Software Engineering
Xinyuan Yang(Chanel)
Qiaowei Li
Yunhao Ye(Edmund)
Shuhan Yuan（Hannah）
Professor Lihua Xu

eDoctor System Final Documentation

Contents

Brief Description-- 1-1
Iteration Process-- 2-3
Specification: User cases-- 3-5
Architecture & Design-- 5-6
Reflections-- 7-10

I. Brief description

Our project is basically an EHR system which is named as E-doctor system. According to the definition of EHR system, EHR system is viewed as a digital vision of health records, and its most obvious characteristic is real-time and patient-oriented. Not only does the EHR system help offer detailed records of a patient’s medical inquiry history, but it also offers a broader overview of the patient so as to help doctors and even patients himself get command of a patient's personal condition faster and much fully. Based on this definition, our group initially separated the system into two major parts: real-time data update and patient’s past medical record history. In terms of real-time data, we conducted two separate pages for doctors to write in-process diagnosis and correct detailed information for the patients, including weight, height, allergies and etc. In terms of past medical record history, we used another page within connection with the back-end database, and the medical records can be searched by keywords which adds large convenience to medical records.

II. Iteration Process

Our team is composed of four team members: Edmund(Yunhao) Ye, Qiaowei Li, Chanel(Xinyuan) Yang and Hannah(Shuhan) Yuan. The process development structure that was mainly applying is the XP programming and the team conducted weekly meetings for in-group discussion which ensures collaborative development. Due to the fact that none of the group members have past experience on software construction and development, so in the first few weeks all the members were all conducting self-learning and present weekly reflection during the meeting in order to try the best to get command of all the basic knowledge. Starting from the second month, the whole group started to divide the software construction into three parts including front-end, back-end and middle connections. In the process of the work, structure workers dynamically designed and adjusted user stories and user cases and conducted feature code designing, development and tested in repeated cycles (once two weeks) in order to make the development iterative. In the final few weeks, after ensuring the basic software model functioned well, the whole team devoted themselves to further-testing and refactoring part, in order to make the product easy for understanding and optimize the code structure.

Timeline:

- September
    - Contents: Self-learning and weekly reflection
- October & November
    - Contents: Starting from this month, every group member has picked their own part of work and started to construct the basic structure of the work.
    - Division of the task:
        - Edmund: Front-end worker
            - HTML & Layui
        - Chanel & Qiaowei: structure worker (Pair programming)
            - mySQL->mongodb & Robot3t
            - Part of basic front-end, Layui
            - Js,py files conduction
            - Iteration recorder
            - Testing (random generation part)

        - ■ Hannah: Process-check person
            - Profile/ search function co-editor
            - Check whether we have reached our weekly goals
- December
    - Contents: Testing, refactoring and final software furnishment.

III. Specifications: User Cases

A.Use Case Diagram:

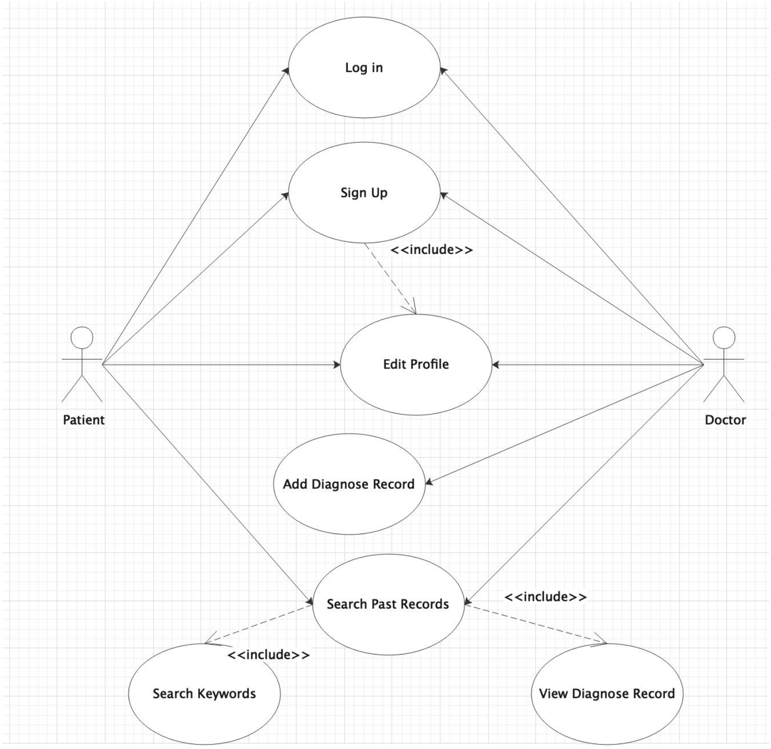


Figure 1. Use Case Diagram

B.Fully Dressed Use Case:

- 1.Edit Profile
    - Preconditions:
        - 1. The user has already logged in.
        - 2. The user has created his/her personal basic information before.
    - Main Flow:
        - After the patient logs into the EHR system, he/she can open up his/her personal profile and see that some personal information is modifiable, like height and weight[S1]. When the patient got the corresponding new values from the family scale or the annual

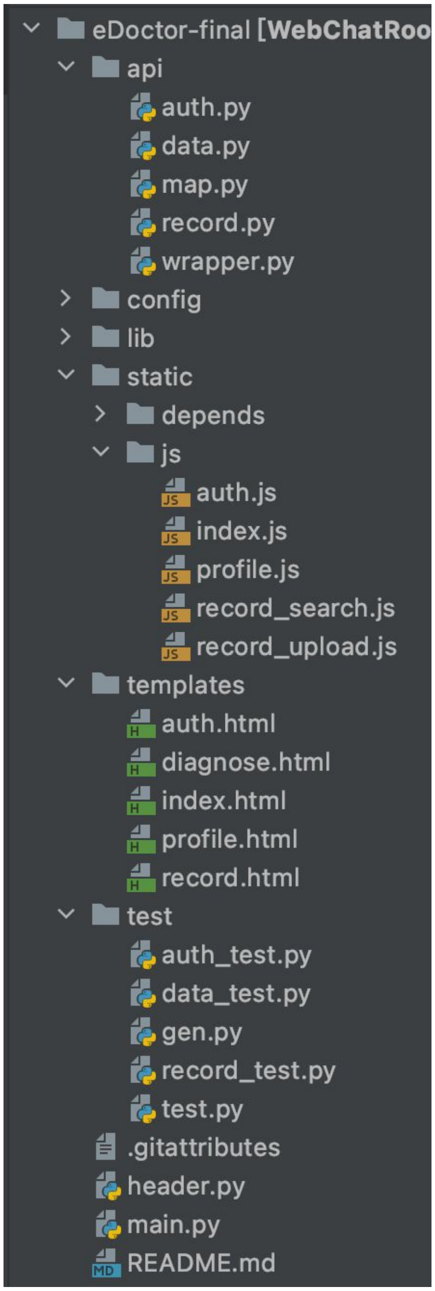
health screen, he/she can update these values into his/her personal profile[E1]. After editing the personal information, the patient can save the latest version to the system[S2].

- Subflows:
    - [S1] When the patient opens up his/her personal information, he/she can see the previous information in the blocks, and they can modify any information he/she wants and keep those left the same.
    - [S2] There will be two buttons at the end of the page. The “submit” button is used to save the profile, the user can click it to save his/her modification, and all the information needs to be filled before submit. The “reset” button is used to clear all the information blocks, the user can click it and fill them again.
- Alternative Flows:
    - [E1] When the patient writes down a wrong type of information to his/her personal profile, like adding some letter into the height information, the system will automatically show a red cross sign as well as words to tell the patients. Also the wrong type information can not be saved.

- Search Past Records
    - Preconditions:
        - 1. The user has already logged in.
        - 2.The user has previous diagnosis records.
    - Main Flow:
        - After the user logs into the system, he/she can search for its previous diagnosis records. The user can type in some keyword to narrow down the searching scales [S1][E1]. And then the search results would be displayed below, then the user can view the details of each of them [S2].
    - Subflows:
        - [S1] The user can choose the disease category and types in the disease name to narrow down the searching scale. And after the user clicks the “search” button, the results will be shown under the searching boxes. Only the diagnose records in which the user participates as patient or doctor will be displayed.
        - [S2] After the searching results are displayed, the user can click the icon of each record to view the details of it. Like the name of patients and doctors, and the symptoms and the medicine.
    - Alternative Flows:

- [E1] When the user doesn’t have any diagnose records or there is no result based on the current keywords, there will be a popup to tell the user there is no result.

IV. Architecture & Design



```
eDoctor-final [WebChatRoo
  api
    auth.py
    data.py
    map.py
    record.py
    wrapper.py
  config
  lib
  static
    depends
    js
      auth.js
      index.js
      profile.js
      record_search.js
      record_upload.js
  templates
    auth.html
    diagnose.html
    index.html
    profile.html
    record.html
  test
    auth_test.py
    data_test.py
    gen.py
    record_test.py
    test.py
  .gitattributes
  header.py
  main.py
  README.md
```

Figure 2. Project Architecture Overview

The basic architecture of our program is shown in the screenshot on the left. Our program follows the Flask framework, and is structured as follows:python files in the api folder for functions related to search/update/insertion in the database and mapping of api; html files in the template folder that will be rendered as the frontend user interface of the system; javascript files in the static/js folder that will interact with the frontend html and backend database, do some input validation work and transfer data as json packages; extra python files in test folder that will auto generate batch of account registration and conduct auto unittests for our project. We also have header.py for database(mongodb) connection and main.py for running the entire project.

In view of the usability, the framework we are using is basically managing different functions into suitable pages so that when the users open the software, they can clearly know on which pages what kind of functions they can realize. According to our structure, we have five different html pages: the auth page is mainly used for log-in/out and register page, while at the same time offering places for basic personal information to be written and delivered in to database; the index page is mainly used for presenting basic personal information and several latest medical records; the profile page is mainly used for editing and updating detailed vision of personal information; the diagnosis page functions as the in-process diagnosis write-in and presentation; and the record page is used for users to conduct medical records search, whereas they can view a detailed table of their medical history.

Class Diagram:

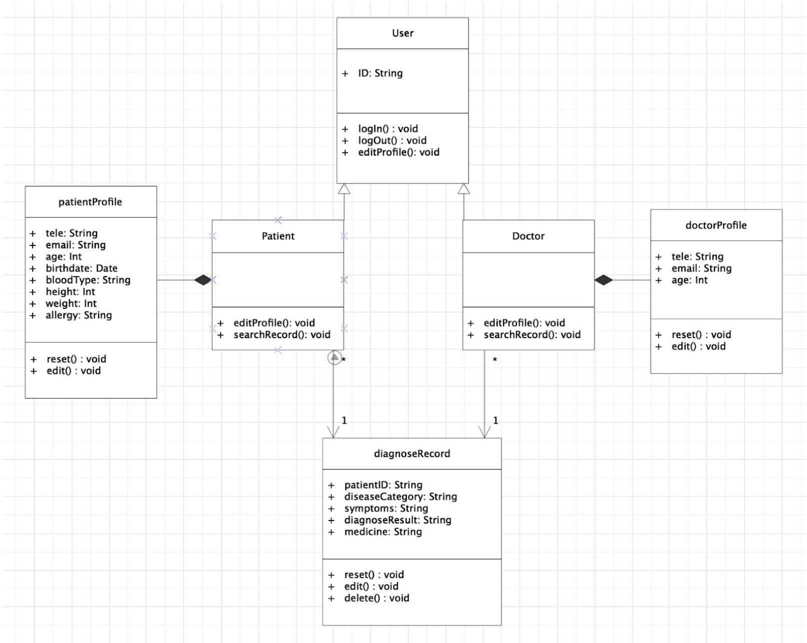


Figure 3. Class Diagram

Sequential Diagram:

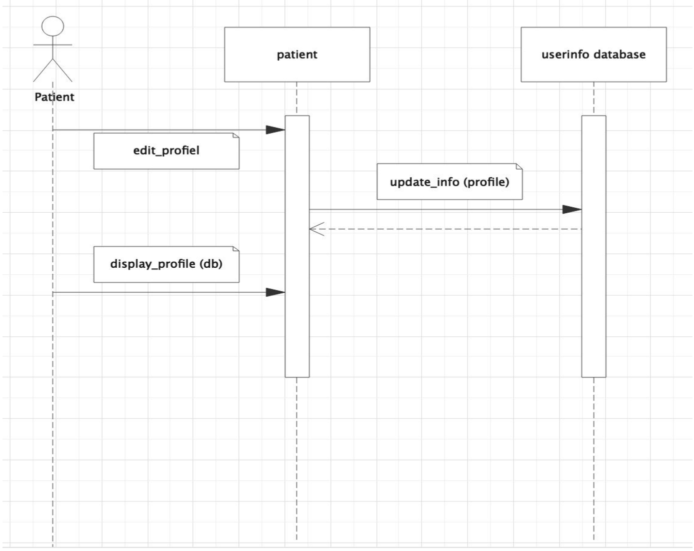


Figure 4. Sequential Diagram

V. Reflections

First thing first, everyone please stay healthy! When designing the database, we found out that there are many components that can affect a person’s health, and the process of diagnosis is usually tiring and time-costing, thus although we design the software as an eHR system, we hope people can more use it to self-monitor their health instead of diagnosis.

Chanel Yang:

This semester's software engineering program was an incredible experience. It deepened my knowledge in the field of computer, especially software engineering, and I also found out that usually when we gently choose a software on the screen, maybe order a take-away, or maybe read an e-book, behind are countless user cases of negation and re-creation, countless lines of code with countless refactoring, and they are all collaboration of countless software engineers' wisdom. Through the development, execution and production of a semester-long project for it, until the birth of the final product, I saw how the group's joint efforts eventually turned into a visual outcome. Two major lessons I learned from this project are always being prepared and teamwork. Before construction, our group spent enough time on user story designing and user case construction. In the preparation stage of our software project, we explored deeper into our software structure which granted us a not only a global view of our final software work but aware us of detailed information that we need to pay attention during the process. A clear logic is a must for software development so that we can always keep in mind what we need to do and what we should avoid. Besides, we formed the habit of always making copies of current visions of software engineering. I have to say this helped a lot, if we accidentally typed something wrong, we didn’t have to repeat the past work all over again while instead having an opportunity to just restart from the beginning of the mistake. Teamwork is another key concept that we benefited a lot, so that cooperation helped us to accelerate the speed of software development and refactoring, as we can always remind each other of the details that they ignored. In conclusion, I was really grateful to the experience of this project and very appreciate the help from my team members especially my pair programming co-worker Qiaowei, and teaching assistants and professors.

Qiaowei Li:

One of the most meaningful things I learned in this semester's project was to learn to dynamically adjust the original plan. For example, at the beginning, we planned to used SQL type in the database design based on the recommendation from seniors, but in the group discussion we found that most of the group members were unfamiliar with this type of database design and development, which would hinder the further development of the project, so after the suggestion of the teaching assistant, we switched to the file-oriented Mongodb as our database, so that we could build the software model as soon as possible. Mongodb as an nosql database is easier for us to build and use, the syntax and structure are more friendly to us beginners. Besides, this whole semester we were conducting teamwork. Teamwork helps a lot and is a must in a large project; I've already felt the importance of teamwork in this project.

The process of making such a web system ourselves from the very beginning is a very precious experience. We start from planning, designing, and then implementing. Having a good overview of the entire project is very important, since it could be a guidance for future works. However, it seems that there should be an extra step in the earlier stage: to set priority of all the tasks in the project. For the whole project we have different steps and functions, for instance the record search function and the web chat room are two of the main functions we planned for our electronic health system. As we are beginners for software development, the implementation of both of the functions need to be studied and takes a long time, so when we have a time limit we should set priority of the tasks and choose one to concentrate on. In brief, too much learnt, both practically and theoretically.

Shuhan Yuan:

The most exciting thing during this project experience for me is always exploring and learning new things. During our project development process, we learnt a lot of useful things from the presentation level to the business logic level and the database level. As in this project, I took the main responsibility of the testing part, I realize that designing the test beforehand and afterwards have the same importance. Although in the first few weeks, we are still in the learning stage, we already have a rough idea about some verification conditions in our system. For example, at the first meeting, we decided to divide our users into three groups: patient, doctor and the third party. So in our system, we add an extra information to indicate different identities of users, and this is our very primary designing idea about "testing". And with the development of our project, more and more tests were designed, the most typical part is for the user's registration authentication. We design tests for various aspects including email account testing, password testing, phone number verification and others, and based on these new designs, we better improved our edoctor system. Actually, testing is a work which needs teamwork very much. It needs all team members to be consistent about the testing design and then implement the system in accordance with these tests. The related implementation span from the database to the front-end presentation. So, from the perspective of testing, each iteration is a process for us to design, test, modify according to the test result, retest and redesign. Thanks to this great project experience, it helps me have an awareness of how testing is important for a project development which I never heard before.

Yunhao Ye:

The experience during this semester has taught me a lot about software engineering. I now know that software engineering is much more complicated than a coding cooperate project. Based on my experience, I think always having a clear plan on the project and keeping in close touch with teammates are very important during the whole process.

At the beginning of this semester, we have designed use case diagrams and UML diagrams for our project. And these diagrams have helped a lot both building the software structures and implementing different functions. Also, at each iteration meeting, we would update our journal together and make it clear on what we want to implement in the next iteration. Making what we have done and what we should do clear can potentially save a lot of time for us.

Another benefit of iteration meeting is that it binds us together, thus we can communicate with each other frequently, of which I think is quite significant. Firstly, by explaining our own codes to others, the frontend and backend can have a clear understanding of each other's work. Secondly, the comments and suggestions raised can lead our project towards a better direction. Lastly, teams can usually make better decisions than individuals based on the current situation.

I am quite new to being a software engineer, so I have encountered many problems and also learnt a lot at the same time. I have known some mainstream frontend frameworks along with their functions. And I chose LayUI to help me write the html, with JS to connect with the backend. Also, I have known the structure of the backend coding, since I need to read teammates' codes to make the software work and pay attention to the details of every request to avoid any bug may occur. In general, I have learnt a lot through this semester, both on programming and on software management, and there is still a long way to go to be an experienced engineer.