TRIBHUVAN UNIVERSITY

INSTITUTE OF ENGINEERING

PASHCHIMANCHAL CAMPUS, POKHARA

Lamachaur, Pokhara-16

A REPORT ON

"DATABASE PROJECT"

Submitted by:

NAVIN ARYAL PAS077BEI025

NAVRAJ AWASTHI PAS077BEI026

NISCHAL OJHA PAS077BEI027

NISHANTA SHARMA CHAPAGAIN PAS077BEI028

Submitted To:

Department of *Electronics Communication & Information* Engineering

2023-08-06

Abstract:

XYZ, a newly started music company in Pokhara, wants their data to be digitalized for efficiency in adding new data, editing and removing their existing data, and viewing those data. We four Nabin, Navraj, Nischal and Nishanta are acting as a team to build a database with interactive GUI for this company. We are using CustomTkinter, a module of Python to create front end as well as back end of the project. Also, we are using MySQL connector a module of python for creating database. We were kind of successful in making the Music Database Management System in form of a GUI which uses MySQL database meeting the minimum requirements of the client. Our prototype is able to create system users, admins, musician, producer and author account giving them their freedom and ability adding, changing and deleting the records of their songs, albums or instrument they play.

Table of Content:

Project Introduction:

A newly started music company in Pokhara, namely XYZ, is thinking to create a database. Firstly, they want to add new data, edit and remove their existing data, and view those data in user friendly manner. Currently, they are working with data like Musician names, address and phone number, album names, id and the name of producer, songs included in the album with its author, copyright data, song format, and instrument available in their company with its musical key. They want these data to be stored in efficient tables such that the information could be easily extracted when needed. They want a GUI where they can easily visualize the data, if needed create users, grant access levels to do different tasks and manage its data for long run.

We four Nabin, Navraj, Nischal and Nishant are acting as a team to build a database with interactive GUI for this company. We are using CustomTkinter a module of Python to create front end as well as back end of the project. Also, we are using MySQL connector a module of python for creating database. The main objective to develop Music Management System is to overcome the manual errors and make a computerized system. In this project there are various types of modules available to manage Albums, Musicians, and Songs. We can also generate report for Song, Album and Musicians. We can create, add, view, update and delete all attributes from this module. It ensures that the information is safe and secure from unauthorized access as the admin of the database can only access to it.

We created following 4 tables to store all those data, where data duplication is the least. With the use of these primary keys, it is easy to search for certain records in the tables.

Problem Statement:

XYZ wants their data to be managed properly like in adding new data, editing and removing their existing data, and viewing those data. Following is the information given about their needs and desires.

- They decided to store information about musicians who perform on its albums (as well as other company data) in a database. The company has wisely chosen to hire us as a database designer.
- Each musician that records at XYZ has an SSN, a name, an address, and a phone number.
- Each instrument used in songs recorded at XYZ has a unique identification number, a name (e.g., guitar, synthesizer, flute, etc.) and a musical key (e.g., C, B-flat, E-flat, etc.).
- Each album recorded on the XYZ label has a unique identification number, a title, a copyright data and an album identifier.
- Each song recorded a title and an author.
- Each musician may play several instruments, and a given instrument may be played by several musicians.
- Each album has a number of songs on it, but no song may appear on more than one album.
- Each song is performed by one or more musicians, and a musician may perform a number of songs.
- Each album has exactly one musician who acts as its producer. A musician may produce several albums, of course.

Now, we need to design a database based on needs stated above. Also, we need to integrate this database with a GUI.

Objective:

The main objectives of this project are:

- To develop a Music Management System that eradicates the error and easy implementation like add, delete, edit and view the records.
- To design an interactive GUI with Database for the ease to XYZ Music Company.

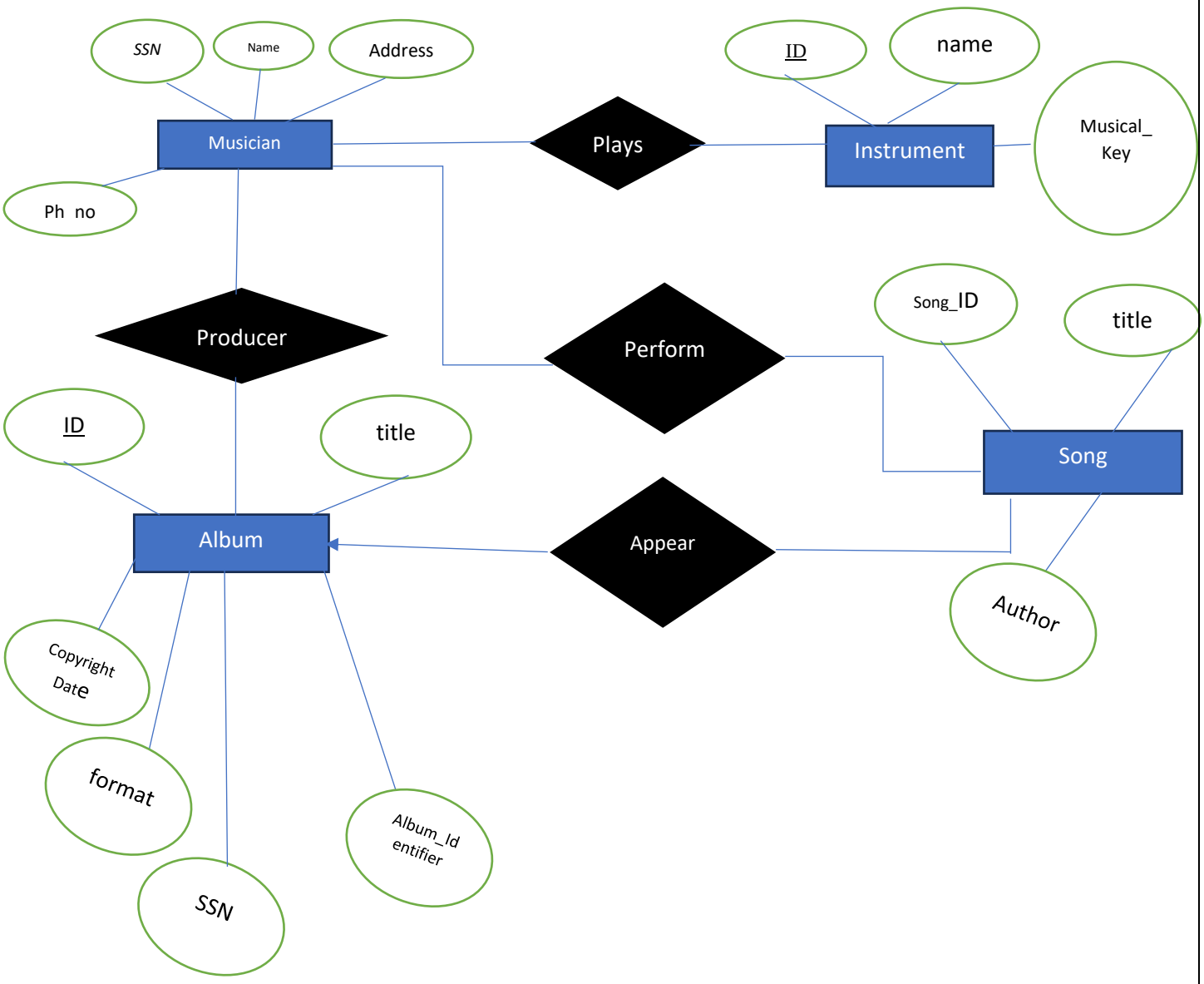
Methodology:

- ***ER DIAGRAM***

ER Diagram stands for entity relationship diagram, also known as entity relationship model that displays the relationship of entity sets stored in a database. ER diagrams are created on three basic concepts: Entities, Attributes and Relationships.

ER diagrams looks similar to the flowchart. However ER diagram includes many specialized symbols, and its meanings makes this model unique. The purpose of ER diagram is to represent the entity framework infrastructures.

Following is the main ER Diagram that is the base of our database used in this project.



- *Relational Schema*

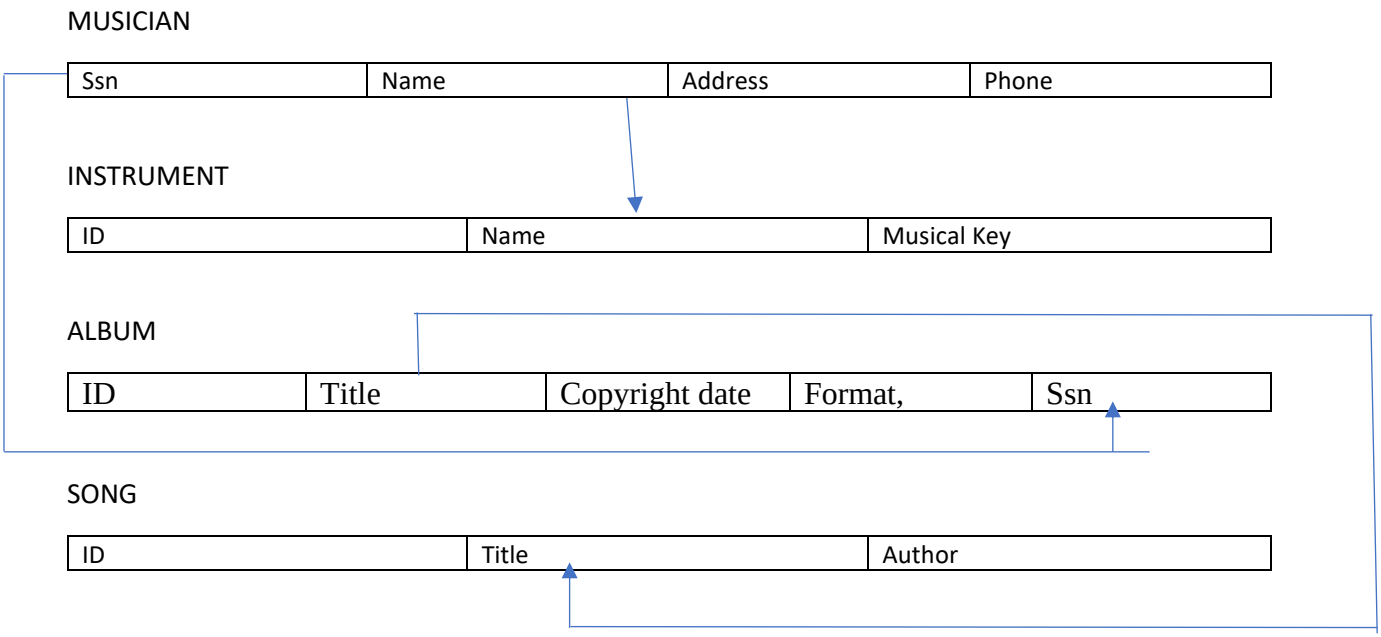
A relational schema is a collection of connected relational table definitions, such as stored base tables or derived views, constraints, and derivation rules. Definitions, constraints, and derivation rule (stored base tables or derived views). All of the base tables, views, indexes, domains, user roles, stored modules, and other items that a user creates to fulfill the data needs of a particular enterprise or set of applications belong to one schema.

A relational schema is a blueprint used in database design to represent the data to be entered into the database and describe how that data is structured in tables (called relations in relational schema). In a relational database program, a relational schema describes the database relationships and structure. It can be displayed graphically or written in the Structured Query Language (SQL) used to build tables in a relational database. In the relational schema, the table, or relation, consists of a set of named, but unsorted, columns (called attributes in relational schema) and an undefined number of unnamed and unsorted rows (called tuples in relational schema). Each row is unique, but the rows can be moved around as needed and stored in any order, modified, or deleted without impacting the efficient operation of the database.

Database Schema of our project "Music Management System" is demonstrated below:

- Musicians (Symbol number, Name, Address, Phone number)
- Instruments (ID, Instrument name, Musical key)
- Songs (ID, Title, Author)
- Album (Album ID, Title, Copyright date, Format, Symbol number)

Relational Schema Diagram: The relational schema diagram of above database schema is shown below:



- *SQL:*

SQL stands for structured Query language. SQL became a standard of the American National Standards Institute (ANSI) in 1986, and of the International Organization for Standardization (ISO) in 1987.SQL is a domain-specific language used in programming and designed for managing data held in a relational database management system, or for stream processing in a relational data stream management system.

The scope of SQL includes data insert, query, update and delete, schema creation and modification, and data access control. SQL is used to communicate with a database.

SQLs used to create our database:

1.Create the database

```
CREATE DATABASE project_sample;
```

Use the database

```
USE project_sample;
```

2.Create the Musician table

```
CREATE TABLE Musician (
    SSN INT PRIMARY KEY,
    Name VARCHAR(100) NOT NULL,
    Address VARCHAR(200) NOT NULL,
    PhoneNumber VARCHAR(15) NOT NULL);
```

3.Create the Instrument table

```
CREATE TABLE Instrument (
    InstrumentID INT PRIMARY KEY,
    Name VARCHAR(50) NOT NULL,
    Musical_Key VARCHAR(10) NOT NULL);
```

4.Create the Album table

```
CREATE TABLE Album (
    Album_ID INT PRIMARY KEY,
    Title VARCHAR(100) NOT NULL,
    CopyrightDate DATE NOT NULL,
    Format VARCHAR(10) NOT NULL,
    Album_Identifier VARCHAR(50) NOT NULL,
    Producer_SSN INT,
```

FOREIGN KEY (Producer_SSN) REFERENCES Musician (SSN) ON DELETE CASCADE);

5.Create the Song table

```
CREATE TABLE Song (
   Song_ID INT PRIMARY KEY,
   Title VARCHAR(100) NOT NULL,
   Author VARCHAR(100) NOT NULL);
```

6.Create the MusicianInstrument table for the many-to-many relationship

```
CREATE TABLE Musician_Instrument (
   Musician_SSN INT,
   InstrumentID INT,
   PRIMARY KEY (Musician_SSN, InstrumentID),
   FOREIGN KEY (Musician_SSN) REFERENCES Musician (SSN) ON DELETE CASCADE,
   FOREIGN KEY (InstrumentID) REFERENCES Instrument (InstrumentID) ON DELETE CASCADE);
```

Create the AlbumSong table for the many-to-many relationship

7.

```
CREATE TABLE Album_Song (
   Album_ID INT,
   Song_ID INT,
   PRIMARY KEY (Album_ID, Song_ID),
   FOREIGN KEY (Album_ID) REFERENCES Album (Album_ID) ON DELETE CASCADE,
   FOREIGN KEY (Song_ID) REFERENCES Song (Song_ID) ON DELETE CASCADE);
```

Result:

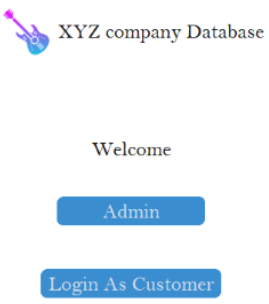


Fig: Front Page

Fig: Admin Page

Fig: View Musician's data

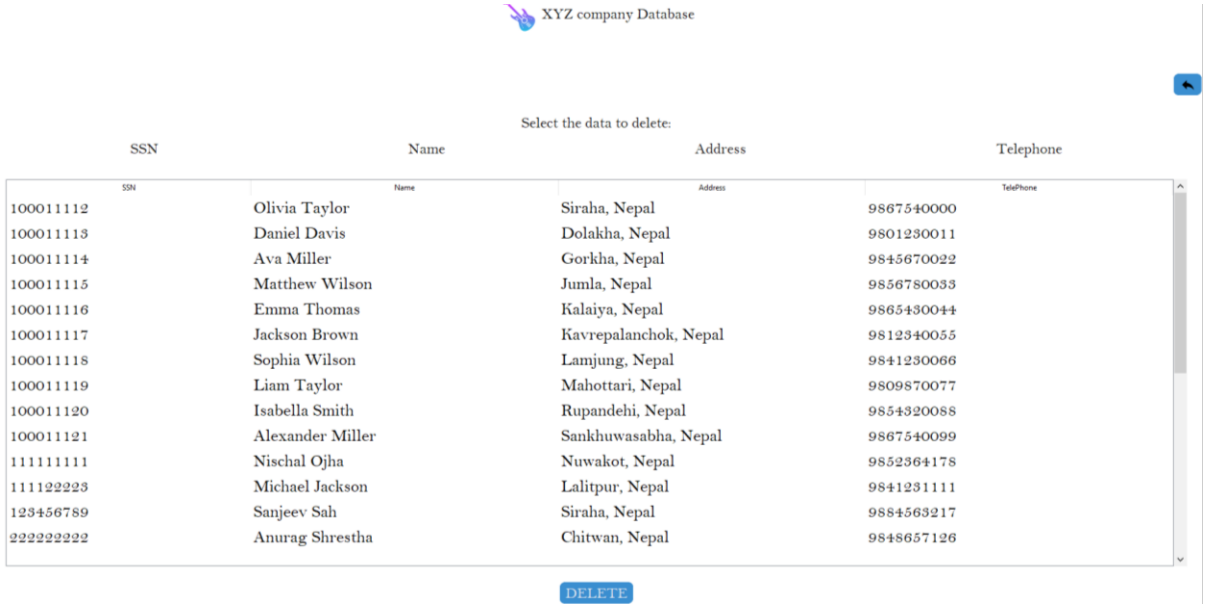


Fig: Delete Musician's Data

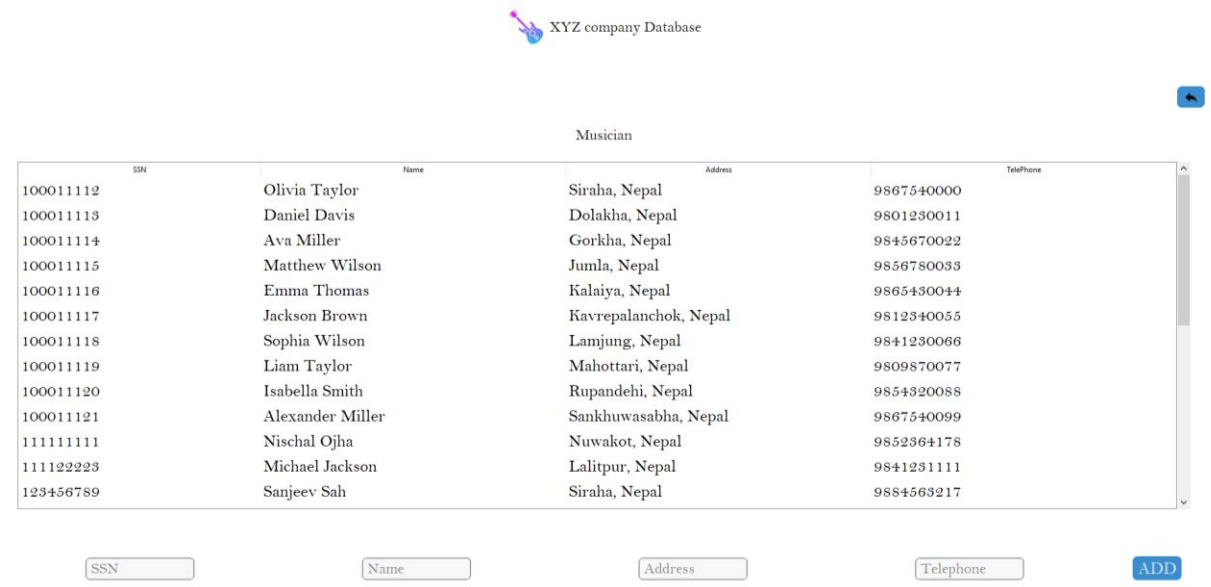


Fig: Add Musician's Data

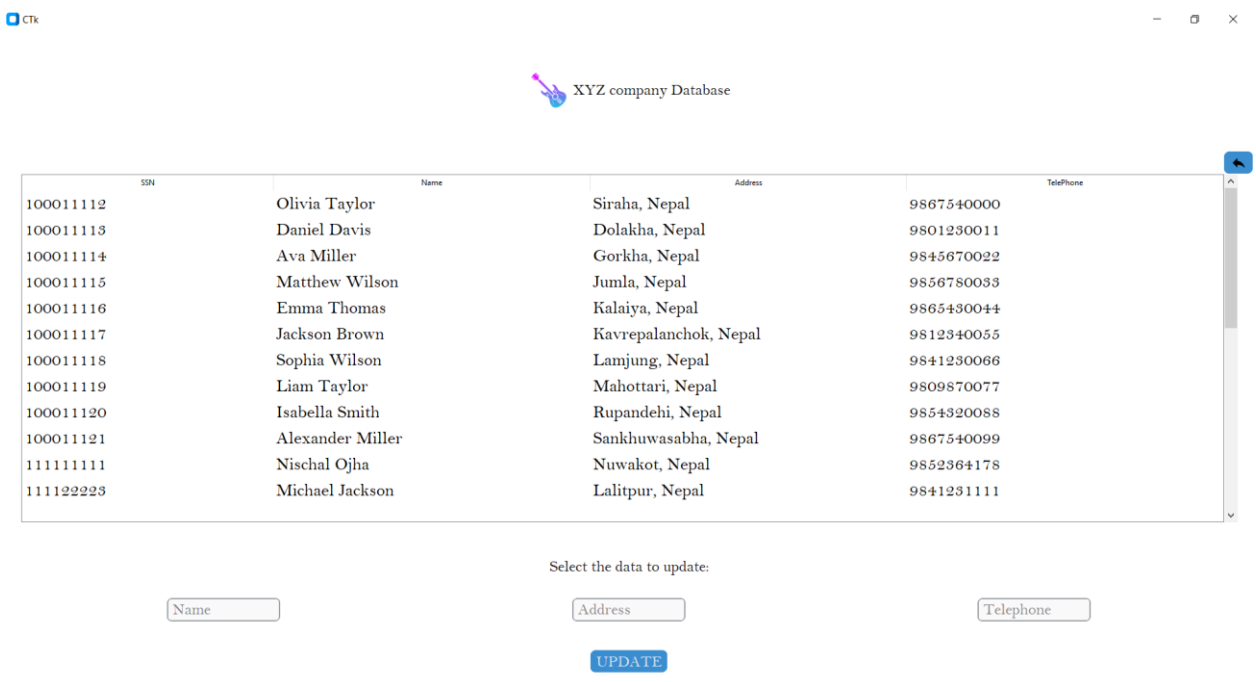


Fig: Update Musician's Data

Fig: View Song's Data

Fig: Update Song's Data

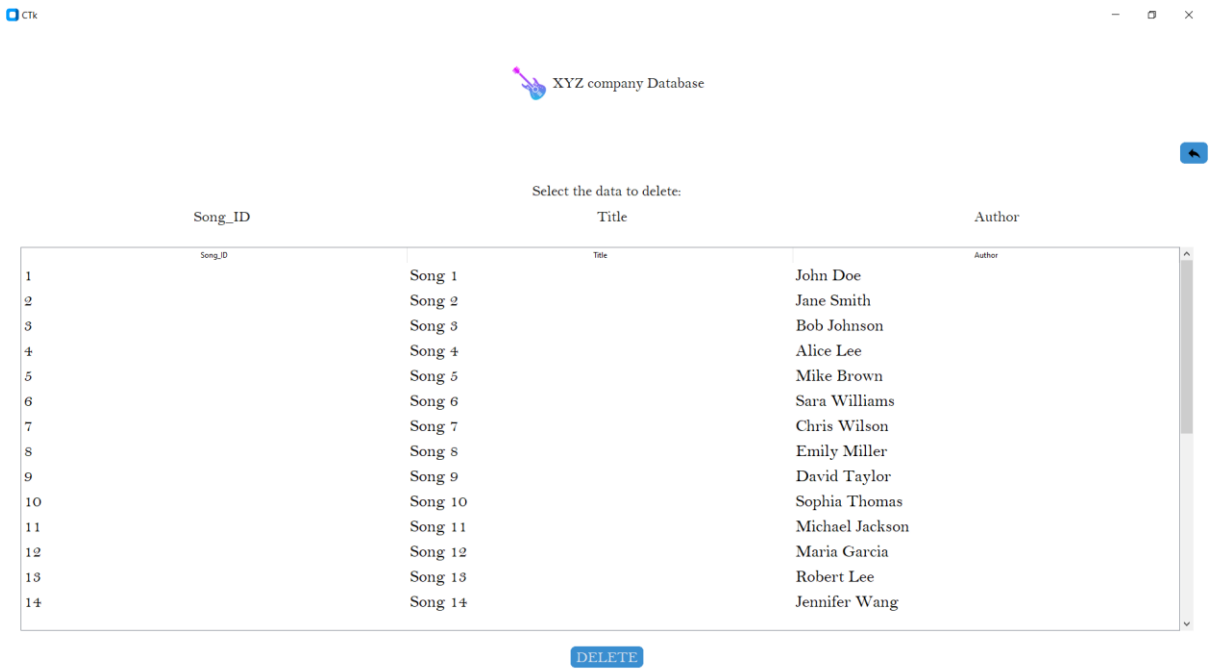


Fig: Delete Song's Data

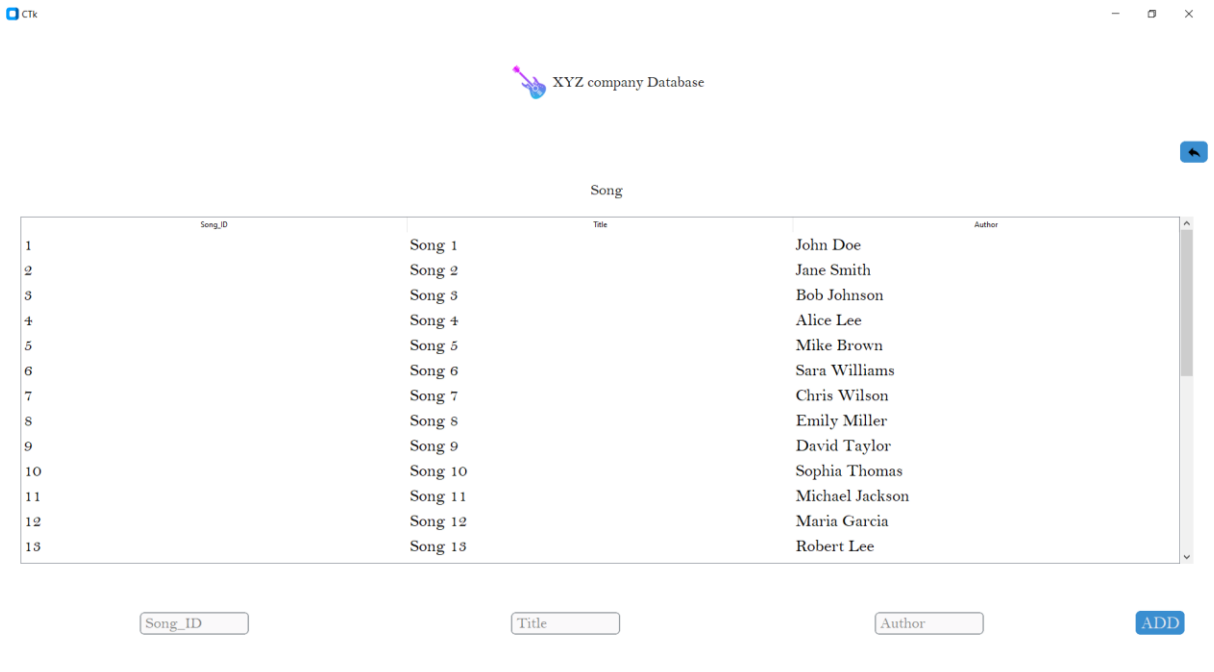


Fig: Add Song's Data

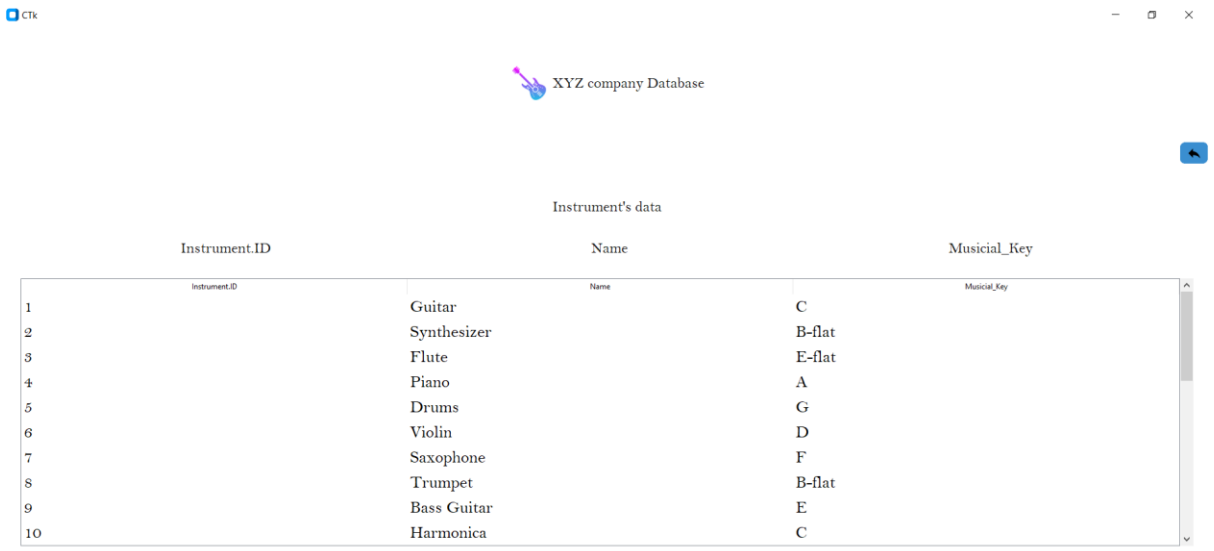


Fig: View Instrumnet Data

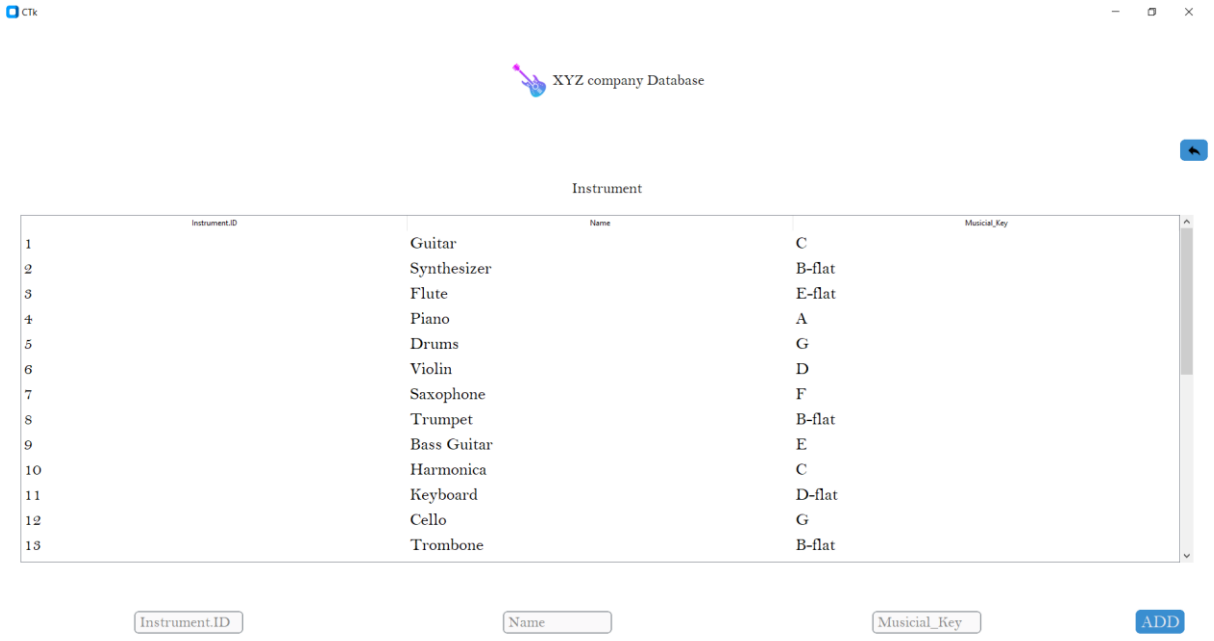


Fig: Add Instrument Data

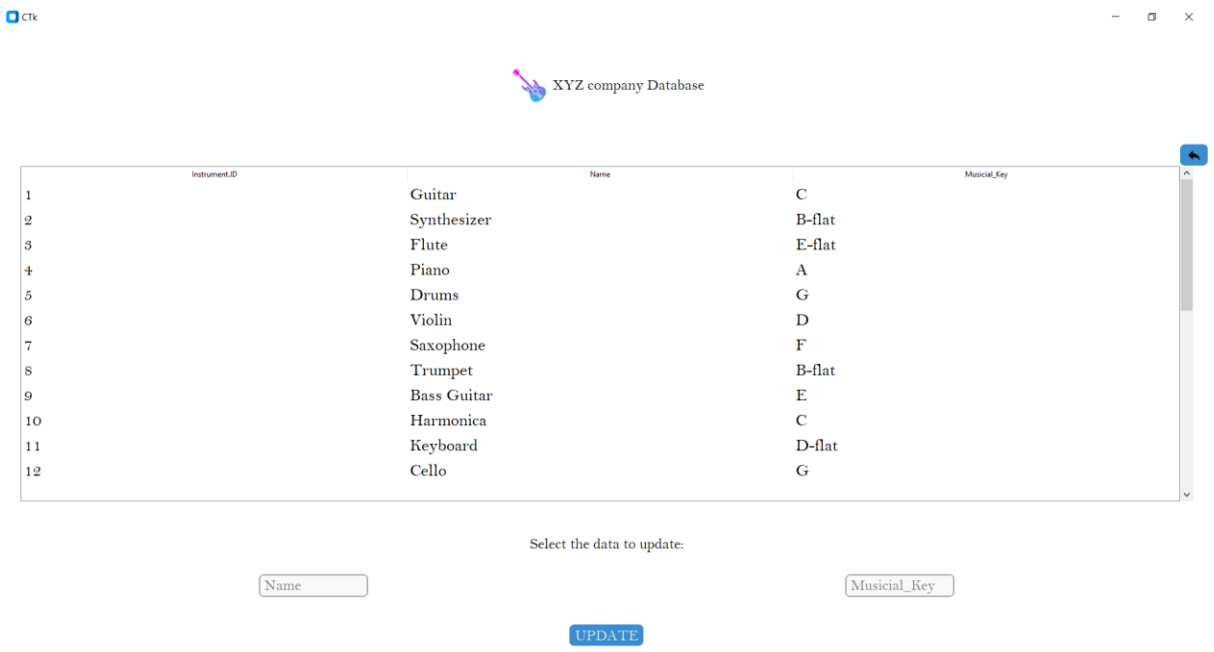


Fig: Update Instrument's Data

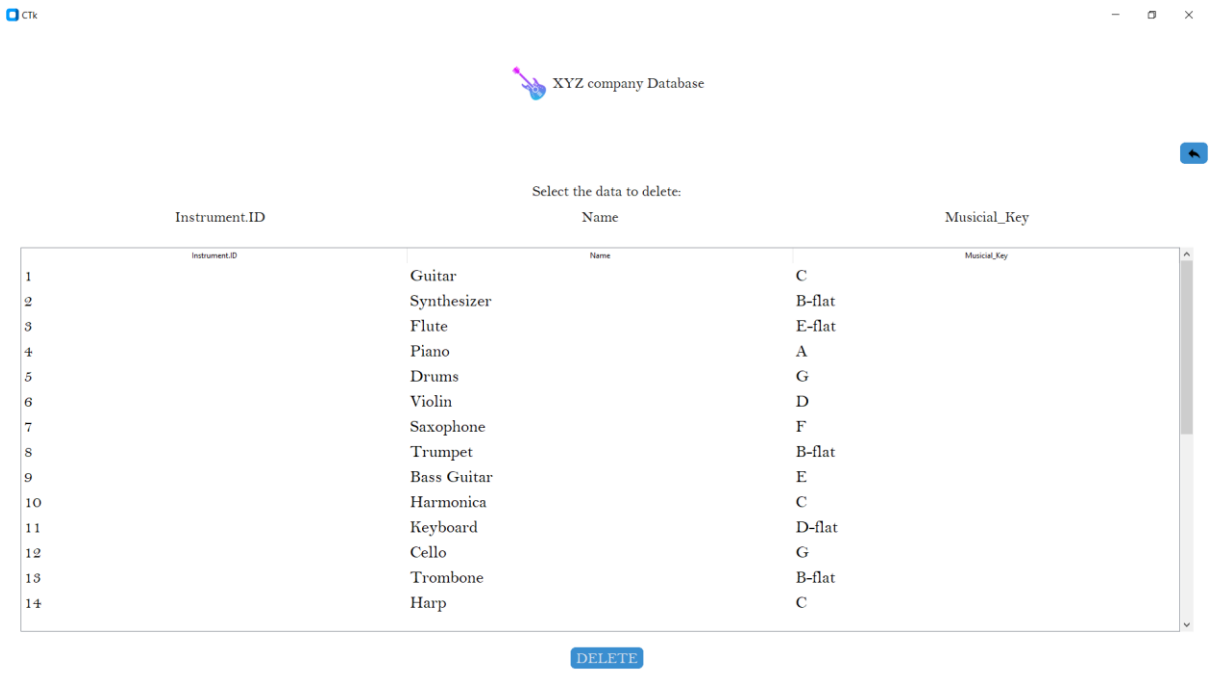


Fig: Delete Instrument's Data

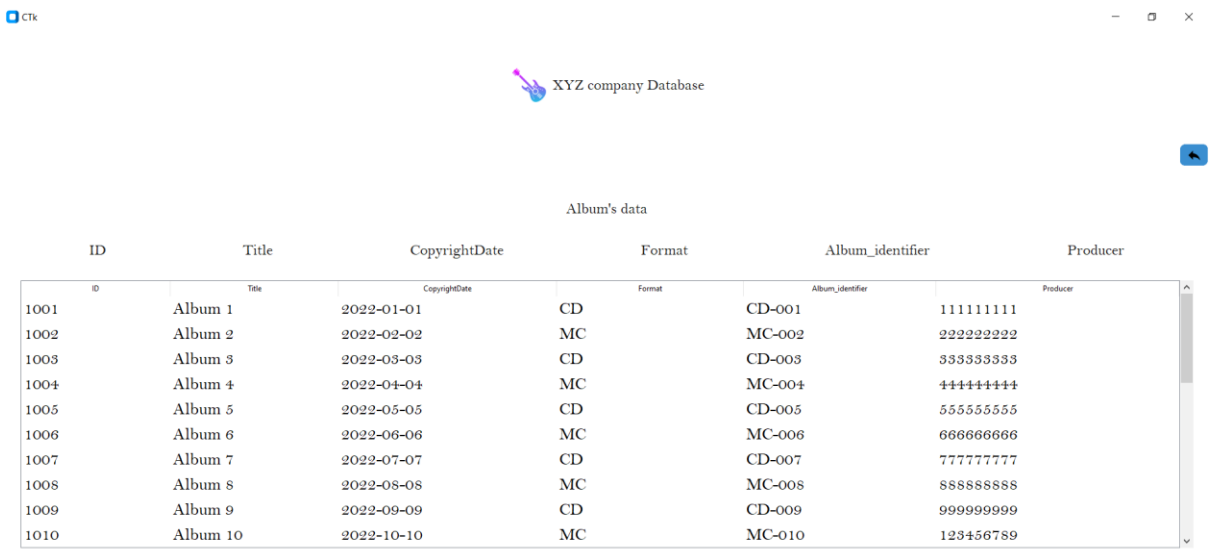


Fig: View Album's Data

Fig: Add Album's Data

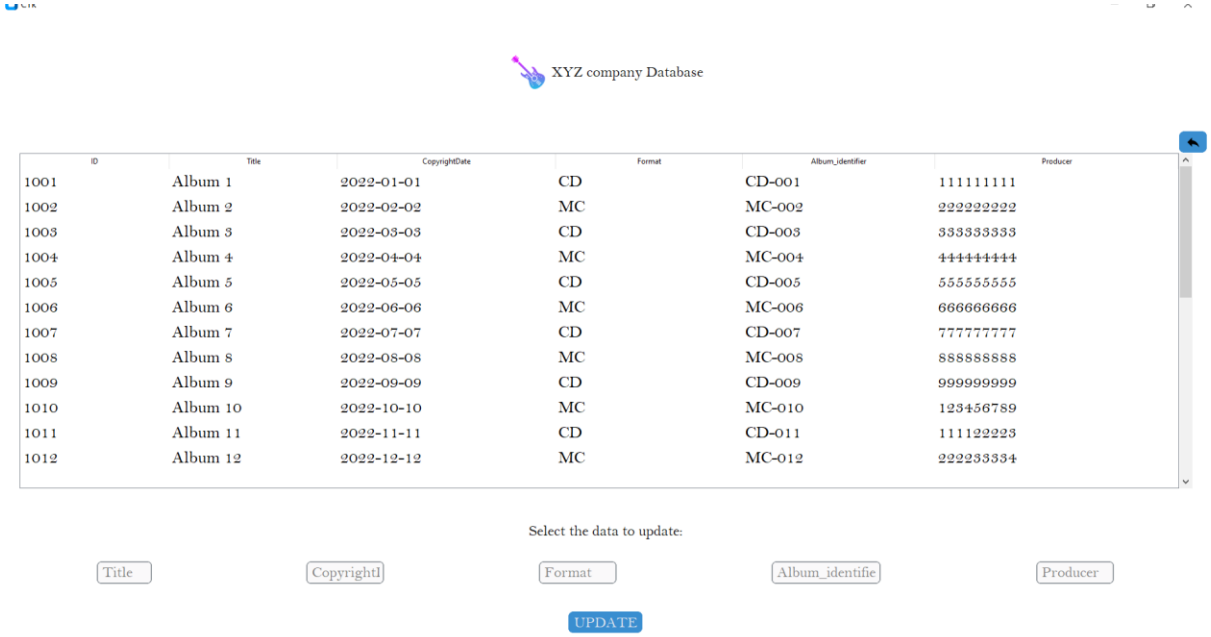


Fig: Update Album's Data

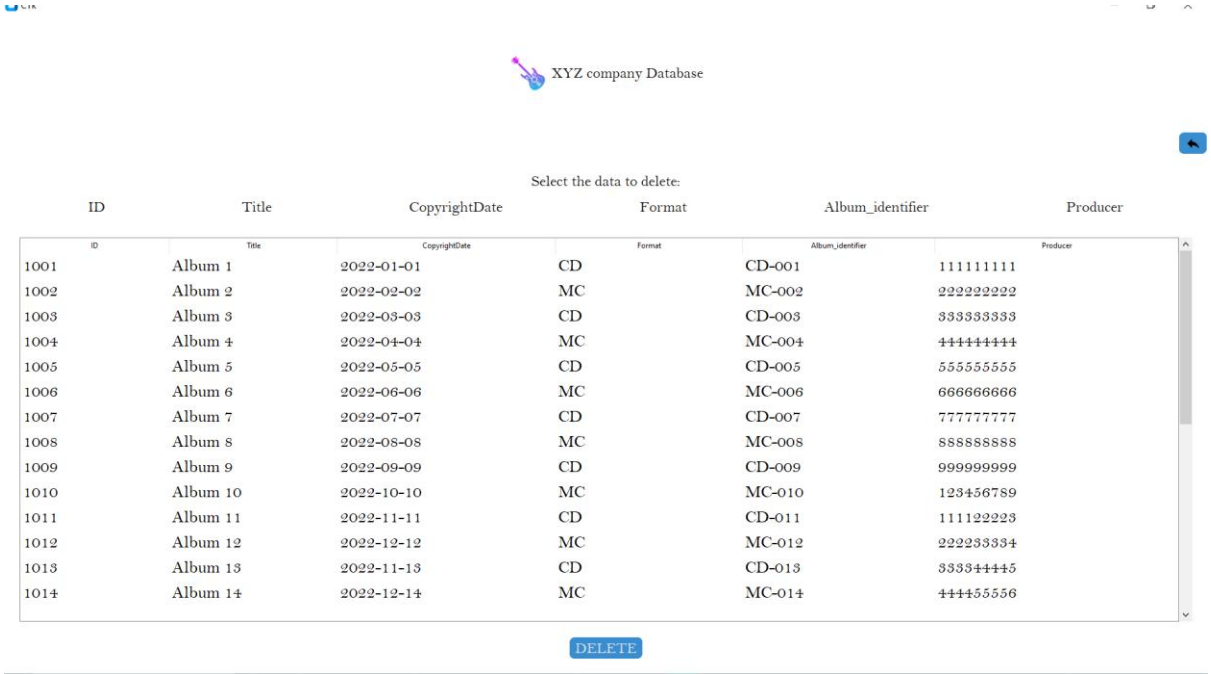


Fig: Delete Instrument's Data

Conclusion:

We were kind of successful in making the Music Database Management System in form of GUI which uses MySQL database meeting the minimum requirements of the client. Our system is able to solve the current problem of the music company XYZ. Our prototype is able to create system users, admins, musician, producer and author account giving them their freedom and ability adding, changing and deleting the records of their songs, albums or instrument they play. Now, after the client's approval we could proceed further with their tech team integrating this database and this GUI we created in their website. Hence, we completed the project making the GUI with integrated MySQL database for managing data of XYZ.