**KESHAV MEMORIAL INSTITUTE OF TECHNOLOGY**



**(AN AUTONOMOUS INSTITUTE)**

 **Accredited by NBA & NAAC, Approved by AICTE, Affiliated to JNTUH, Hyderabad**

**A.Y 2025-2026**

**Department of Computer Science & Engineering (DS)**

**Lab Internal I**

**Subject Name: Software Engineering Subject Code: 23CC501PC**

**Year and Semester: III / I Branch /Section: CSD-B Faculty:** Y Deepthi  **Lab Internal: 10/09/2025**

**SET-2: Clinic Management System (CMS)**

**Part I – Software Requirement Specification (SRS) [10 Marks]**

A small clinic wants to develop a **Clinic Management System (CMS)** to handle patient registrations, doctor appointments, treatment records, and billing. Currently, staff manually record visits and generate bills, leading to delays and errors. Patients often miss appointments because reminders are not automated.

**Answer the following:**

1. What is the **purpose** of this project? Write a short abstract explaining it. [2M]
2. List **functional requirements** (minimum 5) to solve the clinic’s problems. [2M]
3. List **non-functional requirements** (minimum 5) describing how the system should behave. [2M]
4. Identify the different **users** of the system and explain how each would interact with it. [2M]
5. Suggest **modules** to divide the CMS application into manageable parts. [2M]

**Part II – Maven Web Application Development [25 Marks]**

You are given a Maven-based Clinic Management System project.

<https://github.com/vam1207/CMS.git>

The project allows managing patients, doctors, appointments, and billing.

**Answer the following:**

1. The project fails to build. How will you **fix the pom.xml** to resolve dependency errors? [8M]
2. The application was using Java 8, but now it should run on **Java 19**. How will you update Maven configuration? [2M]
3. A teammate updated a dependency, but your local Maven cache uses the old version. How do you **force Maven to update**? [2M]
4. You want to **skip tests** during Maven build. Which command or configuration will you use? [5M]
5. Maven shows a **dependency conflict**. How can you resolve it? [5M]
6. You want to use **JSTL in JSP pages**. Which Maven dependencies will you include? [3M]

**Part III – Git & GitHub Integration [15 Marks]**

You are working with GitHub on the CMS project. Handle these scenarios:

1. You start coding a new feature but the project is not under version control. How do you **initialize Git and push to GitHub**? [5M]
2. You accidentally wrote “Added Appointment Pagee” instead of “Added Appointment Page” in the commit message. You haven’t pushed yet. How do you fix it? [2M]
3. Check which files have been modified but not committed. Which Git command will you use? [1M]
4. See commit history in a compact view. Which command will you use? [2M]
5. You deleted PatientServlet.java by mistake but haven’t committed. How do you recover it? [2M]
6. You want to **clone a teammate’s repo** and switch directly to feature/appointment-module branch. What steps will you follow? [3M]

**Part IV – Git Collaboration, Patch & Merge Conflict Resolution [20 Marks]**

Two developers are working on CMS from different GitHub accounts:

1. You are assigned to add a **doctor availability feature** without affecting main. Create branch feature/availability and switch to it. [3M]
2. List all **local and remote branches**. Which command will you use? [2M]
3. You accidentally created the branch in the wrong location. How do you **delete feature/availability** safely? [2M]
4. A bug is found in BillingModule.java. Create a **patch file** and share it with your teammate. [5M]
5. Your teammate sent you a patch. How do you **apply it** locally? [3M]
6. You want to **combine multiple commits into one** before pushing. Which Git command helps? [3M]
7. Make a local copy of a remote repository to work on it. Which command will you use? [2M]

**Part V – Dockerization of Maven Application [15 Marks]**

You want to containerize CMS:

1. Write a **Dockerfile** to build a Docker image for CMS using Maven. [5M]
2. Build the image and run the container. Verify CMS works at http://localhost:9090. [5M]
3. Push the Docker image to Docker Hub to share with teammates. [5M]

**Part VI – Docker Compose Multi-Container Setup [15 Marks]**

CMS should work with a **database backend**:

1. Write docker-compose.yml with two services: [5M]
   * Service 1: CMS Docker image
   * Service 2: MySQL database with username clinic\_user, password clinic\_pass, and DB name clinic\_db
2. How do you **run both containers**? [2M]
3. If you update CMS Docker image, how do you **rebuild the service** using Docker Compose? [3M]
4. Verify that **patient and appointment data persists** after restarting containers. [3M]
5. Stop all running containers but keep the data volume. Which command will you use? [2M]

**Answer Key – Set-2 (CMS)**

**Part I – SRS [10M]**

1. **Abstract (Purpose)** [2M]  
   The Clinic Management System (CMS) is designed to manage patient registrations, doctor appointments, treatment history, and billing. It reduces manual errors, automates reminders, and provides accurate reports for better clinic operations.
2. **Functional Requirements** [2M]

* Patient registration and record management
* Appointment scheduling and cancellation
* Doctor availability and schedule management
* Billing and invoice generation
* Automated SMS/email reminders for patients

1. **Non-Functional Requirements** [2M]

* Secure login and role-based access
* Fast response time (<2 sec per action)
* Data should be backed up daily
* Scalable to handle 1000+ patients
* User-friendly interface for staff and doctors

1. **Users** [2M]

* **Patients** → Book appointments, view history, pay bills
* **Doctors** → Manage schedules, view patient details, update treatments
* **Receptionist/Staff** → Register patients, manage appointments, billing
* **Admin** → Manage users, system settings, reports

1. **Modules** [2M]

* Patient Management
* Appointment Scheduling
* Doctor Availability
* Billing & Payments
* Reports & Notifications

**Part II – Maven Web Application Development [25M]**

1. **Fix pom.xml errors** [8M]

* Check for missing <dependencies> or <plugins> tags
* Add correct <groupId>, <artifactId>, <version>
* Ensure packaging = war
* Example dependency (JSTL):
* <dependency>
* <groupId>javax.servlet</groupId>
* <artifactId>jstl</artifactId>
* <version>1.2</version>
* </dependency>

1. **Update Java version to 19** [2M]

<properties>

<maven.compiler.source>19</maven.compiler.source>

<maven.compiler.target>19</maven.compiler.target>

</properties>

1. **Force Maven to update dependencies** [2M]

mvn clean install -U

1. **Skip tests in Maven build** [5M]  
   Command:

mvn clean package -DskipTests

1. **Resolve dependency conflict** [5M]

* Use <dependencyManagement> to force a version
* Or use <exclusions> to remove transitive conflict

1. **Add JSTL dependency** [3M]

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>jstl</artifactId>

<version>1.2</version>

</dependency>

**Part III – Git & GitHub Integration [15M]**

1. **Initialize Git & push** [5M]

git init

git add .

git commit -m "Initial commit: CMS project"

git branch -M main

git remote add origin https://github.com/username/CMS.git

git push -u origin main

1. **Fix wrong commit message** [2M]

git commit --amend -m "Added Appointment Page"

1. **Check modified but uncommitted files** [1M]

git status

1. **Compact commit history** [2M]

git log --oneline

1. **Recover deleted file before commit** [2M]

git checkout -- PatientServlet.java

1. **Clone and switch branch** [3M]

git clone https://github.com/teammate/CMS.git

cd CMS

git checkout feature/appointment-module

**Part IV – Git Collaboration, Patch & Merge Conflict [20M]**

1. **Create feature branch** [3M]

git checkout -b feature/availability

1. **List all branches** [2M]

git branch -a

1. **Delete local branch** [2M]

git branch -d feature/availability

1. **Create patch file** [5M]

git diff > bugfix.patch

1. **Apply teammate’s patch** [3M]

git apply bugfix.patch

1. **Combine commits (squash)** [3M]

git rebase -i HEAD~3

1. **Clone remote repo** [2M]

git clone https://github.com/username/CMS.git

**Part V – Dockerization of Maven Application [15M]**

1. **Dockerfile** [5M]

FROM maven:3.9.6-eclipse-temurin-19 AS build

WORKDIR /app

COPY . .

RUN mvn clean package -DskipTests

FROM tomcat:10.1-jdk19

COPY --from=build /app/target/CMS.war /usr/local/tomcat/webapps/CMS.war

EXPOSE 9090

CMD ["catalina.sh", "run"]

1. **Build & Run** [5M]

docker build -t cms-app .

docker run -p 9090:9090 cms-app

1. **Push to Docker Hub** [5M]

docker tag cms-app username/cms-app:latest

docker push username/cms-app:latest

**Part VI – Docker Compose Multi-Container Setup [15M]**

1. **docker-compose.yml** [5M]

version: '3'

services:

cms-app:

image: username/cms-app:latest

ports:

- "9090:9090"

depends\_on:

- mysql-db

mysql-db:

image: mysql:8

environment:

MYSQL\_ROOT\_PASSWORD: rootpass

MYSQL\_USER: clinic\_user

MYSQL\_PASSWORD: clinic\_pass

MYSQL\_DATABASE: clinic\_db

volumes:

- mysql\_data:/var/lib/mysql

volumes:

mysql\_data:

1. **Run both containers** [2M]

docker-compose up -d

1. **Rebuild service after update** [3M]

docker-compose build cms-app

docker-compose up -d

1. **Verify data persistence** [3M]  
   Restart containers:

docker-compose down

docker-compose up -d

Data remains due to volume.

1. **Stop containers but keep data** [2M]

docker-compose down