**Patient Information System**

Team name: ***Asteroid***

陈双(Richard Chen) No.6130310104 Tel: 15754601458

苏学斌(Xuebin Su) No.1133710519 Tel: 15754600771

万向阳(Frank Wan) No.6120310103 Tel: 18646086303

**Section 5.0 Database Application Implementation**

**Section 5.1 Data Definition Language (DDL)**

SET @MYSQLDUMP\_TEMP\_LOG\_BIN = @@SESSION.SQL\_LOG\_BIN;

SET @@SESSION.SQL\_LOG\_BIN= 0;

SET @@GLOBAL.GTID\_PURGED='4d14a7b2-19cc-11e6-bb14-ec388f73a168:1-394';

DROP TABLE IF EXISTS `epide\_analysis`;

SET @saved\_cs\_client = @@character\_set\_client;

SET character\_set\_client = utf8;

SET character\_set\_client = @saved\_cs\_client;

--

-- Table structure for table `specialty`

--

DROP TABLE IF EXISTS `specialty`;

CREATE TABLE `specialty` (

`Specname` varchar(30) NOT NULL,

`pnum` int(11) DEFAULT NULL,

`dnum` int(11) unsigned NOT NULL DEFAULT '0',

`nnum` int(11) NOT NULL DEFAULT '0',

PRIMARY KEY (`Specname`),

UNIQUE KEY `Specname\_UNIQUE` (`Specname`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--

-- Table structure for table `tb\_diagnosis`

--

DROP TABLE IF EXISTS `tb\_diagnosis`;

CREATE TABLE `tb\_diagnosis` (

`ID` int(11) NOT NULL AUTO\_INCREMENT,

`Test` tinytext,

`DiseaseCate` tinytext,

`Suggestion` tinytext,

`MedicalHis` tinytext,

`recorded\_on` datetime NOT NULL DEFAULT CURRENT\_TIMESTAMP,

`Pid` int(11) DEFAULT NULL,

`Did` int(11) DEFAULT NULL,

PRIMARY KEY (`ID`),

UNIQUE KEY `ID\_UNIQUE` (`ID`),

KEY `foreign2\_idx` (`Did`),

KEY `foreign1\_idx` (`Pid`),

CONSTRAINT `foreign1` FOREIGN KEY (`Pid`) REFERENCES `tb\_patients` (`PID`) ON DELETE NO ACTION ON UPDATE NO ACTION,

CONSTRAINT `foreign2` FOREIGN KEY (`Did`) REFERENCES `tb\_doctors` (`Did`) ON DELETE NO ACTION ON UPDATE NO ACTION

) ENGINE=InnoDB AUTO\_INCREMENT=2 DEFAULT CHARSET=utf8;

--

-- Table structure for table `tb\_doctors`

--

DROP TABLE IF EXISTS `tb\_doctors`;

CREATE TABLE `tb\_doctors` (

`Did` int(11) NOT NULL,

`name` varchar(45) DEFAULT NULL,

`title` int(11) DEFAULT NULL,

`specname` varchar(30) DEFAULT NULL,

`email` varchar(30) NOT NULL,

PRIMARY KEY (`Did`),

UNIQUE KEY `Did\_UNIQUE` (`Did`),

UNIQUE KEY `email\_UNIQUE` (`email`),

KEY `specname\_idx` (`specname`),

CONSTRAINT `specname` FOREIGN KEY (`specname`) REFERENCES `specialty` (`Specname`) ON DELETE NO ACTION ON UPDATE NO ACTION

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--

-- Table structure for table `tb\_nurses`

--

DROP TABLE IF EXISTS `tb\_nurses`;

CREATE TABLE `tb\_nurses` (

`Nid` int(11) NOT NULL,

`name` varchar(45) DEFAULT NULL,

`pos` varchar(45) DEFAULT NULL,

`gender` tinyint(4) DEFAULT NULL,

`specname` varchar(30) DEFAULT NULL,

`email` varchar(30) NOT NULL,

PRIMARY KEY (`Nid`),

UNIQUE KEY `Nid\_UNIQUE` (`Nid`),

UNIQUE KEY `email\_UNIQUE` (`email`),

KEY `specname\_idx` (`specname`),

CONSTRAINT `foreignkey1` FOREIGN KEY (`specname`) REFERENCES `specialty` (`Specname`) ON DELETE NO ACTION ON UPDATE NO ACTION

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

--

-- Table structure for table `tb\_nursingrecord`

--

DROP TABLE IF EXISTS `tb\_nursingrecord`;

CREATE TABLE `tb\_nursingrecord` (

`ID` int(11) NOT NULL AUTO\_INCREMENT,

`evaluation` tinytext,

`State` tinytext,

`Nid` int(11) DEFAULT NULL,

`Pid` int(11) DEFAULT NULL,

`date` date DEFAULT NULL,

PRIMARY KEY (`ID`),

UNIQUE KEY `ID\_UNIQUE` (`ID`),

KEY `Nid\_idx` (`Nid`),

KEY `nursing\_foreign2\_idx` (`Pid`),

CONSTRAINT `nursing\_foreign1` FOREIGN KEY (`Nid`) REFERENCES `tb\_nurses` (`Nid`) ON DELETE NO ACTION ON UPDATE NO ACTION,

CONSTRAINT `nursing\_foreign2` FOREIGN KEY (`Pid`) REFERENCES `tb\_patients` (`PID`) ON DELETE NO ACTION ON UPDATE NO ACTION

) ENGINE=InnoDB AUTO\_INCREMENT=10 DEFAULT CHARSET=utf8;

--

-- Table structure for table `tb\_patients`

--

DROP TABLE IF EXISTS `tb\_patients`;

CREATE TABLE `tb\_patients` (

`PID` int(11) NOT NULL AUTO\_INCREMENT,

`Married` tinyint(1) DEFAULT NULL,

`age` int(11) DEFAULT NULL,

`name` varchar(45) DEFAULT NULL,

`outdate` date DEFAULT NULL,

`indate` date DEFAULT NULL,

`addr` varchar(45) DEFAULT NULL,

`gender` tinyint(4) DEFAULT NULL,

`Did` int(11) DEFAULT NULL,

`Nid` int(11) DEFAULT NULL,

`email` varchar(30) NOT NULL,

`is\_recovered` tinyint(1) DEFAULT '0',

`password` varchar(30) DEFAULT NULL,

`specname` varchar(30) DEFAULT NULL,

PRIMARY KEY (`PID`),

UNIQUE KEY `PID\_UNIQUE` (`PID`),

UNIQUE KEY `email\_UNIQUE` (`email`),

KEY `Did\_idx` (`Did`),

KEY `Nid\_idx` (`Nid`),

KEY `foreign1\_pa\_idx` (`specname`),

CONSTRAINT `Did` FOREIGN KEY (`Did`) REFERENCES `tb\_doctors` (`Did`) ON DELETE NO ACTION ON UPDATE NO ACTION,

CONSTRAINT `Nid` FOREIGN KEY (`Nid`) REFERENCES `tb\_nurses` (`Nid`) ON DELETE NO ACTION ON UPDATE NO ACTION,

CONSTRAINT `foreign1\_pa` FOREIGN KEY (`specname`) REFERENCES `specialty` (`Specname`) ON DELETE NO ACTION ON UPDATE NO ACTION

) ENGINE=InnoDB AUTO\_INCREMENT=129 DEFAULT CHARSET=utf8;

--

-- Table structure for table `tb\_prescription`

--

DROP TABLE IF EXISTS `tb\_prescription`;

CREATE TABLE `tb\_prescription` (

`Psid` int(11) NOT NULL AUTO\_INCREMENT,

`Tid` int(11) DEFAULT NULL,

`recorded\_on` datetime NOT NULL DEFAULT CURRENT\_TIMESTAMP,

`drug` tinytext,

`dosage` tinytext,

PRIMARY KEY (`Psid`),

UNIQUE KEY `PrescripID\_UNIQUE` (`Psid`),

KEY `pre\_foreign1\_idx` (`Tid`),

CONSTRAINT `pre\_foreign1` FOREIGN KEY (`Tid`) REFERENCES `tb\_treatment` (`id`) ON DELETE NO ACTION ON UPDATE NO ACTION

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

-- Table structure for table `tb\_symptom`

--

DROP TABLE IF EXISTS `tb\_symptom`;

CREATE TABLE `tb\_symptom` (

`ID` int(11) NOT NULL AUTO\_INCREMENT,

`symptom` tinytext,

`complain` tinytext,

`PID` int(11) DEFAULT NULL,

`DID` int(11) DEFAULT NULL,

`recorded\_on` datetime NOT NULL DEFAULT CURRENT\_TIMESTAMP,

PRIMARY KEY (`ID`),

UNIQUE KEY `ID\_UNIQUE` (`ID`),

KEY `sym\_foreign2\_idx` (`DID`),

KEY `sym\_foreign1\_idx` (`PID`),

CONSTRAINT `sym\_foreign1` FOREIGN KEY (`PID`) REFERENCES `tb\_patients` (`PID`) ON DELETE NO ACTION ON UPDATE NO ACTION,

CONSTRAINT `sym\_foreign2` FOREIGN KEY (`DID`) REFERENCES `tb\_doctors` (`Did`) ON DELETE NO ACTION ON UPDATE NO ACTION

) ENGINE=InnoDB AUTO\_INCREMENT=3 DEFAULT CHARSET=utf8;

--

-- Table structure for table `tb\_treatment`

--

DROP TABLE IF EXISTS `tb\_treatment`;

CREATE TABLE `tb\_treatment` (

`id` int(11) NOT NULL AUTO\_INCREMENT,

`InHospital` tinyint(1) DEFAULT NULL,

`IsOperation` tinyint(1) DEFAULT NULL,

`Remarks` tinytext,

`patient\_id` int(11) DEFAULT NULL,

`doctor\_id` int(11) DEFAULT NULL,

`recorded\_on` datetime NOT NULL DEFAULT CURRENT\_TIMESTAMP,

PRIMARY KEY (`id`),

UNIQUE KEY `TID\_UNIQUE` (`id`),

KEY `Pid\_idx` (`patient\_id`),

KEY `Foreignkey1\_idx` (`doctor\_id`),

CONSTRAINT `foreign1\_treatment` FOREIGN KEY (`doctor\_id`) REFERENCES `tb\_doctors` (`Did`) ON DELETE NO ACTION ON UPDATE NO ACTION,

CONSTRAINT `foreign2\_treatment` FOREIGN KEY (`patient\_id`) REFERENCES `tb\_patients` (`PID`) ON DELETE NO ACTION ON UPDATE NO ACTION

) ENGINE=InnoDB AUTO\_INCREMENT=2 DEFAULT CHARSET=utf8;

--

-- Final view structure for view `epide\_analysis`

--

SET @@SESSION.SQL\_LOG\_BIN = @MYSQLDUMP\_TEMP\_LOG\_BIN;

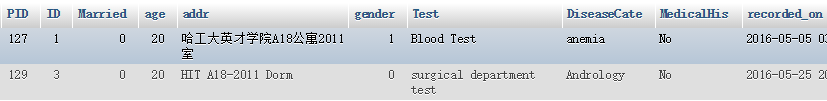
**Section 5.2 Triggers**

In our project, we utilize trigger to complete the function of automatically updating the number of people in different speciality departments. In detail, there are six triggers in our project triggered by the modifications from three tables (*tb\_doctors, tb\_patients* and *tb\_nurses*). Once triggered, the *speciality* table which contains the number of doctors, patients and nurses of each *speciality* is updated automatically to ensure the data is correct. Also, we try to keep the integrity constraints between different tables. Hopefully, we find that mysql server can do it for us if we enable the foreign key constraint check function.

**Section 5.3 Views**

In our project, we utilize views to provide the outbound service considering the security issues. Since the logical model of our database should be hidden from some kind of users. Meanwhile they may not concern about our real logical model. So we provide the database views to users as outbound service.

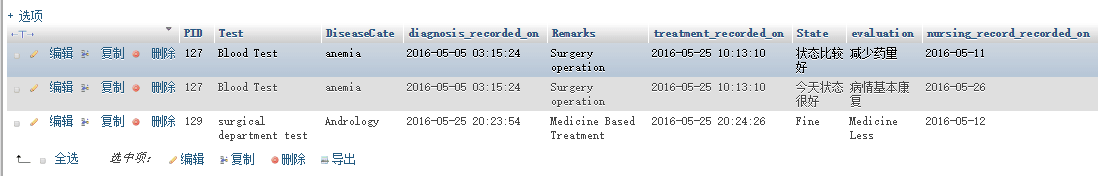
In detail, we create three views: *epide\_analysis* and *clinical\_process.*

The *epide\_analysis* can provide the information about certain patient with certain kind of disease should take some kinds of tests.

And Here is the *clinical\_process* view, which combines the whole workflow of one patient’s examination in clinic into a set of information including symptoms, diagnosis, tests, and treatments, which reduces the effort for creating patient’s health record history.



Finally, the *treatment\_eval* view, created to providing APIs for researchers to evaluating a specific treatment method, which might be helpful for clinical research.



**Section 5.4 Indexes**

Indexes are added for frequently queried tables’ columns. By default, the primary key are set in indexes. Moreover, we set the email column in *tb\_patients*, *tb\_doctors*, *tb\_nurses* indexes since they are frequently used. Considering frequently updated attributes may get complex and slow if indexes are added, indexes are not added to the following tables:

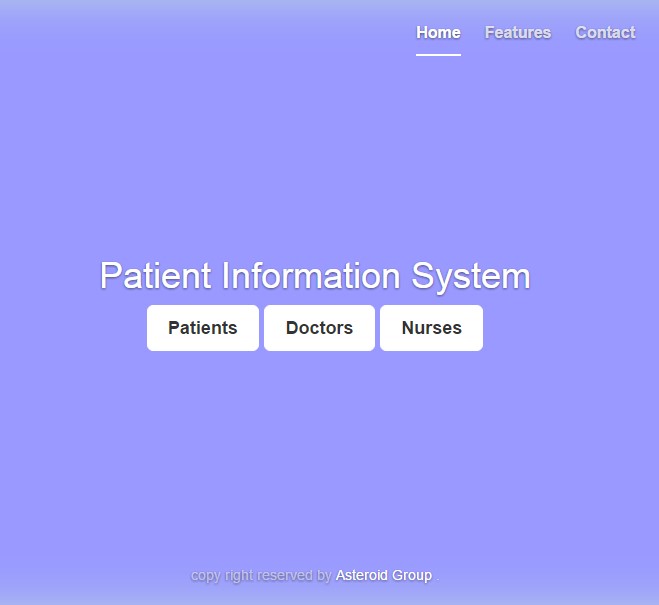
*tb\_diagnosis, tb\_nursingrecord, tb\_prescription, tb\_symptom, tb\_treatment*

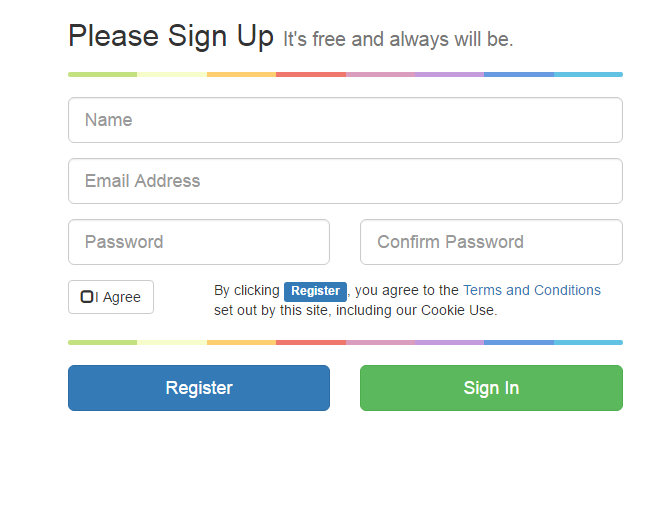
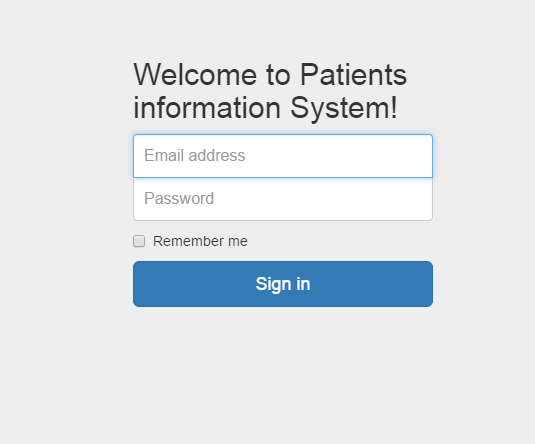
**Section 5.5 User Interfaces (Web frontend)**

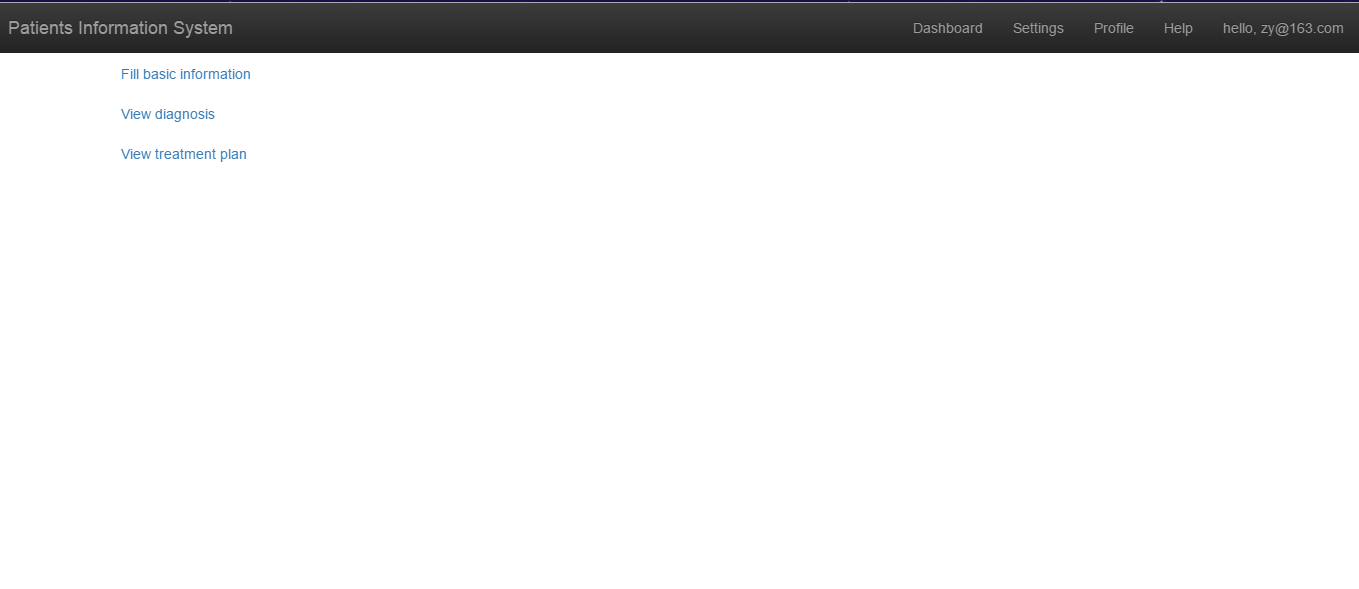
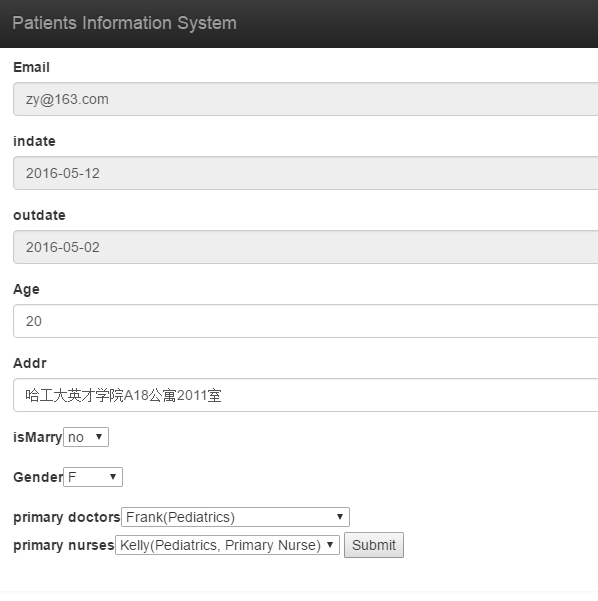
**Section 5.5.1 GitHub**

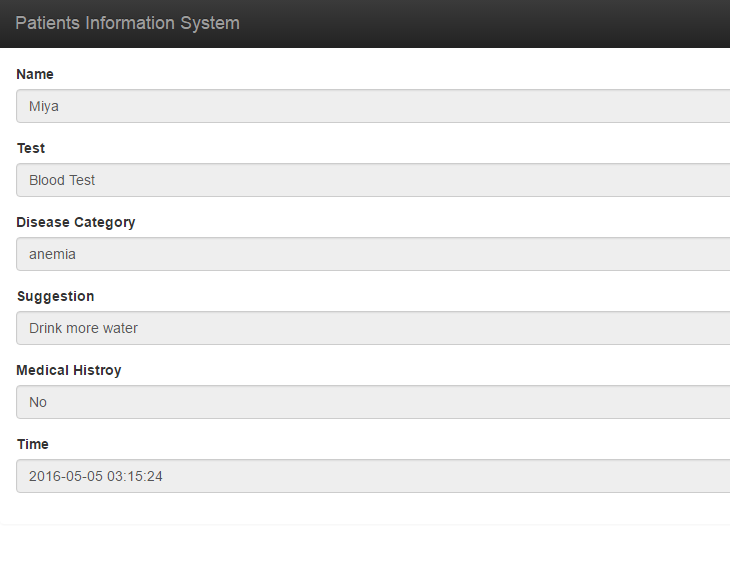
<https://github.com/CHENShuang1994/Patient-Information-System>

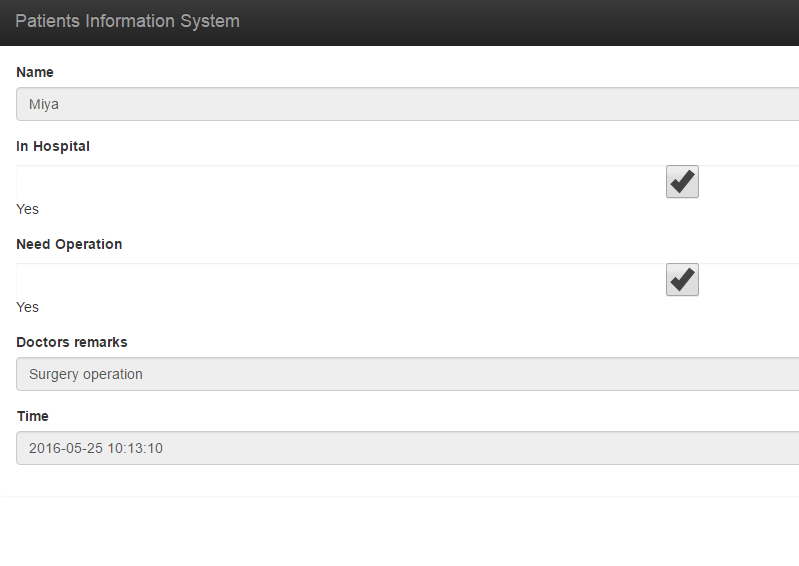
**Section 5.5.2 Web frontend**

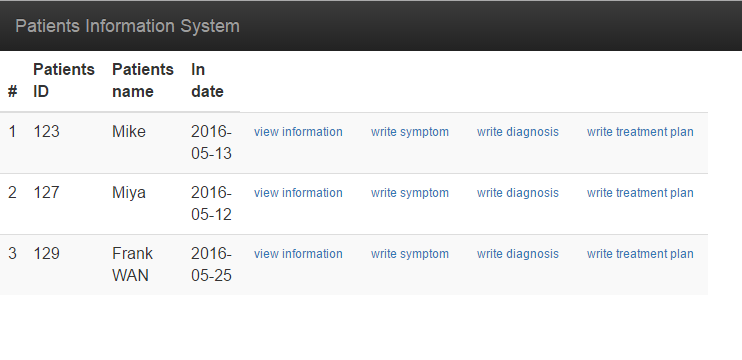
**Remarks:** Details will be shown in the presentation.

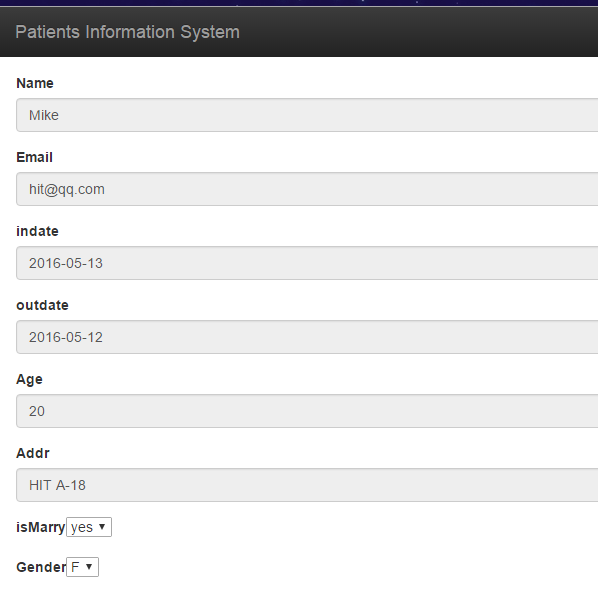
****

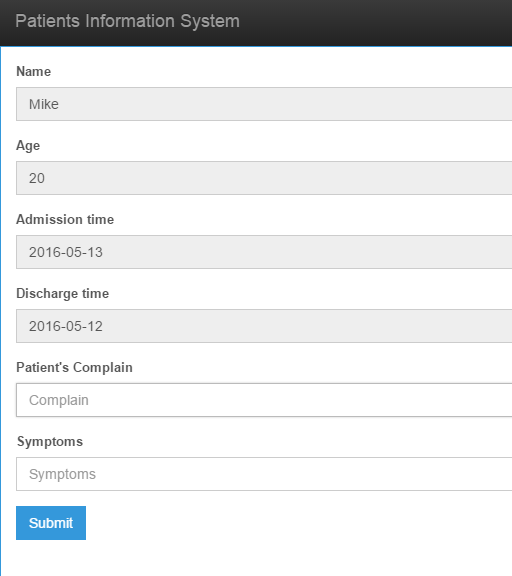
****

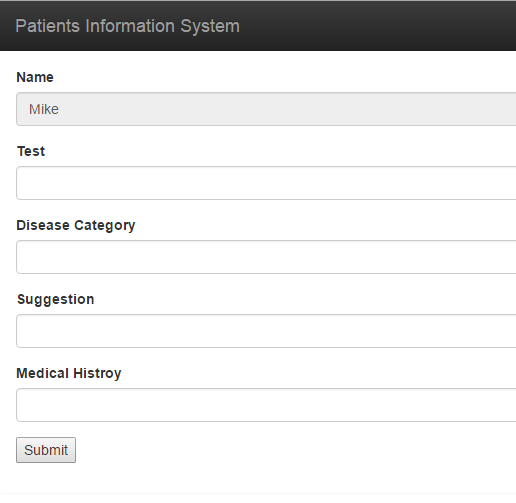
****

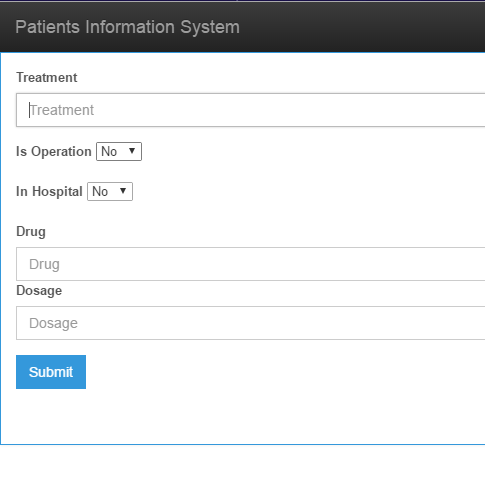
****

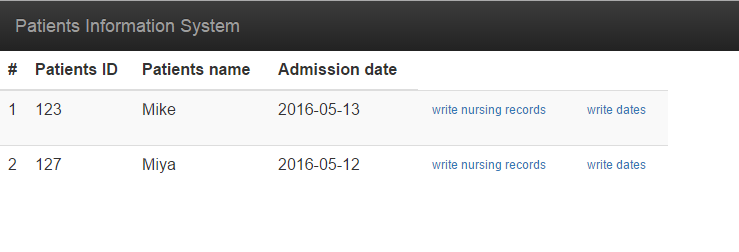
****

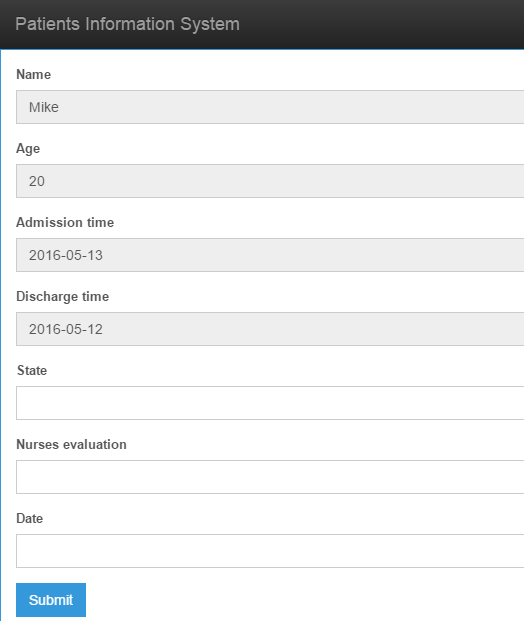
****

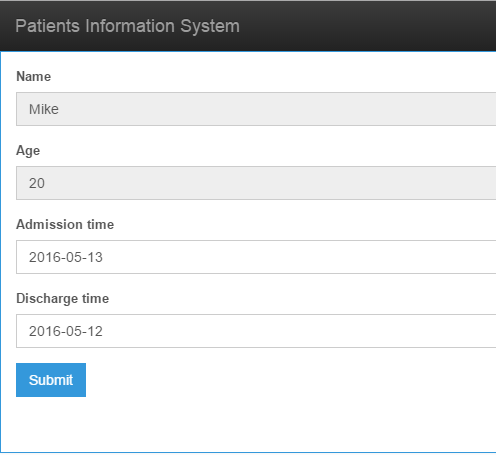
****

****

****

****

****

****

**Section 5.6 Reports**

1. Requirements Planed in Check#1
   * Patients shall register and update their personal information since they enter in the hospital.
   * Patients should view their personal information and health history.
   * Doctors should describe patients’ physical signs based on the physical test figures.
   * Doctors should also make a diagnosis and make a treatment for patients.
   * Nurses should write nursing records daily.
   * Patient information should be accessible by researchers and other PISes
2. Requirement Completed in Final Check

* Patients:
  + patients login/ sign up
  + view/modify basic information
  + view/modify basic information
  + view/modify basic information
* Doctors:
  + view patients information
  + view patients information
  + view patients information
  + view patients information
* Nurses:
  + view patients information
  + view patients information view patients information

Hopefully, we have complete almost all of the requirements gathered. Also we organize them in a nice way that we divide them into three roles. Each role has their own system entrance interface. And the function are separately designed which is more clear and more secure.

**Section 6 Conclusion**

In this project, we have experience a whole cycle for develop a system which focus on the database design. We have learnt a lot through this project especially in the database design process together with our software development techniques from our software engineering. This is a good project to associate these two course together.

Hopefully, we have finished our Patient Information System successfully. From my respective, what differs this project from our software engineering class project is that we not only focus on the usability of our system but also on the efficiency of our database system. The experiences are shown following.

First, we realize the importance of requirement gathering, that is we fully analysis the problem we are going to solve which is to maintain various kinds of data of patients in the hospital. So we think from the source of patient information along with the categories of the data.

Second, we have learnt to describe our thinking using graphs. For example, we use E-R diagram to illustrate the function of different entities, the relation between them and the interaction with other system.

Third, we have experience the design cycle of database design. First of all, we extract the entities from our requirements and analysis the relation between them (One-to-One, One-to-many or many-to-many). We illustrate this relationship using Dia diagram. After that we should collect the attributes that we should save in each table. And we should implement the various referential relationship using foreign keys. We illustrate and design them in EER Diagram using Mysql Workbench. After that we convert our logical design model into the physical tables using DDL. Then in order to maintain the referential integrity constraints and some automatic function associate the database modification, we use the trigger to achieve this goal. Moreover, we utilize the indexes to optimize our database and increase the efficiency of it. Meanwhile, we have consider the security issues solved by views. Finally it is the front part design and database connection.  
 Fourth, through this project, we have learnt the importance of communication. Sometimes the problems occurs because of the less communication among partners and supervisors. In contrast, when communication goes well, project will make progress very quickly.

* Lessons learned from each of your team member

*Richard*

Benefits:

I learnt a lot from this database development cycle and have a deeper observation towards the database optimization methods. Meanwhile, as the team leader, I have learnt how to organize the whole project and divide this project into different parts, which can concurrently develop.

Limitations:

I should learn some philosophy of management particularly how to increase the Enthusiasm of partners.

*Frank Wan*

Benefits:

Learned about php, sql, several softwares, integrity constrains, also the importance of authentication & authority

Limitations:

lack of the ability of efficiently searching information.

*Su Xuebin*

Benefits:

Mastering the concepts and basic skills for designing efficient and less redundant databases and building my teamwork experience.

Limitations:

Lacking of time management experience and falling short in scheduling the project.

**Section 6.1 Your Meeting log**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No | Date and Time | Duration | Location | Discussion topic | Attendees |
| 1 | 19/05/2016 | 2hours | Online | Tables structure design | all |
| 2 | 22/05/2016 4:00pm | 2hours | ZhengXin | Division of labor | all |
| 3 | 23/05/2016 6:30pm | 2hours | ZhengXin | Problems discussion | Xue bin, Richard |
| 4 | 25/05/2016 2:30pm | 4hours | ZhengXin,  then Gewu,  then library | Final implementation | all |