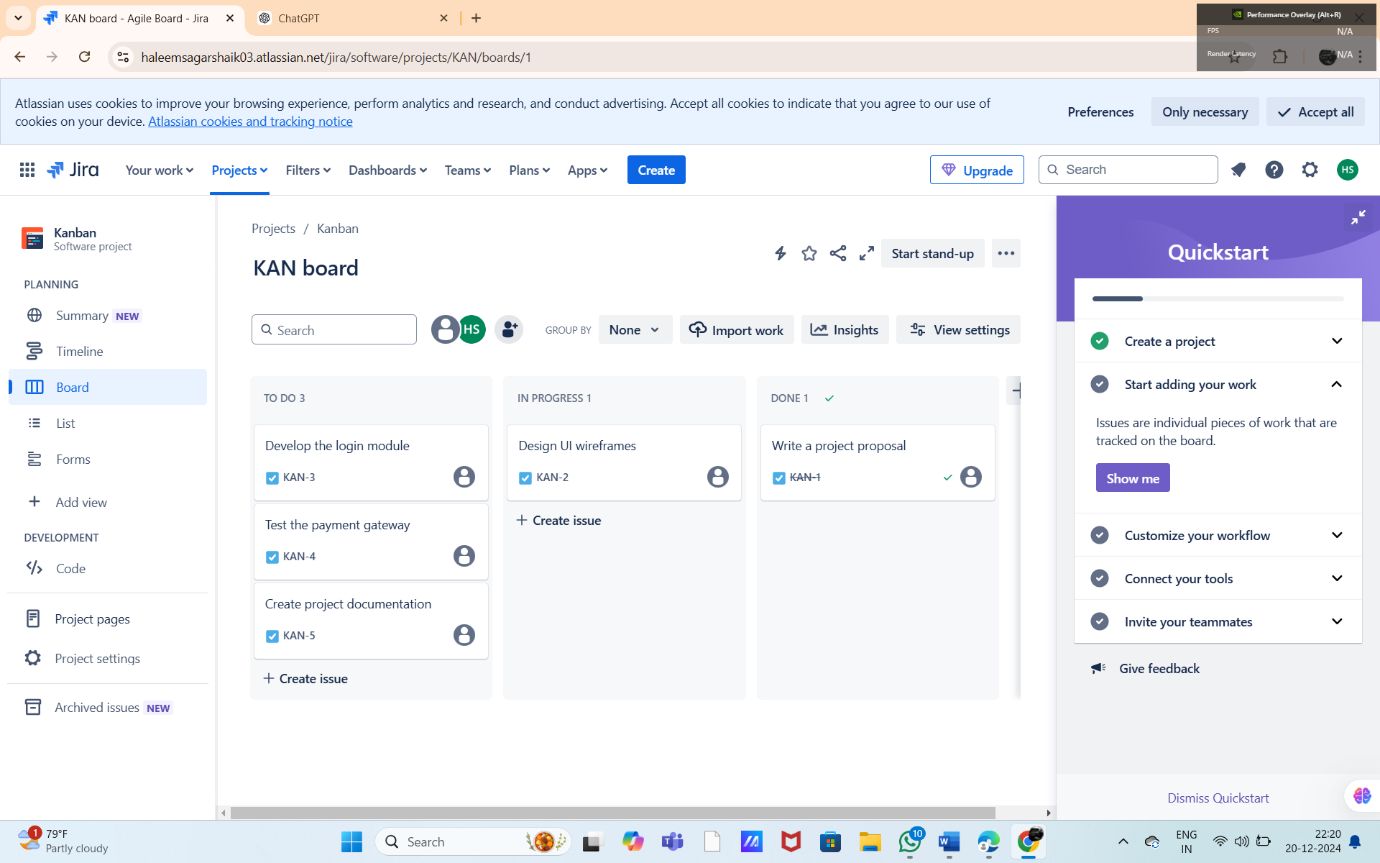
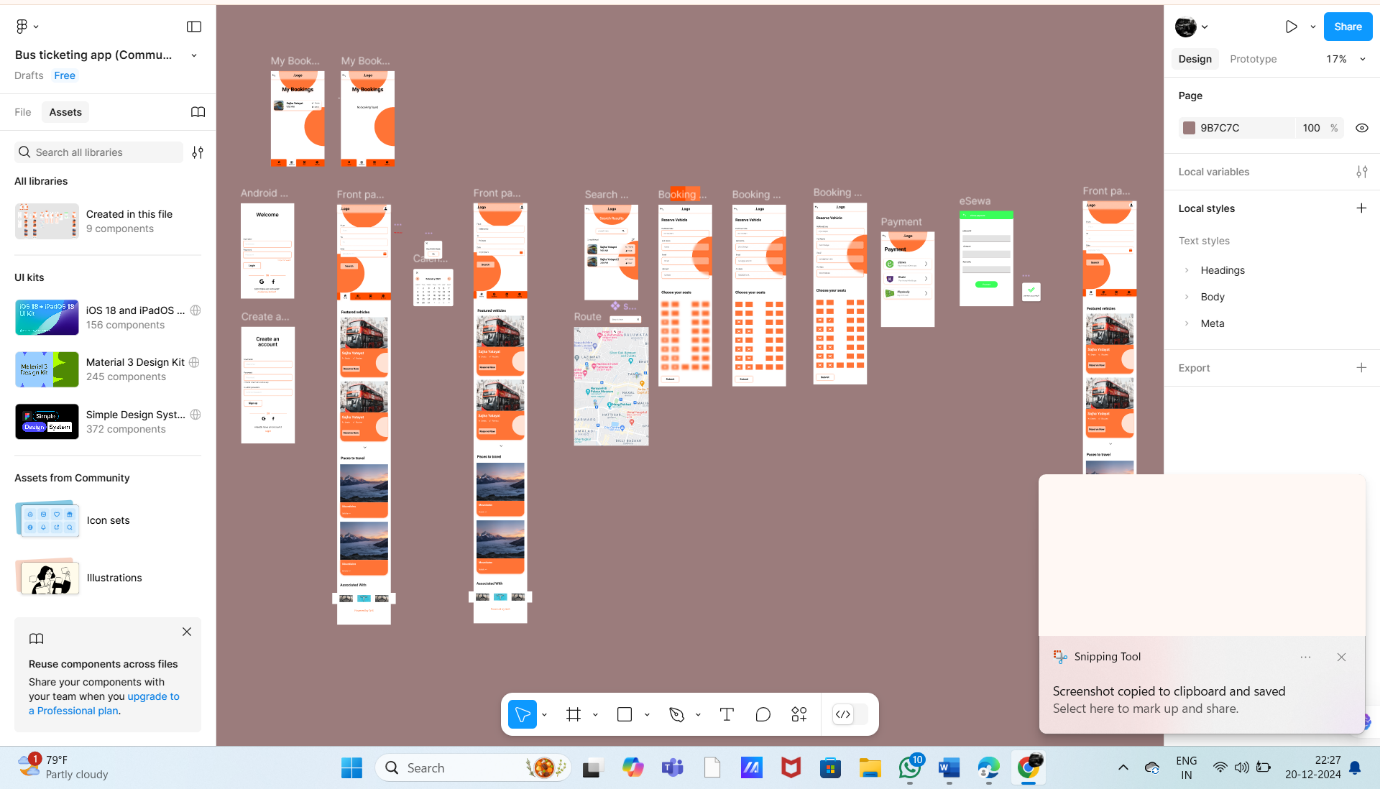
Exp:1

1.Create a Kanban Board to Visualize the Tasks. • Create Columns for To Do, In-Progress and Done. • Add Atleast 5 Sample Tasks • Move the Tasks across the Columns to Simulate the Workflow



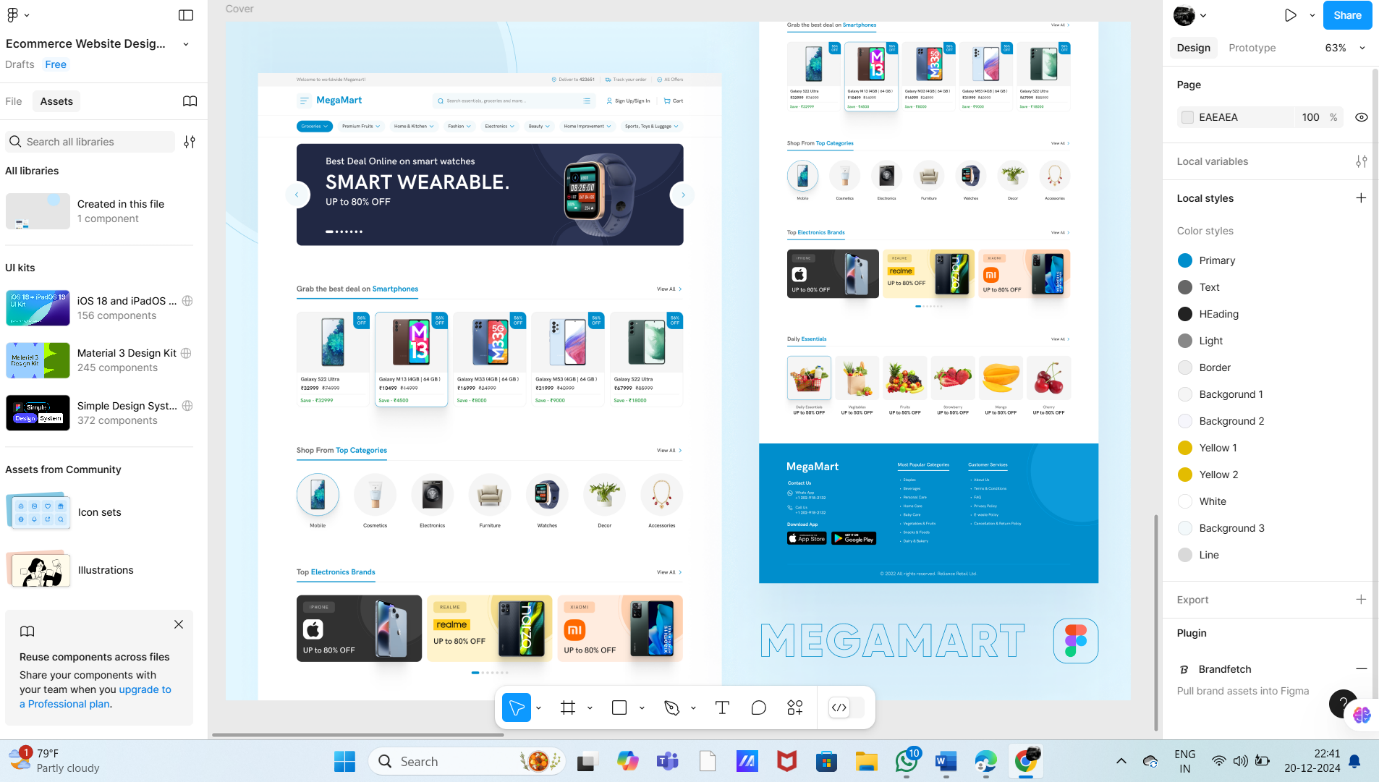
Exp:2

Sketch a Simple Prototype of a Bus Ticket Booking System using Figma Too



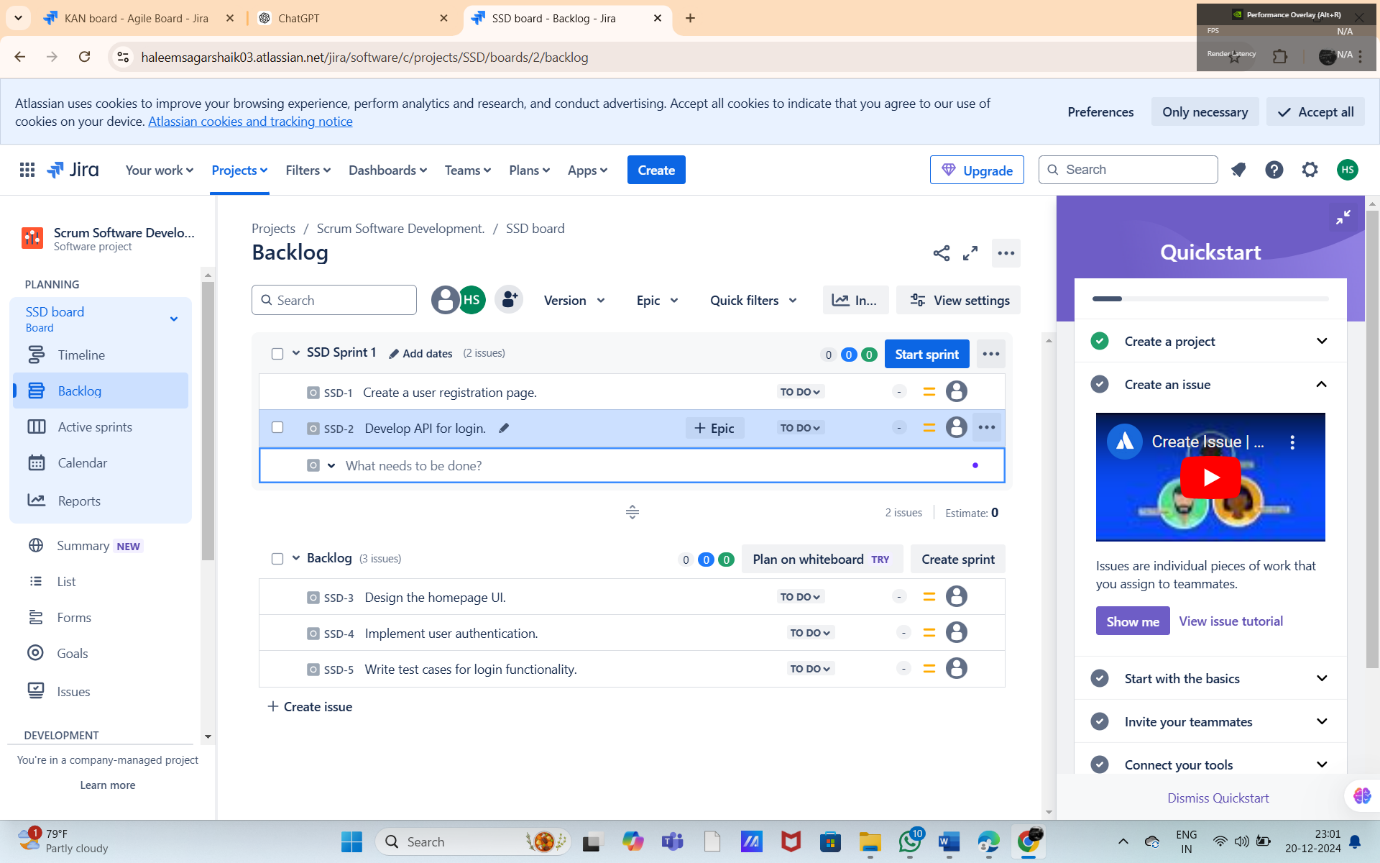
Exp:3

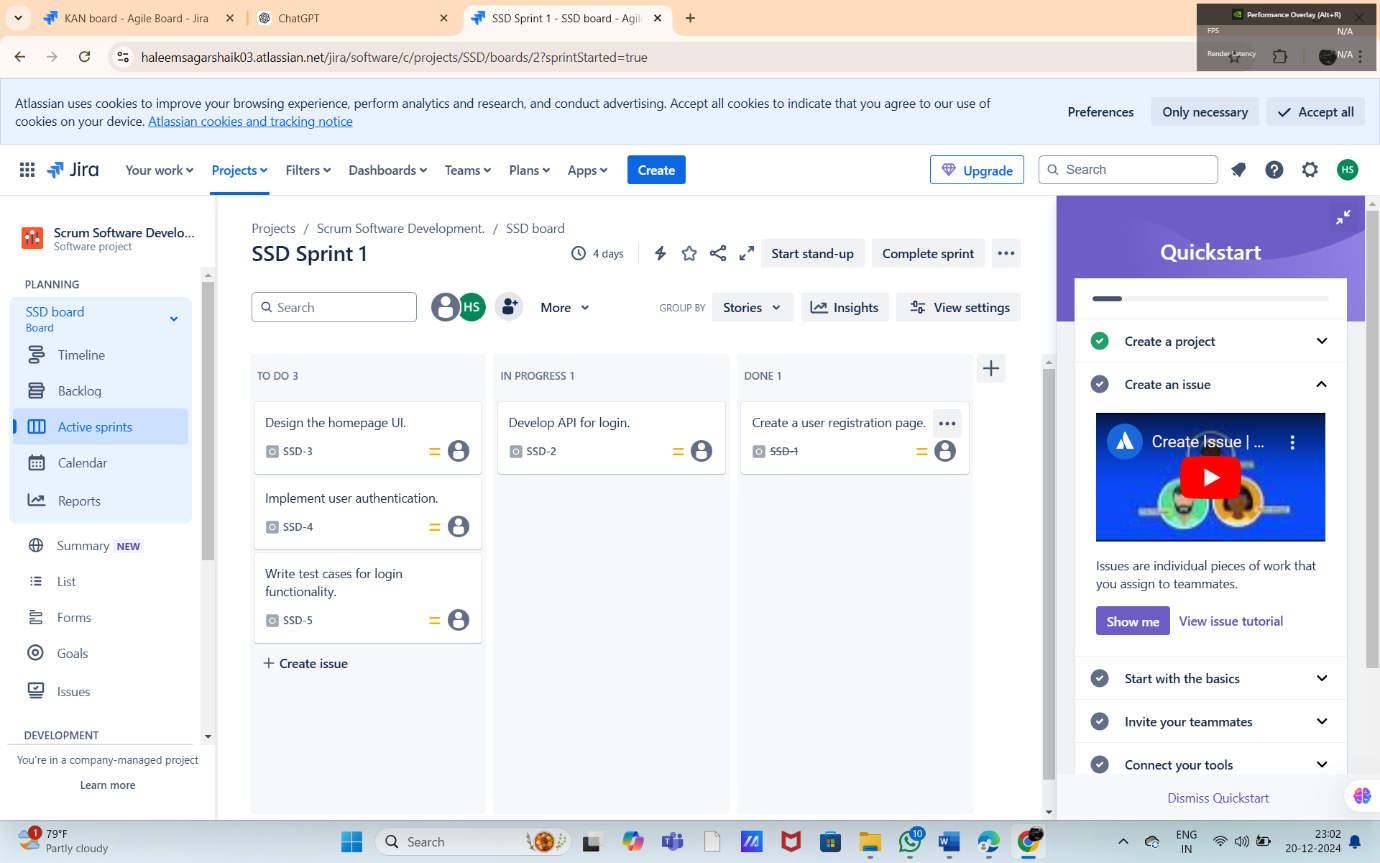
The stakeholders have conflicting views on the user interface design for an E-Commerce mobile app. Create a prototype using Figma tool to discuss with the stakeholders to get their feedback and approval.



Exp:4

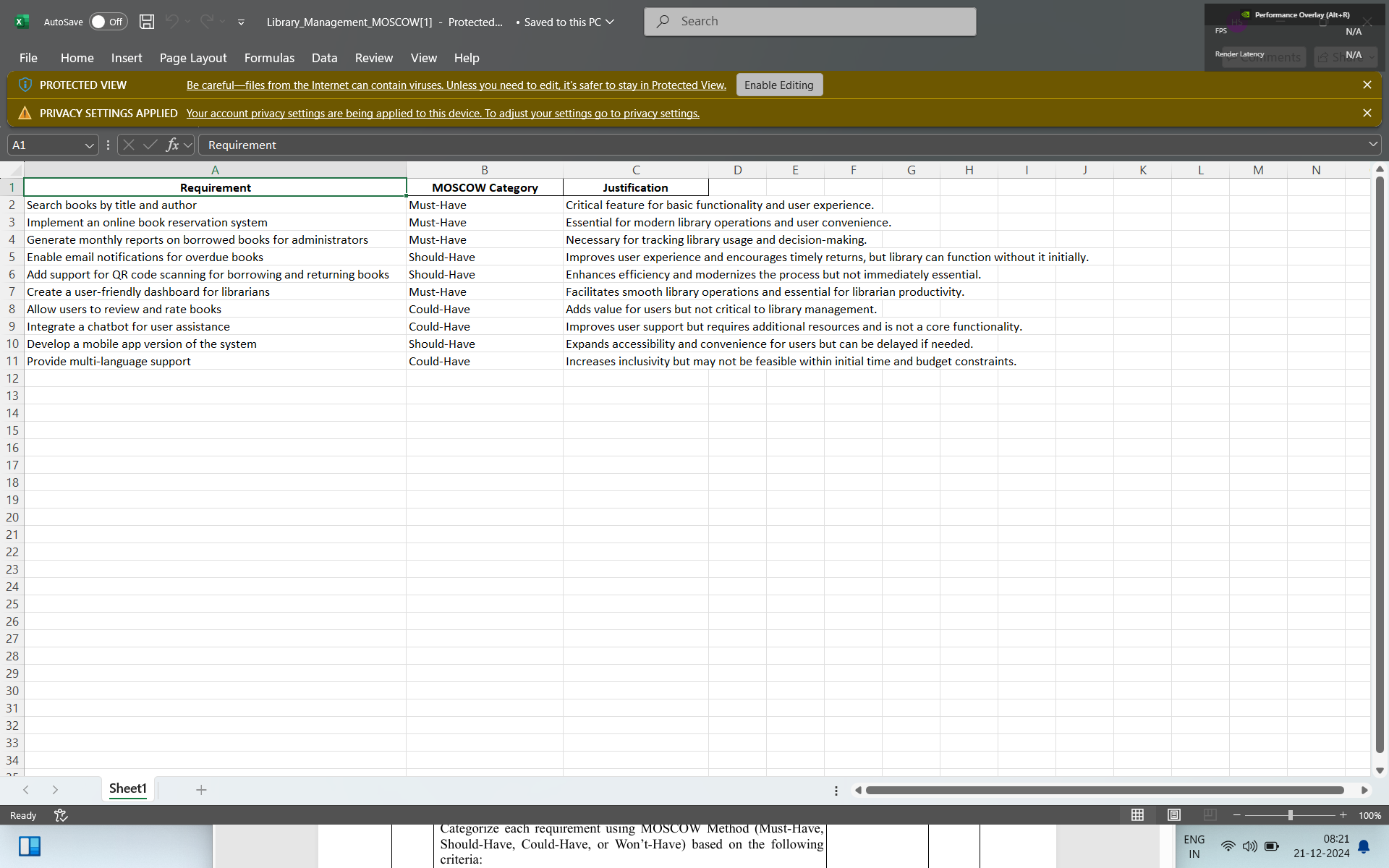
Create a Scrum Project in Jira. • Add a backlog with at least 5 items (e.g., "Create user registration page", "Develop API for login"). • Prioritize the backlog and create a 1-week sprint. • Move backlog items into the sprint and start the sprint. • Finally show the Screenshot of the sprint board at the start and end of the sprint.





Exp:5

Use the following requirements for a Library Management System: • Add a feature to search books by title and author. • Implement an online book reservation system. • Generate monthly reports on borrowed books for administrators. • Enable email notifications for overdue books. • Add support for QR code scanning for borrowing and returning books. • Create a user-friendly dashboard for librarians. • Allow users to review and rate books. • Integrate a chatbot for user assistance. • Develop a mobile app version of the system. • Provide multi-language support. Categorize each requirement using MOSCOW Method (Must-Have, Should-Have, Could-Have, or Won’t-Have) based on the following criteria: • Impact on the users and stakeholders. • Feasibility considering time, budget, and resource constraints. Finally Submit the completed Google Sheet or Excel file with all requirements categorized and justified.



Exp:6

Link Jira tasks with Confluence to streamline task tracking and progress monitoring for the Library Management System development.

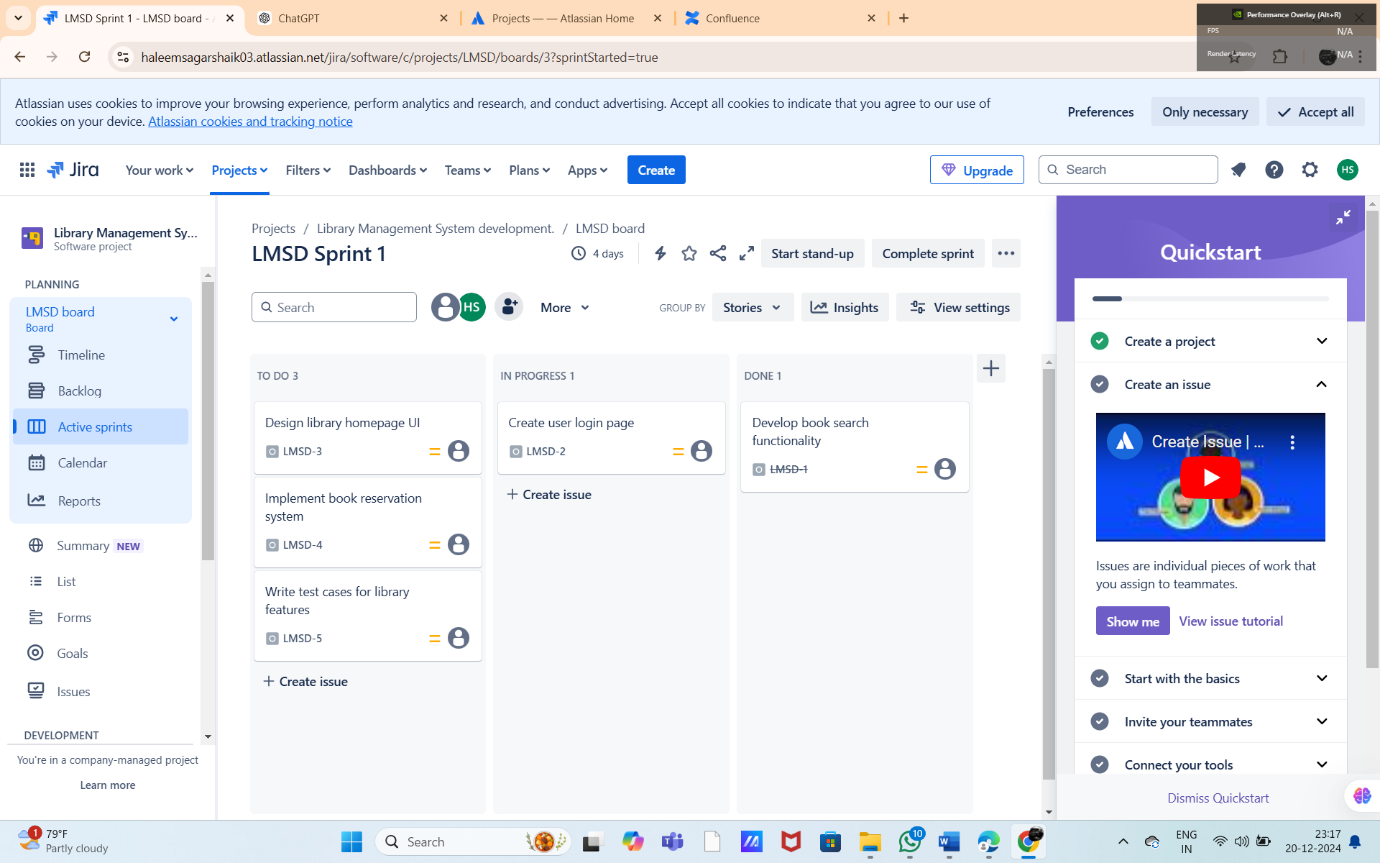
• Create a new page in Confluence titled "Library Management System Project Overview."

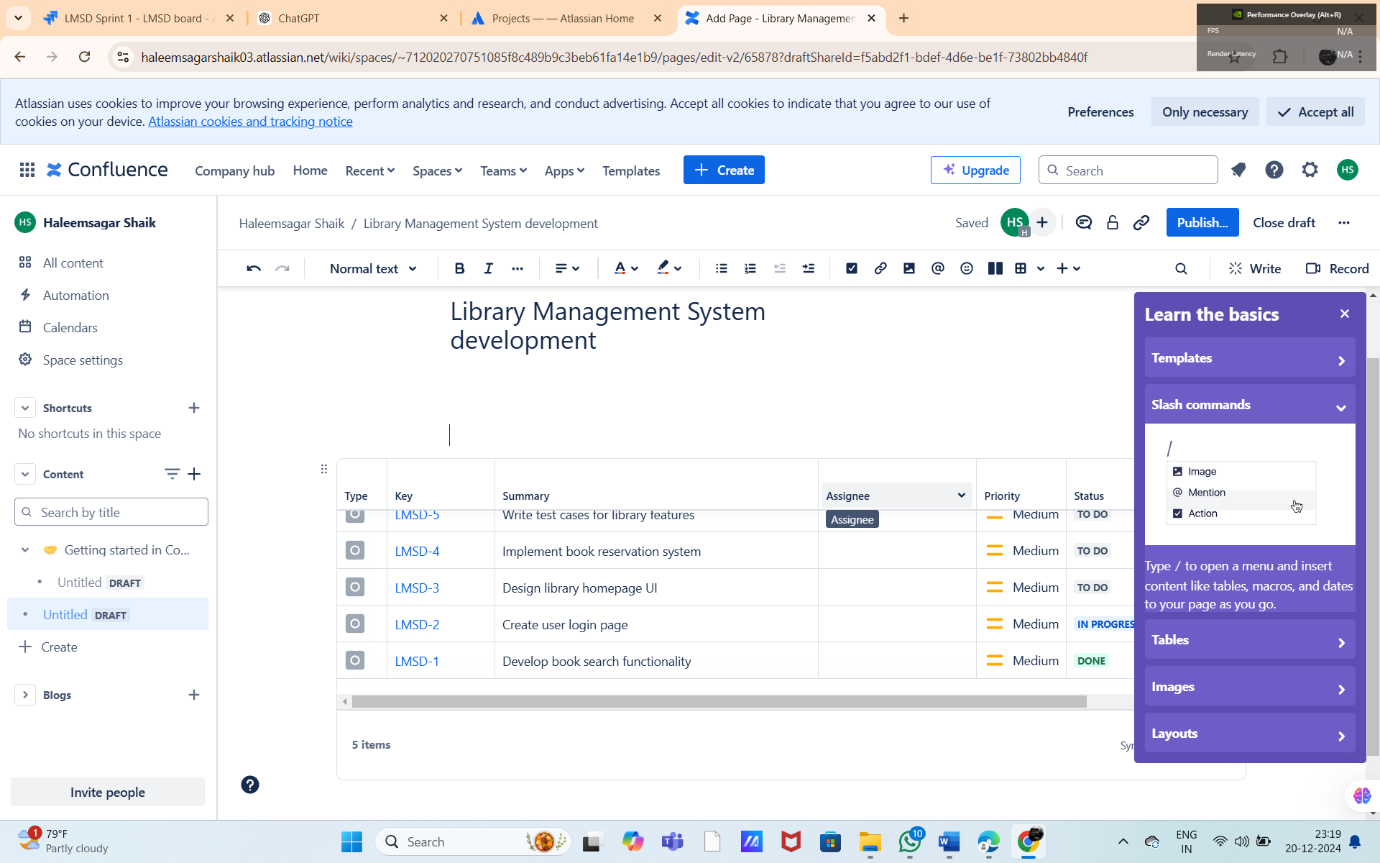
• Embed at least 5 Jira issues related to the development of the Library Management System (e.g., tasks from the sprint like "Develop book search functionality," "Create user login page," etc.).

• Use the Jira macro to display issues with status (e.g., "To Do," "In Progress," "Done").

• Add a progress bar in the Confluence page to visually track the completion of each embedded Jira task (e.g., percentage of tasks completed in the sprint).

• Submit a screenshot of the Confluence page showing the embedded Jira tasks and the progress bar.





Exp:7

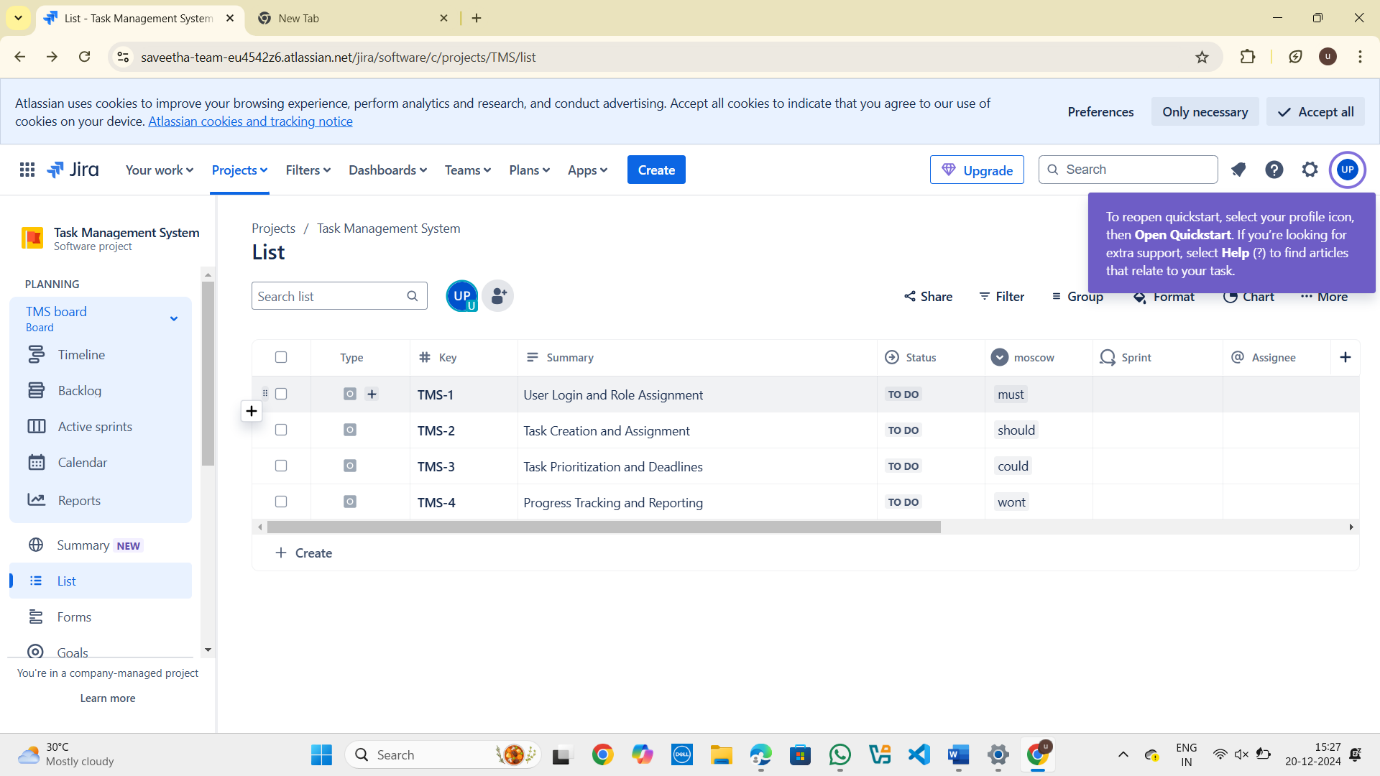
You are designing a Task Management System for a small team. The system should include the following features:

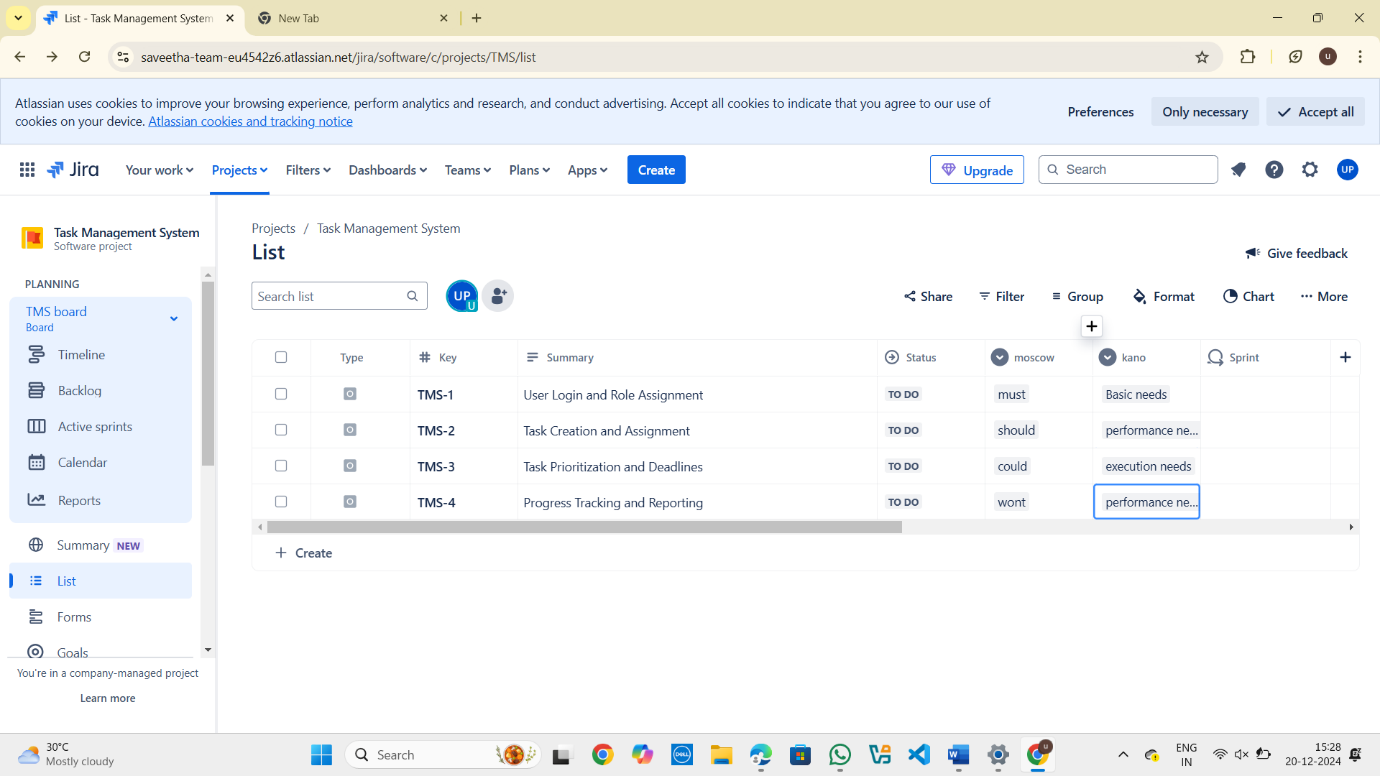
1.User Login and Role Assignment

2.Task Creation and Assignment

3.Task Prioritization and Deadlines

4.Progress Tracking and Reporting Prioritize these requirements using the MoSCoW and Kano models in Jira.





Exp:8

You are tasked with developing an Online Learning Platform. The platform should include the following functionalities:

1. Course Enrollment and Registration

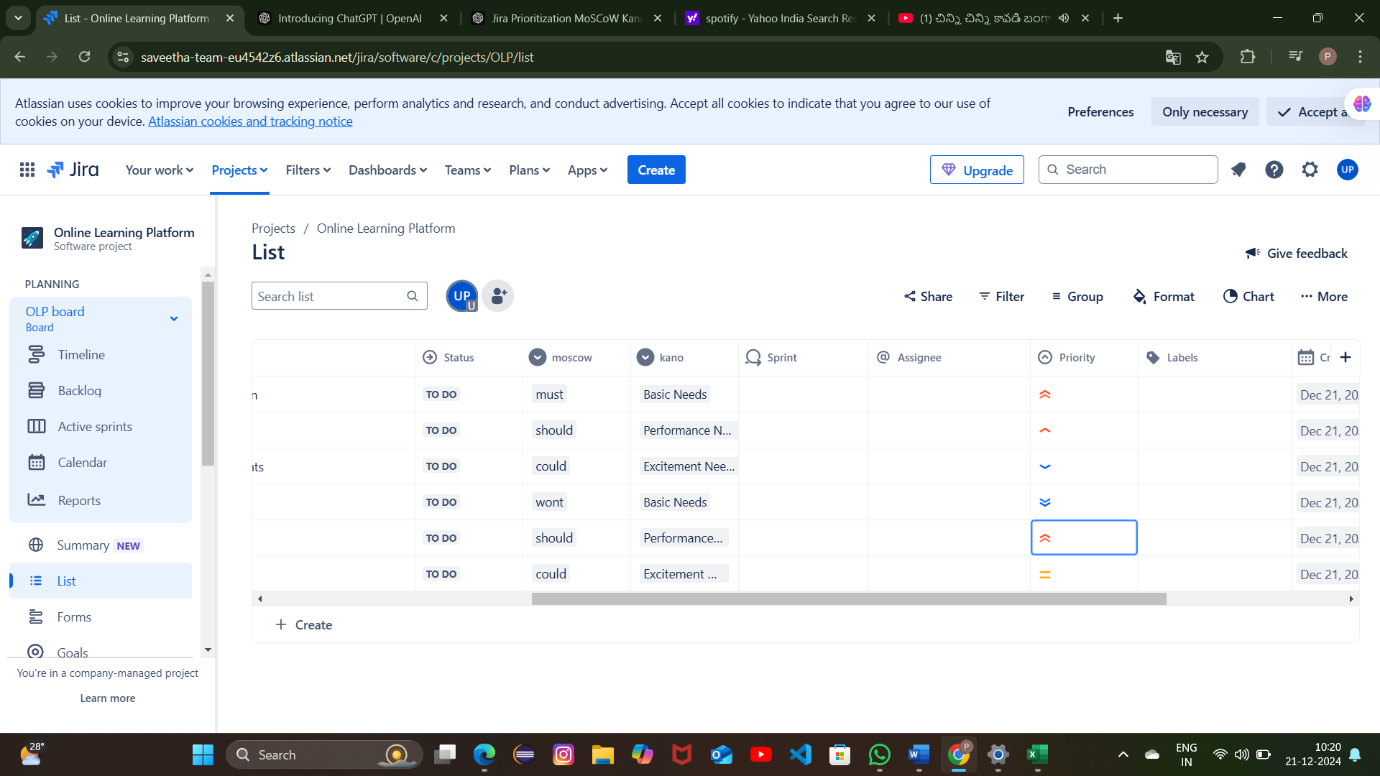
2. Video Lecture Streaming

3. Interactive Quizzes and Assignments

4. Progress Tracking Dashboard

5. Peer-to-Peer Discussion Forums

6. Certificate Generation Use Jira to categorize and prioritize these requirements using MoSCoW and Kano techniques.



Exp:11

Create a Static Website and Containerize, Build & Serve it using Docker. Tasks:

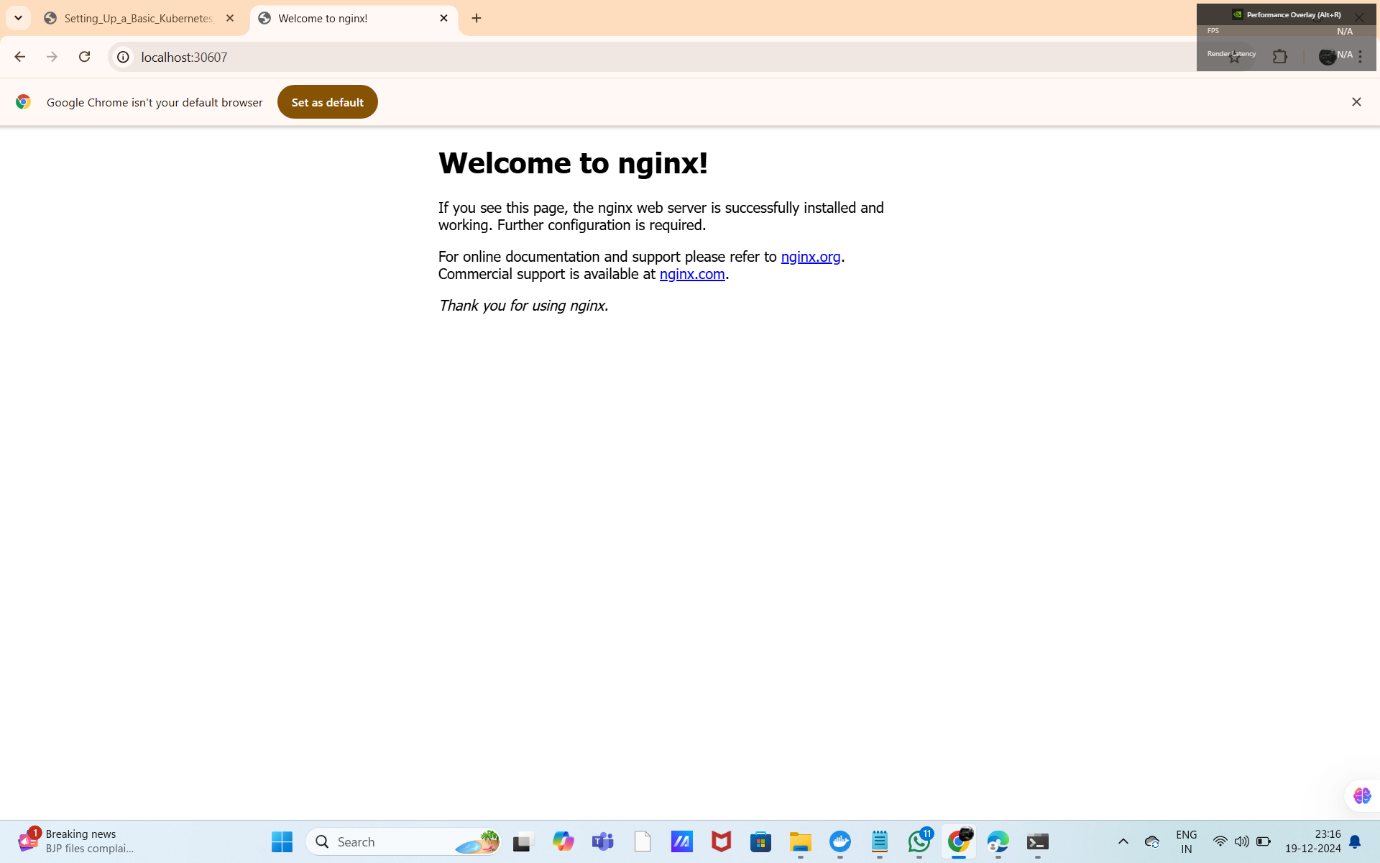
1. Create a Simple Static Website (index.html file) with basic HTML content.

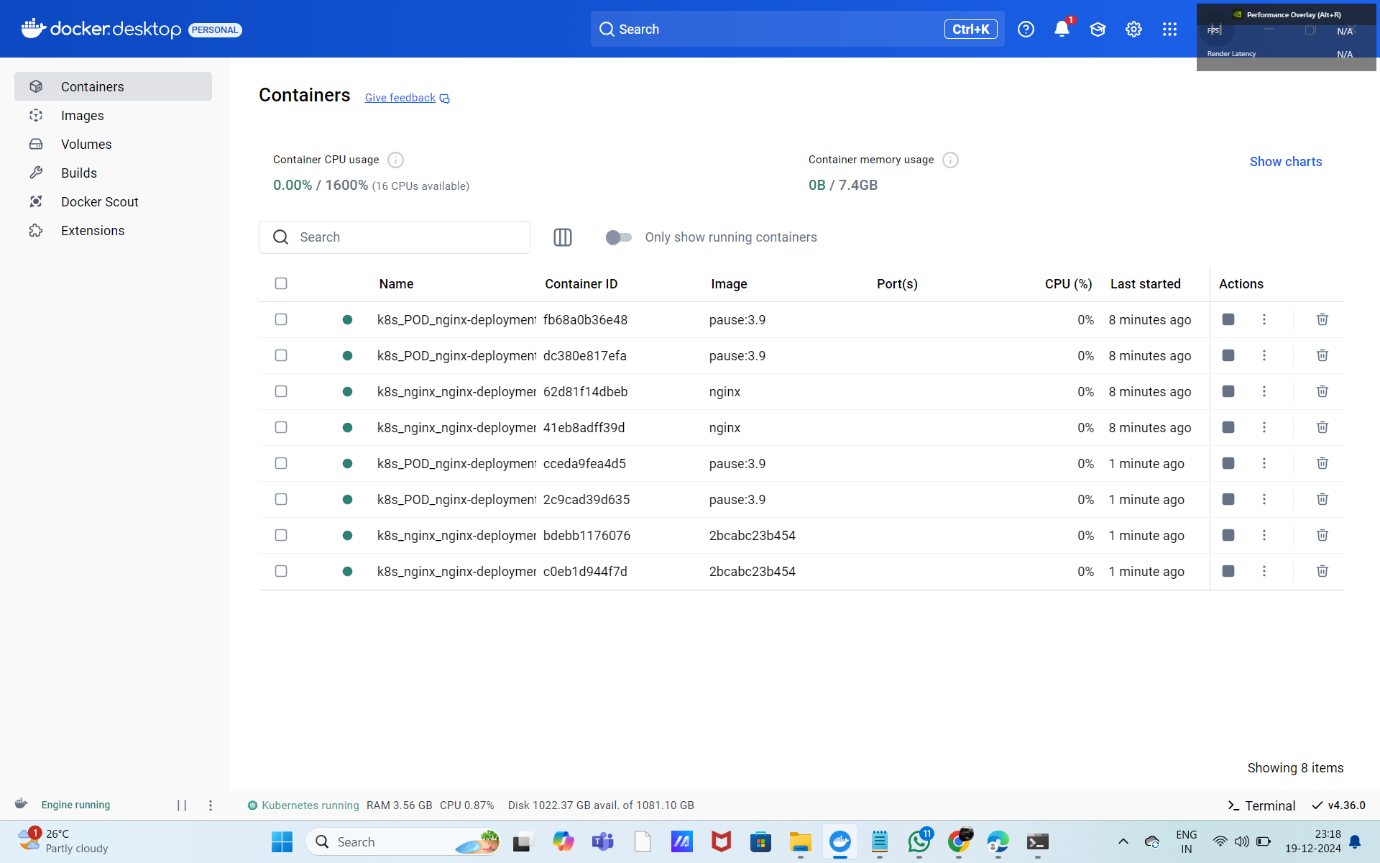
2. Write/create a Dockerfile to serve the website using Nginx.

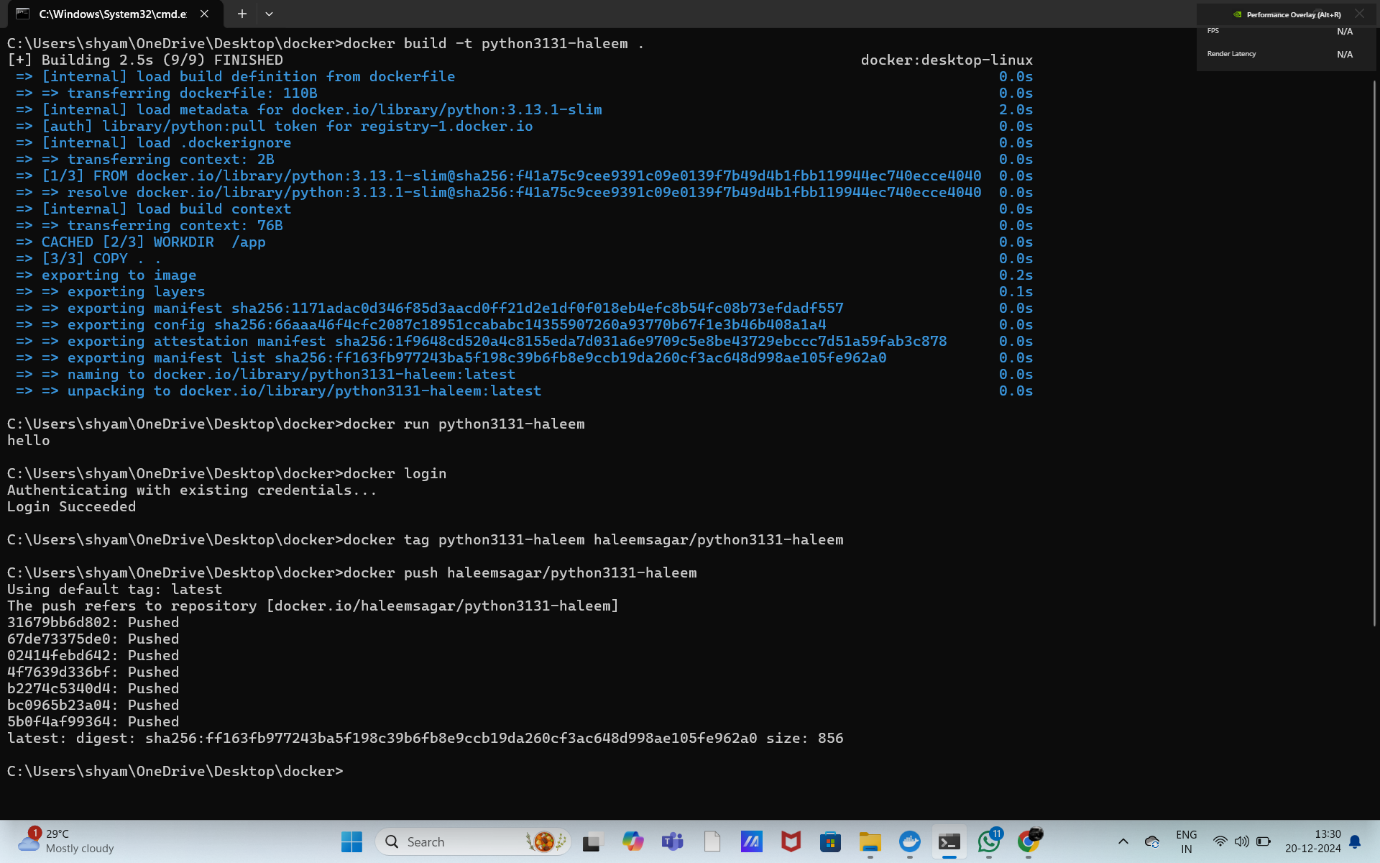
3. Build the Docker Image

4. Run the container:

5. Access the Website using a Browser







Exp:12

Create a Simple Python Flask API, Containerize the Application, Build & Push the Image using Docker and Deploy the Application using Kubernetes. Tasks:

1. Create a Simple Flask API by writing a Python file (app.py) with basic endpoints.

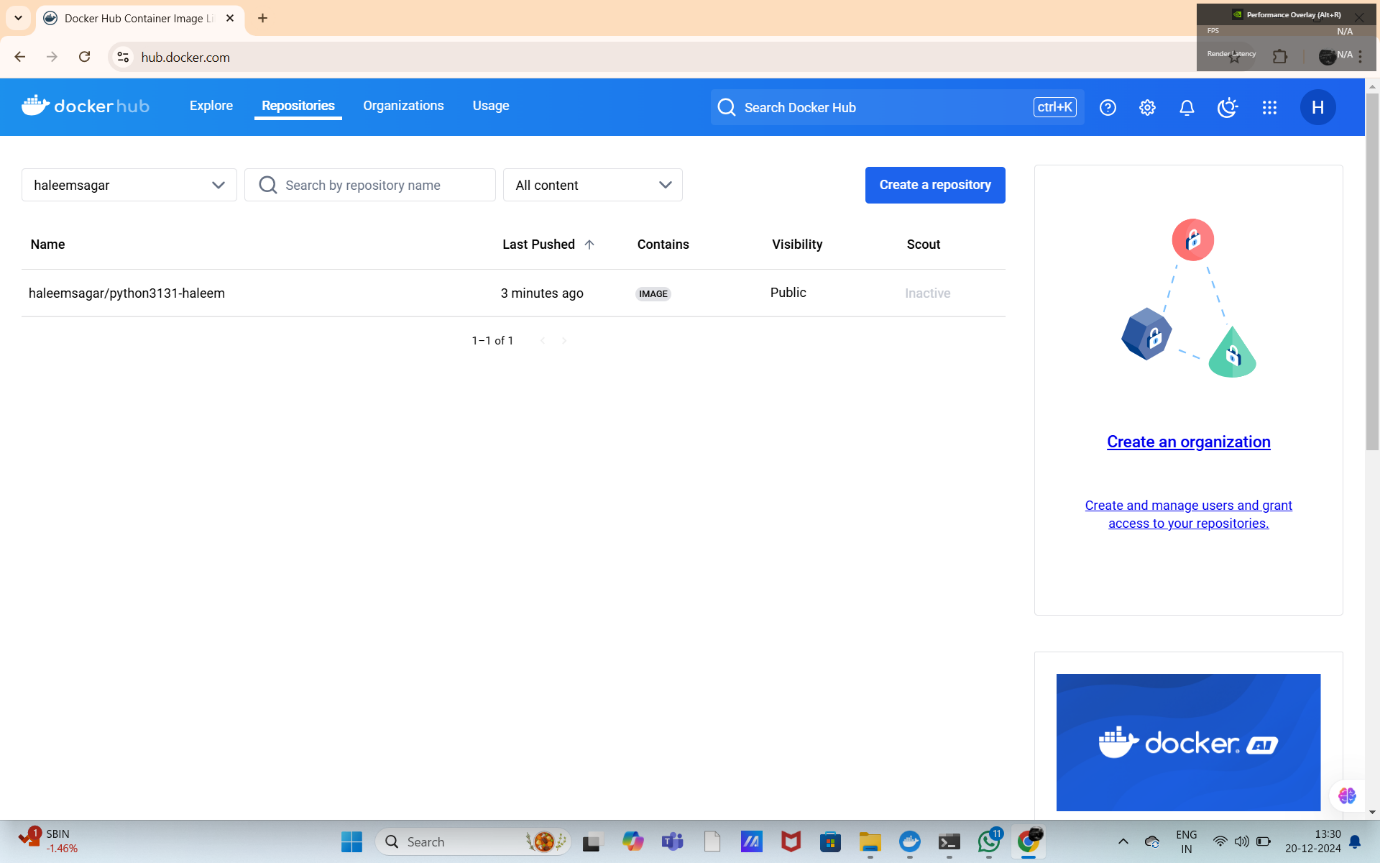
2. Containerize the Flask App using Dockerfile.

3. Build the Image using Docker.

4. Push the Image to Docker Hub.

5. Create Kubernetes manifests (Deployment YAML & Service YAML) to deploy the application.

6. Apply the Manifests and Access the API via NodePort.



EXP 13:

Set up a CI/CD pipeline to automate the building, testing, and deployment of a containerized application.

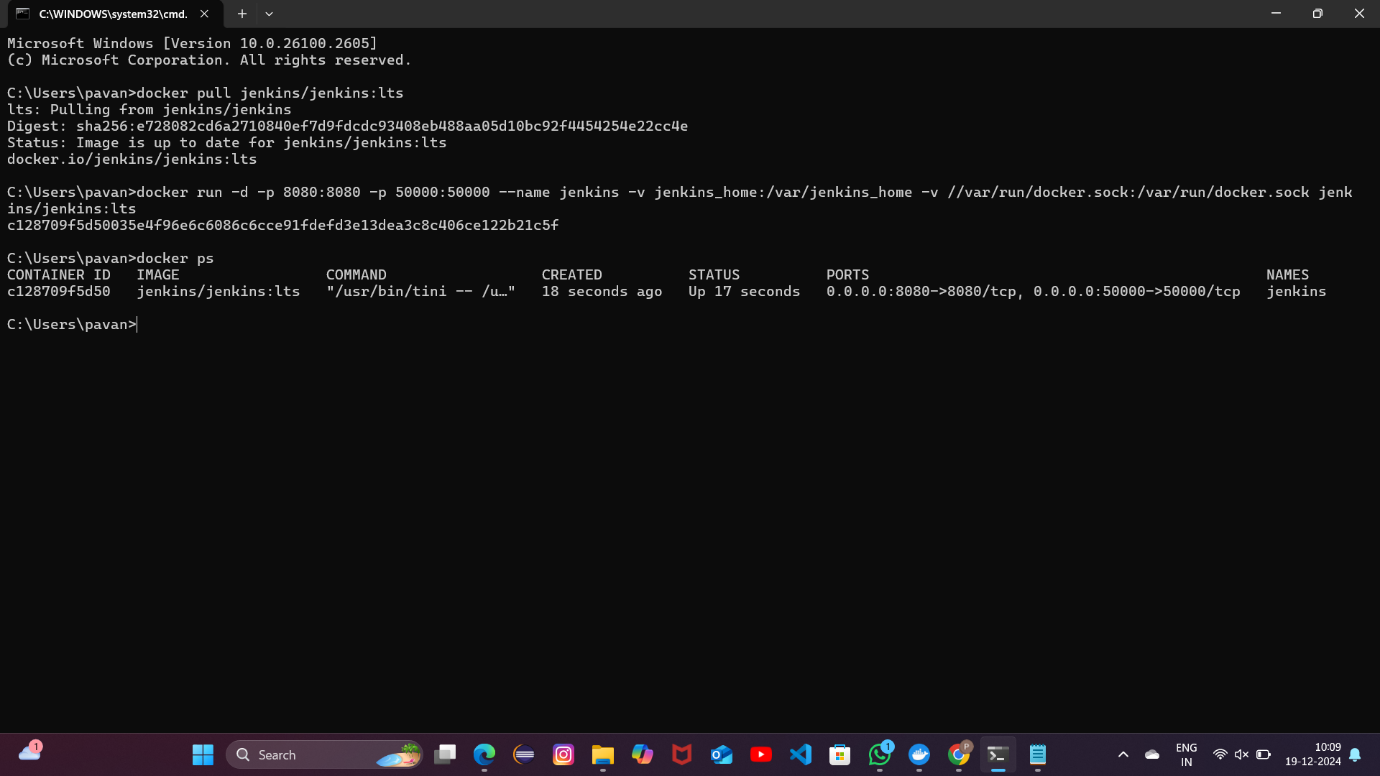
Tasks:

1. Set up Jenkins on a local machine or server. Docker CO4

2. Create a Dockerfile to containerize a sample application.

3. Write a Jenkinsfile to automate the process of building the Docker container, running tests, and deploying to a cloud platform (e.g., AWS or GCP).

4. Configure Jenkins to trigger builds upon code commits or pull requests.



EXP 17:

Push and Pull Docker Images Using Docker Hub

Tasks:

• Create a Docker image for a static HTML website.

• Tag and push the image to Docker Hub.

• Pull the image from Docker Hub and run it on another machine.

• Submit Docker commands and screenshots for each step

