- **(4 points) Database design**
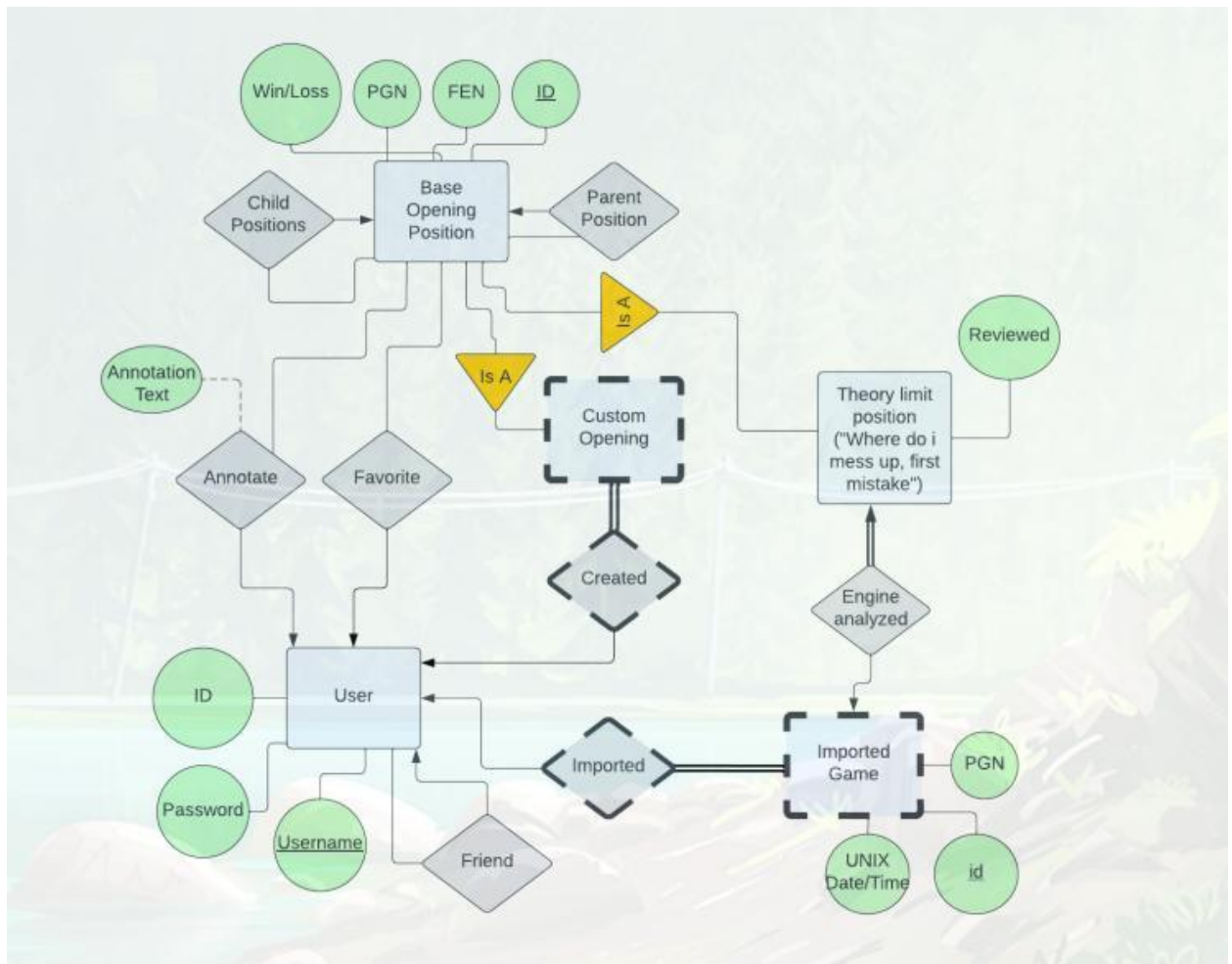 - (2 points) — Include a final version of your E-R diagram

Note final change - win loss was replaced by name and FEN was removed all together as attributes of Base Opening Position

  - (2 points) — Include a final version of your tables written in schema statements

Included in the schema.sql file (just in case also at the end of this doc as well)

Eric Kharitonashvili (eak6sv)

- **(10 points) Database programming**
  - (1 point) — Specify where you host your database

    On my local machine port 3306

  - (1 point) — Specify where you host your app — the deployment environment needed to deploy and run your project

    On my local machine in app.py when I run it

  - (2 points) — Describe and include all instructions / steps needed to deploy and run your project

    Git clone THE REPOSITORY. Create a maria db and edit the make_trees.py file. This will initialize all the starting openings. Use the initialize sql query as well.

    Python start app.py and the html files should work on any local machine

  - (6 points) — Discuss how the advanced SQL commands are incorporated in your app, what features of your app use them, and how they reflect the database

    ```
    ALTER TABLE Account
    ADD CONSTRAINT CK_pword
    CHECK (LEN(username) >= 3)

    CREATE PROCEDURE PgnGetOP myPGN@ varchar(255)
    AS
    Select OpeningID from Openings Where myPGN = @PGN
    GO;
    ```

    The top command ensures usernames are higher than 3 characters. The bottom command allows easy pgn (chess move order) identification in the

==database. The bottom command is helpful for developers avoiding copy paste application layer code, and the top is useful to prevent typo's when creating user accounts.==

- (8 points) Database security at the database level
  - (5 points) — Discuss how you set up security at the database level (access control). Also specify whether the security is set for ==developers== or end users.

    ==I do not allow anyone except root access to the Accounts database. This is for account recovery purposes and no average developer/code can attempt to password crack.==

  - (3 points) — Submit the SQL commands you use to limited / set privileges

    ==GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY "9789";==
    ==Revoke ALL PRIVILEGES ON Chess.Accounts TO erickhar==

- (8 points) Database security at the application level
  - (5 points) — Discuss how database security at the application level is incorporated in your project

    ==My application hashes the passwords so that they are never stored in plain text in the real database==

  - (3 points) — Submit code snippet(s) to illustrate how security aspect is implemented and to support your discussion

When validating logins

```
    password_hash = str(hash(request_data["password"]))
    query = "Select * From Account where Username = '{}' and Password =
'{}'".format(request_data["username"],password_hash)
    cur.execute(query)
```
When creating users - same as above except
```
    query = "Insert into Account (Username,Password) values ('{}',
'{}')".format(request_data["username"],password_hash)
```

Earlier mentioned written schema

```
/*
 To create all the tables and fields
 in the chess database
 */
/* BCNF Entities*/
CREATE TABLE
  Account(
    Username varchar(255) NOT NULL,
    Password varchar(255) NOT NULL,
    PRIMARY KEY (Username)
  );

CREATE TABLE
  Opening(
    OpeningID int NOT NULL,
    PGN varchar(255),
    Name varchar(255),
    Custom bool,
    PRIMARY KEY (OpeningID)
  );

CREATE TABLE
  TheoryLimit(
    Username varchar(255) NOT NULL,
    OpeningID int NOT NULL,
    Reviewed BOOLEAN NOT NULL,
    PGN varchar(255),
    FOREIGN KEY (Username) references Account(Username),
    FOREIGN KEY (OpeningID) references Opening(OpeningID)
  );
```

Eric Kharitonashvili (eak6sv)


```
CREATE TABLE
  CustomOpening(
    Username varchar(255) NOT NULL,
    OpeningID int NOT NULL,
    PGN varchar(255),
    FOREIGN KEY (Username) references Account(Username),
    PRIMARY KEY (OpeningID)
  );

CREATE TABLE
  ImportedGame(
    Username varchar(255) NOT NULL,
    OpeningID int NOT NULL,
    PGN varchar(255),
    FOREIGN KEY (Username) references Account(Username),
    FOREIGN KEY (OpeningID) references CustomOpening(OpeningID)

  );

/*
 CREATE TABLE PGN_to_Hash(
 Hash BINARY(16) NOT NULL,
 PGN varchar(2048) NOT NULL,
 PRIMARY KEY (PGN),
 FOREIGN KEY (HASH) references Opening(OpeningID)
 );
 */
/* BCNF Relations*/
CREATE TABLE
  Favorites(
    Username varchar(255) NOT NULL,
    OpeningID int NOT NULL,
    FOREIGN KEY (Username) references Account(Username),
    FOREIGN KEY (OpeningID) references Opening(OpeningID)
  );

CREATE TABLE
  Annotated(
    Username varchar(255) NOT NULL,
    PGN varchar(255),
```

```
   Annotation varchar(255),
   FOREIGN KEY (Username) references Account(Username)
 );

CREATE TABLE
 OPChildren(
   ParentID int NOT NULL,
   ChildID int NOT NULL,
   FOREIGN KEY (ParentID) references Opening(OpeningID)
 );

CREATE TABLE
 OPParent(
   ChildID int NOT NULL,
   ParentID int NOT NULL,
   FOREIGN KEY (ChildID) references Opening(OpeningID)
 );

CREATE TABLE
 UserFriend(
   Username varchar(255) NOT NULL,
   Friend varchar(255) NOT NULL,
   FOREIGN KEY (Username) references Account(Username)
 );
```