|  |
| --- |
| Southern New Hampshire University |
| **Gas Price Tracker** |
| **A CS231 Project** |
|  |
|  |
|  |
|  |

|  |
| --- |
|  |

|  |  |
| --- | --- |
| **Created:** | 1/16/2023 |
| **Authors:** | Cole Garboski |
|  |  |
| **Last Updated:** | 4/4/23 |
| **Version #:** | 1.3 |

# Glossary & Acronyms

|  |  |
| --- | --- |
| Term /  Acronym | Explanation |
| SDLC | Software Development Life Cycle |
| DBMS | Database Management System |
| SE | Software Engineering |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# Overview of Project

This simple app will be able to display gas stations and the prices of their gas that they sell. Gas station managers can update, add, and remove gas prices with ease.

# Current State

The gas tracker app currently supports basic functionality to display gas stations and the prices of the gas they sell. The app allows gas station managers to update, add, and remove gas prices with ease. The app also displays additional information such as the types of gas available and any other amenities like a store or EV charger. Furthermore, the data can be filtered by various parameters like company and fuel type.

# Future State

In the future, the gas tracker app could be scaled up to allow users to add their own gas prices as they get gas. This would reduce the workload on gas station managers and provide more accurate pricing information for users. Additionally, the app could include more detailed descriptions of the gas stations, such as the number of pumps available, busy times, and the best time to get gas. The app could also provide a feature to filter gas stations based on distance, availability of amenities, and a price limit.

The app could also implement authentication and authorization mechanisms to ensure data security and privacy. The users should have different levels of access, depending on their roles, such as guest, registered user, gas station manager, and app administrator. Logged in users could also do things like store favorite stations and have premade preferences based on the vehicle they drive, for example.

# Detailed Business Rules

## **A. General**

|  |  |
| --- | --- |
| **No.** | **General Requirements** |
| 1001 | A manager can ADD, REMOVE, or EDIT their stores prices. |
| 1002 | A store will have data about its types of gas, each price, and other amenities offered by the store in a short description. |
| 1003 | The prices should be available to everyone, while editing privileges are not. |
| 1004 | Data about gas companies will be stored, such as their name, value, and their main gas station name. |
| 1005 | Many fuel types can be added to stations, and they can each have individual prices |
| 1006 | The app can provide a feature for users to search for gas stations based on specific criteria, such as availability of EV charging, car wash, or snacks. |
|  |  |

ER Diagram

Diagram

Description automatically generated

Relational Schema

**Company Table**

* CompanyID (Primary Key)
* CompanyName
* CompanyDateCreated
* CompanyValue
* StationName

**Station Table**

* StationID (Primary Key)
* StationName
* StationAddress
* CompanyID (Foreign Key referencing Company.CompanyID)

**FuelType Table**

* FuelTypeID (Primary Key)
* FuelTypeName

**Price Table**

* PriceID (Primary Key)
* StationID (Foreign Key referencing Station.StationID)
* FuelTypeID (Foreign Key referencing FuelType.FuelTypeID)
* Price

SQL DDL and DML

To Create Tables

CREATE TABLE Company (

CompanyID INT AUTO\_INCREMENT PRIMARY KEY,

CompanyName VARCHAR(50),

CompanyDateCreated DATE,

CompanyValue DECIMAL(10,2),

StationName VARCHAR(50)

);

CREATE TABLE Station (

StationID INT AUTO\_INCREMENT PRIMARY KEY,

StationName VARCHAR(50),

StationAddress VARCHAR(100),

CompanyID INT,

FOREIGN KEY (CompanyID) REFERENCES Company(CompanyID)

);

CREATE TABLE FuelType (

FuelTypeID INT AUTO\_INCREMENT PRIMARY KEY,

FuelTypeName VARCHAR(50)

);

CREATE TABLE Price (

PriceID INT AUTO\_INCREMENT PRIMARY KEY,

StationID INT,

FuelTypeID INT,

Price DECIMAL(10,2),

FOREIGN KEY (StationID) REFERENCES Station(StationID),

FOREIGN KEY (FuelTypeID) REFERENCES FuelType(FuelTypeID)

);

CREATE TABLE User (

UserID INT AUTO\_INCREMENT PRIMARY KEY,

UserName VARCHAR(30),

PasswordHash VARCHAR(64),

Role VARCHAR(20)

);

To Insert some sample data

INSERT INTO Company (CompanyID, CompanyName, CompanyDataCreated, CompanyValue, StationName)

VALUES (1, 'Shell', '2021-01-01', 50000, 'Shell');

INSERT INTO Company (CompanyID, CompanyName, CompanyDataCreated, CompanyValue, StationName)

VALUES (2, 'ExxonMobil', '2021-02-01', 60000, 'Mobil');

INSERT INTO Station (StationID, StationName, StationAddress, CompanyID)

VALUES (1, 'Shell Station 1', '1234 Main St, Anytown, USA', 1);

INSERT INTO Station (StationID, StationName, StationAddress, CompanyID)

VALUES (2, 'ExxonMobil Station 1', '5678 Elm St, Anytown, USA', 2);

INSERT INTO FuelType (FuelTypeID, FuelTypeName)

VALUES (1, 'Regular Gasoline');

INSERT INTO FuelType (FuelTypeID, FuelTypeName)

VALUES (2, 'Premium Gasoline');

INSERT INTO Price (PriceID, StationID, FuelTypeID, Price)

VALUES (1, 1, 1, 2.50);

INSERT INTO Price (PriceID, StationID, FuelTypeID, Price)

VALUES (2, 1, 2, 3.00);

INSERT INTO Price (PriceID, StationID, FuelTypeID, Price)

VALUES (3, 2, 1, 2.60);

INSERT INTO Price (PriceID, StationID, FuelTypeID, Price)

VALUES (4, 2, 2, 3.10);

Some sample select statements

SELECT \* FROM Price;

SELECT \* FROM Station WHERE CompanyID = 1;

Sample update statement

UPDATE Price SET Price = 2.70 WHERE StationID = 1 AND FuelTypeID = 1;

Sample delete statement

DELETE FROM Company WHERE CompanyID = 2;