Technical Design Document
Project: Band Gear Exchange Forum
Course: Database Systems and Web Programming 2024 Summer

Overview

This plan briefly describes the system's conceptual analysis and database design. Additionally, it presents a few key use cases and the database queries they require. Finally, there is a short description of the practical implementation's architectural solutions.

The Band Gear Exchange Forum is a web-based service that allows users, referred to as "Bidders", to create and accept offers for exchanging band equipment.

This document also outlines the functionalities, user types, and operations of the service. The technical implementation of the application is based on the Flask web framework and SQLAlchemy, which provides an ORM interface layer for the PostgreSQL database. The user interface is implemented with HTML pages.

Image 1: High level architecture of the system (will be updated here later on).

Service Functions

User Registration

User Types

1. **Registered Bidders**: Users who have created an account and logged in.

2. **Unregistered Bidderss**: Users who browse the service without logging in.

System Functions

The system maintains, searches, suggests, and sorts exchange offers
for band equipment. Users interested in a gear can send questions to the provider via email.

Exchange Process

- **Offer Creation**: Bidders can create offers to exchange band equipment.

- **Offer Acceptance**: Bidders can accept any exchange offers.

- **Mutual Agreement**: A successful exchange requires mutual acceptance of each other's offers. Bidders can modify their offers to better suit the exchange operation.

Communication

Questions and answers about the band equipment are exchanged via email or
phone between the parties involved.

Financial Transactions

No money is exchanged through the service.
Any financial arrangements are made separately, outside the service, e.g.via phone or email.

User Operations via Web Browser

Bidders can perform the following operations by using their web browsers:

1. **Create Offers**: Registered Bidders can create new exchange offers.

2. **Browse Offers**: Both registered and unregistered Bidders can browse existing offers.

3. **Accept Offers**: Registered Bidders can accept offers made by others.

4. **Modify Offers**: Bidders can modify their offers to better suit the exchange deal.

5. **Send Inquiries**: Interested users can send inquiries to Bidders via email.

Bidders can register and list their exchange offers in various categories, such as:

- Instruments
- Amplifiers
- Microphones
- Studio Equipment
- Etc.

Offer Details

Bidders input the details of their exchange offers as follows:

- Define the start and end dates of the exchange offer.
- Set compensation and delivery preferences.
- Maintain the status of their exchange offers and adjust the content as needed.
- Remove offers once the exchange is completed.

System Functions

The system maintains, searches, suggests, and sorts exchange offers for band equipment. It includes:

- Creation and maintenance of band equipment lists.
- Maintenance and printing of open offers list.
- Status update of offers.

Listing and Status Management

- Lists offers by Bidder, status, category, date, and content.
- Changes the status of offers that lead to exchanges to “completed” but does not delete them.

Database Entities

Bidder Information

- **User ID**: Unique identifier for the Bidder.
- **Name**: Bidder's name.
- **Nickname**: Bidder's nickname.
- **Password**: Bidder's password.
- **Email**: Bidder's email address.
- **Phone Number**: Visible only to registered users.

Offer-Related Bidder Information

Offer Group Table

- **Offer Group Code**: Unique code for the offer group.
- **Offer Group Name**: Name of the offer group.

Offer Content Table

- **Offer ID**: Unique identifier for the offer.
- **Offer Status**: Status of the offer.
- **Number of Equipment Rows**: Number of equipment rows in the offer.
- **Equipment Row Details**:
- **Quantity Offered**: Quantity of the equipment offered.
- **Offer Price**: Price of the equipment offered.
- **Vintage Product Life Story** (optional): History and background of the vintage product.
- **Original Price**: Original price of the product.
- **Comments and Additional Information**: Any additional details about the offer.

Exchange Offer Table ("Purchase Offer")

- **Offer ID**: Unique identifier for the offer.
- **Offer Status**: Status of the offer.
- **Bidder ID**: Unique identifier for the provider.
- **Bidder Contact Information**: Contact details of the provider.

- **Bidder Preferences and References**: Preferences and references of the provider.
- **Start and End Dates**: Start and end dates of the offer.
- **Number of Product Rows**: Number of product rows in the offer.
- **Offered Equipment IDs**: IDs of the offered equipment.
- **Delivery Method**: Method of delivery for the equipment.
- **Exchange Preferences**: Desired products for exchange.
- **Additional Offer Information**: Any other relevant details about the offer.

Equipment Group Table

- **Equipment Group Code**: Unique code for the equipment group.
- **Equipment Group Name**: Name of the equipment group.

Equipment Table

- **Equipment Code**: Unique code for the equipment.
- **Equipment Name and Type Details**: Name and type details of the equipment, such as model year.
- **Detailed Description**: Detailed description of the equipment, including condition, color, sound, etc.
- **Vintage-gear lifestory (option)**
- **Amount offered**
- **Original price of the gear**
- **Comments and other relevant info**

Historical Statistics and Views

- **Yearly and Monthly Exchange Statistics by BidGenre**: Statistics on exchanges categorized by BidGenre for each month.
- **Bidder Statistics and List**: Statistics and list on users.

Database and Conceptual Model Documentation

Introduction
The following sections provide implementation design of each table and their relations.

Database Schema (DRAFT 1.0 for prototyping, to be updated later on)

1. Bidder Table

The Bidder table stores information about users who participate in the bidding system.

- **id**: A unique identifier for each bidder (Primary Key).
- **bidderName**: The name of the bidder (must be unique).
- **password**: The password for the bidder's account (optional).
- **admin**: A boolean flag indicating if the bidder has administrative privileges (default is FALSE).

```
CREATE TABLE Bidder
(
    id SERIAL NOT NULL PRIMARY KEY,
    bidderName VARCHAR(30) NOT NULL UNIQUE,
    password VARCHAR(20) DEFAULT NULL,
    admin BOOLEAN DEFAULT FALSE
);
```

2. BidGenre Table

The BidGenre table categorizes the types of bids.

- **id**: A unique identifier for each bid genre (Primary Key).
- **bidGenre**: An integer representing the genre.
- **bidder**: A foreign key referencing the Bidder table.
- **bidGenreHeader**: A unique header for the bid genre.

```
CREATE TABLE BidGenre
(
    id SERIAL NOT NULL PRIMARY KEY,
    bidGenre INT,
    bidder INT,
    bidGenreHeader VARCHAR(100) NOT NULL UNIQUE,
    FOREIGN KEY(bidder) REFERENCES Bidder(id)
);
```

3. BidHeader Table

The BidHeader table contains information about individual bids.

- **id**: A unique identifier for each bid header (Primary Key).
- **initialBidder**: A foreign key referencing the Bidder table.
- **bidGnre**: A foreign key referencing the BidGenre table.

- **bidHder**: A unique header for the bid.
- **numberOfBids**: The number of bids made.
- **startedTime**: The timestamp when the bid started.

```
CREATE TABLE BidHeader
(
    id SERIAL NOT NULL PRIMARY KEY,
    initialBidder INT,
    bidGnre INT,
    bidHder VARCHAR(100) NOT NULL UNIQUE,
    numberOfBids INT NOT NULL,
    startedTime TIMESTAMP NOT NULL,
    FOREIGN KEY(initialBidder) REFERENCES Bidder(id),
    FOREIGN KEY(bidGnre) REFERENCES BidGenre(id)
);
```

4. BidContent Table

The BidContent table stores the content of individual bids.

- **id**: A unique identifier for each bid content (Primary Key).
- **bidderID**: A foreign key referencing the Bidder table.
- **bidHead**: A foreign key referencing the BidHeader table.
- **bidderName**: The name of the bidder.
- **bidContent**: The content of the bid.
- **editedTime**: The timestamp when the bid was last edited.

```
CREATE TABLE BidContent
(
    id SERIAL NOT NULL PRIMARY KEY,
    bidderID INT,
    bidHead INT,
    bidderName VARCHAR(30) NOT NULL,
    bidContent VARCHAR(500) NOT NULL,
    editedTime TIMESTAMP NOT NULL,
    FOREIGN KEY(bidHead) REFERENCES BidHeader(id)
);
```

GearGroup Table

- **id**: A unique identifier for each gear group (Primary Key).
- **gearGroup**: An integer representing the genre.
- **gearGroupHeader**: A unique name for the gear group.

- **bidder**: A foreign key referencing the Bidder table.

```
CREATE TABLE GearGroup
(
  id SERIAL NOT NULL PRIMARY KEY,
  gearGroup INT,
  bidderID INT,
  gearGroupHeader VARCHAR(100) NOT NULL UNIQUE,

  FOREIGN KEY(bidder) REFERENCES Bidder(id)
);
```

Equipment Table

- **Equipment Code**: Unique code for the equipment.
- **Equipment Name and Type Details**: Name and type details of the equipment, such as model year.
- **Detailed Description**: Detailed description of the equipment, including condition, color, sound, etc.
- **Story of the Vintage-gear (option)**
- **amount offered**
- **original price estimate**
- **Comments and important details**

```
CREATE TABLE Gear
(
  id SERIAL NOT NULL PRIMARY KEY,
  gearCode INT,
  gearStatus INT,
  bidderID INT,
  bidHead INT,
  gearName VARCHAR(100) NOT NULL,
  gearDetails VARCHAR(500) NOT NULL,
  gearStory VARCHAR(500) NOT NULL,
  amountOffered INT NOT NULL,
  origPrice INT NOT NULL,
  comments VARCHAR(200) NOT NULL,
  editedTime TIMESTAMP NOT NULL,

  FOREIGN KEY(bidderID) REFERENCES Bidder(id)
  FOREIGN KEY(bidHead) REFERENCES BidHeader(id)
 );
```

5. BidExchange Table (liitostaulu, to be updated later on)

The BidExchange table manages the exchanges between bidders.

- **id**: A unique identifier for each bid exchange (Primary Key).
- **salesBidder**: A foreign key referencing the Bidder table (initial bidder).
- **purchaseBidder**: A foreign key referencing the Bidder table (purchasing bidder).
- **equipmentOffer**: A description of the equipment offered.
- **equipmentRequested**: A description of the equipment requested.
- **agreedDeal**: A boolean flag indicating if the deal is mutually agreed upon (default is FALSE).
- **dealTimestamp**: The timestamp when the deal was made.

```
CREATE TABLE BidExchange
(
    id SERIAL NOT NULL PRIMARY KEY,
    salesBidder INT NOT NULL,
    purchaseBidder INT NOT NULL,
    equipmentOffer VARCHAR(100) NOT NULL,
    equipmentRequested VARCHAR(100) NOT NULL,
    agreedDeal BOOLEAN DEFAULT FALSE,
    dealTimestamp TIMESTAMP NOT NULL,
    FOREIGN KEY(salesBidder) REFERENCES Bidder(id),
    FOREIGN KEY(purchaseBidder) REFERENCES Bidder(id)
);
```

The relationships between the tables are as follows:

- **Bidder**: Represents users in the system. Each bidder can create multiple bid genres and bid headers.
- **BidGenre**: Categorizes bids and is linked to a specific bidder.
- **BidHeader**: Contains the main information about a bid, including the initial bidder and the genre of the bid.
- **BidContent**: Stores the detailed content of each bid and is linked to both the bidder and the bid header.
- **BidExchange**: Manages the exchanges between bidders, including the details of the equipment offered and requested.

Appendix:

Prototype-Database creation script:

```
CREATE TABLE Bidder
(
  id SERIAL NOT NULL PRIMARY KEY,
  bidderName VARCHAR(30) NOT NULL UNIQUE,
  password VARCHAR(20) DEFAULT NULL,
  admin BOOLEAN DEFAULT FALSE
 );

CREATE TABLE BidGenre
(
  id SERIAL NOT NULL PRIMARY KEY,
  bidGenre INT,
  bidder INT,
  bidGenreHeader VARCHAR(100) NOT NULL UNIQUE,

  FOREIGN KEY(bidder) REFERENCES Bidder(id)
 );

CREATE TABLE BidHeader
(
  id SERIAL NOT NULL PRIMARY KEY,
  initialBidder INT,
  bidGnre INT,
  bidHder VARCHAR(100) NOT NULL UNIQUE,
  numberOfBids INT NOT NULL,
  startedTime TIMESTAMP NOT NULL,

  FOREIGN KEY(initialBidder) REFERENCES Bidder(id),
  FOREIGN KEY(bidGnre) REFERENCES BidGenre(id)
 );

CREATE TABLE BidContent
(
  id SERIAL NOT NULL PRIMARY KEY,
  bidderID INT,
  bidHead INT,
  bidderName VARCHAR(30) NOT NULL,
  bidContent VARCHAR(500) NOT NULL,
  editedTime TIMESTAMP NOT NULL,
```

```
  FOREIGN KEY(bidHead) REFERENCES BidHeader(id)
 );

CREATE TABLE GearGroup
(
  id SERIAL NOT NULL PRIMARY KEY,
  gearGroup INT,
  bidderID INT,
  gearGroupHeader VARCHAR(100) NOT NULL UNIQUE,

  FOREIGN KEY(bidder) REFERENCES Bidder(id)
 );

CREATE TABLE Gear
(
  id SERIAL NOT NULL PRIMARY KEY,
  gearCode INT,
  gearStatus INT,
  bidderID INT,
  bidHead INT,
  gearName VARCHAR(100) NOT NULL,
  gearDetails VARCHAR(500) NOT NULL,
  gearStory VARCHAR(500) NOT NULL,
  amountOffered INT NOT NULL,
  origPrice INT NOT NULL,
  comments VARCHAR(200) NOT NULL,
  editedTime TIMESTAMP NOT NULL,

  FOREIGN KEY(bidderID) REFERENCES Bidder(id)
  FOREIGN KEY(bidHead) REFERENCES BidHeader(id)
 );

CREATE TABLE BidExchange
--(draft v1, to be updated later on to LIITOSTAULU)
(
  id SERIAL NOT NULL PRIMARY KEY,
  salesBidder INT NOT NULL, -- ID of the initial bidder (sales-bidder)
  purchaseBidder INT NOT NULL, -- ID of the bidder making the purchase
offer
  equipmentOffer VARCHAR(100) NOT NULL, -- Description of the equipment
offered
  equipmentRequested VARCHAR(100) NOT NULL, -- Description of the equip-
ment requested
  agreedDeal BOOLEAN DEFAULT FALSE, -- Indicates if the deal is mutually
```

```
agreed upon
  dealTimestamp TIMESTAMP NOT NULL, -- Timestamp when the deal was made

  FOREIGN KEY(salesBidder) REFERENCES Bidder(id),
  FOREIGN KEY(purchaseBidder) REFERENCES Bidder(id)
);
--Explanation for table BidExchange:
--salesBidder and purchaseBidder columns store the IDs of the bidders involved in the exchange.
--equipmentOffer describes the equipment offered by the sales bidder.
--equipmentRequested describes the equipment requested by the purchaser.
--agreedDeal indicates whether both parties have agreed on the deal (initially set to FALSE).
--dealTimestamp records the time when the deal was made.
--Remember to adjust the column definitions according to your specific requirements.
--This table will allow bidders to exchange equipment based on their bids.

CREATE VIEW BidGenreList AS
SELECT bg.id, bg.bidGenreHeader, COUNT(bc.id) as count, MAX(bc.editedTime) AS latest
FROM BidGenre bg
LEFT JOIN BidHeader bhr ON (bg.id = bhr.bidGnre)
LEFT JOIN BidContent bc ON (bhr.id = bc.bidHead)
GROUP by bg.id, bg.bidGenreHeader;

CREATE VIEW BidList AS
SELECT bh.id, bh.initialBidder, bh.bidGnre, bh.bidHder,
COUNT(*) AS count, MIN(bc.editedTime) AS edited,
MAX(bc.editedTime) AS latest
FROM BidHeader bh
LEFT JOIN BidContent bc ON (bh.id = bc.bidHead)
GROUP BY bh.id, bh.bidHder, bh.initialBidder, bh.bidGnre
ORDER BY MAX(bc.editedTime) DESC;
```