1) Project Proposal

1) **Group Members**

Zane Garvey

2) **Project Description**

Description:

This project involves creating a web application and database for guitar enthusiasts to manage, view, and share details about their guitar collections. The app will allow users to create entries for individual guitars, record specifications, add photos, and list maintenance records. Users can explore each other's collections, share favorite setups, and connect with fellow enthusiasts.

Tech Stack (as of now (because I may actually turn this into something) :

Frontend:

- Vite (Build tool)
- React (UI library)
- TypeScript (Type safety)
- TailwindCSS (Utility-first CSS)
- MaterialUI (Component library)

Backend:

- Next.js API routes (Vercel serverless backend)
- MySQL
- Prisma (type-safe ORM)
- NextAuth.js
- bcrypt (maybe)
- JWT

Database:

- MySQL (Through the school PHPMyAdmin)
- Prisma migrations

Development:

- ESLint
- Prettier
- dotenv
- Vercel
- Vercel ci/cd through GitHub

Potential Uses:

- Guitar collectors can use it to track and organize their instruments and maintenance history.
- Musicians can view other users' setups and specifications for inspiration.
- Guitar stores can use it to showcase rare or vintage guitars.

Primary Users: Guitar collectors, musicians, music shop owners, and enthusiasts.

3) Requirements and Business Rules

Requirements:

- Users can create an entry for a guitar with its details and specifications
- Users can view other collections and share guitars within the platform
- Each guitar entry can have multiple maintenance records (e.g., string changes, setups)
- Users can update and delete guitars and maintenance logs
- All CRUD operations (Create, Retrieve, Update, Delete) are supported

Business Rules:

- Each guitar must have a unique serial number
- Each maintenance entry must include a date
- Users can only update or delete guitars in their collection
- A guitar may have multiple maintenance entries but cannot have the same maintenance entry on the same date
- Users can tag guitars with specific genres or playing styles.

4) **Database Outline**

Entities and Attributes:

User

- user_id (Primary Key)
- username
- email
- password

Guitar

- guitar_id (Primary Key)
- serial_number (unique)
- brand
- model
- year
- user_id (Foreign Key referencing User)
- genre (optional tag for the guitar's best-suited genre)
- Body type (acoustic, electric, classical)

Maintenance

- maintenance_id (Primary Key)
- date
- type (e.g., string change, fret polish)
- notes
- guitar_id (Foreign Key referencing Guitar)

Photos

- photo_id (Primary Key)
- url (link to image)
- guitar_id (Foreign Key referencing Guitar)
- caption **UPDATED FROM ORIGINAL PROPOSAL**

Connections

- connection_id (Primary Key) **UPDATED FROM ORIGINAL PROPOSAL**
- user1_id (Foreign Key referencing User)
- user2_id (Foreign Key referencing User)
- relationship (e.g., "friend" or "follower")

Relationships:

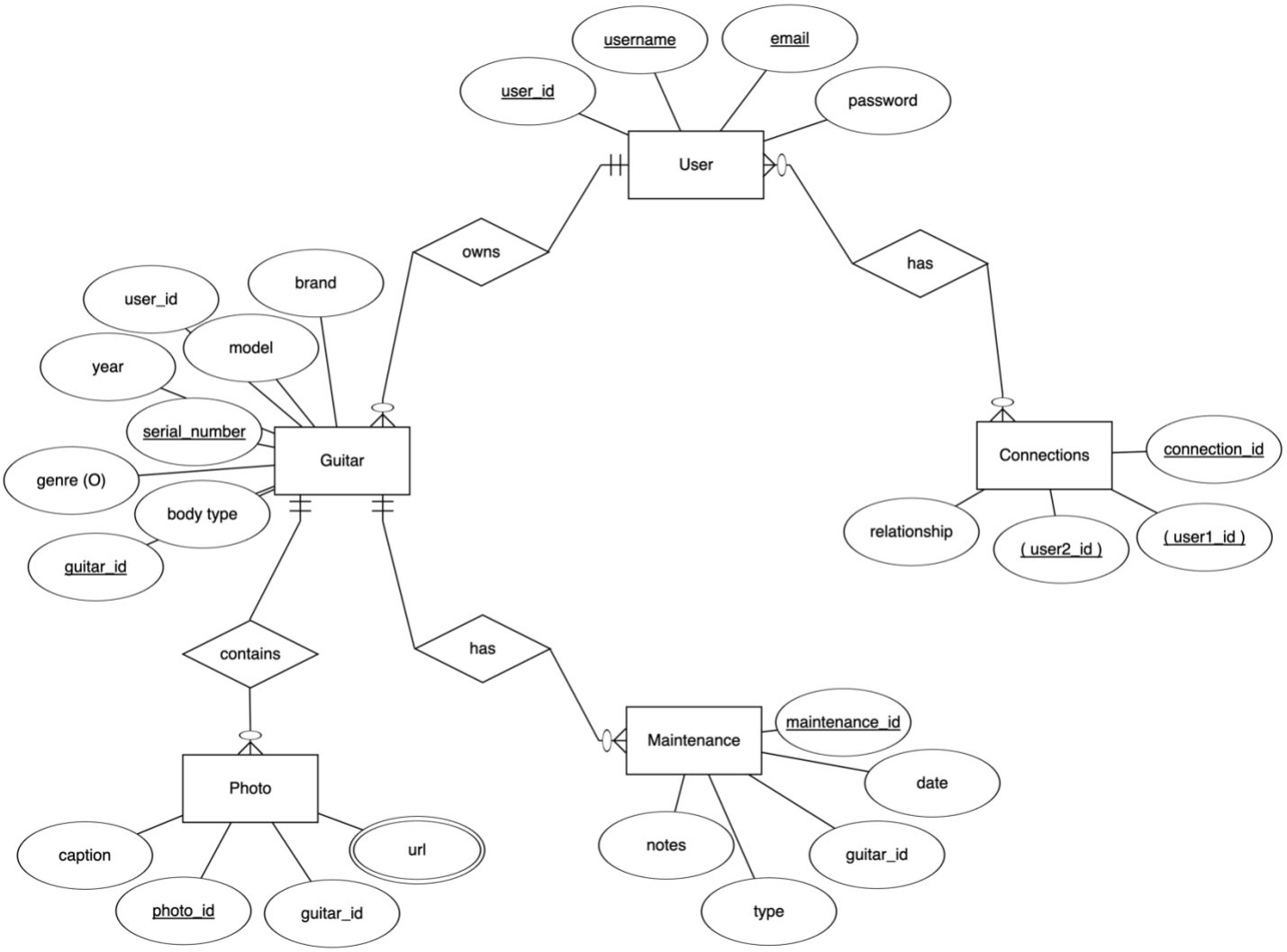
Own: A one-to-many relationship between User and Guitar.

Maintains: A one-to-many relationship between Guitar and Maintenance.

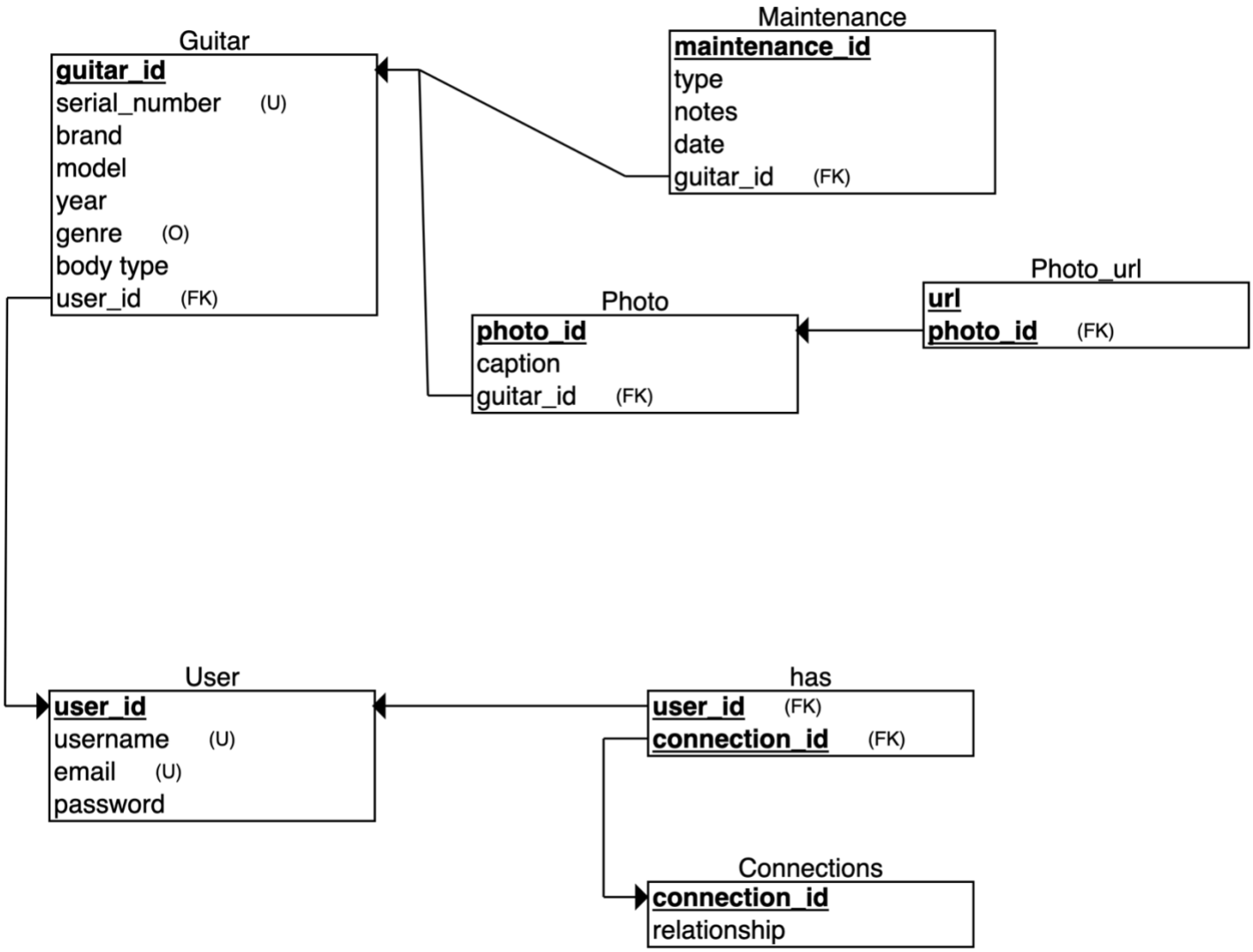
Photographs: A one-to-many relationship between Guitar and Photos.

Connections: A many-to-many relationship between Users with attributes specifying the relationship type.

2) ER Diagram



3) Relational Schema



4) Foreign Key Operations

Foreign Key: *user_id* in the *Guitar* table

- DELETE: Cascade
  - If a user is deleted, all of their guitars should also be deleted as the guitars are closely tied to the user's collection
- UPDATE: Restrict
  - Changing the user ID should not be allowed because it would indicate a change in ownership, which I won't support in this implementation

Foreign Key: *guitar_id* in the *Maintenance* table

- DELETE: Cascade
  - If a guitar is deleted, all of its maintenance records should also be deleted as they become irrelevant without the guitair
- UPDATE: Restrict
  - The guitar_id should not be updated because maintenance records belong to specific guitars

Foreign Key: *guitar_id* in the *Photos* table

- DELETE: Cascade
  - If a guitar is deleted, its associated photos should also be deleted because they are tied to the guitar
- UPDATE: Restrict
  - Updating guitar_id should not be allowed to avoid mismatching photos with guitars

Foreign Keys: *user1_id* and *user2_id* in the *Connection* table

- DELETE: Set to null
  - If a user is deleted, the connection record can be kept with a null reference, to preserve the integrity of other connections
- UPDATE: Cascade
  - If a user ID changes (maybe due to reassignments), the changes should cascade to maintain consistency across relationships

Foreign Key: *guitar_id* in the *Photos* table

- DELETE: Cascade
  - Photos should be deleted if the associated guitar is deleted
- UPDATE: Restrict
  - Do not allow updates, as photos are tied to specific guitars