# PoS Organization

## Backend

* Category
  + Table
    - Field
      * Notes
* User Management
  + User ID
  + User
    - User Type (I.E. manager or employee)
    - Name
    - DOB
    - Fields for contact info
    - Availability
  + Invoice
    - Invoice ID
      * There will only be one invoice with all possible information and depending on the type of invoice needed, i.e. user, managerial, client, the appropriate data will be displayed
    - Reference to appropriate users, clients, and jobs
      * All data should be available with the aforementioned references as they have references to the correct data
      * Discount/promotions
  + EO?
    - Not sure what this is?
  + Job
    - Job ID
    - Location
    - Time Frame
    - Labor
    - References to users associated with it
    - Notes associated with job
    - Type of work (Non for profit, commercial)
    - References to associated client and users
  + Client
    - Client ID
    - Name
    - Company
    - State of residency (For taxes)
    - Other relevant info
  + Payment
    - Payment ID
    - Reference to client (Since to my understanding, no user should be putting money through the system)
    - Payment type
    - Reference to invoice linked to this payment method
    - Reference to job, since once the job is linked so is all of is data
      * Side note, the way you would get the invoice is to search the invoice table for the job linked to the payment method. This would pull up the correct invoice while not having redundant data
    - Payment Information (I.E. package, day event, individual piece of equipment)
  + Inventory Management
  + Items
    - Item ID
    - Category
    - Equipment State
    - Price to client per day
    - Price
    - Model
    - More fields in time
  + Package
    - Package ID
    - Category
    - Price
    - Fields that would point to Item IDs / Package IDs that are part of this package

## Front End / User interaction

* Category
  + Table
    - Field
      * Notes
  + Look at Chris’s CM\_CLIENT\_NOTES\_1 for more detailed breakdown
* Login System
  + Encryption
  + Javascript session class needs to be researched and implemented
  + Logic should be able to be created without front end OR backend built. Temporary database can be set up with XAMPP with just user and password rows to test out encryption and logic
* Implement inventory management system
  + Create new items
  + Read previously created items
  + Edit previously created items and packages
  + Delete previously created items and packages
* Implement scheduling system
  + This will basically pull from all tables required and map the dates that apply to the schedule such as employee hours, or whatever else is selected, to one of the various views in Chris’ notes.
  + This might be a good thing to work on once DB is implemented and front end is somewhat put together as it is dependent on both of these things
  + Must also implement trigger warning and thing so this nature
  + Need a front end design to put all of these dates on
* Implement booking system
  + May be better to separate from the scheduling system but also follow the restrictions of other schedules.
  + Allow the manager to add clients, jobs, and other things of that nature.
* Implement payment system
  + Would involve selecting payment method, handling that payment type, selecting appropriate job and handling all that information securely
  + Encryption is a HUGE part of this
  + This would also involve taxing and all of the legalities associated with that so this is one that is going to involve a large amount of time
* Implement invoicing
  + Can use placeholder data for whatever is not done yet, like a fake credit card number as a string in place of the last four digits that would be found in the database.
  + This would basically pull all relevant information from the database and export it into a human readable page that we could print
  + IF Denizard allows for AJAX, we can email it as well
  + Also allow for export to quickbooks
* If we have time AND Denizard allows for it, we can set up the emailing system as AJAX can handle this!