**A PRELIMINARY REPORT ON**

**POLD APPOINTMENT BOOKING SYSTEM**

SUBMITTED TO THE CDAC ACTS, BENGALURU

IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE AWARD OF THE DEGREE

OF

**THE POST GRADUATE DIPLOMA IN ADVANCED COMPUTING**

##### **SUBMITTED BY**

**OMKAR NATE 190850120062**

**PRANAV LENGULE 190850120047**

**LALIT MAHAJAN 190850120048**

**DEBARSHI PAL 190850120063**



## **DEPARTMENT OF ADVANCED COMPUTING**

**C-DACs**

**ADVANCED COMPUTING TRAINING SCHOOL**

No.1, Old Madras Road Above Bank of Mysore, Near NGEF, Bengaluru,

Karnataka 560038, India.

**CERTIFICATE**



This is to certify that the project report entitled

**“POLD APPOINTMENT BOOKING SYSTEM”**

Submitted by

**OMKAR NATE 190850120062**

**PRANAV LENGULE 190850120047**

**LALIT MAHAJAN 190850120048**

**DEBARSHI PAL 190850120063**

are the bonafide students of this institute and the work has been carried out by them under the supervision of **Mr. Srinivas P.**and **Mr. Shanmuganathan.** It is approved for the partial fulfillment of the requirement of CDAC ACTS, for the award of the degree of **Post Graduate Diploma** of **Advanced Computing**.

**Mr. Srinivas P.** **Mr. Shanmuganathan**

Guide, Guide,

Department of Advanced Computing Department of Advanced Computing

**Mrs. Binu George**

Course Coordinator,

CDAC ACTS, Bengaluru 560038

Place: Bengaluru Date:

**ACKNOWLEDGEMENT**

It is a matter of great pleasure for us to submit this project report on " **POLD Appointment Booking System**", as a part of the curriculum for “The Post Graduate Diploma in Advanced Computing(PG-DAC)” of CDACs ACTS. We are thankful to our guide Mr. Srinivas P. and Mr. Shanmuganathan, for their constant encouragement and able guidance. We are also thankful to Mrs. Binu George, Course Coordinator of CDAC ACTS Bengaluru, for her valuable support.

We take this opportunity to express our deep sense of gratitude towards those who have helped us in various ways, for preparing our seminar. At the last but not least, we are thankful to our parents, who had encouraged and inspired us with their blessings.

**ABSTRACT**

POLD focuses on easing the doctor appointment booking system. It helps patients to reach out to the hospital from an online portal. The patient can search for a doctor he/she wishes to consult, and pick a date for requesting an appointment to a specific doctor. The doctor are managed in the specialities tab as per their respective speciality. The patient can also update his profile details if required. POLD also has an in-house web portal. The in-house web portal is for hospital only. The reception, the doctor and the admin have access to this portal. The doctor page displays the doctor's appointments in a managed way. The doctor specific appointments are managed as today & future appointments, and appointment requests. The reception page helps patients to search for their appointment when they reach the hospital. The admin page is for the admin who can manage patients and doctors in the hospital.

**Project Workflow**

This is a scenario where the patient is a new user and starts with registering himself.

* **PATIENT**

1. The new patient goes to the online web portal. Checks out the tabs on the website.
2. When the patient goes on the specialities tab, he/she sees the information of the specialities available at the hospital and the doctors associated with the speciality.
   1. Client side sends an HTTP GET Request to the Server at ‘api/specialities\_view’ with the speciality name as a parameter.
   2. On the server side ‘Getspecialities\_view(speciality\_name)’ function of class specialities\_viewController is called.
   3. This function will call a function in the DAO Layer and return the information of the speciality and the doctors related to it.
3. Every doctor entry will have a link which will ask the patient to login. The patient clicks on the link, he/she gets redirected to the home page where the login division is displayed.
4. As the patient is a new user, he has to do the registration so he clicks on register button.
5. The patient is then redirected to the registration page. Here the patient fills his personal details viz. name, email-id, mobile number, DOB, gender, blood group, address, password.
6. After clicking on the register button, he/she is redirected to Home page/Login page.
   1. Client side sends an HTTP POST Request to the Server at ‘api/patients/newpatient’ with the registration details binded to an object.
   2. On the server side ‘Postpatient\_data(newPatient)’ function of class patient\_dataController is called and it validates the data.
   3. This function will call a function in the DAO Layer and create a new entry in the database and send back an acknowledgement that the entry is inserted
   4. Server will respond back to the client with a message saying registration successful.
7. Then he/she does login with the mobile no. and his/her password.
   1. Client side sends an HTTP POST Request to the Server at ‘api/patients/login’ with the login details binded to an object.
   2. On the server side ‘patientLogin(verifyPatient)’ function of class patient\_dataController is called.
   3. This function will call a function in the DAO Layer and search for patient\_mobileNo and password, if found patient\_data will be sent back to client side.
8. Once the patient enters his correct details for login, he is redirected to the patient homepage where he is able to see all his appointment’s status which he requested. As the patient is new, he does not have any appointments.
   1. Client side sends an HTTP POST Request to the Server at ‘api/patients/appointments’ with the patients object.
   2. On the server side ‘patientAppointment(patient)’ function of class patient\_dataController is called.
   3. This function will call a function in the DAO Layer and search for patient appointment data from appointment table, if found appointment\_data will be sent back to client side.
9. The patient is provided with two buttons at the home page, ‘Book Appointment’ and ‘Cancel Appointment’.
10. The user clicks on ‘Book Appointment’, he is redirected to the specialities page. Here he/she will see the doctor details. He/she choose the specialty and the doctor according to the requirement.
11. The user clicks on the ‘Book Appointment’ button on the specialities page of a specific doctor, he is redirected to a page where he/she will have to choose the appointment date.
12. The patient selects the appointment date according to his /her will and he/she sends a request to that specific doctor and has to wait until the doctor confirms the appointment.
    1. Client side sends an HTTP POST Request to the Server at ‘api/patients/appointmentreq’ with the appointment object having doctors data , patient’s data and appointment date.
    2. On the server side ‘bookAppointment(newAppointment)’ function of class patient\_dataController is called.
    3. This function will call a function in the DAO Layer and create an entry in appointment data table and sends a message saying appointment requested successfully.
13. The status of all the appointments is displayed on the homepage of the patient.
14. The patient logs out.

The doctor is provided with his login id and password given by the admin.

* **DOCTOR**

1. The doctor goes on the homepage of the in-house web portal and clicks on the doctor login tab.
2. On the doctor login page, the doctor enters the login credentials provided by the admin. If the credentials are correct, the doctor is redirected to the doctor home page.
3. At the doctor’s homepage, the doctor can see his appointments scheduled for that day.
4. The doctor’s homepage is provided with three tabs, ‘Today’s Appointment’, ‘Future Appointments’ and ‘Appointment Requests’.
5. When a patient’s appointment starts the doctor clicks on start appointment session of that patient which lasts for roughly 15 minutes.
6. Once the 15 minutes timer is completed, the patient’s appointment is completed.
7. The doctor clicks on appointment ‘Appointment Requests’ tab and the future appointment requests are displayed.
8. The doctor confirms the appointment for all the patients and their appointment status is changed to confirm.
9. Once the doctor confirms the appointments under ‘Appointment Requests’, the confirmed appointments will be displayed under ‘Future Appointments’ tab.
10. If the doctor wishes to cancel one of his future appointment, he/she goes to the ‘Future Appointments’ tab and cancel the appointment of that patient.

**Conclusion**

* We learned about the three tier architecture which is required to create a full stack application.
* Client side:
  + Angular 8:
    - We learned about child routing, activated routes, router navigation.
    - Maintaining folder structure in angular.
    - UI designing
  + Maps:
    - We also learned about integrating google map api with angular.
* Server side:
  + ASP.Net:
    - Use of web api to accept http requests and response messages.
    - Use of Entity Framework for O2O (Object to Object) Mapping with DAO layer.
    - Linq to operate on the DAO layer data models.
    - CORS(Cross Origin Resource Sharing) to integrate ASP.Net with Angular 8.
* DAO Layer:
  + MSSQL Server:
    - Use of MSSQL Server to manage database.
    - To create triggers, views, etc.