**SHRI G.S. INSTITUTE OF TECHNOLOGY & SCIENCE, INDORE**

**Department of Information Technology**



**SUBJECT CODE: IT38504**

**SUBJECT NAME: WEB ENGINEERING**

**NAME: INSTRUCTOR:**

**KURNIKA BHAWEL DR. LALIT PUROHIT SIR**

**UPENDRA SINGH SIR**

**ENROLLMENT NO:**

**0801IT211045**

**ASSIGNMENT - 01**

**Web Interface:** The first step involved creating a user-friendly web interface where the HR team  could input job requirements. This interface is to be developed using HTML and CSS, ensuring it  is responsive and accessible on various devices. JavaScript is to be used to add interactivity,  allowing for dynamic addition or removal of criteria fields such as required technical skills  (programming languages, platform, etc.) years of experience, and educational qualifications.

**Candidate Application Portal:** A separate portal for candidates to apply for positions is to be  developed. It included a form where applicants can provide their details su as their qualifications,  upload their resume, and specify their availability. JavaScript is to be used to ensure the form data  is correctly captured and stored for processing.

**Resume Upload and Parsing:** Candidates can upload their resumes through a separate web page.  The system should accept resumes in PDF and Word formats. JavaScript, along with a third-party  library, is to be use to parse the uploaded resumes, extracting key information such as technical  skills, previous job roles, and education.

**Matching Algorithm:** The core of the system is a matching algorithm that compare the parsed  resume data against the job requirements set by the HR team. Each resume is to be scored based  on how well it matches the requirements, including the presence of specific skills, relevance of  past experience, and educational background.

**Validation and Feedback:** To ensure accuracy and relevance, the system includes validation  checks and feedback mechanisms. If the HR team enteres contradictory or impossible criteria (e.g.,  requiring 10 years of experience in a programming language that had only been around for 5 years),  the system will alert them and suggest corrections. Candidates should receive automated feedback  on missing or weak areas in their resumes based on the job criteria.

**Lab Assignment #1 (HTML and CSS)**

Write the HTML and CSS part of above case study.

**ASSIGNMENT - 01**

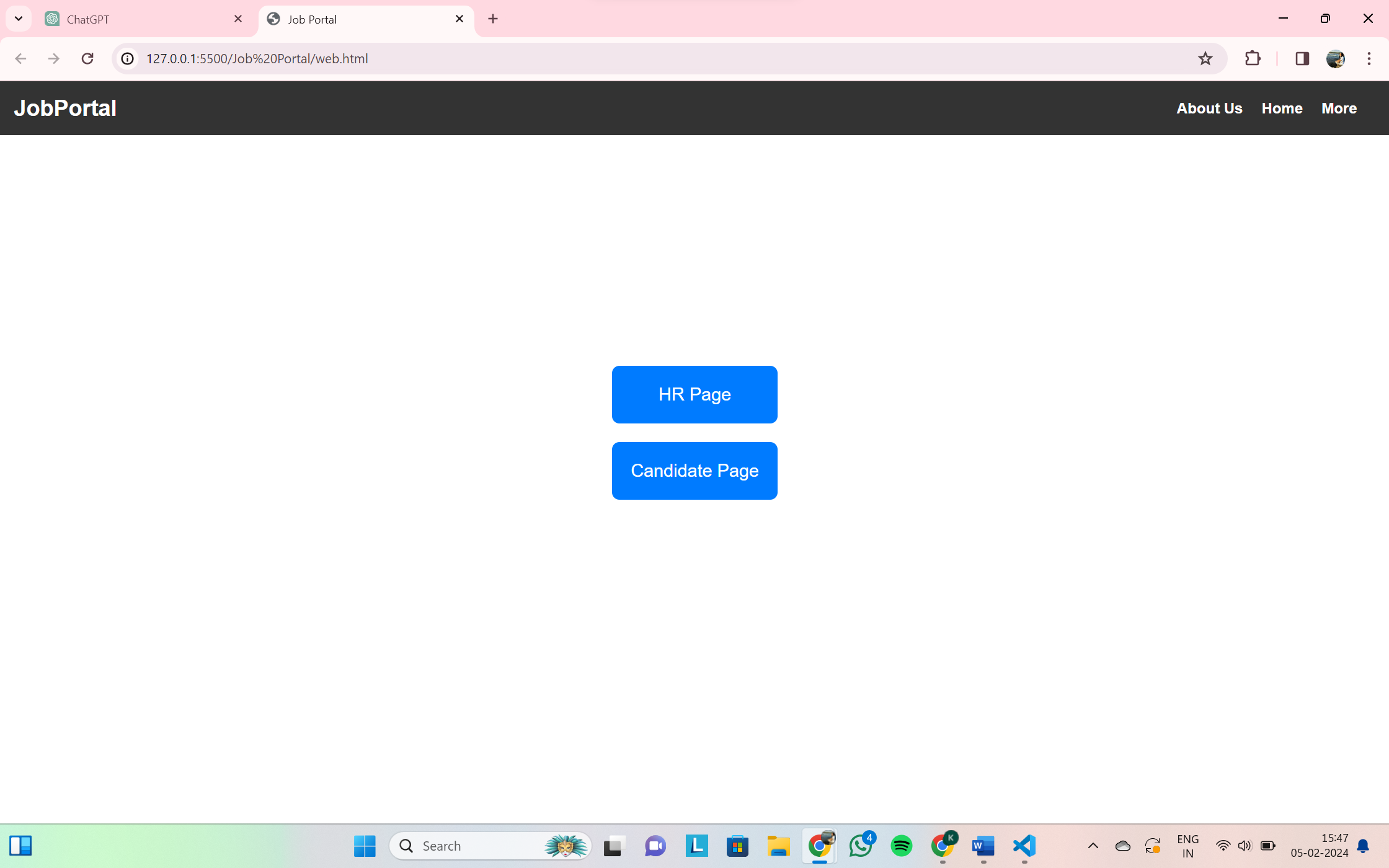
1. **Objectives**

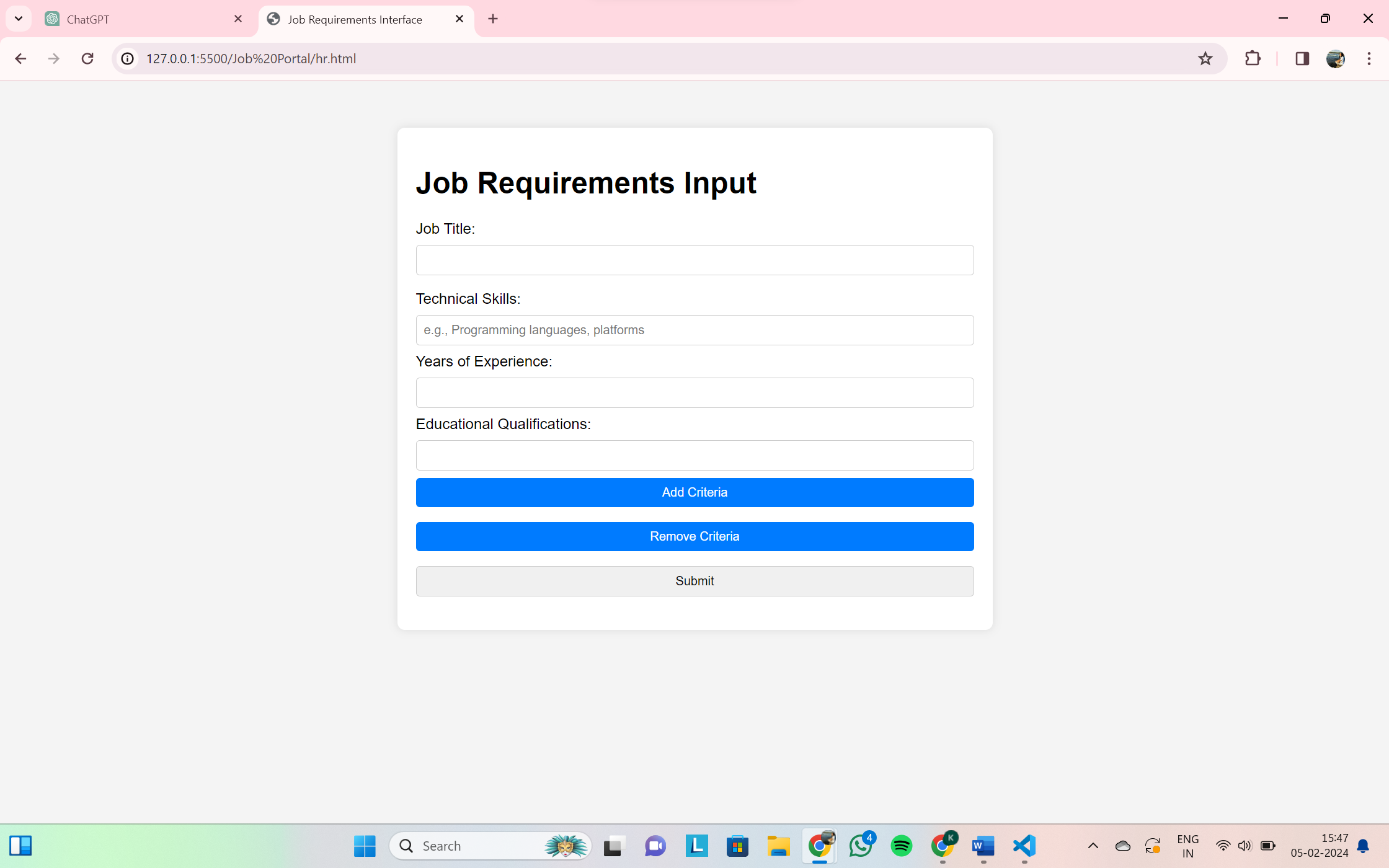
Create a responsive web-based recruitment system using HTML and CSS to streamline HR processes. Develop an intuitive interface for job requirement input and a Candidate Application Portal for efficient applicant submissions. Design a visually appealing and accessible platform without the need for external technologies, focusing solely on HTML and CSS for seamless development.

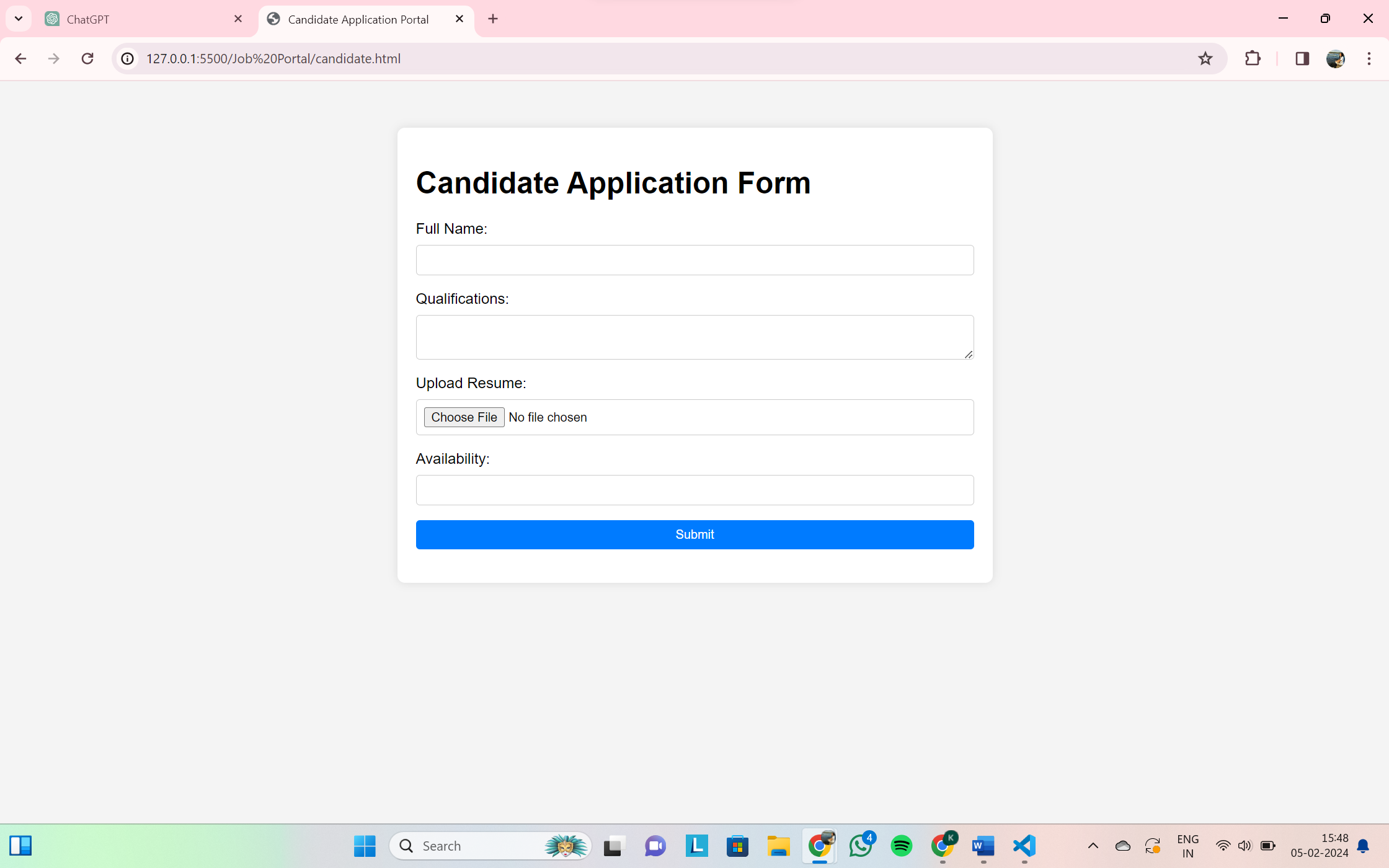
1. **Problem Statement**

Current recruitment processes lack an efficient and user-friendly digital platform, causing inefficiencies for HR teams and candidates. Manual job requirement input is time-consuming, and existing application portals lack a standardized approach. Resumes are often submitted in non-uniform formats, hindering effective parsing. There is a need for a streamlined and responsive web-based recruitment system developed in HTML and CSS to address these challenges and enhance the overall recruitment experience.

1. **Outcomes**

****

****

****

1. **Software & Hardware Requirements**

* Vs Code

1. **Theory-Concept**

* HTML: A markup language that defines the structure and meaning of web content. HTML is used to create the content of a page, such as written text. HTML files can include CSS code.
* CSS: A style sheet language that is responsible for the design or style of a website. CSS translates the display and design of HTML elements. CSS is made up of tags that surround content.

1. **Technology/Tool**

* HTML
* CSS

1. **DESIGN/EXECUTION STEPS**

* HTML and CSS codes in attached zip file – 0801IT211045\_Kurnika\_Bhawel.zip

1. **CONCLUSION/ANALYSIS**

Webpage is successfully created using HTML and CSS.

1. **Oral Questions**

What are the essential HTML tags you would use to create the user interface for the HR team's input form?

How would you ensure that the web interface is responsive and accessible on various devices using CSS?

Assignment-02

# OBJECTIVES:

# • Develop a user-friendly web interface tailored for HR to input job requirements effectively.

# • Establish a dedicated portal for candidates to seamlessly apply for positions.

# • Implement resume upload and parsing capabilities for enhanced candidate profiling.

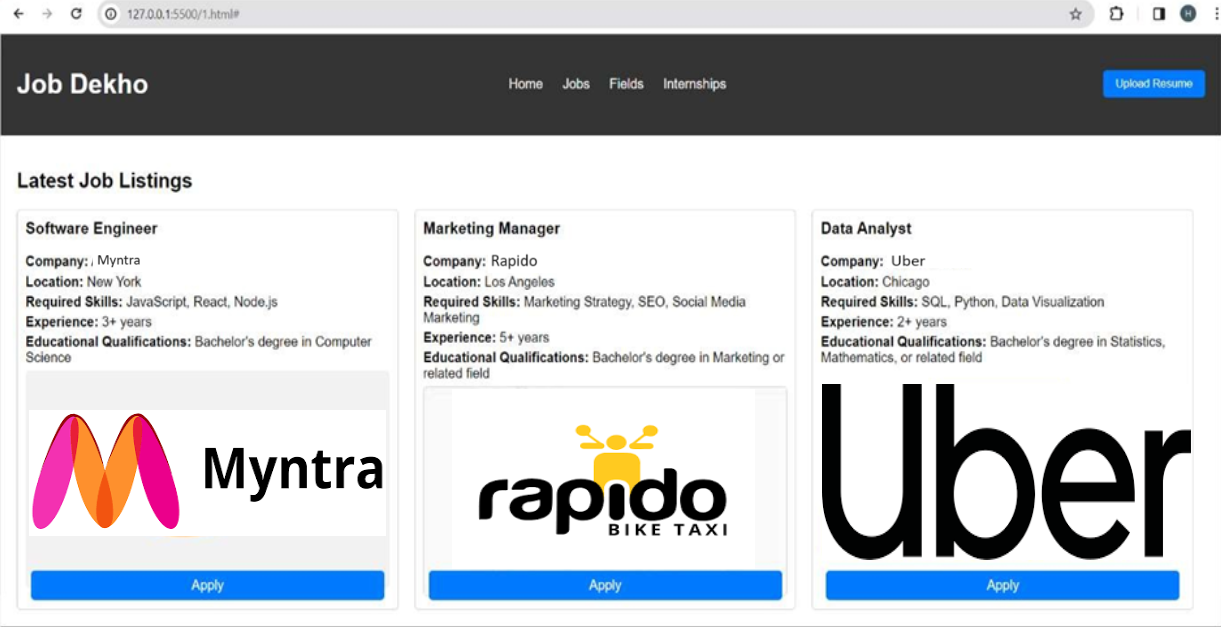
# • Design and deploy a sophisticated matching algorithm to evaluate candidate resumes against job specifications.

# • Integrate robust validation checks and feedback mechanisms to ensure accuracy and relevance throughout the process.

# PROBLEM STATEMENT:

Create a comprehensive system enabling HR to input job listings, candidates to apply, upload resumes, parse them, and match against job requirements seamlessly.

1. **OUTCOMES:**



1. **SOFTWARE & HARDWARE REQUIREMENTS:**

* Software: Web browser, VS Code.
* Hardware: Laptop with internet connectivity.

# THEORY-CONCEPT:

# HTML, CSS, and JavaScript for front-end development.

# TECHNOLOGY/TOOL:

* HTML
* CSS
* JavaScript.

# DESIGN/EXECUTION STEPS:

JavaScript code in attached zip file – 0801IT211045\_Kurnika\_Bhawel.zip

# TEST CASES:

# Tested the web interface on various devices to ensure responsiveness.

# CONCLUSION/ANALYSIS:

# The implemented system simplifies the recruitment process for both HR and candidate.

1. **Oral Questions**

How would you implement the functionality to accept resume uploads in PDF and Word formats using HTML and JavaScript?

Can you provide an example of how JavaScript would be used to validate input fields in the HR team's input form?

**Assignment-03**

**Case Study**

**Web Interface**: The first step involved creating a user-friendly web interface where the HR team could input job requirements. This interface is to be developed using HTML and CSS, ensuring it is responsive and accessible on various devices. JavaScript is to be used to add interactivity, allowing for dynamic addition or removal of criteria fields such as required technical skills (programming languages, platform, etc.) years of experience, and educational qualifications.

**Candidate Application Portal**: A separate portal for candidates to apply for positions is to be developed. It included a form where applicants can provide their details such as their qualifications, upload their resume, and specify their availability. JavaScript is to be used to ensure the form data is correctly captured and stored for processing.

**Resume Upload and Parsing:** Candidates can upload their resumes through a separate web page. The system should accept resumes in PDF and Word formats. JavaScript, along with a third-party library, is to be use to parse the uploaded resumes, extracting key information such as technical skills, previous job roles, and education.

**Matching Algorithm:** The core of the system is a matching algorithm that compare the parsed resume data against the job requirements set by the HR team. Each resume is to be scored based on how well it matches the requirements, including the presence of specific skills, relevance of past experience, and educational background.

**Validation and Feedback:** To ensure accuracy and relevance, the system includes validation checks and feedback mechanisms. If the HR team enters contradictory or impossible criteria (e.g., requiring 10 years of experience in a programming language that had only been around for 5 years), the system will alert them and suggest corrections. Candidates should receive automated feedback on missing or weak areas in their resumes based on the job criteria.

**Lab Assignment #3 (CGI)**

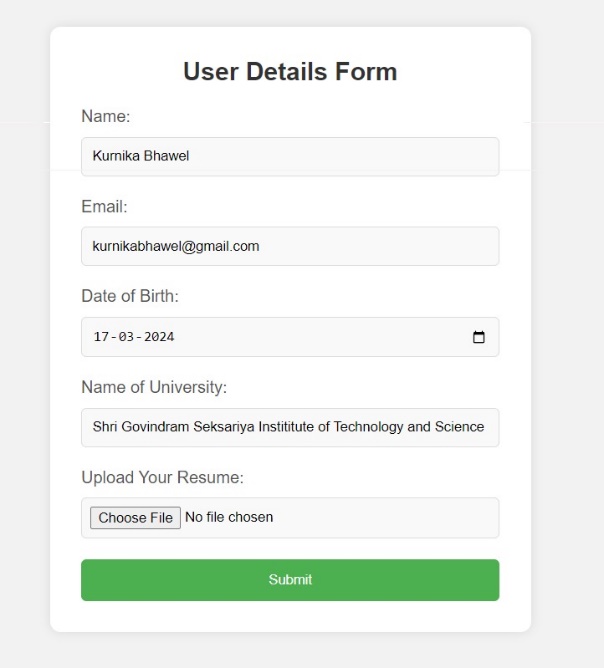
**Write the CGI part of the above case study.**

**Assignment-03**

1. **OBJECTIVES**

The CGI component of the job recruitment portal focuses on elevating the candidate experience. Through a user-friendly Candidate Application Portal, we aim to streamline the application process using HTML, CSS, and JavaScript. This involves facilitating efficient resume uploads in PDF and Word formats, with JavaScript and a third-party library parsing key information. Additionally, interactive features ensure accurate data capture and storage. The implementation includes an automated feedback system, providing candidates with insights into missing or weak resume areas based on job criteria. Overall, the CGI module aims to optimize the candidate journey, making the application process seamless, interactive, and insightful.

1. **PROBLEM STATEMENT**  
   The CGI component of the job recruitment portal faces challenges in providing a seamless candidate experience. Existing issues include a non-user-friendly interface, difficulties in resume upload, limited format support, and a lack of interactive features and feedback mechanisms. The CGI module needs to address these challenges by enhancing user-centric design, optimizing resume parsing, and implementing interactive functionalities for a more efficient and informative candidate application process.
2. **OUTCOMES**

**4. SOFTWARE & HARDWARE REQUIREMENTS**

**Software Requirements:**

* Web server with CGI support (e.g., Apache HTTP Server)
* Python interpreter
* Text editor or IDE for coding

**Hardware Requirements:**

* Computer or server to host the web server software

**5. THEORY-CONCEPT**

The CGI (Common Gateway Interface) is a standard protocol for web servers to execute programs that generate web content dynamically. In this assignment, CGI scripts written in Python will handle form submissions, validate user input, parse uploaded resumes, implement a matching algorithm, and provide feedback to users.

**6. TECHNOLOGY/TOOL**

The technologies/tools required for this assignment include:

* HTML, CSS, and JavaScript for frontend development
* Python for backend scripting (CGI)

**7. DESIGN/EXECUTION STEPS**

* CGI codes in attached zip file – 0801IT211045\_Kurnika\_Bhawel.zip

**9. CONCLUSION/ANALYSIS**

In conclusion, the development of CGI scripts for the web-based job management and candidate application system is essential for its functionality and usability. By implementing features such as validation checks, resume parsing, matching algorithm, and feedback mechanisms, the system can efficiently handle job requirements and candidate applications, ensuring accuracy and relevance in the recruitment process.

This assignment provides practical experience in CGI scripting, web development, and system design, enhancing students' skills in backend programming and web-based application development.

**10. Oral Questions**

1. What is CGI, and how does it relate to web development?
2. How does the CGI process handle user input from web forms?

**Assignment – 04**

**Q1. Let an associative array stores roll number and names of students of your class. If**

**name be the primary key and roll numbers be numeric digits then write a perl program**

**which prints roll number and names of all the students reverse sorted according to the roll numbers.**

**Associative arrays consist - like dictionaries of (key, value) pairs, such that each possible key appears at most once in the collection. Any key of the dictionary is associated (or mapped) to a value. The values of a dictionary can be any type of Python data. So, dictionaries are unordered key-value-pairs. In python, following code is used to define the hash Table :-**

**Sample1 = {“Key1”: 8\_550\_405,**

**“Key2”: 3\_971\_883,**

**“Key3”: 2\_731\_571,**

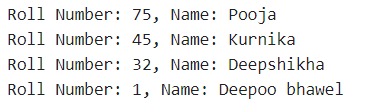
**“Key4”: 2\_720\_546,**

**“Key5”: 667\_137}**

**Define the dictionary for students of your class (See students\_details.txt) and write the above**

**code.**

**Output:**



**Q2. Write a CGI program for users to facilitate the purchase of movie tickets. The CGI**

**should ask users for name, gender, age, movie (from a list) & type of ticket booking**

**(Balcony, VIP etc...) .**

**The application should have following functionality:**

**(i) Echo back the name and age along with a message stating the price of movie.**

**(ii) The price is determined by the age specified by the user in the HTML form.**

** if the age is greater than 17 & less than 45, and the user is college student than 10%**

**discount on movie ticket price is to be considered.**

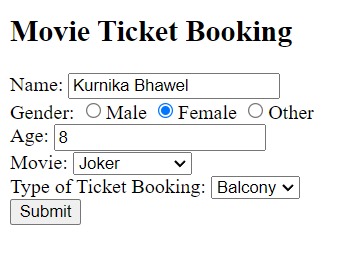
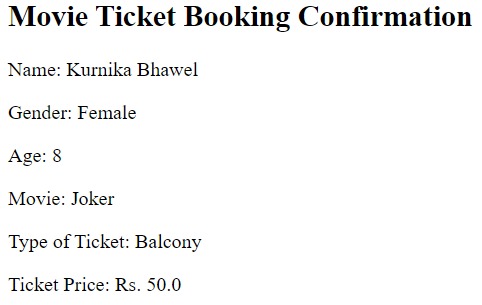
** if the age is greater than 60, than 5% discount on the movie ticket price is to be**

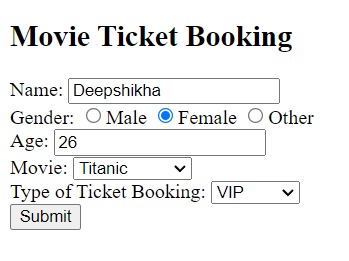
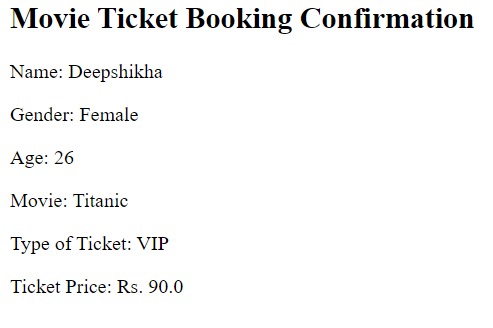
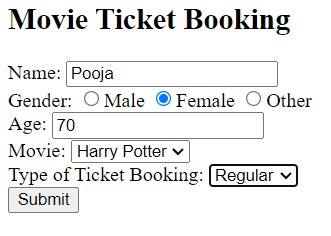
**considered.**

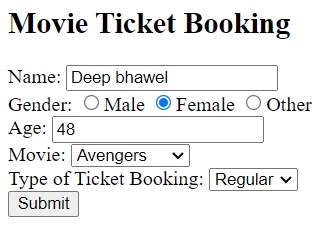
** If the user’s age is less than 10 years old, than 50% discount on the movie ticket price will be applicable.**

** For everyone else, the actual ticket price of Rs 100 is applicable.**

**Output:**

**OBJECTIVES:**

1. Develop a Perl program to print roll numbers and names of students in reverse order based on roll numbers.
2. Implement a CGI program for facilitating the purchase of movie tickets with specified functionalities.

**PROBLEM STATEMENT:**

1. Write a Perl program to sort and print roll numbers and names of students in reverse order based on roll numbers stored in an associative array.
2. Develop a CGI program that asks users for their name, gender, age, selected movie, and type of ticket booking, and echo back the name and age along with the price of the movie ticket based on the specified criteria.

**OUTCOMES:**

1. Sorted list of roll numbers and names of students based on roll numbers in reverse order.
2. CGI program providing a user-friendly interface for purchasing movie tickets with dynamic pricing based on user's age and other criteria.

**SOFTWARE & HARDWARE REQUIREMENTS:**

* Software: Perl interpreter, Web server with CGI support (e.g., Apache), Text editor
* Hardware: Computer system with internet access

**THEORY-CONCEPT:**

* Perl programming for data manipulation and sorting.
* CGI (Common Gateway Interface) for creating dynamic web applications.
* Handling user inputs and generating dynamic content based on specified criteria.

**TECHNOLOGY/TOOL:**

* Perl programming language
* CGI (Common Gateway Interface)
* HTML for form inputs

**DESIGN/EXECUTION STEPS:**

Q1: Read data from students\_details.txt and store it in an associative array.

* + Sort the array based on roll numbers in reverse order.
  + Print roll numbers and names accordingly.

Q2: Create an HTML form to capture user inputs (name, gender, age, movie, ticket type).

* + Implement CGI script to process form data and calculate movie ticket price based on age and other criteria.
  + Echo back the user's name, age, and calculated ticket price.

**TEST CASES:**

* Q1: Verify that the Perl program reads student details correctly and sorts roll numbers and names in reverse order.
* Q2: Test CGI program with different age groups to ensure correct calculation of movie ticket price.

**CONCLUSION/ANALYSIS:**

* Q1: Perl program successfully sorts and prints student details in reverse order based on roll numbers.
* Q2: CGI program effectively processes user inputs and calculates movie ticket prices based on specified criteria.

**Oral Questions for Given Problem Statements:**

1. What is the purpose of CGI in web development?
2. What are the different ways to handle user inputs in a CGI program?
3. How does a web server interact with CGI scripts?

# ASSIGNMENT-05, 06

# Q.1 Write a simple servlet program to say hello to user. Put proper comments in your program. Also write the procedure to create war file.

# Procedure to create war file.

Compile the servlet Java file (DemoServlet .java) using the Java compiler (javac). Make sure you have the servlet API JAR file (servlet-api.jar) in your classpath.

Move the compiled servlet class file (HelloUserServlet.class) into the classes directory.

Create a web.xml file in the WEB-INF directory. This file defines the servlet mapping and other configurations for your web application.

Package your web application into a WAR file. Navigate to the parent directory of the WEB-INF directory and execute the following command:

# jar -cvf YourAppName.war

Replace YourAppName with the desired name of your WAR file.

Your WAR file (YourAppName.war) is now ready for deployment to Apache Tomcat.

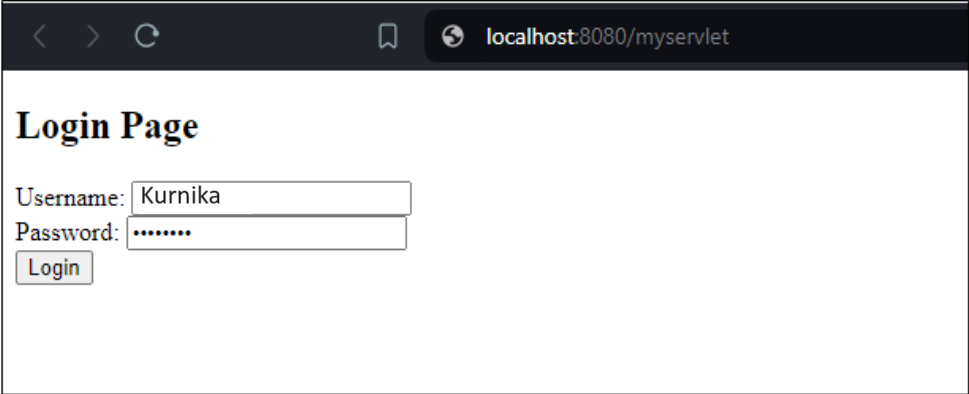


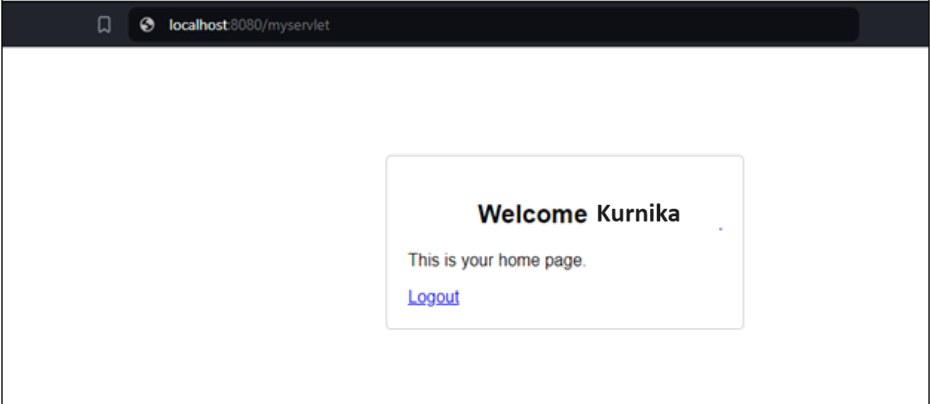
# Q.2 Modify the above program so that :

* **On first request to servlet, login page is generated for the user.**

# When user gets successfully login, his/her name is displayed on the homepage.

**OUTPUT**

****

****

# Q.3 For this assignment, you will

**Write a JSP that takes the user’s name and age from a form.**

# Echo back the name and age along with a message stating the price of movie tickets.

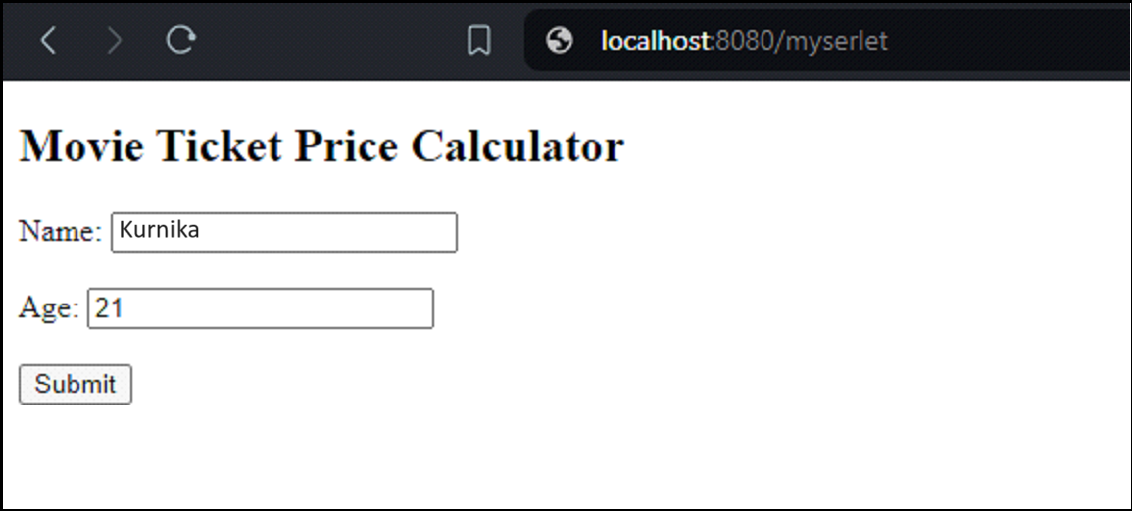
* **The price is determined by the age passed to the JSP.**

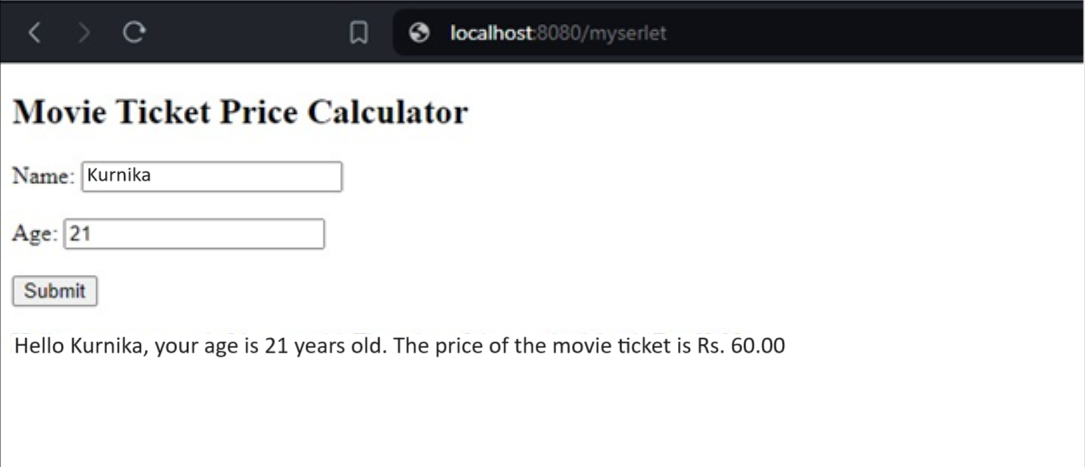
# If the age is greater than 62, the movie ticket price is Rs.45.00.

* **If the user is less than 10 years old, the price is Rs. 30.00.**

# For everyone else, the price is Rs. 60.00.

Output:





**Q.4  MVC and Session Tracking**

**Design a MVC model based web application for college students with following functionality.**

**Accessing Your Servlet:**

**Users should be able to access your application using servlet or directly login.jsp.**

**login.jsp**

**• Captures login and password and passes to the servlet as parameters.**

**• If servlet sees that that there are incoming parameter values for userid and password, it means a new user is logging in.**

**Servlet responsibility [when userid and passwd parameters exist]**

**• DO: invalidate any session information, before checking for valid user.**

**• If NOT valid user, redirect to login.jsp and include phrase: invalid user on the page somewhere**

**• If valid user:**

**o store userid and username in session object.**

**o redirect to home.jsp**

**Registration.jsp**

**If user is a new user, the form has link for new user to get registered. The registration form asks for following details from user**

**- First, middle and last name**

**-  Date of birth**

**-  Address**

**- Phone number**

**-  Pin number**

**-  Department name (As drop down list)**

**-  Roll number (As three drop down list for – Group such as AB/BA/ABRR/BARR,  roll number series year wise such as 18/28/38/48 and three digit roll number starting from 000 to 999)**

**- Year and semester (Drop down list)**

**-  Extra-curricular activity (Select from the list given as checkboxes)**

**home.jsp**

**-Display welcome message to user**

**-Give options to user as follows**

**\* Enter marks details semester wise. (marks\_details.jsp)**

**\* Check aggregate of marks of all semesters. (check\_aggregate.jsp)**

**\* Edit the registration details (registration.jsp with editable text)**

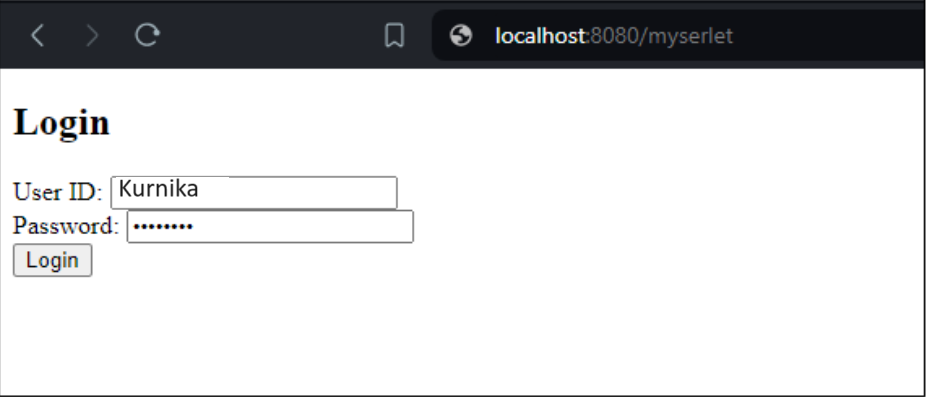
**\* List the subjects of current semester ( From scheme table. All the subjects are stored in the database in the form of scheme table) (list\_subjects.jsp)**

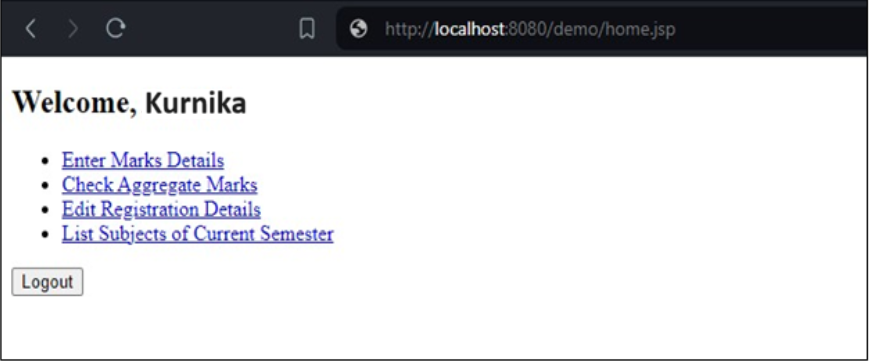
**logout.jsp**

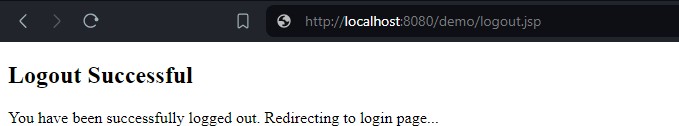
**Called when user click on logout button displayed on home page of user. This terminates the user session and calls login.jsp.**

**You have to maintain user session for performing all the activity.**

**Extra credit:- After logout the students should not be able to go back to home page by pressing back button.**

****

****



**Q.5 Let’s create a movie library application. Here are some screen mocks to help:**

**A movie should have the following information:**

* **Title**
* **Release Date**
* **Director**
* **Actors**
* **Category**
* **Summary**
* **User Comments**

**We need to welcome the user to the application.**

* **If this is the first time they’ve been there, it should ask for their name.**
* **It should remember the name of the last user and welcome them by name when they come back.**
* **It should also say “If you aren’t {name}, click here” and provide a page to either pick an existing user (so keep track of everyone that’s been here) or enter a new name.**

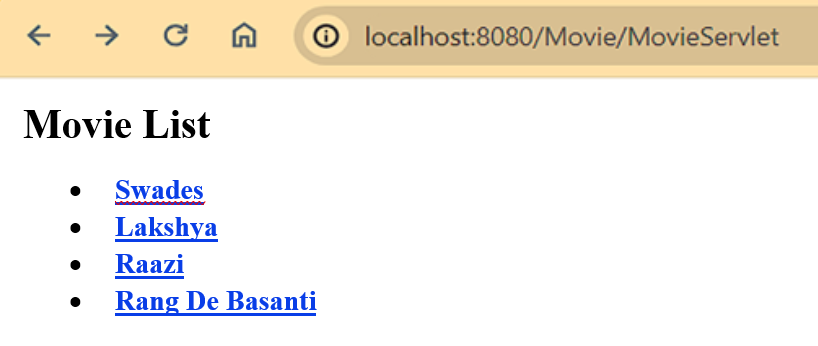
**We need to list all of our movie categories. Let's start with:**

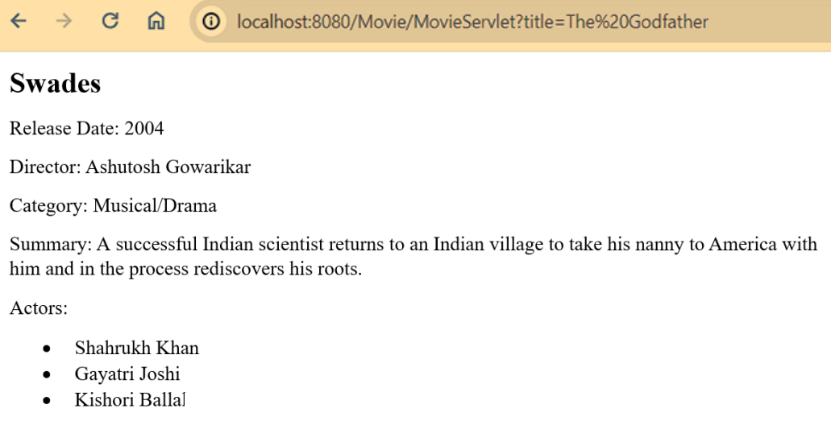
* **Horror**
* **Drama**
* **Comedy**
* **Action/Adventure**
* **Sci-Fi**

**The user should be able to browse and add, edit and delete movies. We'll store the movies in a database. To make this useful though, we'll have to load them from the database when we start the application. What kind of Servlet might be good for this task?**

**The application should keep track of the last 5 movies the user looked at. When they go back to the main page they should see Recently Viewed items (titles only). The main page should also show a recommended movie from the category they viewed the most. (The one that occurs the most in their recent history.) It should not be one of the items in their list.**

**The functionality to be there, but it doesn’t have to be “smooth”. For example a new user coming in doesn’t have a recently viewed history. It might be nice to not display that at all if there’s no information.**





**Objectives:**

1. To develop a simple servlet program to greet users.
2. To modify the servlet program to incorporate a login page and display user's name upon successful login.
3. To create a JSP that takes user's name and age, and echoes back the input along with movie ticket pricing based on age.
4. To design a MVC model-based web application for college students with functionalities including login, registration, home page with options, and session management.
5. To develop a movie library application where users can browse, add, edit, and delete movies, and view recently viewed items and recommendations based on viewing history.

**Problem Statement:**

1. Develop a servlet program to greet users with a simple "Hello" message.
2. Modify the servlet to incorporate a login page. Upon successful login, display the user's name on the homepage.
3. Create a JSP that takes user's name and age, and displays movie ticket pricing based on age criteria.
4. Design a MVC model-based web application for college students with login, registration, and home page functionalities.
5. Develop a movie library application with browsing, adding, editing, and deleting movies, and viewing recommendations based on user's viewing history.

**Outcomes:**

1. Implementation of a simple servlet program to greet users.
2. Enhancement of the servlet to include a login page and display user's name upon successful login.
3. Creation of a JSP to echo back user's input along with movie ticket pricing based on age.
4. Development of a MVC model-based web application for college students with login, registration, and home page functionalities.
5. Implementation of a movie library application with CRUD operations, viewing recently viewed items, and recommendations based on user's viewing history.

**Software & Hardware Requirements:**

* Software: Java Development Kit (JDK), Servlet container (e.g., Apache Tomcat), IDE (e.g., Eclipse), Database (for Q4 and Q5)
* Hardware: Computer system with sufficient RAM and storage capacity.

**Theory-Concept:**

* Servlet programming
* JavaServer Pages (JSP)
* Model-View-Controller (MVC) architecture
* Session management
* CRUD operations in web applications
* Database integration (for Q4 and Q5)

**Technology/Tool:**

* Java Servlet API
* JavaServer Pages (JSP)
* HTML, CSS, JavaScript (for front-end)
* Servlet container (e.g., Apache Tomcat)
* Database (e.g., MySQL, PostgreSQL) for Q4 and Q5

**Design/Execution Steps:**

* Q1: Write a simple servlet program to say hello to the user.
* Q2: Modify the servlet program to include a login page and display user's name upon successful login.
* Q3: Create a JSP to take user's name and age, and display movie ticket pricing based on age.
* Q4: Design a MVC model-based web application for college students with login, registration, and home page functionalities.
* Q5: Develop a movie library application with CRUD operations, viewing recently viewed items, and recommendations based on user's viewing history.

**Test Cases:**

* Q1: Verify that the servlet displays "Hello" message.
* Q2: Test the login functionality by providing valid and invalid credentials.
* Q3: Test the JSP by providing different age inputs and verifying the displayed movie ticket pricing.
* Q4: Test the registration, login, and session management functionalities.
* Q5: Test the CRUD operations, viewing recently viewed items, and recommendations functionality in the movie library application.

**Conclusion/Analysis:**

* Q1: Simple servlet program to greet users was successfully implemented.
* Q2: The servlet program was modified to include a login page and display user's name upon successful login.
* Q3: A JSP was created to echo back user's input along with movie ticket pricing based on age.
* Q4: A MVC model-based web application for college students with login, registration, and home page functionalities was designed and implemented.
* Q5: A movie library application with CRUD operations, viewing recently viewed items, and recommendations functionality was successfully developed.

**Oral Questions:**

1. What is the purpose of servlet programming?
2. Explain the Model-View-Controller (MVC) architecture and its benefits in web development.
3. What are the advantages of using a servlet container like Apache Tomcat?