WEB-BASED DATABASE PROJECT

TITLE: VEHICLE AUCTION SYSTEM

GRADUATE PROJECT REPORT

Submitted by

NAME: PREETHAM PABBA

CWID: A20388769

Under the guidance of

Dr. Johnson Thomas

Department of Computer Science

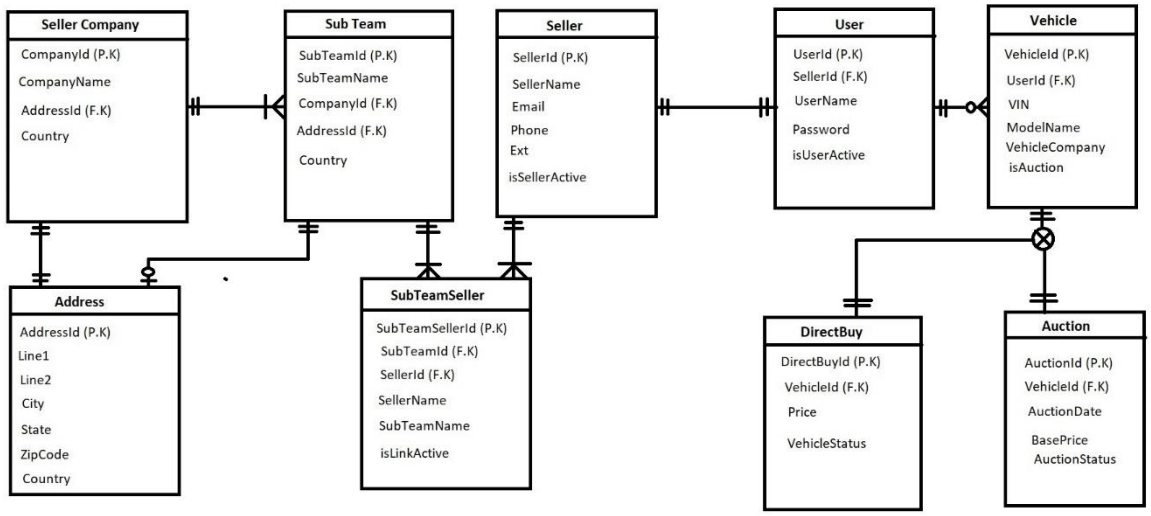
Oklahoma State University

(2022)

INTRODUCTION:

Vehicle Auction System application allows users/ external companies to create their profile under their companies, sub-companies and register their employees and present their vehicles on the website, where buyers can go through the vehicles at respective dates and participate in auction or direct buy to buy the vehicle. The application acts as an interface between sellers and buyers, where seller register their vehicles and buyers can move to the auction place at mentioned data and time to participate in auction, if they are interested in any vehicle.

ER MODEL:



The ER model shows the connection between tables and relation between each entity. The Employee can first create a company, where the address of company will be stored in address table. A Company can have multiple Sub-Companies or Sub-teams based on location, work, projects, etc. Hence, we can create Sub-Team under a company, which can have different address, or if address is not mentioned, we can get the details from the default company address. Now an employee i.e., seller, to access the application, should be registered under a company, where the Seller can be part of multiple Sub-Teams, but should be under one company only. We can understand that the seller can have multiple sub-teams whereas a sub-team can have multiple sellers. To store the relation between a seller and sub-team, we are using the table SubTeamSeller, which mention the link records and the status between the seller and sub-team.

For a seller to register a vehicle in the application, needs access, so each seller will create a user credentials. The user credentials for each seller will be stored in the User table. By using the user credentials, the seller will login to the application to register vehicle and view the status of the vehicle.

The User will have two options when he registers the vehicle, first option is DirectBuy, where if user selects directBuy option, the vehicle won't participate in the auction. If buyer is interested in the vehicle, he will buy the vehicle directly, if he is satisfied the vehicle and vehicle price. The second option is Auction, where the vehicle will go through auction from base price mentioned by seller on the auction date mentioned by the seller. So, the basic vehicle details will be stored in Vehicle table, where based on option other data will be stored in DirectBuy table or Auction table.

When Seller is made Inactive, then the user related to seller should also be inactive. The links between the seller and subteam should also be inactive. Because, if seller (employee) in active due to some reasons, he should not be able to access the application using user credentials and the connection between seller and sub team should also be inactive. If an inactive user tries to login to the application, gives as invalid credentials. If the user wants to access the application, he needs to again active the seller, then the links between the seller and sub team will be made active again and the user will be active to login and create vehicle.

SCHEMA DESCRIPTION:

Seller Company Creation: When a user wants to register their company in the application, should enter details like company name which should be unique, address details. The address will be stored in address table, and we connect the company table and address table using addressId which acts as foreign key in company table.

Sub Team Creation: As mentioned, a company can have different sub teams based on different criteria, user of the respective company can create Sub Team under one company by passing details like sub team name, companyId, address details. Using company Id which acts like foreign key in sub team table, maintains the relation about under which company, the sub team is being created. Address details are mapped in address table and the addressId is stored in sub team data.

Seller Creation: A Seller profile can be created under a company, where a seller can be part of multiple sub teams, as all sub teams come under one company. A Seller can create a profile by entering details like name, email, phone number, and list of sub teams under a company, which he can be part of.

We can understand that a seller can be part of multiple sub teams and a sub team can have multiple sellers. So, to store the link between seller and sub team, we are adding the connections in subTeam-seller table where each row describes a link between a seller and sub team.

User Creation: Each Seller needs to create login credentials to access the application to register vehicles to participate in auction or buy. Each Seller can have only one user credential. Hence, when user enter the details like seller (which doesn't have user), username, password.

Vehicle Creation: Seller will login to the application, with their user credentials, and can add vehicle details. When user adds a vehicle, he can have 2 options, if they want their vehicle to participate in auction or should only be displayed as direct buy by buyer. When he adds vehicle, the primary details will be stored in vehicle table and based on condition, the other details will be stored in Auction table or Direct-Buy table using vehicleId as foreign key to map the details to vehicle details.

Relations and Cardinality:

One-to-many Relation: We can see one-many relations between Seller Company to Sub Team and User to Vehicle. Where a company can have multiple sub teams and each subteam can be only under one company. Similarly, each User can create/ register multiple vehicles, but one vehicle can be owned by under one user/seller.

Many-to-many Relation: The many-to-many relation is maintained between Sub-Team and Seller, where a Sub team can have multiple sellers (i.e., multiple employees working under a team/ sub company) and a seller can also be part of multiple sub teams (an employee can be shifted to different teams/ projects under a company).

Weak Entities: Address table is a weak entity. Because, if there is no company or sub team, there is no identity to the address, which it is pointing to. By adding company or sub team mentions the dependency of address on company or sub team.

Sub-type Entities: Direct-Buy and Auction tables are sub types of Vehicle table, where based on decision of vehicle to be part of auction, the vehicle data completes with one of its subtypes.

Triggers: Created a “sellerStatusUpdate” trigger (type of trigger: AFTER UPDATE) for Seller table. When a seller is made Active or Inactive, it should also affect the status of its respective user and the links from the seller to their sub teams. Because when a seller is made inactive, the respective user shouldn’t be able to access the application and the link to subteams should be made inactive and vice-versa. When user tries to login the application we check if the user is active or not. Only if the user is active then they can login to the application and create vehicles.

Views: Created a view “sellerwithuser” which gets the details of seller who has user credentials. Joined the seller and user table on sellerId which retrieves the results. Have used the view on a query to get the sellers which doesn’t have user credentials. So, these data will be shown in dropdown of creation user’s form. As an employee, we shouldn’t be able to create user multiple times for same seller.

Stored Procedures: Created a stored procedure “createVehicle” which inserts vehicle data to the database and based on the user interest of choosing the option of auction or direct buy, the details will be inserted to Auction table or DirectBuy table for its respective vehicleId.

BCNF (BOYCE-CODD NORMAL FORM):

We can observe that the schemas in ER have only single values attributes, unique name for every attribute which proves the tables are in 1NF. Every non-primary key attribute is dependent on primary key like in seller company table we can observe that country, company name is dependent on companyId. In User table, username, password is dependent on user Id. In vehicle table, vehicle details are dependent on vehicleId (primary key). This proves that the tables are in 2NF.

We can observe there is no transitive dependency in any table. We have already normalized address table, where state, country have transitive dependencies. As we separated from company table, there won't be transitive dependency on comapnyId. These states the tables are in 3NF.

We can observe that there is no functional dependency in the tables, where an attribute depends on non-primary keys. In Vehicle table if we consider model name or VIN which are internally dependent but based on user input, we are considering the model's name and VIN detail separately. So functional dependency does not exist in vehicle table also. As the tables are in 3NF and every functional dependency of X -> Y, X is super key of table, it proves that the tables are in BCNF.

IMPLEMENTATION OF DATABASE, TOOLS AND LANGUAGE USED:

Front End:

Languages: Angular framework, HTML, CSS, TypeScript.

Used HTML to create the web page, forms, actions. Added styling using the CSS. Used typescript to pass data from web page to backend server and display data on webpage from backend.

Created Forms using Angular framework for each component, using the forms passed the data entered as an object to the respective service. From service we call the http request to backend server by any one of the CRUD operations.

Backend:

Languages: Spring boot, Java, Mybatis (Jpa)

Used spring boot framework using java language to handle API requests from webpage and process the request. Used Mybatis jpa to connect to database and pass or retrieve the data.

Executed the queries using mapper, where queries are written inside the mapper. Calling the mapper function from service level to execute the respective queries.

Created MVC structure, where API request is taken at respective Controller and passed to service. At Service level, the request is handled and will call mapper files for data retrieval or to pass data.

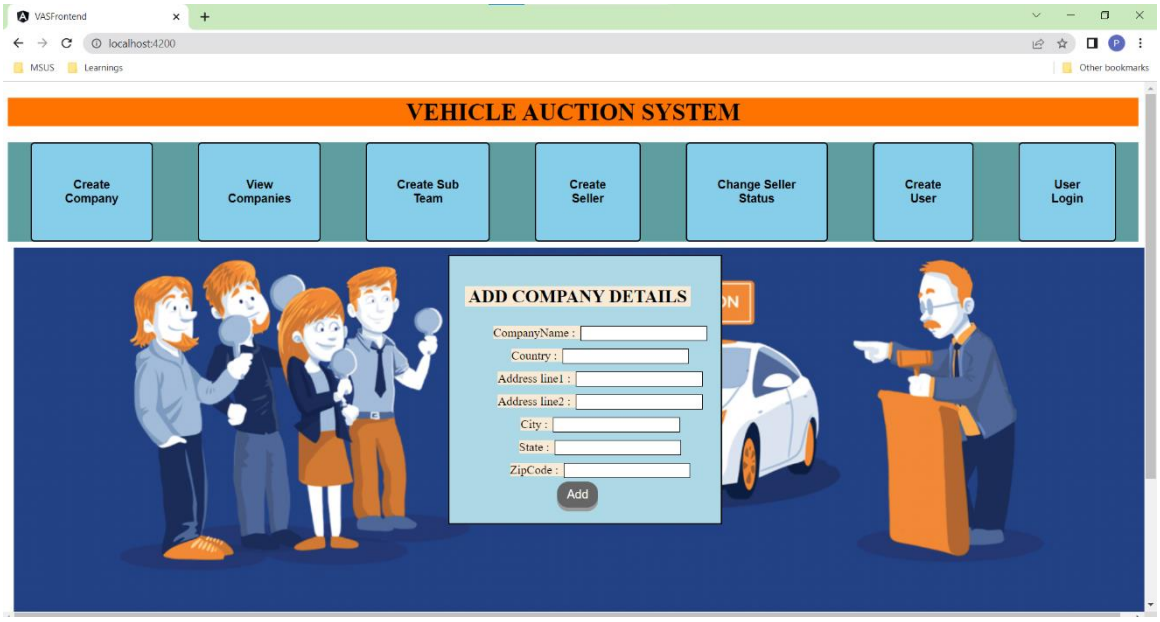
Used MyBatis as JPA framework to connect database and backend. Written queries using mapper structure to execute SQL queries.

Database:

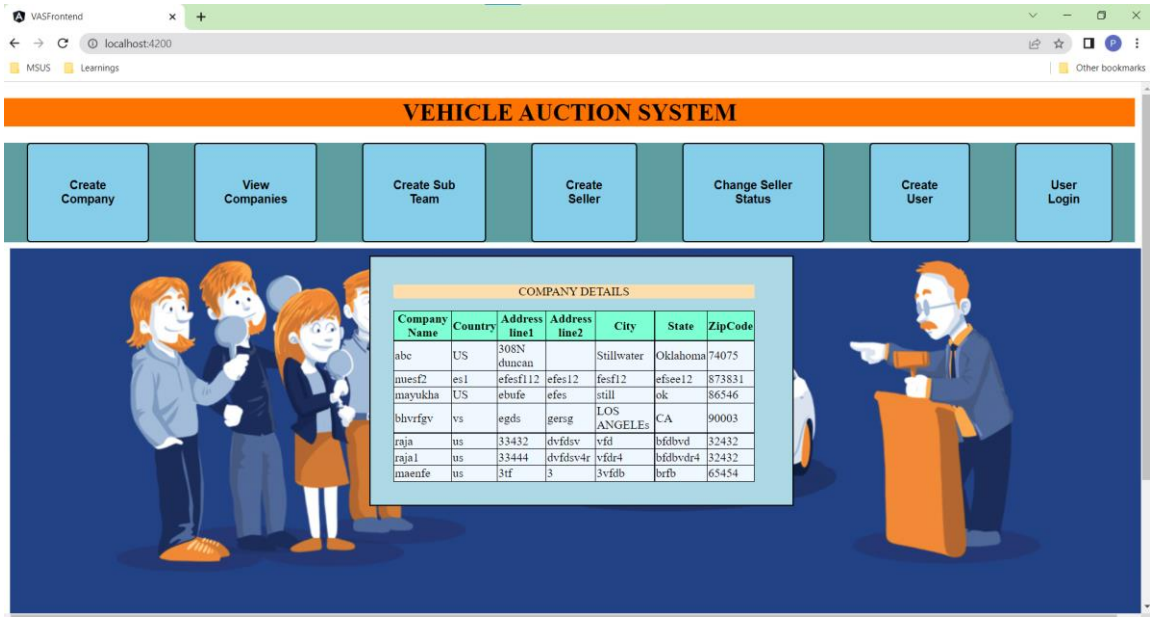
Tool: MySQL Workbench

Created tables in MySQL workbench. Added triggers, views and stored procedures to the tables. Written SQL queries using mybatis mapper to call the database.

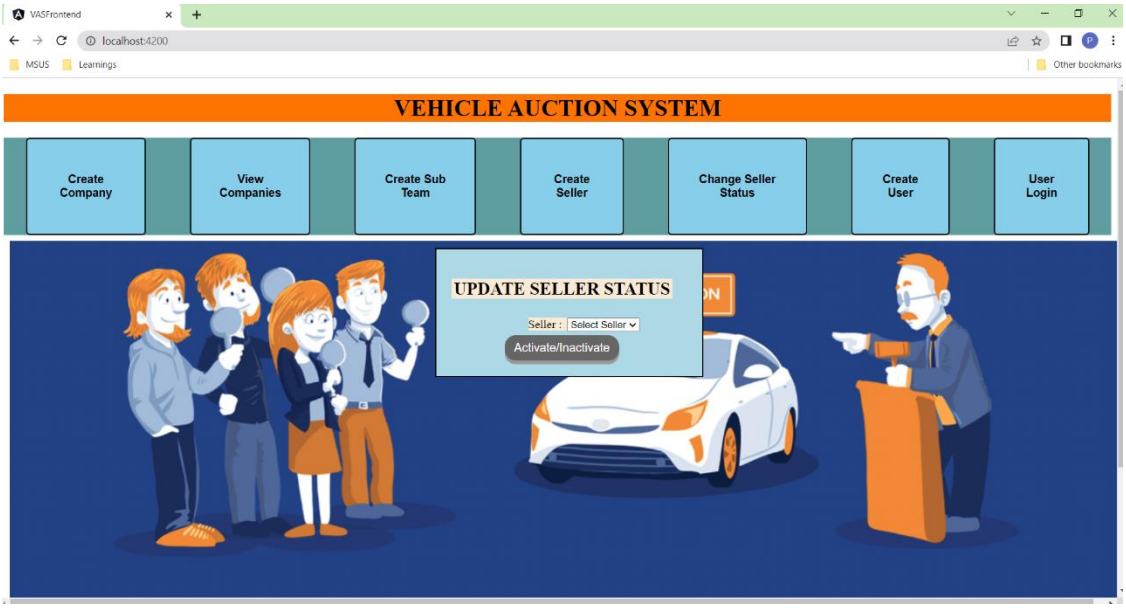
APPENDIX:



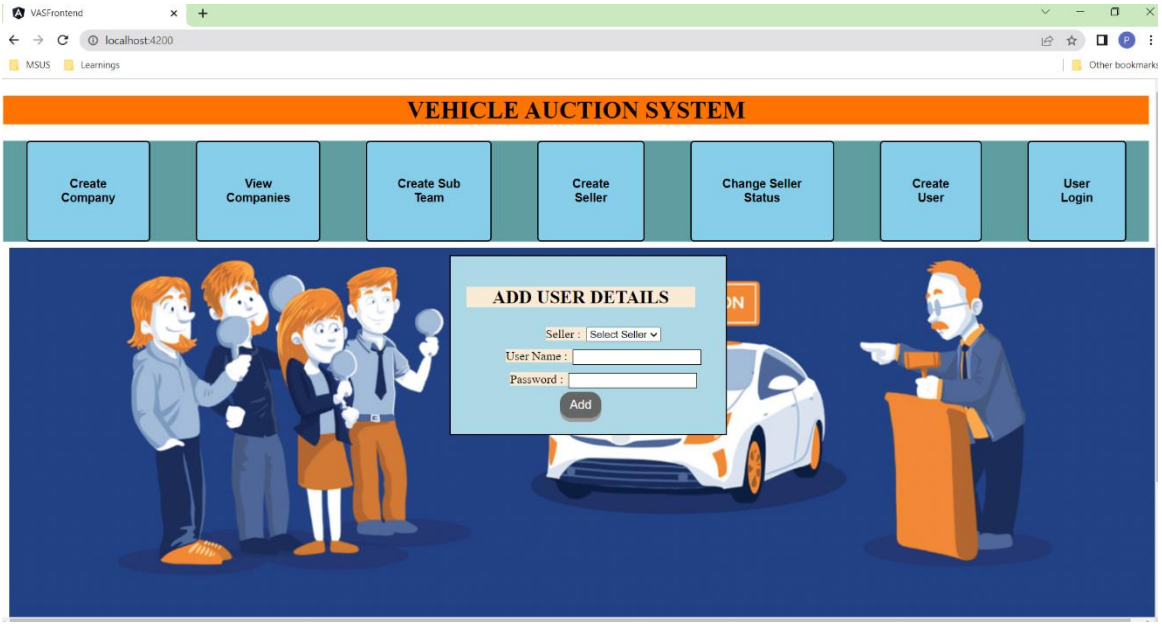
Above picture displays the web page of the application “Vehicle Auction System”. Shows the Create Company form, where user can create the company by entering the details. If user entered correct details, company will be successfully added and shows a success message. If entered invalid details will display error message. Similarly for creation of sub team also replicates the same scenario, where we also need to select the company under which the sub team to be created.



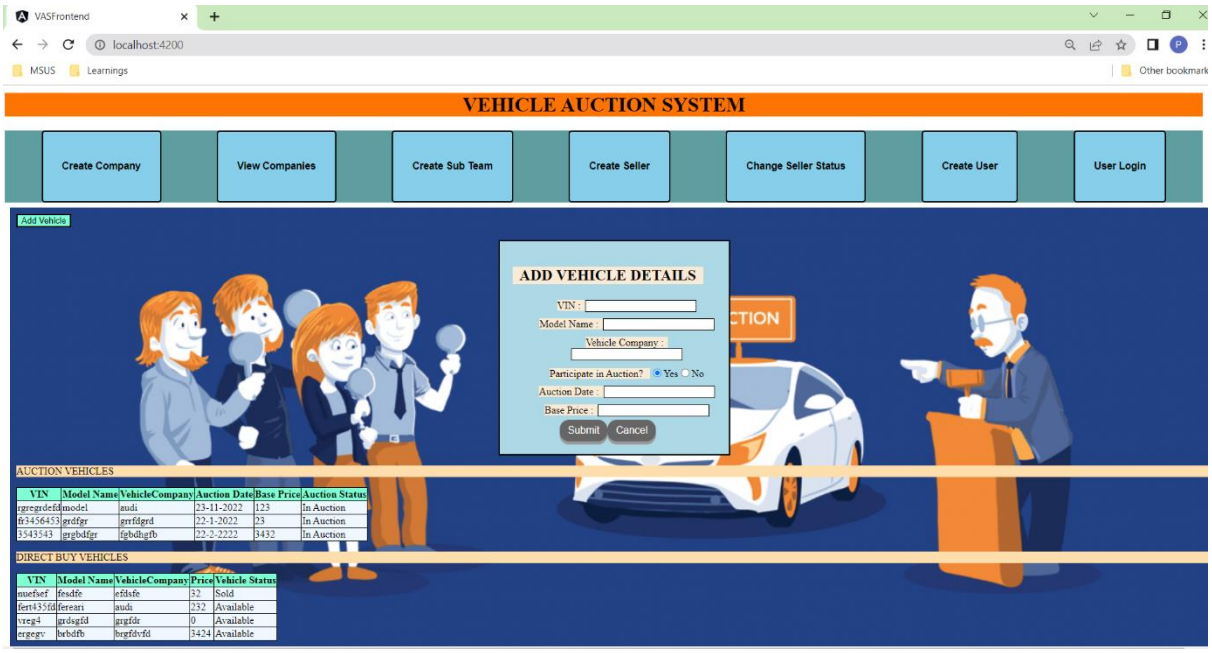
When user clicks on view companies, it displays the companies created and the details of the company.



When user clicks on Change Seller Status, update seller status form displays. User needs to select the seller for which they want to change to seller status from active to inactive or inactive to active. Which also reflects the respective user status and link between sub team and seller status.



Each seller can create user by clicking on Create User by giving details. Once user gets created successfully, can login to the application by clicking on user login to register vehicles and view their previous registered vehicles and its status.



Once user login via user login, it will redirect to above displayed page. It displays Add Vehicle button, whereby clicking on it a vehicle form is displayed. By entering all the details and click on submit, if all entered details are valid, the vehicle will be created and shows a success message, else will show an error message.

User can also observe his Auction Vehicles and Direct Buy vehicles in tables and check the status of the vehicles.