

Information Systems and Data Modeling – IT1090

**Final Assignment**

|  |  |
| --- | --- |
| Title: Construction Management System | |
| Batch Number: 9.2 | Group Number:  **MLB\_09.02\_07** |
| Declaration:  We hold a copy of this assignment that we can produce if the original is lost or damaged.  We hereby certify that no part of this assignment has been copied from any other group’s work or from any other source. No part of this assignment has been written / produced for our group by another person except where such collaboration has been authorized by the subject lecturer/tutor concerned.  Text, letter  Description automatically generatedGroup Members:  …………………………  signature  A drawing on a white surface  Description automatically generated with low confidenceIT22095176 KALUARACHCHI J.I    …………………………  signature  IT22083128LIYANAGE S.D.  Text, whiteboard  Description automatically generatedA blue signature on a black background  Description automatically generated with low confidence  …………………………  signature  IT22095176 WIJEKOON. I.S.    IT22571502 MANCHANAYAKA M.A.N.D.  …………………………  signature  A picture containing text, whiteboard, businesscard  Description automatically generated  IT22096920 TENNAKOON T.A.K.K  …………………………  signature    **Submitted on: 27/05/2023** | |

**Content**

1. **Introduction…………………………………………………………3pg**
2. **Hypothetical Scenario………………………………………………4pg**
3. **Requirement Analysis………………………………………………**

**3.1 Main Requirements**

**3.1.1 Functional Requirements**

**3.2.2 Non-Functional Requirements**

**3.2 Data Requirements**

1. **Entity Relationship (ER Diagram)**
2. **Relational Schema**
3. **SQL Queries**

**6.1 Database Create**

**6.2 Data Store in Database**

1. **Performance Requirement**
2. **Security Requirement**

**Introduction**

A database is essential to construction management systems because it offers a structured and well-organized location to handle and store project-related data. A database functions as a digital backbone in the context of construction management, enabling effective data storage, retrieval, and manipulation and facilitating the efficient functioning of numerous construction processes.

Data from a variety of sources is often stored in construction management databases, including information about the project, the client, the contractor, the subcontractor, the material requirements, cost projections, the project timeline, change orders, and communication records. Construction management systems provide data integrity, consistency, and accessibility for all project stakeholders by organizing data in a database.

A construction management system's database enables users to carry out several functions, including information retrieval and searching, report generation, data analysis, and project progress tracking. Offering a centralized platform for real-time access to and updates on project data, it also enables collaboration and information sharing among team members.

Construction management systems can improve decision-making processes, increase project efficiency, decrease errors, and promote efficient communication and coordination among project participants with the aid of a well-designed and reliable database. In general, the database is a vital part of construction management systems because it allows for efficient management and control of construction projects.

**Hypothetical Scenario**

BuildMaster is a well-known developer of software for managing construction projects and specializes in cutting-edge technological solutions. Their goal is to assist clients in streamlining processes, increasing productivity, and guaranteeing the completion of projects. BuildMaster provides construction management services and software for large-scale commercial projects. BuildMaster has a team of skilled specialists who are knowledgeable about the nuances of the construction business.

BuildMaster gathers project data such as design plans, specifications, and schedules while collaborating directly with the client, architects, and suppliers. They adapt their platform for the construction management system to the needs of the project, offering a central location for teamwork, document exchange, and progress tracking. Throughout the planning and design phases, cooperation is made easier through real-time communication and updates.

BuildMaster provides on-site help and instructs the construction team in the use of the construction management system after the building phase has started. The system has several features, such as document management, change order tracking, progress monitoring, and resource allocation. Through the system, contractors can modify schedules, make modification requests, and submit progress reports. Notifications and alerts ensure a timely response and minimize delays. Project managers can make wise judgments thanks to comprehensive data analytics.

BuildMaster keeps a tight eye on the progress of each project, using cutting-edge tracking and reporting tools to spot bottlenecks, manage risks, and guarantee timeliness and budgetary conformity. Their construction management system facilitates the compilation of comprehensive project documentation, offers real-time access to key project indicators, and connects with other systems for smooth financial administration.

To illustrate how their construction management system enhances collaboration, simplifies project management, and ensures project success, BuildMaster uses this fictitious example. Their platform serves as the hub for all project-related operations and provides real-time visibility, efficient collaboration, and successful resource management to clients to increase productivity and project success.

**Requirement Analysis**

**3.1 Main Requirements**

**3.1.1 Functional Requirements**

The functional requirements give a description of the primary functions of the website and the interactions between users and the system. This BuildMaster construction management system is being used by five users. Specifically, customers, employees, contractors, architects, and suppliers When it relates to them, they use various methods to access this system.

1. **Customer (can access the front-end of the system)**

User Requirements –

* Customer can login to the system by using their own credentials.
* Customer can view details about architecture.
* Customer can edit own account details.
* Customer can cancel a project.
* Customer can monitor the construction process.
* Customer can contact administrator and supplier.
* Customer can access financial data such as budgets, costs and payment status.

System Requirements –

* System should validate the login credentials entered by the customer.
* System should save updated customer account details.
* System should inform the administrator about the cancellation of the project.
* System should display financial data including budgets, costs and payment status.
* System should display architecture details.

1. **Administrator (can access the back-end of the system)**

User Requirements -

* Administrator login to the website by providing own login credentials.
* Administrator can activate and deactivate user accounts.
* Administrator can approve the project.
* Administrator can approve cancelled projects.

System Requirements –

* System should verify the user login credentials.

1. **Supplier, Contractor, and Architect (can access the back-end of the system)**

User Requirements –

System Requirements –

1. **Developer (can access the back-end of the system)**

User Requirements –

System Requirements –

**3.1.2 Non-Functional Requirements**

The characteristics, constraints, and standards that are necessary for a construction management system's overall performance, usability, dependability, security, and other non-functional features are referred to as non-functional requirements. Non-functional needs for a construction management system are briefly described as follows:

* **Performance**: The system should be able to accommodate multiple users concurrently without experiencing any noticeable delays, handle huge volumes of data efficiently, and provide quick responses.
* **Usability**: To make it simple to use for both technical and non-technical users, the system should have a user-friendly interface, easy navigation, and clear instructions.
* **Reliability**: The system must have a high degree of dependability, with little downtime and the capacity to bounce back rapidly from errors or interruptions. Additionally, it must operate with data accuracy and integrity.
* **Security**: To prevent unauthorized access, alteration, or theft of important construction data, the system should include strong security features. It comprises data encryption, access limits, user authentication, and routine security upgrades.
* **Scalability**: The system needs to be scalable to handle growing project requirements, an expanding user base, and growing data volumes. It ought to be able to manage extra resources and adjust to shifting requirements without noticeably degrading performance.
* **Interoperability**: The system should allow for integration with other pertinent systems or tools frequently used in the construction sector, such as communication platforms, project management tools, scheduling software, and accounting systems.
* **Maintainability**: To make upgrades, bug fixes, and enhancements simple, the system should be designed with modularity and maintainability in mind. It comprises clean code, a transparent system architecture, and effective system modification support.
* **Compliance**: The system must abide by all applicable industry norms, regulatory requirements, and data protection standards, including construction rules, safety regulations, privacy legislation, and industry-specific standards.
* **Accessibility**: To ensure inclusive use and to accommodate users with visual, aural, motor, or cognitive impairments, the system should be accessible to people with disabilities.
* **Performance Monitoring**: The system must include mechanisms for tracking its performance, gathering pertinent data, and producing reports that may be used to analyze system utilization, spot bottlenecks, and better allocate resources.

**3.2 Data Requirements**

* **CUSTOMER Entity**
* Cus\_id
* First\_Name
* Last\_Name
* Email
* Gender
* Town
* Street\_Name
* Add\_No
* DOB
* Age
* **ADMINISTRATOR Entity**
* Admin\_id
* First\_Name
* Last\_Name
* Email
* **ACCONT Entity**
* Acc\_id
* Acc\_Password
* Use\_Name
* Acc\_Type
* **SUPPLIER Entity**
* Sup\_id
* Sup\_Name
* Email
* Phone
* Sup\_Com\_Name
* Street\_Name
* Add\_No
* Town
* **CONTRACTOR Entity**
* Cont\_id
* Cont\_Name
* Phone
* Email
* Con\_Com\_Name
* Street\_Name
* Add\_No
* Town
* **ARCHITECTURE Entity**
* Archi\_id
* Arch\_Name
* Phone
* Email
* Arch\_Com\_Name
* Street\_Name
* Add\_No
* Town
* **PAYMENT Entity**
* Pay\_id
* Pay\_Price
* Pay\_Type
* Pay\_Emp
* Receipt
* **PROJECT Entity**
* Pro\_id
* Pro\_Price
* Pro\_Type
* Pro\_Name
* Location
* **DESIGN Entity**
* De\_id
* De\_Name
* Description
* **MATERIAL Entity**
* Mat\_id
* Mat\_Name
* Quantity
* **INVENTORY Weak Entity**
* Inv\_id
* Inv\_Name
* Inv\_Location
* Quantity

**A picture containing drawing, sketch, diagram, pattern

Description automatically generated Entity Relationship (ER Diagram)**

**Relational Schema**

**SQL Queries**

**6.1 Database Create**

/\*Table customer\*/

CREATE TABLE Customer

(

Cus\_id varchar(15) not null,

First\_Name varchar(40) not null,

Last\_Name varchar(40) not null,

Gender varchar(10) not null,

DOB date not null,

Add\_no varchar (30) not null,

Sreet\_Name varchar (30) not null,

Town varchar (35) not null,

Email varchar (40) CHECK (Email LIKE '%\_@\_\_%.\_\_%' )not null,

Admin\_id varchar (15) not null,

CONSTRAINT CUS\_ID\_PK PRIMARY KEY(Cus\_id)

);

/\*Table Customer Phone\*/

CREATE TABLE Customer\_Phone

(

Cus\_id varchar(15) not null,

Phone int not null,

CONSTRAINT CUS\_PHONE\_PK PRIMARY KEY(Cus\_id),

CONSTRAINT CUS\_PHONE\_FK FOREIGN KEY(Cus\_id) References Customer

(Cus\_id)

);

/\*Table Administrator\*/

CREATE TABLE Administrator

(

Admin\_id varchar(15) not null,

First\_name varchar(40) not null,

Last\_Name varchar(40) not null,

Email varchar (40) CHECK (Email LIKE '%\_@\_\_%.\_\_%' )not null,

CONSTRAINT Administrator\_PK PRIMARY KEY (Admin\_id)

);

/\*Table Account\*/

CREATE TABLE Account

(

Acc\_id varchar(15) not null,

Use\_Name varchar(40) not null,

Acc\_Password varchar(15) not null,

Acc\_Type varchar(20) not null,

CONSTRAINT Account\_PK PRIMARY KEY (Acc\_id)

);

/\*Table Supplier\*/

CREATE TABLE Supplier

(

Sup\_id varchar(15) not null,

Sup\_name varchar(40) not null,

Email varchar(40) CHECK (Email LIKE '%\_@\_\_%.\_\_%' )not null,

Sup\_com\_name varchar(20) not null,

Add\_no varchar (30) not null,

Sreet\_Name varchar (30) not null,

Town varchar (35) not null,

CONSTRAINT Supplier\_PK PRIMARY KEY (Sup\_id)

);

/\*Table Supplier Phone\*/

CREATE TABLE Supplier\_Phone

(

Sup\_id varchar(15) not null,

Phone int not null,

CONSTRAINT SUP\_PHONE\_PK PRIMARY KEY(Sup\_id),

CONSTRAINT SUP\_PHONE\_FK FOREIGN KEY(Sup\_id) References Supplier

(Sup\_id)

);

/\*Table Material\*/

CREATE TABLE Material

(

Mat\_id varchar(15) not null,

Mat\_Name varchar(30) not null,

Quantity int not null,

CONSTRAINT Material\_PK PRIMARY KEY (Mat\_id)

);

/\*Table Contractor\*/

CREATE TABLE Contractor

(

Cont\_id varchar(15) not null,

Cont\_Name varchar(40) not null,

Email varchar (40) CHECK (Email LIKE '%\_@\_\_%.\_\_%' )not null,

Con\_com\_Name varchar(10) not null,

Add\_No varchar (30) not null,

Sreet\_Name varchar (30) not null,

Town varchar (35) not null,

Archi\_id varchar (15) not null,

CONSTRAINT Cont\_id\_PK PRIMARY KEY(Cont\_id)

);

/\*Table Contractor Phone\*/

CREATE TABLE Contractor\_Phone

(

Cont\_id varchar(15) not null,

Phone int not null,

CONSTRAINT CONT\_PHONE\_PK PRIMARY KEY(Cont\_id),

CONSTRAINT CONT\_PHONE\_FK FOREIGN KEY(Cont\_id) References Contractor

(Cont\_id)

);

/\*Table Architect\*/

CREATE TABLE Architect

(

Arch\_id varchar(15) not null,

Arch\_Name varchar(40) not null,

Add\_No varchar (30) not null,

Sreet\_Name varchar (30) not null,

Town varchar (35) not null,

Email varchar (40) CHECK (Email LIKE '%\_@\_\_%.\_\_%' )not null,

Arch\_com\_Name varchar(10) not null,

CONSTRAINT Arch\_id\_PK PRIMARY KEY(Arch\_id)

);

/\*Table Architect Phone\*/

CREATE TABLE Architect\_Phone

(

Arch\_id varchar(15) not null,

Phone int not null,

CONSTRAINT ARCH\_PHONE\_PK PRIMARY KEY(Arch\_id),

CONSTRAINT ARCH\_PHONE\_FK FOREIGN KEY(Arch\_id) References Architect

(Arch\_id)

);

/\*Table Payment\*/

CREATE TABLE Payment

(

Pay\_id varchar(15) not null,

Pay\_Type varchar(10) not null,

Pay\_Emp varchar(50) not null,

Pay\_Price FLOAT not null,

Recipt varchar(50) not null,

Pro\_id varchar(15) not null,

Cus\_id varchar(15) not null,

CONSTRAINT Pay\_id\_PK PRIMARY KEY(Pay\_id)

);

/\*Table Project\*/

CREATE TABLE Project

(

Pro\_id varchar(15) not null,

Pro\_Name varchar(40) not null,

Pro\_Type varchar(15) not null,

Pro\_Price FLOAT not null,

Location varchar(20) not null,

CONSTRAINT Pro\_id\_PK PRIMARY KEY(Pro\_id)

);

/\*Table Design\*/

CREATE TABLE Design

(

De\_id varchar(15) not null,

De\_name varchar(40) not null,

Description varchar(200) not null,

Arch\_id varchar(15) not null,

CONSTRAINT De\_id\_PK PRIMARY KEY(De\_id)

);

/\*Customer login Table\*/

CREATE TABLE Customer\_login\_Account

(

Cus\_id varchar(15) not null,

Acc\_id varchar(15) not null,

Login\_time decimal not null,

CONSTRAINT Customer\_login\_PK PRIMARY KEY (Cus\_id),

CONSTRAINT Customer\_login\_FK1 FOREIGN KEY(Cus\_id) References Customer (Cus\_id)

CONSTRAINT Customer\_login\_FK2 FOREIGN KEY(Acc\_id) References Account (Acc\_id)

);

/\*Table Customer contact contractor\*/

CREATE TABLE Customer\_contact\_contractor

(

Cus\_id varchar(15) not null,

Cont\_id varchar(15) not null,

CONSTRAINT Customer\_contact\_contractor\_PK PRIMARY KEY(Cus\_id),

CONSTRAINT Customer\_contact\_contractor\_FK1 FOREIGN KEY(Cus\_id) References Customer (Cus\_id),

CONSTRAINT Customer\_contact\_contractor\_FK2 FOREIGN KEY(Cont\_id) References Contractor (Cont\_id),

);

/\*Table Customer has project\*/

CREATE TABLE Customer\_has\_project

(

Cus\_id varchar(15) not null,

Pro\_id varchar(15) not null,

CONSTRAINT Customer\_has\_project PRIMARY KEY(Cus\_id),

CONSTRAINT Customer\_has\_project\_FK1 FOREIGN KEY(Cus\_id) References Customer (Cus\_id),

CONSTRAINT Customer\_has\_project\_FK2 FOREIGN KEY(Pro\_id) References Project (Pro\_id),

);

/\*Table Customer view architect\*/

CREATE TABLE Customer\_view\_architect

(

Cus\_id varchar(15) not null,

Arch\_id varchar(15) not null,

CONSTRAINT Customer\_view\_architect\_PK PRIMARY KEY(Cus\_id),

CONSTRAINT Customer\_view\_architect\_FK1 FOREIGN KEY(Cus\_id) References Customer (Cus\_id),

CONSTRAINT Customer\_view\_architect\_FK2 FOREIGN KEY(Arch\_id) References Architect (Arch\_id),

);

/\*Table Supplier login account\*/

CREATE TABLE Supplier\_login\_account

(

Sup\_id varchar(15) not null,

Acc\_id varchar(15) not null,

Login\_time decimal not null,

CONSTRAINT Supplier\_login\_account PRIMARY KEY(Sup\_id),

CONSTRAINT Supplier\_login\_account \_FK1 FOREIGN KEY(Sup\_id) References Supplier (Sup\_id),

CONSTRAINT Supplier\_login\_account\_FK2 FOREIGN KEY(Acc\_id) References Account (Acc\_id),

);

/\*Table Supplier order material\*/

CREATE TABLE Supplier\_order\_material

(

Sup\_id varchar(15) not null,

Mat\_id varchar(15) not null,

CONSTRAINT Supplier\_order\_material\_PK PRIMARY KEY(Sup\_id),

CONSTRAINT Supplier\_order\_material\_FK1 FOREIGN KEY(Sup\_id) References Customer (Sup\_id),

CONSTRAINT Supplier\_order\_material\_FK2 FOREIGN KEY(Mat\_id) References Material (Mat\_id),

);

/\*Table Supplier deal contractor\*/

CREATE TABLE Supplier\_deal\_contractor

(

Sup\_id varchar(15) not null,

Cont\_id varchar(15) not null,

CONSTRAINT Supplier\_login\_account PRIMARY KEY(Sup\_id),

CONSTRAINT Supplier\_login\_account \_FK1 FOREIGN KEY(Sup\_id) References Supplier (Sup\_id),

CONSTRAINT Supplier\_login\_account\_FK2 FOREIGN KEY(Cont\_id) References Contractor (Cont\_id),

);

/\*Table Contractor Login Account\*/

CREATE TABLE Contractor\_Login\_Account

(

Cont\_id varchar(15) not null,

Acc\_id varchar(15) not null,

Login\_time decimal not null,

CONSTRAINT Contractor\_Login\_Account PRIMARY KEY(Cont\_id),

CONSTRAINT Contractor\_Login\_Account\_FK1 FOREIGN KEY(Cont\_id) References Contractor (Cont\_id),

CONSTRAINT Contractor\_Login\_Account\_FK2 FOREIGN KEY(Acc\_id) References Account (Acc\_id),

);

/\*Table Inventory\*/

CREATE TABLE Inventory

(

Inv\_id varchar(15) not null,

Inv\_name varchar(40) not null,

Inv\_Location varchar(20) not null,

Quantity int not null,

Cont\_id varchar(15) not null,

Mat\_id varchar(15) not null,

CONSTRAINT Inventory\_PK PRIMARY KEY (Inv\_id)

);

**6.2 Data Store in Database**

**Performance Requirement**

* All the system's modules and features ought should load quickly.
* Quick and effective data retrieval is required.
* The system's many sections should be easy for users to navigate.
* The system ought to be able to manage massive amounts of data.
* There shouldn't be a material decrease in performance when numerous users are using it concurrently.
* The system needs to be scalable to handle growing project sizes and data volumes.
* To reduce downtime and system failures, the system should be extremely dependable.
* It must be equipped with systems that enable speedy restoration after disruptions.
* Users should always have easy access to the system.
* The system should constantly be accessible to users.
* To keep data accurate, it should have strong data validation methods.
* The system ought to offer safe access controls and shield private data from intrusion.
* Other equipment and systems used in the building business should smoothly interface with the system.
* It ought to encourage data sharing and cooperation among many stakeholders.
* Integration with document management systems, BIM programs, scheduling tools, and accounting software should be possible.
* The system ought to provide thorough reporting capabilities on project status, finances, and key performance indicators.
* Users ought to be able to produce accurate and current reports.
* To support data-driven decision-making, customizable dashboards, and data visualization should be offered.
* The interface of the system should be simple and easy to use.
* Users should be able to navigate and complete tasks with the least amount of assistance and training.
* Provide clear instructions and error messages when required.
* The system ought to be usable on mobile devices and accessible.

**Security Requirement**

* Strong authentication and access controls are enforced by the construction management system to confirm user identification and restrict access to legitimate users for project data and functionality.
* Strong encryption methods are used by construction management systems to secure sensitive data, providing complete security for both sent and stored information (data in transit).
* Role-based permissions are used in the construction management system to assign users different levels of access and privileges depending on their roles, preventing illegal access to crucial project data and functions.
* For auditing, analysis, and spotting security lapses or suspect activity, construction management systems keep meticulous logs of user actions, system events, and data changes.
* By using protocols like HTTPS, construction management systems provide secure connectivity, protecting users' devices from hacking and data interception.
* For the construction management system to resolve vulnerabilities, defend against new threats, and ensure long-term security, regular software updates, and patches are essential. Monitoring and upkeep must always be done.
* In addition to a disaster recovery plan for speedy system restoration in the event of events or data loss, the construction management system employs reliable backup techniques to regularly make and store project data copies.
* By educating staff members about best practices for security, the construction management system lowers the risks associated with malicious activity and human error.