Test Document

for

E-Vaidya

Version 1.0

Prepared by

Group 12: **Group Name:** Debuggers

Aman	**210104**	**aman21@iitk.ac.in**
Aniket Borkar	**210135**	**aniketsb21@iitk.ac.in**
Deven Gangwani	**210327**	**devenag21@iitk.ac.in**
Goutam Das	**210394**	**goutamd21@iitk.ac.in**
Kartik Soni	**210496**	**kartiksoni21@iitk.ac.in**
Narendra Singh	**210649**	**narendras21@iitk.ac.in**
Prashant Kumar	**210750**	**kprashant21@iitk.ac.in**
Shrey Bansal	**210997**	**shreyb21@iitk.ac.in**
Shubham Patel	**210709**	**devang21@iitk.ac.in**
Swastik Singhal	**211090**	**sswastik21@iitk.ac.in**

Course: CS253

Mentor TA: *Anuj Shrivastava*

Date: 20/03/23

Index

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
1.0	Debuggers	Initialized the document and added the necessary details.	02/04/23

1 Introduction

Mention the test strategy

We have used manual testing at various levels to check the robustness of our software. We have used Postman to send HTTP requests to different endpoints/backend routes and tallied the received responses against the expected response.

When was the testing conducted?

We conducted the testing in two phases, one of which was manual testing done prior to pushing whenever a new implementation was introduced. The other step was testing conducted concurrently with the implementation. Before committing a change with the main branch, the corresponding developer tested the functionality being added on their own. Pull requests were opened for significant feature/security features, and before the pull requests were merged, careful testing and conflict resolution were performed. We performed manual integration and system testing once again in the second step.

Who were the testers?

The entire team participated in the testing. Testing the backend of the programme was done by the students who worked on the front end, and vice versa.

What coverage criteria were used?

For manual testing, we used the number of lines of code reviewed manually as well as how many functional requirements which were fulfilled by the software. Also while testing, we used the Branch Coverage criteria. Since it was impossible to test all possible errors, we have kept the maximum achieved coverage to 95% and not 100%.

Have you used any tool for testing?

We have used Postman for back-end testing. Postman is a tool which lets the testers to send HTTP requests to the server and allows viewing of the responses in a single unified view. It was used to test all the backend endpoints.

2 Unit Testing

1. POST /login

The backend function for login was tested. All the testable branches functioned as expected.

Unit Details: Backend function to handle Post Request “/login”
Test Owner: [Patel Shubham Devang]
Test Date: [03/19/2023] - [03/19/2023]
Test Results:
Correct Username and Password – “Login Successful”
Correct Username and wrong Password – “Password didn't match”
Incorrect Username – “User not registered”
Structural Coverage: Branch Coverage 95%

2. POST /register

The backend function for registering was tested. All the testable branches functioned as expected.

Unit Details: Backend function to Post Request “/register”
Test Owner: [Swastik Singhal]
Test Date: [03/19/2023] - [03/20/2023]
Test Results:
Existing Username – “A user already registered with same email”
New Username – “Successfully Registered, please login now.”
Structural Coverage: Branch Coverage 95%

3. GET /refresh

The backend function for refreshing was tested. All the testable branches functioned as expected.

Unit Details: Backend function for Get Request “/refresh”
Test Owner: [Kartik Soni]
Test Date: [03/20/2023] - [03/20/2023]
Test Results:
Correct cookies – Access Token is received successfully
JWT cookies absent – 401 Error (Unauthorized)
Wrong JWT cookies – 403 Error (Forbidden)
Structural Coverage: Branch Coverage 95%

4. GET /logout

The backend function for logout was tested. All the testable branches functioned as expected.

Unit Details: Backend function to handle Get Request “/logout”
Test Owner: [Aniket Borkar]
Test Date: [03/20/2023] - [03/20/2023]
Test Results:
Correct cookies – cookies successfully updated
JWT cookies absent – 204 Error
Wrong JWT cookies – 204 Error
Structural Coverage: Branch Coverage 95%

5. POST /request_student

The backend function for requesting appointment by student was tested. All the testable branches functioned as expected.

Unit Details: Backend function to handle Post Request “/request_student”
Test Owner: [Swastik Singhal]
Test Date: [03/20/2023] - [03/20/2023]
Test Results:
Correct request – "Successfully request registered"
Structural Coverage: Branch Coverage 50%

6. GET /new:roll

The backend function for retrieving the upcoming appointments of a student was tested. All the testable branches except one functioned as expected. The server crashed in the last test case, in which invalid details were provided. The bug was fixed following which, all test cases passed.

Unit Details: Backend function to handle Get Request “/new:roll”
Test Owner: [Shrey Bansal]
Test Date: [03/20/2023] - [03/20/2023]
Test Results:
Valid details and appointments present – Upcoming appointments are successfully returned
Valid details and no appointments - "No appointments"
Invalid details – backend crashed
After the bug fix:
Valid details and appointments present – Upcoming appointments are successfully returned
Valid details and no appointments - "No appointments"
Invalid details – "Invalid input”
Structural Coverage: Branch Coverage 95%

7. POST /submitted

The backend function for updating details when the receptionist has submitted the approved request of the student was tested. All the testable branches functioned as expected.

Unit Details: Backend function to handle Post Request “/submitted”
Test Owner: [Deven Gangwani]
Test Date: [03/20/2023] - [03/21/2023]
Test Results:
Valid Appointment details provided: “Done”
Structural Coverage: Branch Coverage 95%

8. GET /doctor:name

The backend function for retrieving the upcoming appointments of a doctor was tested. All the testable branches except one functioned as expected.

Unit Details: Backend function to handle Get Request “/doctor:name”
Test Owner: [Aman]
Test Date: [03/21/2023] - [03/21/2023]
Test Results:
Valid name and appointments present – Upcoming appointments are successfully returned

Valid name and no appointments - "No appointments"
Invalid name – backend crashed
Structural Coverage: Branch Coverage is 100%

9. POST /nurse

The backend function for the nurse to add the patient's vitals was tested. All the testable branches functioned as expected.

Unit Details: Backend function to handle Post Request "/nurse"
Test Owner: [Narendra Singh]
Test Date: [03/20/2023] - [03/20/2023]
Test Results:
Valid roll number, has an active appointment and correct details were sent: "Done"
Valid roll number but has no active appointment allotted: "User must have a registered appointment to update vitals."
Invalid roll number: "User not found"
Structural Coverage: Branch Coverage 100%

10. GET /student_history:roll

The backend function for fetching student history was tested. All the testable branches functioned as expected.

Unit Details: Backend Get Request "/student_history:roll"
Test Owner: [Narendra Singh]
Test Date: [03/19/2023] - [03/19/2023]
Test Results:
Correct roll number and medical history is present: Medical history is returned
Correct roll number but there is no Medical History: Medical history is returned.
Structural Coverage: Branch Coverage 95%

11. POST /complete_appt

The backend function for marking a pending appointment as completed by doctor.

Unit Details: Backend Get Request "/complete_appt"
Test Owner: [Kartik Soni]
Test Date: [03/19/2023] - [03/19/2023]
Test Results:
Correct details: Medical history is updated.
Structural Coverage: Branch Coverage 95%

12. POST /dispense

The backend function for marking a pending prescription as completed by pharmacy.

Unit Details: Backend POST Request “/dispense”
Test Owner: [Prashant Kumar]
Test Date: [03/19/2023] - [03/19/2023]
Test Results:
The correct roll number and UID of the appointment is correct: “ok”
Structural Coverage: Branch Coverage 95%

13. GET /ongoing_appointments

The backend function for returning all ongoing appointments which have been allotted a doctor but have yet to be attended by the doctor.

Unit Details: Backend GET Request “/ongoing_appointments”
Test Owner: [Aman]
Test Date: [03/19/2023] - [03/19/2023]
Test Results:
No ongoing appointments: Empty array is returned
Ongoing appointments are present: Array of ongoing appointments is returned
Structural Coverage: Branch Coverage 100%

14. GET /phamacist

The backend function for returning the pending prescriptions to the pharmacy.

Unit Details: Backend GET Request “/pharmacist”
Test Owner: [Goutam Das]
Test Date: [03/19/2023] - [03/19/2023]
Test Results:
No pending prescription: Empty array is returned
Pending prescriptions are present: Array of pending prescription is returned
Structural Coverage: Branch Coverage 95%

15. POST /doctor_schedule

The backend function for setting the doctor schedule for the next day by the receptionist.

Unit Details: Backend POST Request “/doctor_schedule”
Test Owner: [Shrey Bansal]
Test Date: [03/19/2023] - [03/19/2023]
Test Results:
Array Doctor_list in the backend application is assigned the value which has been sent as a part of the request body.
Structural Coverage: Branch Coverage 100%

16. GET /schedule

The backend function for fetching the doctor schedule for the next day from doctor and student.

Unit Details: Backend GET Request "/schedule"
Test Owner: [Deven Gangwani]
Test Date: [03/19/2023] - [03/19/2023]
Test Results:
Array Doctor_list in the backend application is returned as a part of the response.
Structural Coverage: Branch Coverage 100%

17. POST /report_upload

The backend function for uploading the report, to be done by the receptionist.

Unit Details: Backend POST Request "/report_upload"
Test Owner: [Deven Gangwani]
Test Date: [03/19/2023] - [03/19/2023]
Test Results:
Valid report and correct roll number: The PDF file is uploaded successfully to the database.
Structural Coverage: Branch Coverage 100%

18. GET /report:roll

The backend function for uploading the report, to be done by the receptionist.

Unit Details: Backend POST Request "/report_upload"
Test Owner: [Deven Gangwani]
Test Date: [03/19/2023] - [03/19/2023]
Test Results:
Valid roll number and non zero PDFs for that roll number in DB: All PDFs for that roll number are returned
Valid roll number and zero PDFs for that roll number in DB: Empty array is returned
Invalid roll number: Empty array is returned
Structural Coverage: Branch Coverage 100%

19. GET /report/:roll/:pdfname

The backend function for fetching pdf report of student/patient by sending the report's roll number and PDF name.

Unit Details: Backend GET Request "/report/:roll/:pdfname"
Test Owner: [Shubham Patel]
Test Date: [03/19/2023] - [03/20/2023]
Test Results:

Correct PDF name and roll number: Test Passed. Report is fetched successfully

Structural Coverage: Branch Coverage 95%

3 Integration Testing

1. **Checking the interface between login and Dashboard for User**

   Module Details: login
   Test Owner: Narendra Singh
   Test Date: [03/21/2023] - [03/21/2023]
   Test Results: When entering the correct user login credentials, the software redirects to the corresponding dashboard.

2. **Logout redirects any user to the login page.**

   Module Details: logout
   Test Owner: Shubham Patel
   Test Date: [03/21/2023] - [03/21/2023]
   Test Results: Test successful for all roles

3. **User can not register with an already-used email.**

   Module Details: request
   Test Owner: Kartik Soni
   Test Date: [03/21/2023] - [03/21/2023]
   Test Results: Registration is denied, and prompt appears saying the email is already in use
   Additional Comments: [Enter any summary comments]

4. **Incorrect credentials do not allow login**

   Module Details: login
   Test Owner: Swastik Singhal
   Test Date: [03/22/2023] - [03/22/2023]
   Test Results: Any user cannot log in with incorrect credentials, it shows invalid along with the type of error. The password didn't match, or user not registered.

5. **Students can request an appointment, and this will be reflected on the receptionist page as a requested appointment**

   Module Details: request, setdoctors
   Test Owner: Goutam Das
   Test Date: [03/22/2023] - [03/23/2023]
   Test Results: Successful
   Additional Comments: Doctors of the slot corresponding to the one selected by the student are now displayed on the page. Receptionist can see the requested appointments on this dashboard.

6. **Students can view his/her upcoming appointments.**

   Module Details: upcoming
   Test Owner: Deven Gangwani
   Test Date: [03/22/2023] - [03/22/2023]
   Test Results: Successful. Appointments approved by HC and not completed show when user clicks 'Upcoming Appointments'
   Additional Comments: S

7. **Students can view his/her medical history.**

   Module Details: getHistory, History, setHistory
   Test Owner: Prashant Kumar
   Test Date: [03/23/2023] - [03/24/2023]
   Test Results: Successful, clicking on 'Medical History' redirects user to a new page showing his medical history and medical reports

8. **Students can view doctors' schedules for that day.**

   Module Details: Doctors
   Test Owner: Aman
   Test Date: [03/22/2023] - [03/23/2023]
   Test Results: When a student clicks on the "view Doctor Schedule" button, he is able see the schedule of all the doctors for that day.
   Additional Comments: Students can now see the doctors' schedules along with their slots.

9. **Doctor can see the upcoming appointments for his slot**

   Module Details: Appointment
   Test Owner: Shrey Bansal
   Test Date: [03/23/2023] - [03/24/2023]
   Test Results: Successful; when the receptionist approves a patient's appointment and assigns a doctor by clicking on the 'Allot Doctor' and 'Approve Appointment

Request' button, the appointment details is shown in the upcoming appointments tab of the respective doctor.

Additional Comments: The doctor can see his/her upcoming appointments.

10. Doctor can see the schedule of all the doctors for that day

Module Details: Schedule, set_doctor_list
Test Owner: Swastik Singhal
Test Date: [03/21/2023] - [03/22/2023]
Test Results: Successful; when the doctor clicks on the “Schedule” button, he can see the schedule of all other doctors for that day.

11. Doctor can conduct an appointment

Module Details: Conduct_appointment, setMedicine, onPrescriptionAdd
Test Owner: Patel Shubham Devang
Test Date: [03/23/2023] - [03/24/2023]
Test Results: Successful. Clicking on ‘Conduct appointment’ on an appointment card opens a new page of which shows details of the appointment. The doctor can now prescribe medicines by entering their medicine name and adding it to list. Clicking on the “Medical history” button shows the patient’s medical history finishing the appointment, this sends the prescriptions to pharmacy and add the prescribed medicines to the medical history of patient.

12. Receptionist can see appointment requests

Module Details: AppointmentRequests, setAppts
Test Owner: Narendra Singh
Test Date: [03/24/2023] - [03/25/2023]
Test Results: Successful, when a student requests an appointment by clicking on “book appointment” option, the receptionist can see that by clicking on the ‘View Appointment Requests’ option.

13. Receptionist can allot doctor to an appointment

Module Details: DoctorAllotment, setDoctors, submited
Test Owner: Aniket Borkar
Test Date: [03/24/2023] - [03/25/2023]
Test Results: Successful. receptionist can allot available doctor to an appointment by clicking on “Allot doctor” on appointment card. On selecting a doctor from dropdown allots doctor to appointment. Now the student can see the appointment in ‘Upcoming Appointments’ tab.

14. Receptionist can set Doctor's schedule

Module Details: DoctorAppointment, setMedicine, handleClick
Test Owner: Kartik Soni
Test Date: [03/25/2023] - [03/25/2023]
Test Results: Successful, Receptionist was able to allot a slot (Morning/Evening) of the same day to the doctor. By clicking on 'Set Doctor's schedule'. The receptionist can type the name of doctor to be assigned, select the slot, and type the room no. Clicking on 'Add doctor' adds doctor to array, and finally clicking on 'Add Doctors for the Day' sets the schedule for the day.

15. Receptionist can upload medical reports

Module Details: UploadReports, handleUpload, changeInput, handleSubmit
Test Owner: Prashant Kumar
Test Date: [03/26/2023] - [03/26/2023]
Test Results: Successful
Additional Comments: Receptionists can upload medical reports by entering roll/PF no and then selecting the file to upload; user can then view that medical report in medical history

16. Pending prescriptions showing on pharmacists' dashboard

Module Details: InitialPharmacy, handlePrint, Prescription
Test Owner: Narendra Singh
Test Date: [03/25/2023] - [03/25/2023]
Test Results: When the doctor completes the appointment by clicking the 'complete appointment' button, the prescription shows on the dashboard of the pharmacist

17. Pharmacists can select Medicines that are available and click on dispense prints

Module Details: prescription, handleClick, handlePrint
est Owner: Deven Ganwani
Test Date: [03/26/2023] - [03/26/2023]
Test Results: When pharmacist clicks on a prescription, a new page opens. List of medicines gets printed with the selected medicines showing with checked mark. Now he can print a receipt by clicking on 'Print Receipt' button.

18. Nurse can enter vitals of a patient

Module Details: InitialNurse, Vitals
Test Owner: Goutam Das
Test Date: [03/24/2023] - [03/24/2023]
Test Results: Successful, In the dashboard, Nurse can enter a patient's roll/pf number, click 'Enter' and can enter his/her measured vitals (blood pressure, body

temperature, oxygen level), and click ‘Submit’, which will reflect in student’s appointment profile.

4 System Testing

1. **Requirement:** The patient should be able to book an appointment on the chosen slot and have an option to choose from the available doctors and also view upcoming appointments; we tested the following

    - Login by Patient
    - If not a registered user, then registration
    - POST request by a patient for an appointment request
    - Login by the receptionist
    - Assign Doctor by the receptionist
    - Allot doctor to appointment by the receptionist

Test Owner: [Narendra Singh]
Test Date: [27/03/2023] - [28/03/2023]
Test Results: The test was successfully conducted. The testing begins by login the rom receptionist ID and then assigning doctors to slots of the day. Then log in using the patient ID and fill up the form for the appointment The receptionist then got the appointment request which he allotted an available doctor and confirmed the appointment. The patient was then able to see the appointment in his ‘upcoming appointments’ tab.

2. **Requirement:** Patient registration
Test Owner: [Swastik Singhal]
Test Date: [25/03/2023] - [26/03/2023]
Test Results: [Test was successfully conducted. For testing user had to click on ‘register’ button on login page. When redirected to register page, filled up the registration form. The patient was then able to login to the system using login credentials entered in the registration form
Additional Comments: [We are considering allowing only patients to register]

3. **Requirement:** The doctor can view all the appointments allotted to him, can start conducting any one of those appointments, and can see the student's medical history, after starting he can prescribe medications and diagnostic tests, and put additional remarks

We tested the following

- Login by doctor
- View the appointment by the doctor
- View medical history by the doctor
- Prescribe medications and tests by the doctor
- Vitals entered by Nurse

Test Owner: [Aniket Borkar]
Test Date: [28/03/2023]- [30/03/2023]
Test Results: [The test was successfully conducted. Testing was done dashboard page of a doctor, Nurse. The doctor was able to see the list of pending appointments and then started an appointment. After starting, he was able to see the patient's history. Minor front-end issues were fixed and integrated in the main application.

4. **Requirement:** The Pharmacist can view the prescriptions received from the Doctor and issue necessary medicines to the Patient

We tested the following:

- View of pending prescriptions by the pharmacist
- Dispensing of medicines by a pharmacist

Upon successfully testing all the components, the pharmacist was able to view all pending prescriptions and dispense medicines.
Test Owner: [Shrey Bansal]
Test Date: [03/29/2023] - [03/30/2023]
Test Results: [Test was successfully conducted. It was done on the dashboard of a pharmacist. Minor issues were fixed and integrated in the main application.]

5. **Requirement:** The receptionist can upload the medical reports of the patient, which can then be viewed by patient later-on

Upon successful testing, the receptionist could upload medical reports by entering the Roll/PF no of the patient. The patient was able to see medical reports in the medical history tab.
Test Owner: [Kartik Soni]
Test Date: [03/29/2023]- [03/31/2023]

Test Results: [Test was successfully conducted. The test was done on the dashboard of the receptionist and patient]

6. **Requirement:** The nurse can enter the vitals of a patient (blood pressure, body temperature, oxygen level), which can then be viewed by the doctor and student later-on

Upon successful testing, the nurse was able to enter the patient's measured vitals by entering the patient's roll/pf number. The doctor was able to see the vitals during conduction of the appointment.
Test Owner: [Goutam Das]
Test Date: [03/31/2023]- [03/31/2023]
Test Results: [Test was successfully conducted. The test was done on the dashboard of nurse and doctor]

7. **Requirement:** External interface requirement
We tested the following components.
   - User Interfaces
   - Software Interfaces

   All the pages and databases were tested manually to check whether everything was working properly. The API testing was automated, while some requests were manually tested.

Test Owner: [Aman]
Test Date: [03/31/2023] - [03/31/2023]
Test Results: [The test was successfully conducted. The testing was done on all the HTML pages. All the APIs used were also tested and properly integrated. Minor issues were fixed and integrated in the main application by creating and testing PRs over GitHub.

8. **Requirement:** Non-functional requirements
We tested for the following components.
   - Performance Requirements
   - Safety and Security Requirements

[API response time was tested for smoothness. The page loading time and other security issues were tested.]
Test Owner: [Kartik Soni]
Test Date: [04/01/2023] - [04/01/2023]
Test Results: [The test was successfully conducted. The main aim of the testing was to check whether all API requests were responding in the stipulated time. We even checked whether the login system was secure, whether the access to pages was restricted based on role or not.

9. **Requirement:** The Receptionist can Login after registration, can view all the pending appointments, can view currently ongoing appointments, can approve good appointments, and can allot a doctor, can upload medical reports of patients, and can schedule doctors for current day slot-wise.

   We tested the following:

   - Registration by Receptionist
   - Login by Receptionist
   - Visibility of pending appointments to the Receptionist
   - Approval of good appointments and allotting a doctor by the Receptionist
   - Visibility of ongoing appointments' status to the Receptionist
   - Uploading medical reports by the Receptionist
   - Scheduling doctors for the current day by the Receptionist

Test Owner: [Prashant Kumar]
Test Date: [29/03/2023]- [30/03/2023]
Test Results: [The test was successfully conducted. Testing was done dashboard page of a doctor, Receptionist, and Student. The Receptionist was able to see the list of pending appointments and then allotted doctors to students. After allotting, he was able to see the status of ongoing appointments. Minor front-end and back-end issues were fixed and integrated in the main application.

5 Conclusion

How Effective and exhaustive was the testing?

Testing was quite effective because we ensured that when any member pushed their local changes to the main repo, the code was working as intended, and there were no major flaws.
Our application has 5 users: student, doctor, pharmacist, receptionist (HC admin), and nurse. This eventually led to the creation of many different functions that required a significantly high number of tests. Using postman for back-end testing helped us in automating a portion of the testing, thus relieving us little in exhaustiveness.

Which components have not been tested adequately?

Almost all the components have been tested adequately.

What difficulties have you faced during testing?

Manual testing for integration testing was quite time-consuming. We could not find a feasible way to automate integration testing. We were able to test different functionalities just because we had maintained everything on GitHub in commits and issues testing; testing all those requirements was possible.

How could the testing process be improved?

The GitHub repo has helped us a lot while testing, so maintaining a track of development would improve the testing process, especially for safety-critical systems. Good documentation allows us to identify the portions that need testing. Moreover, completely automating all testing processes can result in a lot of improvement and even save a lot of time. We only guaranteed that the extremely sophisticated queries made to the backend by Postman would function properly for all potential frontend inputs, not just any input. By carrying out the skipped step, we may have further assured the security of backedn.

Appendix A - Group Log

Sr. No.	Date	Activity	Participants
1	18-03-2023	We organised meetings to initiate the testing procedures and in case of any issue.	Aman, Swastik, Kartik, Aniket, Shubham, Goutam, Shrey, Prashant, Deven, Narendra
2	20-03-2023	Finished the Software Implementation Document and resolved queries in the application	Aman, Swastik, Kartik, Aniket, Shubham, Goutam, Shrey, Deven, Prashant, Narendra
3	23-03-2023	Discussed and distributed work related to the issues in the backend testing	Aman, Aniket, Swastik, Narendra, Shrey, Shubham, Prashant, Goutam, Kartik, Deven
4	26-03-2023	Tested the login, role-based authentication, nurse and pharmacy requests using postman and fixed bugs	Aman, Aniket, Swastik, Narendra, Shrey, Shubham, Prashant, Goutam, Kartik, Deven
5	30-03-2023	Tested all remaining requests of students, doctors and receptionists using postman and fixed bugs	Aman, Aniket, Swastik, Narendra, Shrey, Shubham, Prashant, Goutam, Kartik, Deven