Project Design Phase-II
Technology Stack (Architecture & Stack)

Date	02 NOVEMBER 2025
Team ID	NM2025TMID00789
Project name	CRM for Jewel Management
Marks	4 Marks

Technical Architecture:

The CRM for Jewellery Inventory System is designed using a web-based client-server architecture. The system integrates Customer Relationship Management (CRM) with Inventory Control to streamline jewellery business operations.

The architecture includes modules such as:
- Customer Management
- Inventory Management
- Sales & Billing
- Notification & Reports

The system is hosted on a cloud-based backend, ensuring scalability, availability, and data security. It also allows third-party API integration for messaging and analytics.

Reference:
https://aws.amazon.com/architecture/

Table-1 : Components & Technologies

S.No	Component Description	Technology
1.	User Interface – Customers and staff interact through a responsive web dashboard.	HTML5, CSS3, JavaScript, React.js
2.	Application Logic – Customer Module manages customer details, purchase history, and loyalty tracking	Node.js/ Express.js
3.	Application Logic – Inventory Module updates stock in real time Integration on transactions.	Node.js / MongoDB
4.	Application Logic – Sales & Billing handles invoices, payment	Express.js / Razorpay API
5.	Notification System – Sends reminders and promotional offers via SMS/Email	Twilio API / SMTP
6.	Reports & Analytics – Generates dashboards showing sales trends and customer activity.	Chart.js / Power BI Integration
7.	Database – Stores all jewellery, customer, and transaction records.	MongoDB / MySQL
8.	Cloud Hosting – Application hosted on scalable cloud infrastructure	AWS EC2 / Google Cloud
9.	File Storage – Stores invoice PDFs and offer templates.	AWS S3 Bucket
10.	External API Integration – For payment, notifications, and analytics.	REST API
11.	Infrastructure (Server / Cloud) – Backend hosted on secure cloud environment	.AWS Cloud (IaaS)

Table-2 : Application Characteristics

S.no	**Characteristics**	**Description**	**Technology**
1.	Open-Source Frameworks	The solution uses open-source technologies for flexibility and cost-efficiency.	React.js, Node.js, Express.js, MongoDB
2.	Security Implementations	Includes authentication, data encryption, and access control for different user roles.	JWT Authentication, HTTPS, Bcrypt
3.	Scalable Architecture	Designed to support multiple branches and a growing customer base.	Cloud-Based Microservices Architecture
4.	Availability	Ensures high uptime using load balancing and redundant servers.	AWS Cloud Load Balancer
5.	Performance	Optimized database queries and caching ensure quick response time.	Redis, Indexed MongoDB Queries