneha Thakkar
Vivek Vermani
Arvind Vijayakumar
Yucheng Wang
Jayaram Yeluri

Table of Contents

Project Overview

Our goal for this project was to develop a suite of features for iTrust. For background, iTrust is an open-source system which provides a seamless experience for recording and tracking medical records. Specifically, this project consisted of developing an infrastructure for obstetrics - the branch of medicine concerning childbirth. Some of the outlined requirements include an interface for doctors to establish a new patient, keep track of office visits, schedule a child birth visit, and see a report encompassing the patient's pregnancy history. Implementing these features allows iTrust to be used for an entirely new kind of medical record: obstetrics.

Architecture & Design

Overview

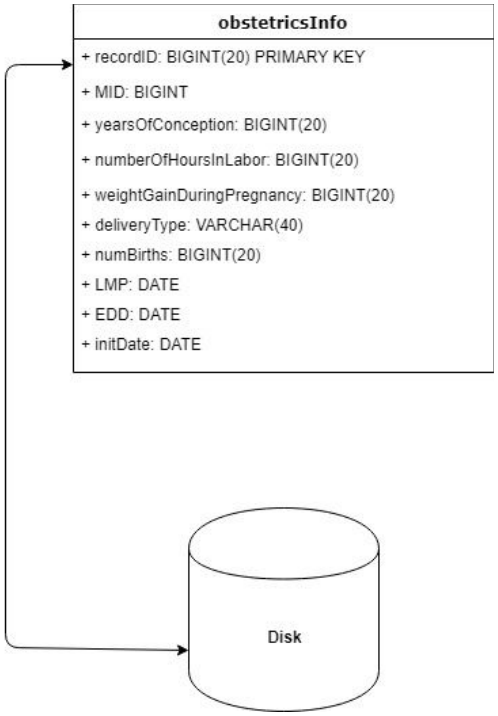
All our implementations of the five use cases are based on the current architecture and design decisions of the original iTrust code. The project follows a MVC (model-view-controller) architecture and uses a SQL database for storage. In general, models in iTrust are represented by `Bean` classes. `Loaders` and `DAOs` (data-access-objects) retrieve information from SQL tables to construct `Beans`. Controllers are represented by `Actions` and facilitate the communication between the views and the models. Lastly, each page that the user sees is handled by a view, which are found under the WebRoot directory. We tend to follow this same architecture while implementing each of our five UCs. Specific implementation regarding each use case is provided in the following sections.

UC-93 *Obstetrics Patient Initialization*

The main objective of UC-93 was to create a patient obstetric record interface after marking a patient eligible for obstetric care. The entry page for this process was `obstetricCare.jsp`, which was modeled after the existing `editPatient.jsp`. After selecting the patient, the OB/GYN can change eligibility status and add or edit obstetric records through `addNewObstetricRecord.jsp` and `viewIndividualObstetricsRecord.jsp` respectively. Information on this form includes year of conception, number of weeks pregnant, number of hours in labor, weight gain during pregnancy, delivery type, number of births, last menstrual period (LMP), and estimated due date (EDD). Of these, all fields are filled in by the HCP except EDD, which is calculated in `ObstetricInfoBean.java` with LMP information as input. Error checking was implemented on this form to ensure that all inputs were numerical values. Once the information is added, the HCP is redirected to the start page where records show up in descending order by initialization date.

Classes created for this use case include the following: `ObstetricInfoBean.java`, `ObstetricInfoLoader.java`, and `ViewObstetricInfoAction.java`, `ObstetricInfoDAO.java`.

Schema:



UC-94 *Obstetrics Office Visit*

The main object of UC-94 was to create functionality to document or edit an obstetrics office visit for a current obstetric patient. The core components are encapsulated in the various subflows:
(*from* the UC-94 specification)

1. The HCP enters a MID [E1] or name of a patient and confirms their selection [E2][E3].
2. An OB/GYN HCP [E4] documents the following information related to an obstetrics office visit and saves the obstetrics office visit record.
3. OB/GYN HCPs can return to an obstetrics office visit and modify or delete the fields of the obstetrics office visit.
4. If multiple obstetrics patient initializations fall within the 49 week window (e.g., two pregnancies within 49 weeks, so two initializations), the most recent obstetrics patient initialization is used.
5. If the patient's RH- flag (UC96) is set (the blood type is RH negative) and the current pregnancy term is past 28 weeks, a notice is displayed that the patient should be given an RH immune globulin shot if they have not already
6. During an ultrasound, an ultrasound record is created. In this record, the HCP can upload a .jpg, .pdf, or .png image of an ultrasound.
7. The next appointment (office visit or delivery visit) will be scheduled for the patient at the end of the appointment.

Schema:

The main schema changes are additional database tables to store obstetric office visits and ultrasound records, along with changes to appointment type to support these new events (Figure 3).

All additional data types are backed by beans, validators, and DAO (incl. loaders) following the existing "old" code model. In summary the newly created data types are:

- `ObstetricOfficeVisit`
- `UltrasoundRecord`

The modified data types are:

- `Patient`
    - Added an RH- flag to indicate if a patient has an RH- blood type
    - Added an RH Immunization flag to indicate if an obstetrics patient has been appropriately immunized
- `ApptType`
    - The existing data type was not inherently modified, but since appointment type is essentially a type enumeration stored in the DB, a new data entry was added to represent an "Obstetric Office Visit" appointment

The main functional additions correspond to the **document** and **edit** flows.

Document:

The entry point to the document flow is through the `addObstetricOfficeVisit.jsp` file. The key backend functionality for this process is provided by the *AddObstetricOfficeVisitAction* class. The scheduling a next appointment functionality is also found in *AddObstetricOfficeVisitAction* but depends on new code added to the `ApptDAO` (mainly, the `scheduleNextAvailableAppt` method).

Edit:

The entry point to the edit flow is through `viewDetailedObstetricOfficeVisit.jsp`. The key backend functionality for this process is provided by the `ViewObstetricOfficeVisitAction` class.

There are additional ultrasound functionalities (**upload, view)** as a part of the **document** and **edit** flows.

Upload:

The add ultrasound functionality (along with image upload) is handled by an *HttpServlet* called `UltrasoundServlet`. The servlet will create and insert an ultrasound record into the database as well as uploading the image to disk storage on the server.

View:

In view obstetric office visit, users are able to view ultrasound images by clicking on the "View Ultrasound" button, which downloads the image file to the local machine. The image download is handled by a separate `HttpServlet` called `DisplayImageServlet`, which uses URL pattern matching to redirect requests.

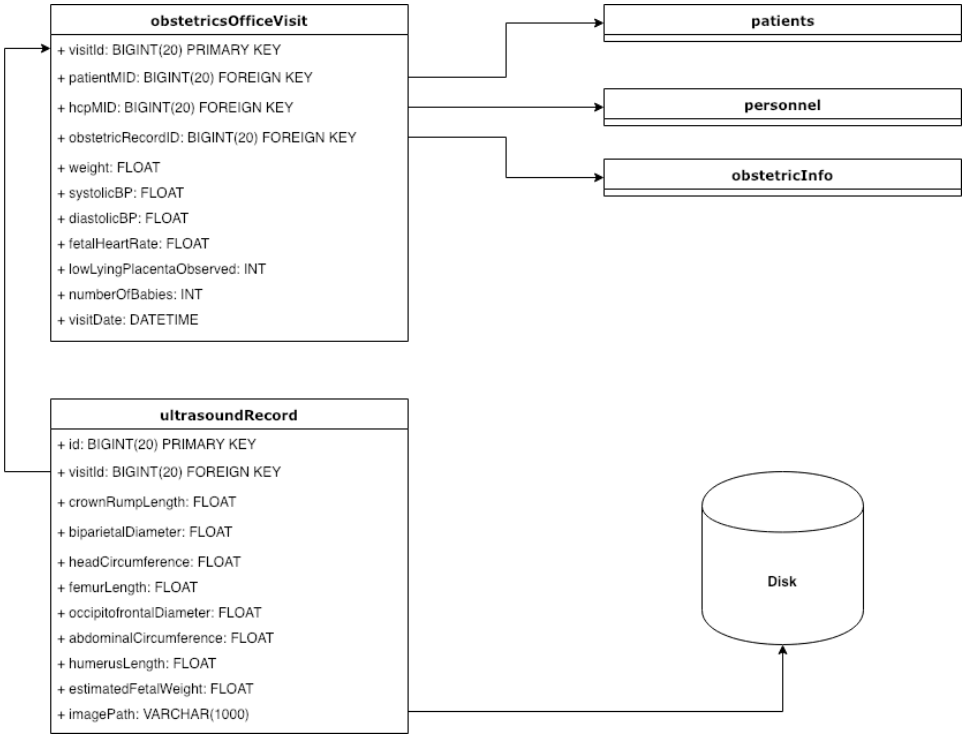


Figure 3: UC-94 Data schema

UC-95 *Labor & Delivery Report*

This Use Case involved formulating a report containing the relevant information about pregnancy and obstetric visit history for a certain patient. The HCP can access this report by selecting the Labor & Delivery report tab under the Patient Information section of the menu, and selecting a specific patient. If the patient is ineligible for obstetrics, the page shows an appropriate error: “This patient is not eligible for obstetric care”. In the case where the patient is eligible and has recorded any obstetric info, a report will be properly generated.

The design for UC-95 is rather simple. It required only two new files, a view and an action. The view is located in `/WebRoot/auth/hcp/viewLaborDeliveryReport.jsp`. The action is found at `src.main.edu.ncsu.csc.itrust.action.LaboryDeliveryReportAction`. Although the action doesn’t do any new logic, we created it to simplify work for the view. First, the view verifies that we have selected a specific patient before proceeding. It retrieves all information regarding prior pregnancies, obstetric office visits, and several other flags from the action. The action’s functionality is also simple. It talks to four DAOs (data access objects) to fetch the information it needs, and that’s it: `PatientDAO`, `ObstetricInfoDAO`, `ObstetricOfficeVisitDAO`, and `AllergyDAO`.

UC-96 *Childbirth Hospital Visit*

The purpose of this use case is to allow healthcare providers to schedule a new childbirth visit through an entry page `addNewChildBirthVisit.jsp` linked from the Obstetrics Care page created in UC-93 to store a record of information associated with the child birth. The main flow is connected to UC-94, when the patient is between 37 and 42 weeks pregnant, but the form can also be accessed in the case of an emergency. The form keeps track of whether the appointment was previously scheduled, the mother's preferred delivery type, if the baby was delivered, and a record of any drugs that had to be administered to the mother during childbirth - specifically pitocin, nitrous oxide, pethidine, epidural anaesthesia, magnesium sulfate, and RH immune globulin. Once a delivery has taken place, another form `addNewBabyDeliveryInfo.jsp` is displayed for each birth which tracks the birth time, gender, delivery type, and whether the birth time is an estimated value (in the case that the baby was born before the mother arrives at the hospital).

Schema:

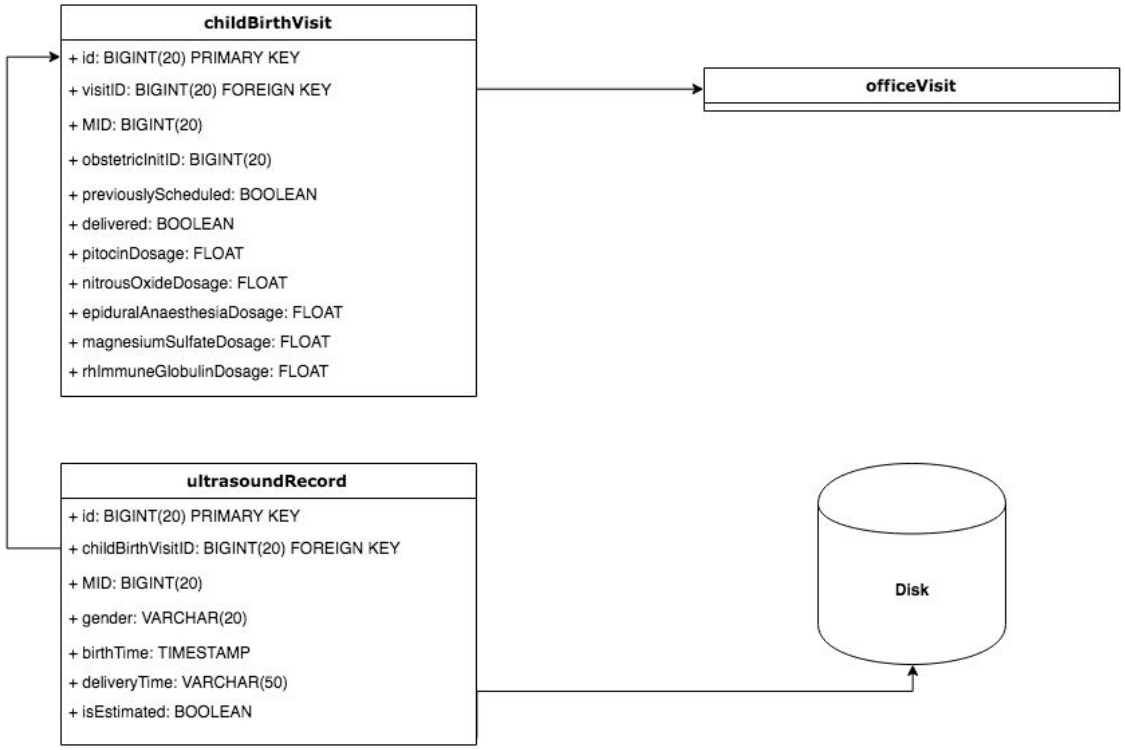


Figure 5: UC-96 Data schema

In implementing this use case, new tables were created in `createTables.sql` to store information for the childbirth visit and baby delivery. In order to read from and write to these tables, `ChildBirthVisitBean.java`, `BabyDeliveryInfoBean.java`, `ChildBirthVisitLoader.java`, `BabyDeliveryInfoLoader.java`, `ChildBirthVisitDAO.java` and `BabyDeliveryInfoDAO.java` were created.

`ChildBirthVisitValidator.java` was added to ensure the preferred delivery type would be set when filling out the form. Actions were made to connect the front end to the back end, modeled after UC-93 (`ViewChildBirthVisitAction.java`, `ChildBirthVisitAction.java`, `AddBabyDeliveryInfoAction.java`, `ViewBabyDeliveryInfoAction.java`). These actions allowed for retrieval of all records associated with a given patient and individual results by the characteristic record ID stored in the database table.

A list of all the childbirth visits and baby delivery information present for each patient can be seen in descending order by date on the `obstetricsCare.jsp` page, and OB/GYN HCPs can edit both the childbirth visits and associated baby delivery information with the `viewIndividualChildBirthVisit.jsp` and `viewIndividualBabyDeliveryInfo.jsp` pages. Another element of this use case is that for every baby delivery documented on `addNewBabyDeliveryInfo.jsp`, a new patient was added to the database. The information stored about the baby was the gender and the Mother's MID, and once the form was submitted, the baby's MID was displayed on the page.

UC-41 *Send Reminders* [Custom Use Case]

The main purpose for this use case is to allow the administrators to send reminders to patients. In `sendApptRem.jsp`, the administrator input a number and click “Send Appointment Reminders”. Then the system will send reminders to patients who have upcoming appointments in next n days. The administrator can view all the reminders sent in the message outbox. Patients can view reminders in their message inbox. Users can view details of messages.

In implementing this use case, we create new file `SendRemindersAction.java` add new methods to `ApptDAO.java`. We also create `messageOutbox.jsp` and `viewMessageOutbox.jsp` in `iTrust/WebRoot/auth/admin/` to allow the administrator to see the sent reminders.

Appendix

Testing

We provide unit tests for all new DAOs we added, including tests for `ObstetricInfoDAO.java` (UC-93), `ObstetricOfficeVisitDAO.java` (UC-94), `UltrasoundRecordDAO.java` (UC-94), `ChildBirthVisitDAO.java` (UC-96), and `BabyDeliveryInfoDAO.java` (UC-96). Each of the unit tests can be found in the corresponding directory in the `test\itrust\unit\dao` directory. The tests were written to achieve 80% line coverage for the DAOs. We also provide test for new method for our custom test case in `ApptDAOTest.java`.

UI testing was also performed through Selenium Tests to ensure functionality of UC-41, UC-93, UC-94, UC-95, and UC-96. Each of these consisted of a success case and failure case.