**Brunel University London**

# NC1600 Level 1 Group Project Reassessment – Software Implementation

Assignment Title: Software Implementation

Date of Submission: 15/08/2024

Submitted To

**Dr Zear Ibrahim**
Associate Lecturer
Department of Computer Science at Brunel University London

Submitted By

**Name: Nafis Alam Tousim**
Student ID: 2249954

## Introduction:

The Brunel Hospital Management System is an application in Java built for maintaining a hospital management. The implementation includes functionality for managing and manipulating reviews of healthcare professionals.

Patient feedback is inevitable in the health care system today not only for process improvement but also to allow the new patient to make the right choice of the healthcare provider. There will be a user-friendly interface through which patients can search for doctors, add reviews, and read feedback. It has an emergent user-friendly design needed for efficient health care review management. The design is made easy accordingly to ensure that all users of the system move around with ease, regardless of their technical expertise.

This report details the software implementation of the Review Management System for Doctors, which features a Java-based GUI application with Java Swing. It will explain all the met requirements, conducted testing, future work, and finally conclude by summarizing overviews of the success of the implementation. In the appendix, it will showcase source code for all the classes implemented.

## Requirements Met:

1. This component will allow patients to provide feedback and ratings for the doctors they have visited. The GUI should support switching between different windows : **Completed.**

2. At the outset, this component should display average ratings and feedback for each doctor. This component should also implement a continuously running sorting mechanism that finds the highest-rated doctors and display these at the top of the average ratings : **Completed.**

3. After an appointment has been completed and once the medicine has been dispensed, patients will have the option to rate their experience and leave comments : **Completed.**

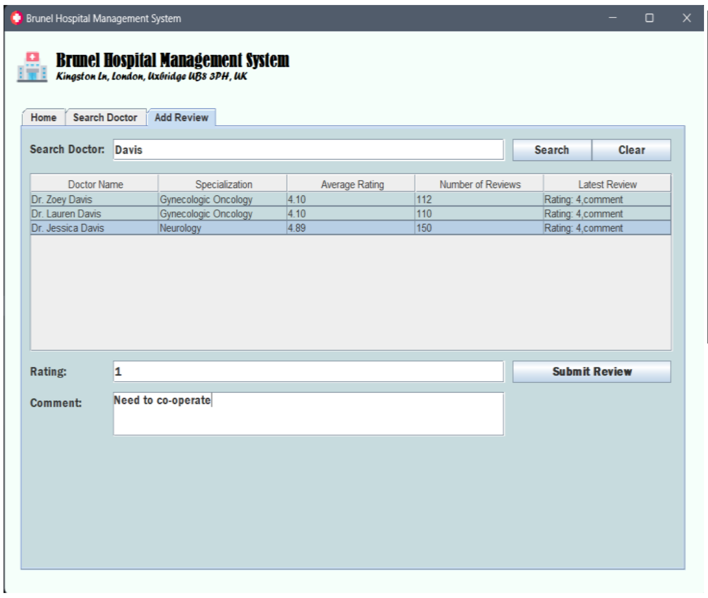
4. The doctor's information and any existing rating are to be retrieved from the review control system. New ratings and comments left by the patient or next of kin are to be attached to the doctor’s records : **Completed.**

5. Error handling needs to be included as well as an interface that is clear and easy to use : **Completed.**

## <u>Testing:</u>

The following tests were conducted to ensure the implementation meets the requirements. JUnit tests and manual tests through the interface were used where applicable.

| Test Number | Component | Duration | Corresponding Requirements | Pass/Fail | Correction |
|---|---|---|---|---|---|
| 1 | *Doctor Review* | Patient’s rating and comment should be included to the corresponding doctor’s overall rating as well as comment. The GUI switch should work. | 1,3, 5 | Pass | N/A |


Brunel Hospital Management System

Brunel Hospital Management System
Kingston Ln, London, Uxbridge UB8 3PH, UK

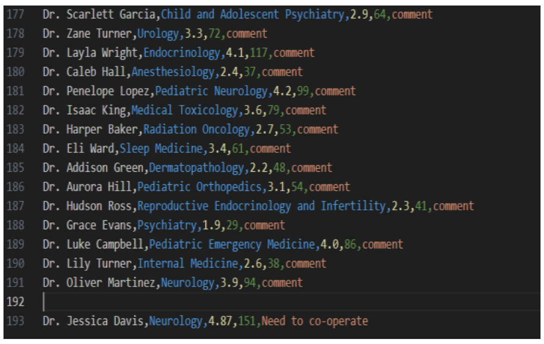
Home | Search Doctor | Add Review

Search Doctor: Davis  [Search] [Clear]

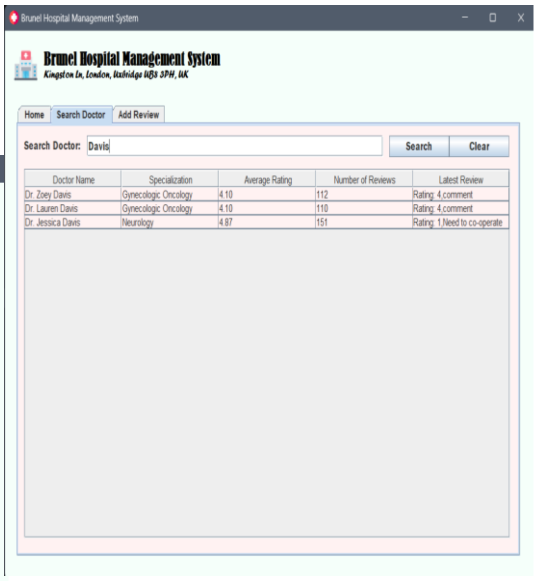
| Doctor Name | Specialization | Average Rating | Number of Reviews | Latest Review |
|---|---|---|---|---|
| Dr. Zoey Davis | Gynecologic Oncology | 4.10 | 112 | Rating: 4,comment |
| Dr. Lauren Davis | Gynecologic Oncology | 4.10 | 110 | Rating: 4,comment |
| Dr. Jessica Davis | Neurology | 4.89 | 150 | Rating: 4,comment |

Rating: 1  [Submit Review]

Comment: Need to co-operate


```
Dr. Scarlett Garcia,Child and Adolescent Psychiatry,2.9,64,comment
Dr. Zane Turner,Urology,3.3,72,comment
Dr. Layla Wright,Endocrinology,4.1,117,comment
Dr. Caleb Hall,Anesthesiology,2.4,37,comment
Dr. Penelope Lopez,Pediatric Neurology,4.2,99,comment
Dr. Isaac King,Medical Toxicology,3.6,79,comment
Dr. Harper Baker,Radiation Oncology,2.7,53,comment
Dr. Eli Ward,Sleep Medicine,3.4,61,comment
Dr. Addison Green,Dermatopathology,2.2,48,comment
Dr. Aurora Hill,Pediatric Orthopedics,3.1,54,comment
Dr. Hudson Ross,Reproductive Endocrinology and Infertility,2.3,41,comment
Dr. Grace Evans,Psychiatry,1.9,29,comment
Dr. Luke Campbell,Pediatric Emergency Medicine,4.0,86,comment
Dr. Lily Turner,Internal Medicine,2.6,38,comment
Dr. Oliver Martinez,Neurology,3.9,94,comment

Dr. Jessica Davis,Neurology,4.87,151,Need to co-operate
```

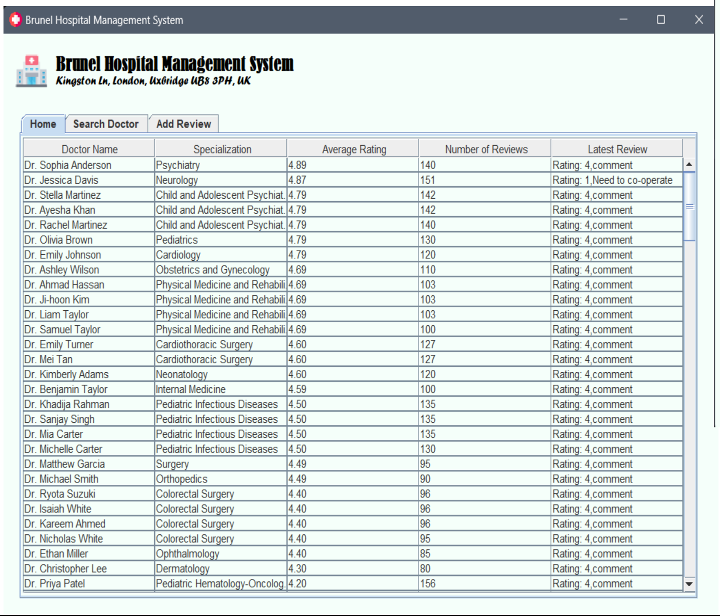

Brunel Hospital Management System

Brunel Hospital Management System
Kingston Ln, London, Uxbridge UB8 3PH, UK

Home | Search Doctor | Add Review

Search Doctor: Davis  [Search] [Clear]

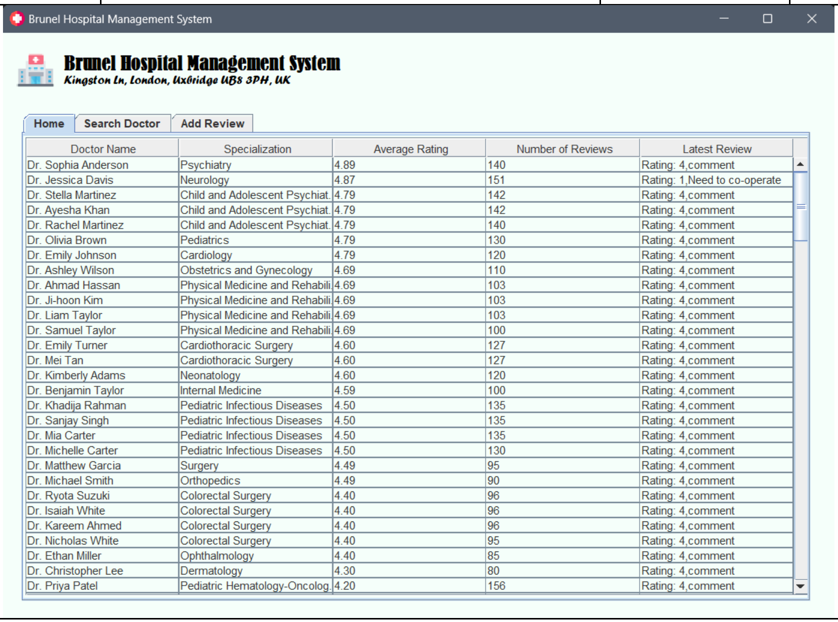
| Doctor Name | Specialization | Average Rating | Number of Reviews | Latest Review |
|---|---|---|---|---|
| Dr. Zoey Davis | Gynecologic Oncology | 4.10 | 112 | Rating: 4,comment |
| Dr. Lauren Davis | Gynecologic Oncology | 4.10 | 110 | Rating: 4,comment |
| Dr. Jessica Davis | Neurology | 4.87 | 151 | Rating: 1,Need to co-operate |


Brunel Hospital Management System

Brunel Hospital Management System
Kingston Ln, London, Uxbridge UB8 3PH, UK

Home | Search Doctor | Add Review

| Doctor Name | Specialization | Average Rating | Number of Reviews | Latest Review |
|---|---|---|---|---|
| Dr. Sophia Anderson | Psychiatry | 4.89 | 140 | Rating: 4,comment |
| Dr. Jessica Davis | Neurology | 4.87 | 151 | Rating: 1,Need to co-operate |
| Dr. Stella Martinez | Child and Adolescent Psychiat. | 4.79 | 142 | Rating: 4,comment |
| Dr. Ayesha Khan | Child and Adolescent Psychiat. | 4.79 | 142 | Rating: 4,comment |
| Dr. Rachel Martinez | Child and Adolescent Psychiat. | 4.79 | 140 | Rating: 4,comment |
| Dr. Olivia Brown | Pediatrics | 4.79 | 130 | Rating: 4,comment |
| Dr. Emily Johnson | Cardiology | 4.79 | 120 | Rating: 4,comment |
| Dr. Ashley Wilson | Obstetrics and Gynecology | 4.69 | 110 | Rating: 4,comment |
| Dr. Ahmad Hassan | Physical Medicine and Rehabili. | 4.69 | 103 | Rating: 4,comment |
| Dr. Ji-hoon Kim | Physical Medicine and Rehabili. | 4.69 | 103 | Rating: 4,comment |
| Dr. Liam Taylor | Physical Medicine and Rehabili. | 4.69 | 103 | Rating: 4,comment |
| Dr. Samuel Taylor | Physical Medicine and Rehabili. | 4.69 | 100 | Rating: 4,comment |
| Dr. Emily Turner | Cardiothoracic Surgery | 4.60 | 127 | Rating: 4,comment |
| Dr. Mei Tan | Cardiothoracic Surgery | 4.60 | 127 | Rating: 4,comment |
| Dr. Kimberly Adams | Neonatology | 4.60 | 120 | Rating: 4,comment |
| Dr. Benjamin Taylor | Internal Medicine | 4.59 | 100 | Rating: 4,comment |
| Dr. Khadija Rahman | Pediatric Infectious Diseases | 4.50 | 135 | Rating: 4,comment |
| Dr. Sanjay Singh | Pediatric Infectious Diseases | 4.50 | 135 | Rating: 4,comment |
| Dr. Mia Carter | Pediatric Infectious Diseases | 4.50 | 135 | Rating: 4,comment |
| Dr. Michelle Carter | Pediatric Infectious Diseases | 4.50 | 130 | Rating: 4,comment |
| Dr. Matthew Garcia | Surgery | 4.49 | 95 | Rating: 4,comment |
| Dr. Michael Smith | Orthopedics | 4.49 | 90 | Rating: 4,comment |
| Dr. Ryota Suzuki | Colorectal Surgery | 4.40 | 96 | Rating: 4,comment |
| Dr. Isaiah White | Colorectal Surgery | 4.40 | 96 | Rating: 4,comment |
| Dr. Kareem Ahmed | Colorectal Surgery | 4.40 | 96 | Rating: 4,comment |
| Dr. Nicholas White | Colorectal Surgery | 4.40 | 95 | Rating: 4,comment |
| Dr. Ethan Miller | Ophthalmology | 4.40 | 85 | Rating: 4,comment |
| Dr. Christopher Lee | Dermatology | 4.30 | 80 | Rating: 4,comment |
| Dr. Priya Patel | Pediatric Hematology-Oncolog. | 4.20 | 156 | Rating: 4,comment |

| | | | | | |
|---|---|---|---|---|---|
| 2 | *Doctor Review* | Should display average rating and feedback for each doctor. Also, should show the doctors' list in descending order with respect to their average rating.<br><br>New ratings are listed accordingly using EventQueue.invokeLater(showDoctorReviews). This ensures thread security as well as to avoid issues like inconsistent UI behavior or crashing. | 2,4,5 | Pass | N/A |
|  | | | | | |

Brunel Hospital Management System

**Brunel Hospital Management System**
*Kingston Ln, London, Uxbridge UB8 3PH, UK*

Home | Search Doctor | Add Review

| Doctor Name | Specialization | Average Rating | Number of Reviews | Latest Review |
|---|---|---|---|---|
| Dr. Sophia Anderson | Psychiatry | 4.89 | 140 | Rating: 4,comment |
| Dr. Jessica Davis | Neurology | 4.87 | 151 | Rating: 1,Need to co-operate |
| Dr. Stella Martinez | Child and Adolescent Psychiat. | 4.79 | 142 | Rating: 4,comment |
| Dr. Ayesha Khan | Child and Adolescent Psychiat. | 4.79 | 142 | Rating: 4,comment |
| Dr. Rachel Martinez | Child and Adolescent Psychiat. | 4.79 | 140 | Rating: 4,comment |
| Dr. Olivia Brown | Pediatrics | 4.79 | 130 | Rating: 4,comment |
| Dr. Emily Johnson | Cardiology | 4.79 | 120 | Rating: 4,comment |
| Dr. Ashley Wilson | Obstetrics and Gynecology | 4.69 | 110 | Rating: 4,comment |
| Dr. Ahmad Hassan | Physical Medicine and Rehabili. | 4.69 | 103 | Rating: 4,comment |
| Dr. Ji-hoon Kim | Physical Medicine and Rehabili. | 4.69 | 103 | Rating: 4,comment |
| Dr. Liam Taylor | Physical Medicine and Rehabili. | 4.69 | 103 | Rating: 4,comment |
| Dr. Samuel Taylor | Physical Medicine and Rehabili. | 4.69 | 100 | Rating: 4,comment |
| Dr. Emily Turner | Cardiothoracic Surgery | 4.60 | 127 | Rating: 4,comment |
| Dr. Mei Tan | Cardiothoracic Surgery | 4.60 | 127 | Rating: 4,comment |
| Dr. Kimberly Adams | Neonatology | 4.60 | 120 | Rating: 4,comment |
| Dr. Benjamin Taylor | Internal Medicine | 4.59 | 100 | Rating: 4,comment |
| Dr. Khadija Rahman | Pediatric Infectious Diseases | 4.50 | 135 | Rating: 4,comment |
| Dr. Sanjay Singh | Pediatric Infectious Diseases | 4.50 | 135 | Rating: 4,comment |
| Dr. Mia Carter | Pediatric Infectious Diseases | 4.50 | 135 | Rating: 4,comment |
| Dr. Michelle Carter | Pediatric Infectious Diseases | 4.50 | 130 | Rating: 4,comment |
| Dr. Matthew Garcia | Surgery | 4.49 | 95 | Rating: 4,comment |
| Dr. Michael Smith | Orthopedics | 4.49 | 90 | Rating: 4,comment |
| Dr. Ryota Suzuki | Colorectal Surgery | 4.40 | 96 | Rating: 4,comment |
| Dr. Isaiah White | Colorectal Surgery | 4.40 | 96 | Rating: 4,comment |
| Dr. Kareem Ahmed | Colorectal Surgery | 4.40 | 96 | Rating: 4,comment |
| Dr. Nicholas White | Colorectal Surgery | 4.40 | 95 | Rating: 4,comment |
| Dr. Ethan Miller | Ophthalmology | 4.40 | 85 | Rating: 4,comment |
| Dr. Christopher Lee | Dermatology | 4.30 | 80 | Rating: 4,comment |
| Dr. Priya Patel | Pediatric Hematology-Oncolog. | 4.20 | 156 | Rating: 4,comment |

## Future Work:

Following are some of the future enhancements and modifications that can be added -

1. Enhanced Input Validation: The mechanism to validate inputs will be further enhanced so that the integrity of the data is properly maintained and no invalid or incomplete data are entered. This would include stricter rating input checks, meaningful comments, prevention against duplicate/erroneous entry, and so on.

2. Advanced Filtering and Sorting: Advanced filtering and sorting need to be introduced in which users can sort their reviews based on rating, date, or look for some specific keywords. This would allow the patient and healthcare administrators to identify some feedback rapidly and relevantly, and track trends over time. It would also respond better to critical issues.

3. User Authentication and Role-Based Access Control: A strong authentication system for users should be developed that supports role-based access control. This will limit most of its functionalities to specific authorized users only, like the administrative users having the right

to delete or modify reviews, while a patient can only create and view a review. This would add more security to the system in terms of safeguarding data.

4. User Interface and User Experience Enhancements: Invest in an easier and more engaging UI interface for the improvement of the overall user experience. It allows for layout optimization, readability improvement, and responsiveness to different devices. A more user-friendly design will make the system easier to use and more accessible, hence encouraging greater frequency and increasing engagement.

## Conclusion:

This Java-based application is designed to make possible meaningful interfaces between the doctors and patients to uplift and enhance healthcare services. It has core features such as searching, submission of reviews, and rating aggregation of the doctors in the home menu. For improving effectiveness, the system has to strengthen data validation so that it can have reliability and credibility; also, it has to implement advanced filtering options to improve user experience and introduce user authentication with role-based access control for improved security. And, the UI/UX refinement will be a constant, leaving the system indispensable. It will continue to add resources to the user and stay relevant in the healthcare domain.

## Appendix:

Source Codes –

1. Follwoing is the source code for GUI.java :

```java
import java.awt.Color;
import java.awt.EventQueue;
import java.awt.Font;
import java.awt.Image;
import java.io.File;
import java.io.FileNotFoundException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;
import javax.swing.ImageIcon;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTabbedPane;
import javax.swing.JTable;
import javax.swing.border.EmptyBorder;
import javax.swing.table.DefaultTableModel;

public class GUI extends JFrame {

    private static final long serialVersionUID = 1L;
    private JPanel contentPane;
    private JTable table;
    private JScrollPane scrollPane;
    private JLabel hospital_name;
    private ImageIcon img;
    private JPanel home, search;
    private Map<String, Doctor> doctors = new HashMap<>();
    /*
    @author Nafis
     */
    public static void main(String[] args) throws FileNotFoundException {
        GUI frame = new GUI();
        frame.setVisible(true);
    }

    public GUI() {
        setTitle("Brunel Hospital Management System");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(900, 670);
        setLocationRelativeTo(null);

        contentPane = new JPanel();
        contentPane.setBackground(new Color(245, 255, 250));
        contentPane.setBorder(new EmptyBorder(5, 5, 5, 5));
        setContentPane(contentPane);
        contentPane.setLayout(null);

        img = new ImageIcon(getClass().getResource("logo.png"));
        this.setIconImage(img.getImage());

        ImageIcon bannerIcon = new ImageIcon(getClass().getResource("Hospital.png"));
        Image image = bannerIcon.getImage();
        Image scaledImage = image.getScaledInstance(40, 35, Image.SCALE_SMOOTH);
        ImageIcon scaledIcon = new ImageIcon(scaledImage);
        JLabel bannerLabel = new JLabel(scaledIcon);
        bannerLabel.setBounds(0, 0, 70, 80);
        contentPane.add(bannerLabel);

        hospital_name = new JLabel("Brunel Hospital Management System");
        hospital_name.setBounds(65, 2, 500, 60);
        hospital_name.setFont(new Font("Bernard MT Condensed", Font.BOLD, 20));
        hospital_name.setForeground(Color.BLACK);
        contentPane.add(hospital_name);

        JLabel address = new JLabel("Kingston Ln, London, Uxbridge UB8 3PH, UK");
        address.setBounds(65, 22, 500, 60);
        address.setFont(new Font("Forte", Font.PLAIN, 13));
        address.setForeground(Color.BLACK);
        contentPane.add(address);

        home = new JPanel();
        home.setBounds(235, 100, 600, 340);
        home.setBackground(new Color(245, 255, 250));
        home.setLayout(null);
        contentPane.add(home);

        scrollPane = new JScrollPane();
        scrollPane.setBounds(2, 2, 835, 488);
        home.add(scrollPane);

        table = new JTable();
        table.setBackground(new Color(245, 255, 250));
        scrollPane.setViewportView(table);

        JTabbedPane tp = new JTabbedPane();
        tp.setBounds(20, 85, 840, 520);
```

```
        tp.addTab("Home", home);
        tp.addTab("Search Doctor", new Search(this));
        tp.addTab("Add Review", new AddReview(this));
        contentPane.add(tp);

        try {
            loadDoctors();
              updateDoctorTable(); //The loadDoctors() method reads doctor data from a
doctors.csv file, creates Doctor objects, and stores them in a Map<String, Doctor>. The
updateDoctorTable() method is used to display the loaded doctor data in a table format,
sorting them by average rating.
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }

        EventQueue.invokeLater(this::showDoctorReviews);
        // Used to maintain thread security. Also, since updating occurs, it is useful
        // when to update the GUI after the application has started.
    }

    private void loadDoctors() throws FileNotFoundException {
        File file = new File("C:\\Implementation\\src\\doctors.csv");

        if (!file.exists()) {
            throw new FileNotFoundException("File not found: " + file.getAbsolutePath());
        }

        Scanner scanner = new Scanner(file);
        doctors.clear();

        if (scanner.hasNextLine()) {
            scanner.nextLine();
        }

        while (scanner.hasNextLine()) {
            String line = scanner.nextLine();
            String[] details = line.split(",");

            if (details.length < 5) {
                System.err.println("Skipping invalid line: " + line);
                continue;
            }

            String name = details[0];
            String specialization = details[1];
            double rating;
            int numberOfReviews;
            String comment = details[4];

            try {
                rating = Double.parseDouble(details[2]);
                numberOfReviews = Integer.parseInt(details[3]);
            } catch (NumberFormatException e) {
                System.err.println("Skipping line due to parsing error: " + line);
                continue;
            }

            // Initialize Doctor object with totalRating and numberOfReviews from CSV
              Doctor doctor = new Doctor(name, specialization, rating * numberOfReviews,
numberOfReviews);
            doctor.addReview(new Review((int) rating, comment));
            doctors.put(name, doctor);
        }

        scanner.close();
    }

    // Method to update the table with doctor data
    public void updateDoctorTable() {
        ArrayList<Doctor> doctorList = new ArrayList<>(doctors.values());
                 doctorList.sort((d1,   d2)   ->   Double.compare(d2.getAverageRating(),
d1.getAverageRating()));

        String[] columnNames = { "Doctor Name", "Specialization", "Average Rating", "Number
of Reviews",
                "Latest Review" };

        Object[][] data = new Object[doctorList.size()][5];
        // Attribute names for the table

        for (int i = 0; i < doctorList.size(); i++) {
            Doctor doctor = doctorList.get(i);
            data[i][0] = doctor.getName();
            data[i][1] = doctor.getSpecialization();
            data[i][2] = String.format("%.2f", doctor.getAverageRating());
            data[i][3] = doctor.getNumberOfReviews();
            Review latestReview = doctor.getLatestReview();
            data[i][4] = latestReview != null ? latestReview.toString() : "No reviews yet";
        }

        DefaultTableModel model = new DefaultTableModel(data, columnNames);
        table.setModel(model);
    }

    public void showDoctorReviews() {
        updateDoctorTable();
        scrollPane.revalidate();
        scrollPane.repaint();
    }
    // Method to display doctor reviews. Also, scrollpane has been refreshed and
    // repainted by inbuilt methods.

    public Doctor getDoctor(String name) {
        return doctors.get(name);
    }
    // Returns a Doctor object from the doctors map by name.

    public Map<String, Doctor> getDoctors() {
        return doctors;
    }
    // Method to get all doctors (Returns the entire doctors map.)
}
```

2. Follwoing is the source code for Search.java :

```
import java.awt.Color;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.KeyEvent;
import java.awt.event.KeyListener;
import java.io.FileWriter;
import java.io.IOException;
import java.util.ArrayList;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.table.DefaultTableModel;

public class Search extends JPanel {
    private JTable doctorTable;
    private JTextField ratingTextField;
    private JTextArea commentTextArea;
    private Font f;
    private JButton searchButton, clearButton;
    private GUI parentGUI;

    public Search(GUI parent) {
        this.parentGUI = parent;
        setLayout(null);
        setBackground(new Color(255, 242, 242));

        // Parent GUI is used for it is a convenient way and code reuseablity is achieved

        f = new Font("Franklin Gothic Demi", Font.BOLD, 14);

        JLabel searchLabel = new JLabel("Search Doctor:");
        searchLabel.setBounds(10, 10, 100, 25);
        searchLabel.setFont(new Font("Franklin Gothic Demi", Font.PLAIN, 14));
        add(searchLabel);

        JTextField searchTextField = new JTextField();
        searchTextField.setBounds(115, 11, 495, 25);
        searchTextField.setFont(new Font("Franklin Gothic Demi", Font.PLAIN, 14));
        add(searchTextField);

        searchButton = new JButton("Search");
        searchButton.setFont(new Font("Franklin Gothic Demi", Font.PLAIN, 14));
        searchButton.setBounds(620, 11, 100, 25);
        add(searchButton);

        clearButton = new JButton("Clear");
        clearButton.setFont(new Font("Franklin Gothic Demi", Font.PLAIN, 14));
        clearButton.setBounds(720, 11, 100, 25);
        add(clearButton);

        JScrollPane scrollPane = new JScrollPane();
        scrollPane.setBounds(10, 50, 812, 425);
        add(scrollPane);

        doctorTable = new JTable();
        doctorTable.setBackground(new Color(255, 242, 242));
        scrollPane.setViewportView(doctorTable);

        // Similar to Parentclass

        searchButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                String searchQuery = searchTextField.getText();
                updateDoctorTable(searchQuery);
            }
        }); // Search button invokes in two way. One case can be using search button, the
other case can be, using key pressed. In maintaining bullet proof coding, it has been
assured. So, both of the listeners work properly.

        searchTextField.addKeyListener(new KeyListener() {
            @Override
            public void keyTyped(KeyEvent e) {
                // Not used
            }

            @Override
            public void keyPressed(KeyEvent  e) {
                if (e.getKeyCode() == KeyEvent.VK_ENTER)
         {
                    String searchQuery = searchTextField.getText();
                    updateDoctorTable(searchQuery);
                }
            }

            @Override
            public void keyReleased(KeyEvent e) {
                // Not used
            }
        });

        clearButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
             searchTextField.setText("");
             updateDoctorTable("NULL");
            }
        }); // Clear button to maintain UI experience cleen and smooth
    }

    private void updateDoctorTable(String searchQuery) {
        ArrayList<Doctor> doctorList = new ArrayList<>(parentGUI.getDoctors().values());
                                        doctorList.removeIf(doctor              ->
!doctor.getName().toLowerCase().contains(searchQuery.toLowerCase()));
        String[] columnNames = { "Doctor Name", "Specialization", "Average Rating", "Number
of Reviews", "Latest Review" };
        Object[][] data = new Object[doctorList.size()][5];

        for (int i = 0; i < doctorList.size(); i++) {
            Doctor doctor = doctorList.get(i);      // Gets the current doctor

            data[i][0] = doctor.getName();
            data[i][1] = doctor.getSpecialization();
            data[i][2] = String.format("%.2f", doctor.getAverageRating());
            data[i][3] = doctor.getNumberOfReviews();
            data[i][4] = doctor.getLatestReview().toString();
        }

        DefaultTableModel model = new DefaultTableModel(data, columnNames);
        doctorTable.setModel(model);
    }
```

```
}
```

3. Follwoing is the source code for AddReview.java :

```
import java.awt.Color;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.KeyListener;
import java.io.FileWriter;
import java.io.IOException;
import java.awt.event.KeyEvent;
import java.util.ArrayList;
import javax.swing.JButton;
import javax.swing.JComboBox;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.JTextArea;
import javax.swing.JTextField;
import javax.swing.table.DefaultTableModel;

public class AddReview extends JPanel {
    private JTable doctorTable;
    private JTextField ratingTextField;
    private JTextArea commentTextArea;
    private Font f;
    private JButton searchButton, clearButton;
    private GUI parentGUI;

    public AddReview(GUI parent) {
        this.parentGUI = parent;
        setLayout(null);
        setBackground(new Color(199, 219, 222));

        f = new Font("Franklin Gothic Demi", Font.BOLD, 14);

        JLabel searchLabel = new JLabel("Search Doctor:");
        searchLabel.setBounds(10, 10, 100, 25);
        searchLabel.setFont(new Font("Franklin Gothic Demi", Font.PLAIN, 14));
        add(searchLabel);

        JTextField searchTextField = new JTextField();
        searchTextField.setBounds(115, 11, 495, 25);
        searchTextField.setFont(new Font("Franklin Gothic Demi", Font.PLAIN, 14));
        add(searchTextField);

        searchButton = new JButton("Search");
        searchButton.setFont(new Font("Franklin Gothic Demi", Font.PLAIN, 14));
        searchButton.setBounds(620, 11, 100, 25);
        add(searchButton);

        clearButton = new JButton("Clear");
        clearButton.setFont(new Font("Franklin Gothic Demi", Font.PLAIN, 14));
        clearButton.setBounds(720, 11, 100, 25);
        add(clearButton);

        JLabel ratingLabel = new JLabel("Rating:");
        ratingLabel.setBounds(10, 260, 80, 25);
```

```java
ratingLabel.setFont(new Font("Franklin Gothic Demi", Font.PLAIN, 14));
add(ratingLabel);

ratingTextField = new JTextField();
ratingTextField.setBounds(115, 260, 495, 25);
ratingTextField.setFont(new Font("Franklin Gothic Demi", Font.PLAIN, 14));
add(ratingTextField);

JLabel commentLabel = new JLabel("Comment:");
commentLabel.setBounds(10, 295, 80, 25);
commentLabel.setFont(new Font("Franklin Gothic Demi", Font.PLAIN, 14));
add(commentLabel);

commentTextArea = new JTextArea();
commentTextArea.setFont(new Font("Franklin Gothic Demi", Font.PLAIN, 14));

JScrollPane commentScrollPane = new JScrollPane(commentTextArea);
commentScrollPane.setBounds(115, 295, 495, 50);
add(commentScrollPane);

JButton submitButton = new JButton("Submit Review");
submitButton.setFont(f);
submitButton.setBounds(620, 260,200, 25);
add(submitButton);

JScrollPane scrollPane = new JScrollPane();
scrollPane.setBounds(10, 50, 812, 200);
add(scrollPane);

doctorTable = new JTable();
doctorTable.setBackground(new Color(199, 219, 222));
scrollPane.setViewportView(doctorTable);

searchButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        String searchQuery = searchTextField.getText();
        updateDoctorTable(searchQuery);
    }
});

searchTextField.addKeyListener(new KeyListener() {
    @Override
    public void keyTyped(KeyEvent e) {
        // Not used
    }

    @Override
    public void keyPressed(KeyEvent  e) {
        if (e.getKeyCode() == KeyEvent.VK_ENTER)
 {
            String searchQuery = searchTextField.getText();
            updateDoctorTable(searchQuery);
        }
    }

    @Override
    public void keyReleased(KeyEvent e) {
        // Not used
    }
});
```

```
        submitButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                addReview();
                updateDoctorTable("NULL");
                searchTextField.setText("");
                ratingTextField.setText("");
                commentTextArea.setText("");
            }
        });

        clearButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
             searchTextField.setText("");
             ratingTextField.setText("");
             commentTextArea.setText("");
             updateDoctorTable("NULL");
            }
        });
    }
private void updateDoctorTable(String searchQuery) {
        ArrayList<Doctor> doctorList = new ArrayList<>(parentGUI.getDoctors().values());
                                                doctorList.removeIf(doctor            ->
!doctor.getName().toLowerCase().contains(searchQuery.toLowerCase()));

        String[] columnNames = { "Doctor Name", "Specialization", "Average Rating", "Number
of Reviews", "Latest Review" };
        Object[][] data = new Object[doctorList.size()][5];

        for (int i = 0; i < doctorList.size(); i++) {
            Doctor doctor = doctorList.get(i);
            data[i][0] = doctor.getName();
            data[i][1] = doctor.getSpecialization();
            data[i][2] = String.format("%.2f", doctor.getAverageRating());
            data[i][3] = doctor.getNumberOfReviews();
            data[i][4] = doctor.getLatestReview().toString();
        }

        DefaultTableModel model = new DefaultTableModel(data, columnNames);
        doctorTable.setModel(model);
    }
    // Till this, the code is very much similar to Search.java


    private void addReview() {
        int selectedRow = doctorTable.getSelectedRow();

         // Here one thing is that, by searching, there is a possibility of having two
doctors with the same name or nickname. In this case, specification determines their
differences and the patient gets to select which of the doctor he/she wants to select and
review. For, he must review by selecting the particular row.

        if (selectedRow == -1) {
            System.err.println("No doctor selected");
            return;
        }

        String selectedDoctor = (String) doctorTable.getValueAt(selectedRow, 0);
        String ratingText = ratingTextField.getText();
        String commentText = commentTextArea.getText();
        int rating;

        // System.out.println(selectedDoctor);

        try {
            rating = Integer.parseInt(ratingText);
            if (rating < 1 || rating > 5) {
                        JOptionPane.showMessageDialog(null,  "You  must  rate  within  1-5
!","WARNING!!!", JOptionPane.WARNING_MESSAGE);
                // System.err.println("Invalid rating: " + rating);
                return;
            }
        } catch (NumberFormatException e) {
            // System.err.println("Invalid rating input: " + ratingText);
            JOptionPane.showMessageDialog(null, "You must rate within 1-5 !","WARNING!!!",
JOptionPane.WARNING_MESSAGE);
            return;
        }

        // Retrieve the doctor object from the parent GUI
        Doctor doctor = parentGUI.getDoctor(selectedDoctor);
        if (doctor != null) {
            // Add the new review to the doctor
            doctor.addReview(new Review(rating, commentText));
            // Update the doctor table and refresh the view
            parentGUI.updateDoctorTable();
            parentGUI.showDoctorReviews();
            // Clear input fields
            ratingTextField.setText("");
            commentTextArea.setText("");
             JOptionPane.showMessageDialog(null, "Review Added Successfully!","Success in
Submission !", JOptionPane.INFORMATION_MESSAGE);

            // Append the review to the CSV file
           try (FileWriter writer = new FileWriter("C:\\Implementation\\src\\doctors.csv",
true)) {
                       writer.write(String.format("\n%s,%s,%.2f,%d,%s",  doctor.getName(),
doctor.getSpecialization(),    doctor.getAverageRating(),    doctor.getNumberOfReviews(),
commentText));
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }
}
```

4. Follwoing is the source code for Doctor.java :

```
import java.lang.reflect.Method;
import java.util.ArrayList;
import java.util.List;

public class Doctor {
    private String name;
    private String specialization;
    private double totalRating;
    private int numberOfReviews;
    private List<Review> reviews;
```

```
        public   Doctor(String   name,   String   specialization,   double   totalRating,   int
    numberOfReviews) {
            this.name = name;
            this.specialization = specialization;
            this.totalRating = totalRating;
            this.numberOfReviews = numberOfReviews;
            this.reviews = new ArrayList<>();
        }

        public void addReview(Review review) {
            reviews.add(review); // Add the new review to the list
            totalRating += review.getRating();
            numberOfReviews++; // Increment the number of reviews
        } // Method to add a review for the doctor

        public String getName() {
            return name;
        }

        public String getSpecialization() {
            return specialization;
        }

        public double getAverageRating() {
            return numberOfReviews > 0 ? totalRating / numberOfReviews : 0;
        }

        public int getNumberOfReviews() {
            return numberOfReviews-1;
        }

        public Review getLatestReview() {
            // Return the last review if there are any, otherwise return null
            return reviews.size() > 0 ? reviews.get(reviews.size() - 1) : null;
        }
        // Method to get the latest review for the doctor
    }
```

5. Follwoing is the source code for Review.java :

```
public class Review {
    private int rating;
    private String comment;

    public Review(int rating, String comment) {
        this.rating = rating;
        this.comment = comment;
    } //constractor

    public int getRating() {
        return rating;
    }

    public String getComment() {
        return comment;
    } //getters

    @Override
    public String toString() {
        return "Rating: " + rating + "," + comment;
    }
 }
```