An Interactive Health Web Application: MEDIORA

Umang Agarwal[1], Shubham Bhardwaj[2], Nitesh Dewan[3]

[1,2,3] *School Computer Science and Engineering, VIT Vellore*

---***---

Abstract - *The purpose of this paper is to overcome the limited hour service availability in the current medical scenario. Saving time for both doctors and patients. Eliminating the time wasted sitting at the clinics. Good for doctors who cannot practice in clinics. In essence we are going to create a platform where people can come online and share ideas, raise discussions, issue complaints, create suggestion/reviews for doctor's services and encourage the use of e-prescriptions and any-time personalized services to patients. By eliminating the waiting time to meet the doctor we can saves both doctor's and patient's time and emergency services can be easily available.*

***Key Words*: Health care system, web application, interactive, e-prescriptions, platform**

1.INTRODUCTION

The product is aimed at providing comfortable medical assistance to patients by connecting them to doctors via a web application. The patients can issue requests which are then handled by the doctors and appropriate responses is generated. The patients if not satisfied with the response may query the doctors further till they are fulfilled with the solution. The software will enable the following function to be performed:

- **Users (Patients) :** Users should be able to create new account, log-in to their existing accounts which will give them the authority to use the services provided by the system. Authenticated users should be able to issue service request, check request status, browse through other services and post feedback. Authenticated users should be able to create search doctors for their particular ailment request services. Users can to create groups where users can share their experiences; discuss common problems, and the possible solution.

- **Doctors :** Municipal authorities can log-in to their accounts as created by administrator. Authorities can access all the service requests. Doctors can issue e-prescriptions. Doctor can chat with patients real-time.

- **System Administrators :** Create, and monitor accounts of doctors and patients.Filter the content reported as inappropriate and handle misuse of doctors information for illicit purpose. Handle complaints about improper response by doctors.

2. RELATED WORKS

2.1 WebMD:

It is a corporation which provides news on human health and information that leads to our well being. WebMD is known primarily for the news and information that is published relating to human health and well-being. The site displays information related to drugs. WebMD is among the top healthcare websites in terms of unique visitors. WebMD services are opted by physicians and private clients. Service like private Health management programs and decision-support portals are provided by WebMD health services. WebMD Health and other health-related sites together with: Medscape, MedicineNet, eMedicine, eMedicineHealth, RxList, theheart.org, Medscape Education, and other-owned WebMD sites are maintained by the WebMD Health Network . These sites provide related services to those of WebMD.

2.2 1mg:

It is an online pharmacy e-commerce health care company. 1 mg is an health care company established in Gurgaon, Haryana, India. 1 mg when initially founded was called HealthkartPlus, a subsidiary of HealthKart, by Prashant Tandon, Sameer Maheshwari, Gaurav Agarwal and Vikas Chauhan. In April 2015, HealthkartPlus was later separated and is called 1 mg now. 1 mg is an online store where you can buy medicines, moreover get a medical appointments and diagnostic test bookings at the comfort of one's home.

2.3 American Health Network:

Provides independent health insurance, Hosts the Health Network Community American Health Network (AHN) is a large physician group practice with over 70 offices and more than 200 primary care physicians and specialists in Indiana and Ohio. The main motive of the organization is to provide high-quality, cost-effective care, sustained by a electronic infrastructure which is strong and durable and in its mission to provide exceptional care to its patients

2.4 University Health Network:

University Health Network (UHN) is a health care and medical research organization in Toronto, Ontario, Canada. It is considered a largest research organization in Canada and North America, have the highest amount of research funding. It was named Canada's top research hospital by Research Infosource in 2015. All four hospitals are affiliated with the Faculty of Medicine at the University of Toronto and serve as teaching hospitals for resident physicians, nurses, and other health care professions. UHN also operates The Michener Institute, an applied health sciences college that opened in 1958

2.5 Practo:

Practo is an online-system where you can book appointments with doctors, upload medical, order medicines. Practo helps to find the right doctor for you based on your exact health issue, helps to book diagnostic tests, get medicines right at your door step, storing past health records or educating you with the ways to live a healthy life

3. SYSTEM ARCHITECTURE

3.1 Architectural Design

In Fig1.0 the various modules of the sytem are shown. It also shows which module interacts with the other.

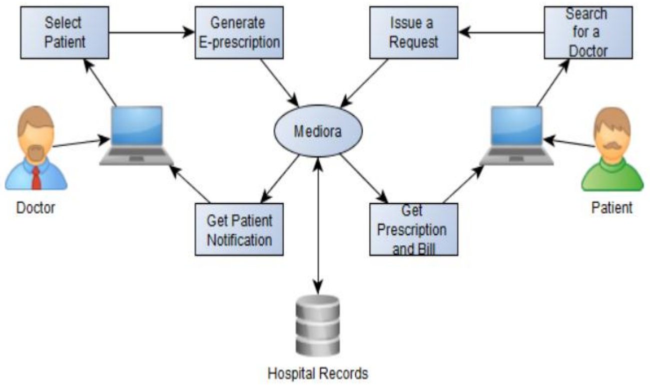


Fig.1.0 Modular structure of the software.

3.2 Decomposition Description

In Fig2.0 the different functions provided by the system are decomposed hierarchically into sub-functions.

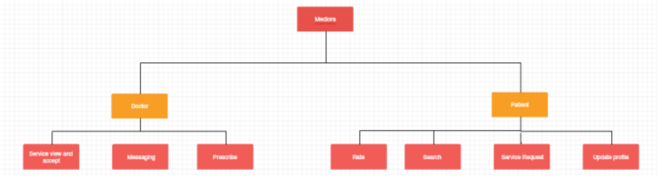


Fig.2.0 Hierarchical decomposition of the application functionalities

4. DESIGN PROCEDURE

4.1 Data Description

1. The form input information from the onboarding page is translated into a dictionary and stored in the users collection in the mongodb database as following dictionary - {'photo_link':'../'+ photo_link, 'username': username, 'password': password, 'email': email, 'name': fname+' '+lname, 'services': [], 'contact': contact, 'desig': 'patient', 'signup': 0, 'social_accounts': {}, 'joined_on': time} where photo_link points to the default photo if the users doesnt upload any photograph.

2. When the user logs in we use first authenticate his profile by checking his credentials from the database and use the resulting id from the search query to set a client side cookie. just incase he closes his tabs and want to not log into his account multiple times.

3. Incase of service request from patient to doctor we use the following data structure to record the information in the serviceRequests collection -

```
{'aliases':[{'fromid':s},{'toid':ObjectId(recvInfo['_id'])}],'Service':[{"accepted":0},{'service':service},{"sentby":userInfo["username"]},{"recievedby":recvInfo["username"]},{"time":time}],'photo_link':userInfo["photo_link"]}
```

4. When the doctor accepts the request we use the following query -

```
db.serviceRequests.update({'_id':ObjectId(sid)},{'$set':{'Service.0.accepted':1}})
```

5. Doctors have the liberty to update the patients prescription incase the default or want to write another medicine they can do so using the query -

```
db.prescriptions.update({'aliases':[{'fromid':ObjectId(s)},{'toid':ObjectId(obId)}]},{'$push':{'medicines':{'mn':mednames[i],'morning':mornings[i],'afternoon':afternoons[i],'evening':evenings[i],'daycount':int(daycount)}}})
```

4.2 Data Dictionary

Providing the list for all the classes:

1. AcceptServicesHandler
2. DoctorDashboardHandler
3. DoctorOnboardingHandler
4. IndexHandler
5. LoginHandler
6. PatientDashboardHandler
7. PatientOnboardingHandler
8. PatientProfileHandler
9. PrescriptionHandler
10. SearchHandler
11. ServiceRequestHandler
12. SignupHandler
13. UserProfileHandler

5. PROPOSED SYSTEM

An online platform where you can search doctors , apply for at the comfort of your home. The patient after logging in can contact doctor directly about their health issues/ symptoms and the doctor can prescribe medications based on the past health record of the patient or book an appointment to meet the doctor physically. A database of patients and doctors is maintained.

New users will be able to sign-up with their gmail/yahoo id or log-in to their current accounts and use the functionalities of the website. Users that are verified can raise a service request, check status of request, browse through other functionalities and can provide a feedback that will help improve the user-experience. Verified users for their particular ailment can search for doctor and request the particular doctor for his/her services. Users can also create groups where they will be able to give feedback for the particular treatment they received; discuss common complications, and can come up with a potential solution. Municipal authorities can log-in to their accounts as created by administrator. Authorities then will be to use the functionalities of the website. Patient will be able to receive an e-prescription. Doctor can do one-to-one with patients real-time. Create, and monitor accounts of doctors and patients. Take action on the inappropriate content and safeguard the information of the doctor. Handle complaints from patients who were not treated fairly by the doctors. User's data will be safe and not shared with anyone. 24 X 7 availability. Better component design to get better performance at peak time. Flexible service based architecture will be highly desirable for future extension

6. IMPLEMENTATION

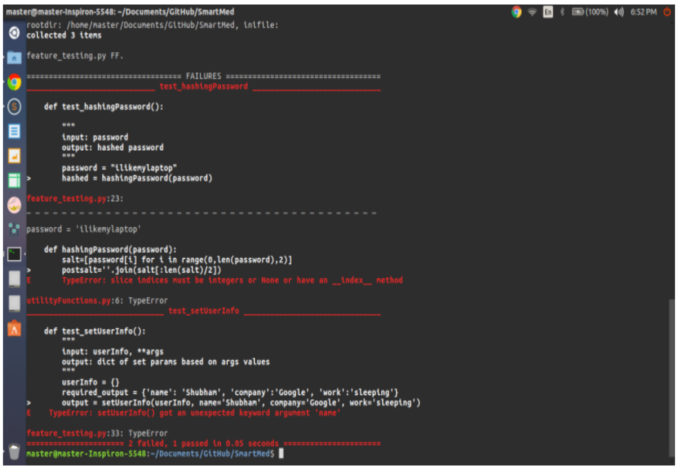
The front-end is a website and the back-end is made using AJAX. Tornado web-server is used and the database is made using NoSQL. Other tools used are XML, MongoDB, Materialize and POSTMAN. The testing was done using Python2.7.

7. TESTING

The testing for the system was done. The failed test cases are shown in Fig.3.0-5.0 show the failed test cases. The functions leading to failed test case were updated. The test case then passed successfully and is shown in Fig.6.0 Below is the test cases code, in Python2.7

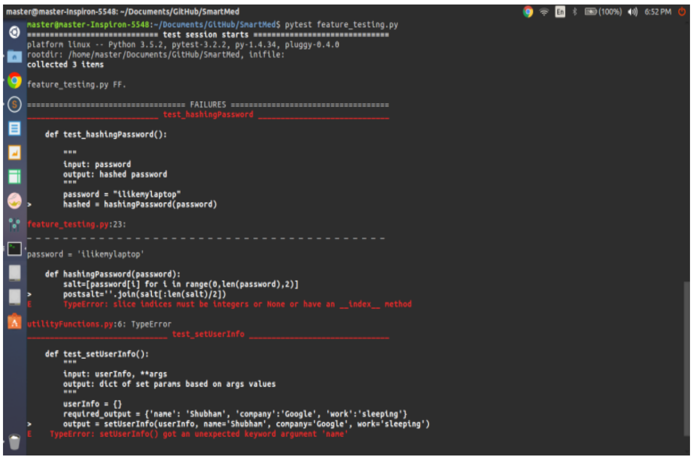
Testing function:

```
from utilityFunctions import hashingPassword, setUserInfo, sendMessage

# native dependencies

import logging

def test_hashingPassword():

"""

input: password

output: hashed password

"""

password = "ilikemylaptop"

hashed = hashingPassword(password)

assert hashed != password

def test_setUserInfo():

"""

input: userInfo, **args

output: dict of set params based on args values

"""

userInfo = {'name': 'Kamal', 'company':'goodboy', 'work':'ask_him'}

required_output = {'name': 'Shubham', 'company':'Google', 'work':'sleeping'}

output = setUserInfo(userInfo, 'name', 'company', 'work')

assert output != required_output

def test_sendMessage():

"""

input: number, text_message

output: success_text_message

"""

status_msg = sendMessage(9790625528, "please help me test my modules")

        assert status_msg == "success"
```



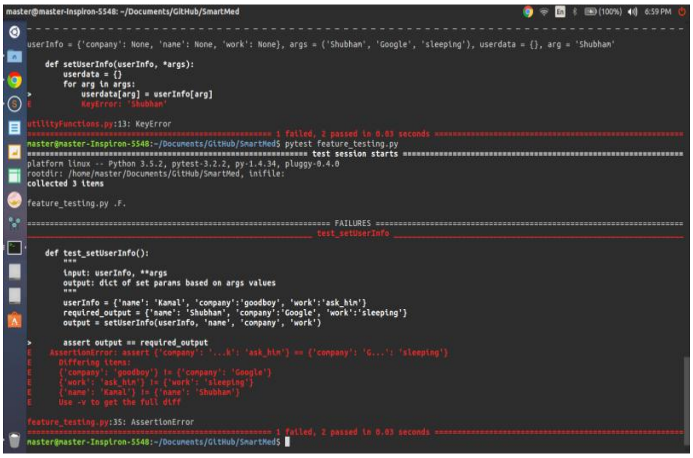
```
master@master-Inspiron-5548: ~/Documents/GitHub/SmartMed
rootdir: /home/master/Documents/GitHub/SmartMed, inifile:
collected 3 items

feature_testing.py FF.

================================== FAILURES ===================================
____________________________ test_hashingPassword _____________________________

    def test_hashingPassword():

        """
        input: password
        output: hashed password
        """
        password = "ilikemylaptop"
>       hashed = hashingPassword(password)

feature_testing.py:23:
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

password = 'ilikemylaptop'

    def hashingPassword(password):
        salt=[password[i] for i in range(0,len(password),2)]
>       postsalt=''.join(salt[:len(salt)/2])
E       TypeError: slice indices must be integers or None or have an __index__ method

utilityFunctions.py:6: TypeError
______________________________ test_setUserInfo _______________________________

    def test_setUserInfo():
        """
        input: userInfo, **args
        output: dict of set params based on args values
        """
        userInfo = {}
        required_output = {'name': 'Shubham', 'company':'Google', 'work':'sleeping'}
>       output = setUserInfo(userInfo, name='Shubham', company='Google', work='sleeping')
E    TypeError: setUserInfo() got an unexpected keyword argument 'name'

feature_testing.py:33: TypeError
===================== 2 failed, 1 passed in 0.05 seconds ======================
master@master-Inspiron-5548:~/Documents/GitHub/SmartMed$
```

Fig.3.0 Testing- failed tests



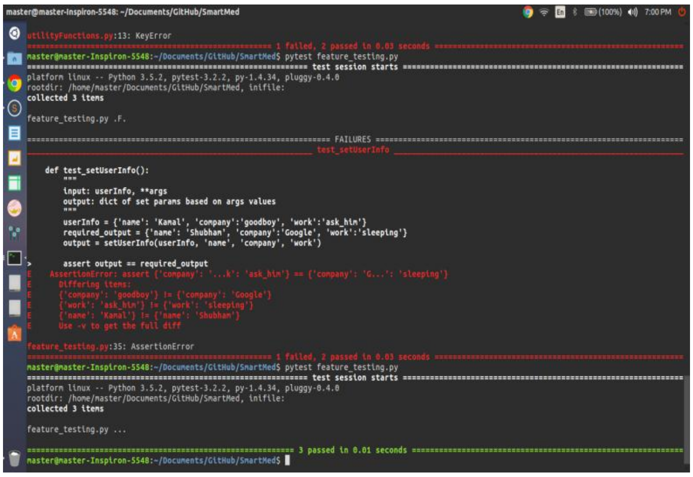
```
master@master-Inspiron-5548: ~/Documents/GitHub/SmartMed
master@master-Inspiron-5548:~/Documents/GitHub/SmartMed$ pytest feature_testing.py
============================= test session starts ==============================
platform linux -- Python 3.5.2, pytest-3.2.2, py-1.4.34, pluggy-0.4.0
rootdir: /home/master/Documents/GitHub/SmartMed, inifile:
collected 3 items

feature_testing.py FF.

=================================== FAILURES ===================================
_____________________________ test_hashingPassword _____________________________

    def test_hashingPassword():

        """
        input: password
        output: hashed password
        """
        password = "ilikemylaptop"
>       hashed = hashingPassword(password)

feature_testing.py:23:
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

password = 'ilikemylaptop'

    def hashingPassword(password):
        salt=[password[i] for i in range(0,len(password),2)]
>       postsalt=''.join(salt[:len(salt)/2])
E       TypeError: slice indices must be integers or None or have an __index__ method

utilityFunctions.py:6: TypeError
_______________________________ test_setUserInfo _______________________________

    def test_setUserInfo():
        """
        input: userInfo, **args
        output: dict of set params based on args values
        """
        userInfo = {}
        required_output = {'name': 'Shubham', 'company':'Google', 'work':'sleeping'}
>       output = setUserInfo(userInfo, name='Shubham', company='Google', work='sleeping')
E    TypeError: setUserInfo() got an unexpected keyword argument 'name'
```

Fig.4.0 Testing- failed tests



```
master@master-Inspiron-5548: ~/Documents/GitHub/SmartMed
_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

userInfo = {'company': None, 'name': None, 'work': None}, args = ('Shubham', 'Google', 'sleeping'), userdata = {}, arg = 'Shubham'

    def setUserInfo(userInfo, *args):
        userdata = {}
        for arg in args:
>           userdata[arg] = userInfo[arg]
E           KeyError: 'Shubham'

utilityFunctions.py:13: KeyError
===================================================== 1 failed, 2 passed in 0.03 seconds =====================================================
master@master-Inspiron-5548:~/Documents/GitHub/SmartMed$ pytest feature_testing.py
============================================================== test session starts ==============================================================
platform linux -- Python 3.5.2, pytest-3.2.2, py-1.4.34, pluggy-0.4.0
rootdir: /home/master/Documents/GitHub/SmartMed, inifile:
collected 3 items

feature_testing.py .F.

=================================================================== FAILURES ====================================================================
_______________________________________________________________ test_setUserInfo ________________________________________________________________

    def test_setUserInfo():
        """
        input: userInfo, **args
        output: dict of set params based on args values
        """
        userInfo = {'name': 'Kamal', 'company':'goodboy', 'work':'ask_him'}
        required_output = {'name': 'Shubham', 'company':'Google', 'work':'sleeping'}
        output = setUserInfo(userInfo, 'name', 'company', 'work')

>       assert output == required_output
E    AssertionError: assert {'company': '...k': 'ask_him'} == {'company': 'G...': 'sleeping'}
E      Differing items:
E      {'company': 'goodboy'} != {'company': 'Google'}
E      {'work': 'ask_him'} != {'work': 'sleeping'}
E      {'name': 'Kamal'} != {'name': 'Shubham'}
E      Use -v to get the full diff

feature_testing.py:35: AssertionError
===================================================== 1 failed, 2 passed in 0.03 seconds =====================================================
master@master-Inspiron-5548:~/Documents/GitHub/SmartMed$ 
```

Fig.5.0 Testing- failed tests

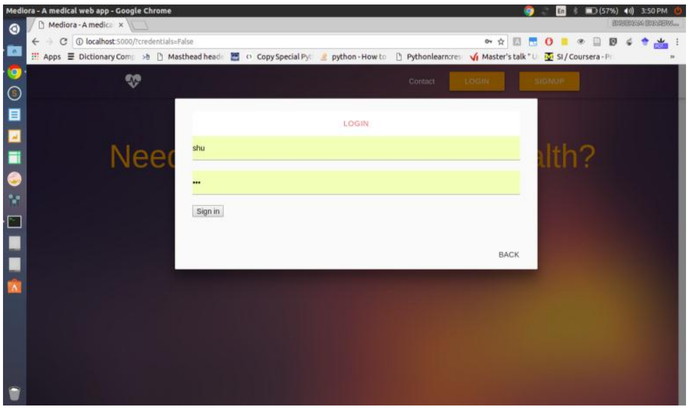


```
master@master-Inspiron-5548: ~/Documents/GitHub/SmartMed
utilityFunctions.py:13: KeyError
===================================================== 1 failed, 2 passed in 0.03 seconds =====================================================
master@master-Inspiron-5548:~/Documents/GitHub/SmartMed$ pytest feature_testing.py
============================================================== test session starts ==============================================================
platform linux -- Python 3.5.2, pytest-3.2.2, py-1.4.34, pluggy-0.4.0
rootdir: /home/master/Documents/GitHub/SmartMed, inifile:
collected 3 items

feature_testing.py .F.

=================================================================== FAILURES ====================================================================
_______________________________________________________________ test_setUserInfo ________________________________________________________________

    def test_setUserInfo():
        """
        input: userInfo, **args
        output: dict of set params based on args values
        """
        userInfo = {'name': 'Kamal', 'company':'goodboy', 'work':'ask_him'}
        required_output = {'name': 'Shubham', 'company':'Google', 'work':'sleeping'}
        output = setUserInfo(userInfo, 'name', 'company', 'work')

>       assert output == required_output
E    AssertionError: assert {'company': '...k': 'ask_him'} == {'company': 'G...': 'sleeping'}
E      Differing items:
E      {'company': 'goodboy'} != {'company': 'Google'}
E      {'work': 'ask_him'} != {'work': 'sleeping'}
E      {'name': 'Kamal'} != {'name': 'Shubham'}
E      Use -v to get the full diff

feature_testing.py:35: AssertionError
===================================================== 1 failed, 2 passed in 0.03 seconds =====================================================
master@master-Inspiron-5548:~/Documents/GitHub/SmartMed$ pytest feature_testing.py
============================================================== test session starts ==============================================================
platform linux -- Python 3.5.2, pytest-3.2.2, py-1.4.34, pluggy-0.4.0
rootdir: /home/master/Documents/GitHub/SmartMed, inifile:
collected 3 items

feature_testing.py ...

========================================================== 3 passed in 0.01 seconds ===========================================================
master@master-Inspiron-5548:~/Documents/GitHub/SmartMed$ 
```

Fig.6.0 Testing- passed tests

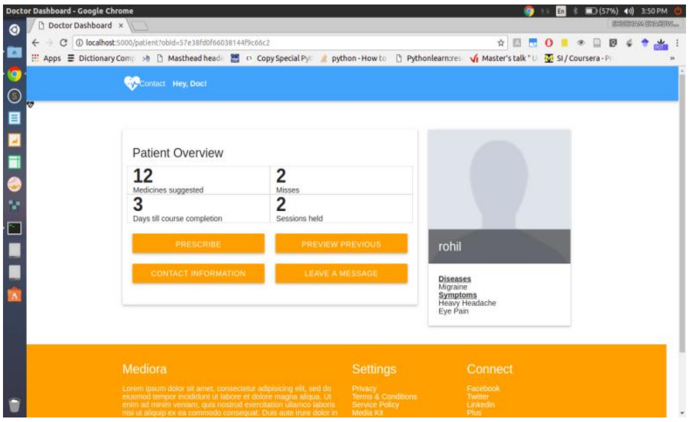
8. RESULTS AND DISCUSSIONS

Users were able to create new account and log-in to their existing accounts. Authenticated users were able to issue service request, check the status of their status, browse through other services and post feedback. Authenticated users can search doctors for their particular ailment and request services. Users can also create groups where they can share their experiences, discuss common problems, and the possible solution. Municipal authorities can log-in to their accounts as created by administrator. Authorities can access all the service requests. Doctors can issue e-prescriptions and can chat with patients in real-time. Administrators can create, and monitor accounts of doctors and patients, filter the content reported as inappropriate and handle misuse of doctors information for illicit purpose. Also, handle complaints about improper response by doctors. Fig.7.0-11.0 show the various instances of the application.



Mediora - A medical web app - Google Chrome
Mediora - A medica
localhost:5000/?credentials=False
Contact
LOGIN
SIGNUP
Need
alth?
LOGIN
shu
•••
Sign in
BACK

Fig.7.0 Login Screen

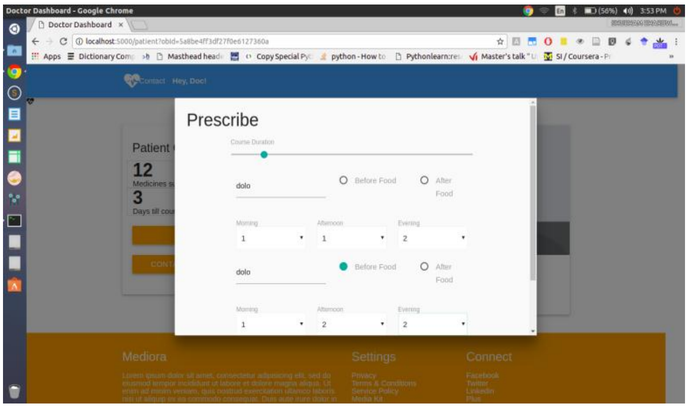


Patient Overview
12 Medicines suggested
2 Misses
3 Days till course completion
2 Sessions held

rohil
Diseases
Migraine
Symptoms
Heavy Headache
Eye Pain

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in

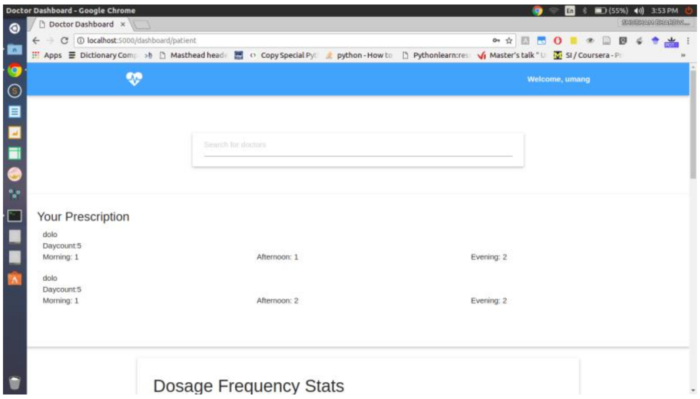
Fig.8.0 Doctor’s view of the patient.



Doctor Dashboard - Google Chrome
Doctor Dashboard
localhost:5000/patient?obId=5a8be4ff3df27f0e6127360a
Contact Hey, Doc!
Mediora
Settings
Connect

Prescribe
Course Duration
dolo ○ Before Food ○ After Food
Morning 1 Afternoon 1 Evening 2
dolo ● Before Food ○ After Food
Morning 1 Afternoon 2 Evening 2

Fig.9.0 Doctor prescribing the patient.



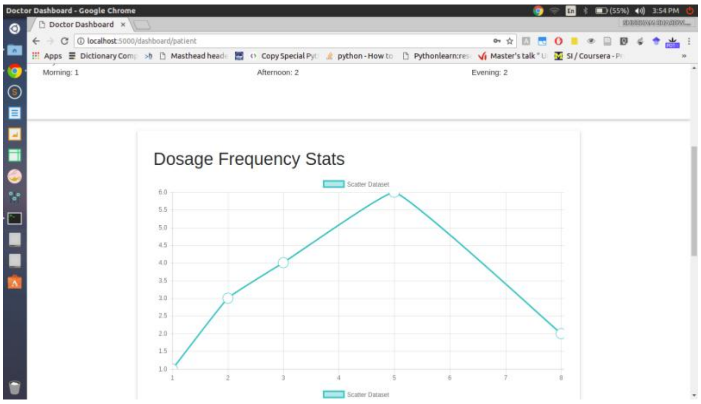
Your Prescription

dolo
Daycount:5
Morning: 1 Afternoon: 1 Evening: 2

dolo
Daycount:5
Morning: 1 Afternoon: 2 Evening: 2

Dosage Frequency Stats

Fig.10.0 Patients view of the prescription received.



Morning: 1 Afternoon: 2 Evening: 2

Dosage Frequency Stats

Fig.11.0 Patient's dosage frequency graph.

9. CONCLUSION

The website is to bring the medical practice online and enabling patients as well as the doctors to benefit from the Internet. Site acts as an interface between patients and doctors. Here the user can avail medical assistance at comfort of his/her home. It tackles the problem of not being able to visit the clinic as the doctor's availability does not match with the patient's schedule. With this website in place, patients can push his request at a time of his convenience and the doctor can reply whenever he has time. The system also brings the information of the doctors in a single place. Patients may choose the doctor after referring to their details, making, making a more informed choice. In future more features can be added to websites like a video chat interface, where the doctor and the patient can fix a time to have a one-to-one video call.

ACKNOWLEDGEMENT

We would like to acknowledge our professor Prabu S, for immense support throughout the project

REFERENCES

[1] Abdul Hakim H.M. Mohamed, Hissam Tawfik, Lin Norton, Dhiya Al-Jumeily (2013), A Technology Acceptance Model for a User-Centred Culturally-Aware E-Health Design.

[2] H. Balaji & A. Govardhan (2015), Precision improvement in healthcare information extraction.

[3] H. Dhand & A. Verma (2008), Healthcare kiosk.

[4] Sarah Masud Preum, Abu Sayeed Mondol, Meiyi Ma , Hongning Wang , John A. Stankovic (2017), Demo Abstract: Conflict Detection in Online Textual Health Advice.