The github link is: <https://github.com/MengDeC4o/MedicalSystem/tree/master> .

The team log link is:

<https://docs.google.com/document/d/1CIlXrVjjtTmOzWw4o1IZmBANy0ODB7YPJ7iImVi9-Jc/edit?usp=sharing>

This is the backend documentation by Siyan Li.

Backend structure:



For a more clear diagram, please look at “Class\_Diagram\_final.png” in the submission.

All the “management” classes contain APIs that can return value to the front end.

All the “Mapper” classes contain DAOs that can access and modify the database.

“Drug”, “Patient”, “Room”, “Doctor”, “Nurse” are five basic data structures used in our program.

BackEnd Test method:

For the development of every little component, we use unit test to test it. Also, we use the software “postman” to perform automatic test. In postman, I made several sequence of API running. Each sequence represents a different path of program running. “Postman” can help to run our backend program in the sequence that is preset by us. Then we can look at the returning value to see whether the program is running as what we want it to run. The postman file is attached in postman\_test.zip.

Here is an example of a complex path of program running.



Backend important APIs

Login：//登录

“/user/login”:

Parameter: String account // 账号

String password // 密码

String status // 登陆者身份：nurse/doctor/patient/guest

Return value: Map<String,Object>: “success”, “login status: <status>”

“failure”, “account not found!”

“failure”, “wrong password!”

Patient account registration (Permission: all status)：//病人账号注册（权限：所有人）

“/user/account\_register”:

Parameter: String account //账号

String password //密码

Return value: Map<String,Object>: “error”, ” account already exists!”

“success”, ” account successfully created!”

Action: put new registered patient’s information in the database

Procedure：//看病流程

Intialization of rooms：// 初始化

“/Queue/init”:

Parameter: None

Return value: Map<String,Object>: “result: ”, ” successful initialization!”

Patient registration (Permission: doctor, nurse): // 病人挂号（权限：医生，护士）：

“/Queue/patient\_register”:

Parameter: patient\_account\_id

Return value: Map<String,Object>: “error: ”, “patient not found!” // if patient account is not in the database

“success: ”,”patient registered” // else

Action: put the patient’s instance to the current “in-hospital” patient\_list. // 将该病人的instance加入当前活跃的patient\_list当中

Assign room to the registered patient (Permission doctor, nurse): //病人挂完号后给其分配房间（权限：医生，护士）：

“/Queue/patient\_registered”:

Parameter: int patient\_id

String operation // patient 病人下一步要做的事情

Return value: Map<String, Object> "result: ","patient is added to the queue after registration!"

Action: Add the next room to the patient’s RoomeQueue, and add the patient to the corresponding room’s PatientQueue. // 给病人的RoomQueue加上下一个房间，并给相应房间的PatientQueue加上此病人。

病人离开一个房间（权限：医生，护士）：

“/Queue /patient\_finish”:

Parameter: int patient\_id

int room\_id // the room patient is leaving

String[] next operation //病人接下来要做的事情

Return value: Map<String,Object>: "result:","the patient can leave!" // 若病人没有要做的事情了

"success", "next operations successfully inserted" // 病人被安排到接下来的对应房间

Action: 病人的Queue中移除此房间，根据next\_op增加之后要去的房间.

对应房间的Queue加入此病人。

Information access and modify: 信息获取和修改：

Get all patient’s information in the database: // 获取所有数据库里病人的信息：

“/Patient /getAllPatients”:

Parameter: None

Return value: Map<String, Object> : “patients",list\_of\_patients

Get a patient from the patient\_id: //根据patient\_id获取数据库里某个病人的信息：

“/Patient/patient\_search\_by\_patient\_id”:

Parameter: int patient\_id

Return value: Map<String, Object> “patient", instance of that patient

Get a patient’s id from the patient\_account\_id: // 根据patient\_account\_id获取数据库里某个病人的patient\_id：

“Patient /patient\_id\_searchByAccount”:

Parameter: String patient\_account\_id

Return value: Map<String, Object> “id", patient\_id of the patient

Get information of all drugs:

“/Drug/ /getAllDrugs”

Parameter: None

Return value: Map<String, Object> “list",list\_of\_drugs

Insert a new drug to database:

“/Drug/insert\_newDrug”

Parameter: String drug\_name

Int amount

Return value: Map<String, Object> ("result","already exist, increment!") if already exists, ("result","successful insertion!") otherwise.

Action: insert the new Drug with its amount into the database

Stock change of a drug:

“/Drug /stock\_change2”

Parameter: int new\_stock

String drug\_name

Return value: Map<String, Object>: "result","success!". "failure", "no such drug in record!" If no such drug in database.