RAMAIAH
Institute of Technology

Department of Computer Science & Engineering

QUESTION BANK FOR VI SEMESTER (Term: SEP-DEC 2025)

Full Stack Development Laboratory (CSL57)

I.A. Marks: 50 **Exam Hours: 03**
Credits : 0:0:1 **Exam Marks: 50**

Q.no	Problem Statements	COs	POs
1.	(a) Write a function translate () that will translate a text i.e, double every consonant and place an occurrence of "o" in between. For example, translate("this is fun") should return the string "tothohisos isos fofunon". (b) Write a Complaint Management API using Node.js, Express, and MongoDB with the following features: • Each complaint should include: Complaint ID, User Name, Issue, and Status. • Implement a POST route to submit a new complaint. • Implement a PUT route to update the status of a complaint (e.g., "In Progress", "Resolved"). • Implement a GET route to retrieve all complaints that are currently pending.	2,3	1,2,3,5
2.	(a) Write a java script program to convert month number to month name using closures. • If the user enters a number less than 1 or greater than 12 or a non-number, have the function write "Bad Number" in the monthName field. • If the user enters a decimal between 1 and 12 (inclusive), strip the decimal portion of the number. (b) write a Node.js application using Express and MongoDB with the following functionality: • Accept student details via a web form: Student_name, USN, Semester, and Exam_fee. • Store the submitted data in a MongoDB collection. • Implement a feature to delete all students from the database who have not paid the exam fee (Exam_fee = 0 or null).	2,3	1,2,3,5
3.	(a) Write a JavaScript program using closures to manage student academic data. Accept USN, Subject Code, Subject Name, CIE Marks, and SEE Marks. Use closures to keep CIE and SEE marks private. write a method to compute and return the total marks. Display student details with total. (b) Develop a Node.js application using Express and MongoDB to perform the following tasks: • Create a database named HR with a collection called employees. • Each employee document should include the following fields: emp_name, email, phone, hire_date, job_title, and salary.	2,3	1,2,3,5

	• Design a web form to collect this information from the user and store it in the MongoDB database. • Implement a GET route to display all employee records where the salary is greater than 50,000.		
4.	(a) Write an REACT program to print Name, Address and Company of an Employee. When you Click on CHANGE button, the name and address should be changed. (b) Build an Internship Tracking System using Node.js, Express, and MongoDB with the following requirements: • Create a MongoDB collection to store internship details with fields: Student_ID, Name, Company, Duration, and Status. • Accept internship data through a web form and store it in the database. • Implement a GET route to display all students interning at "Infosys". • Implement a PUT route to update the status when a student's internship is marked as completed.	1,3	1,2,3,5
5.	(a) Write a java script function named pluralize that: • takes 2 arguments, a noun and a number. • returns the number and pluralized form, like "5 cats" or "1 dog". • Make it handle a few collective nouns like "sheep" and "geese". (b) Develop a Node.js application using Express and MongoDB to manage student records with the following features: • Accept student details from a web form: Name, USN, Department, and Grade. • Store the submitted information in a MongoDB database. • Implement a PUT route to update the grade of a student by specifying the Name. • Implement a GET route to display all student records from the database.	2,3	1,2,3,5
6.	(a) Write an REACT program which accepts the Name from the form. As you type, it updates the Name in the page with an h1 tag. (b) Develop a Node.js application using Express and MongoDB to manage hospital data with the following requirements: • Accept and store hospital details: Hospital_ID, Name, Location, Total_Beds, and Occupied_Beds using a web form. • Store this information in a MongoDB collection. • Implement a GET route to display all hospitals where available beds (Total_Beds - Occupied_Beds) are less than 10. • Implement a POST route to admit a patient, which will increment the Occupied_Beds count for the specified hospital.	1,3	1,2,3,5
7.	(a) Write a Node.js program using Express framework to display different branch information offered in an Engineering College with different background color and fonts (Note: Use Routing, Min: 3 branches) (b) Create a Node.js application using Express and MongoDB to manage course enrollments with the following features: • Accept enrollment details through a web form: Student_ID, Name, Course_Name, Duration, and Status.	1,3	1,2,3,5

	• Store the enrollment data in a MongoDB collection. • Implement a GET route to display all active enrollments (Status: "active"). • Implement a PUT route to update the status of an enrollment to "completed" based on Student_ID or Course_Name.		
8.	(a) Write an npm script having a function vowelCount() that takes a string as input and counts number of occurrences of each vowel in the string. (Hint: run the program through npm start) For. Eg. Input : vowelCount('Le Tour de France') Output: a, e, i, o, and u appear, respectively, 1, 3, 0, 1, 1 times (b) Using Node.js, Express, and MongoDB, build a product management system with the following requirements: • Accept product details: Product_ID, Name, Price, Discount, and Stock from a web form. • On insertion, calculate the Final Price using the formula: Final_Price = Price - (Price × Discount / 100) and store it along with the product details in MongoDB. • Implement a GET route to display all products where the Final_Price is less than ₹1000.	3	1,2,3,5
9.	(a) Write a JavaScript function called notBad that takes a single argument, a string. • It should find the first appearance of the substring 'not' and 'bad'. • If the 'bad' follows the 'not', then it should replace the whole 'not'...'bad' substring with 'good' and return the result. • If it doesn't find 'not' and 'bad' in the right sequence (or at all), just return the original sentence. For example: • notBad('This dinner is not that bad!'): 'This dinner is good!' • notBad('This dinner is bad!'): 'This dinner is bad!' (b) Create a Node.js application using Express and MongoDB with the following features: • Accept student details from a web page: User_Name, Branch, and Semester. • Store the data in a MongoDB collection. • Implement a GET route to display all students who belong to the 6th Semester and are from the CSE branch.	3	1,2,3,5
10.	(a) Write a node.js Express program to create a custom middleware functions for i. Logger ii. No. of time the visitor visited the website (b) Develop a Node.js application using Express and MongoDB to create a portal for recording student startup ideas with the following features: • Accept the following details from a web form: ID, Team_Name, Title, Domain, and Funding_Required. • Store the submitted data in a MongoDB collection. • Implement a GET route to display all startup ideas in the "EdTech" domain where the Funding Required exceeds ₹5 lakhs.	3	1,2,3,5

	• Required and Store records. Display all startup ideas in "EdTech" domain needing funding > ₹5 lakhs		
11.	(a) Create a web page with the following characteristics using BOX Model i. h1's have 1px red solid borders, background color #D18C1D, and 10px of space between the content and the border (padding) ii. List items have 15px extra space around them (margin) and background color #C0A9DB iii. Paragraphs are contained in 600px by 400px boxes with 2px black dotted borders and background color #D1D631 (b) Develop an Attendance Management System using Node.js, Express, and MongoDB with the following features: • Create a student database with appropriate fields such as: ○ Student_ID, Name, Course, Total_Attendance, Classes_Attended, and Attendance_Percentage. • Calculate the Attendance_Percentage as: Attendance_Percentage = (Classes_Attended / Total_Attendance) * 100. • Implement a GET route to display all students whose attendance is below 75%.	1,3	1,2,3,5
12.	(a) Write a java script program to implement Stack and Queue using modules (b) Develop an Exam Management System using Node.js, Express, and MongoDB with the following functionality: • Create a student database with appropriate fields such as: Student_ID, Name, Subject, Marks, and Eligibility_Status. • Store the student data in a MongoDB collection. • Implement logic to mark students as "Not Eligible" if their Marks < 20. • Provide a GET route to display the list of students who are not eligible for the exam based on this criterion.	2,3	1,2,3,5

Marks Distribution

Conduction and Result	Write Up	Execution	Viva	Change of Program	Total
Part – a	8	15 Marks	7 Marks	- 5 Marks	50 Marks
Part – b		20 Marks			

Reviewed by

HoD, Dept. of CSE