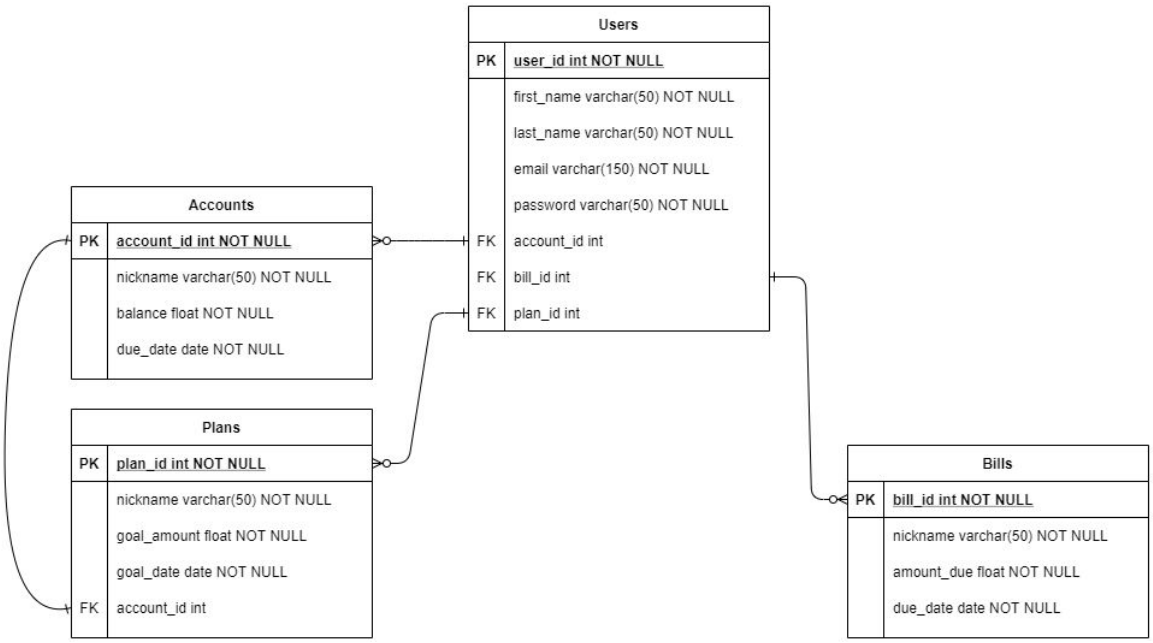
Henry Vu



For the MoneyJournal application, I will be using PostgreSQL as my relational database. The User will have multiple Accounts, Bills, and Plans that can simply be done with the use of one-to-many relationships. Users can perform CRUD actions on the database.

The tables required are Users, Accounts, Bills, and Plans.
Each column is displayed in the diagram with its respective data type.

Users: need to register with first and last names, email, and password. When they add an account, bill, or plan, a foreign key will reference it to the added table's primary key.

Accounts: has nickname, balance, and due date. The balance can be negative indicating a debt/liability that will cost the user to pay off. A positive balance indicates income/assets that can be used to save or pay off debts.

Bills: This has changed since the last design. There are no longer subscriptions and Bills will have a nickname, amount due, and due date.

You can see from the ERD the foreign key relationships are laid out from the User to the other tables.

Plans have a one-to-one relationship with Accounts because Plans can be made utilizing data from the Accounts table. Say, the user plans to pay off a loan. They can create a relationship with that loan's account to see how much needs to be paid off. For the stretch feature, plans will also have retirement accounts. These can be added as permanent rows that can only be updated and not deleted (ie. Index funds, Roth, 401(k))