Functional Requirements

* User can create an account
* User can login/authenticate
* User can CRUD customers
* User can CRUD appointments
* User can CRUD notes
* User can CRUD services
* User can CRUD invoices
* User can see a map of the local area
* The map shows a path from their current location to the next appointment
* User can “Take Payments”
  + No actual payment processor will be involved, this will be faked

Non-Functional Requirements

* This will be a GUI based software
* Unit testing will be the primary source of documentation
* This can either be desktop based or web based depending on group (web based groups will be responsible for their own hosting)

Language – Python

Database – SQLite

Directory Structure – Tree Structure

Naming Conventions

* Classes – Pascal Case
* Methods – Snake Case
* Constants – Upper Snake Case
* Variables – Snake Case

Quick start for developers – We are creating an application which will let the user manage dog grooming appointments for their personal business

Sprint 1 information:

* user\_manager – handles the database and storage and access of created user accounts
* user\_manager\_test – tests the UserManager class and all of the methods within it using fish themed tests
* Gui – creates the gui and the elements and handles the different interactions caused by button presses, entry boxes, etc.

Users can create an account by pressing the button on the home page, and that will prompt them with a first and last name, email address (which doubles as the username), and a password. Pressing enter into the password entry box will return them to the home while saving the information in all the entry boxes.

Sprint 2 information:

* customer object – creates a python object from a raw db record, holding customer data (first name, last name, email, phone, address)
* customer service – connects to the db, can create, edit, view, soft-delete customers
* gui – added a ‘customers’ menu option, showing a list of customers with view/edit/delete options, and a button to create a new customer (form fields for first/last name, email, phone, address). view/edit forms load/modify data as needed

Sprint 3 information:

* appointments – introduced an appointment object and an appointment service to handle creating, editing, viewing, and soft-deleting appointments
* gui – added an ‘appointments’ menu option with a list view similar to customers. can add new appointments with fields like title, description, start time, end time, location, and link it to a customer. view/edit pages load/modify data.
* notes – introduced notes attached to appointments and customers. notes can be created, viewed, edited, deleted. each note limited to 100 chars snippet in list view.

Sprint 4 information:

* notes – extended functionality of notes to also connect to customers and appointments.
* gui – integrated notes into customers and appointments views. notes appear in a table, sorted by most recent. can click to view full note, add new note, edit, delete.
* testing – ensured all new note-related functions are unit tested as well.

Sprint 5 information:

* services – introduced a services table and service object (name, description, price)
* notes for services – just like customers/appointments, services can have notes
* gui – added a ‘services’ menu. services page lists all services with name, short description, price, and view/edit/delete. can create new service with a form.
* tested all service-related operations.

Sprint 6 information:

* invoices – introduced invoices with invoice\_header and invoice\_detail tables. each invoice links to a customer and includes line items (services and their costs), total amount, due date, and paid status. also supports notes.
* invoice service – handles creation, viewing, editing (updating line items, due date, notes), soft deletion, and marking invoices as paid/unpaid. fetches all invoices or only unpaid ones, sorted by recent date.
* gui – added ‘invoices’ menu option. invoices page shows a list of invoices (date, invoice #, client, total, paid status) with view/edit/delete icons. can toggle to show only unpaid or all. view page shows invoice info, customer info, line items, total, notes, and a button to mark as paid. edit page pre-fills invoice data for changes. can add a new invoice with a form selecting a customer, due date, services, and notes.

Design Document Details:

Colors:

|  |  |  |  |
| --- | --- | --- | --- |
| White | | Off White | |
|  | Hex: #ffff |  | Hex: #fefefe |
| Black |  |  |  |
|  | Hex: #1010 |  |  |
| Primary Accent |  | Primary Highlight |  |
|  | Hex: #156082 |  | Hex: #1d81af |

Fonts:

Arial

Color: black

Headings

Comic Sans MS

Color: Primary Accent if it is a page title, black if it isn’t

Form Inputs:

2 px border, primary accent color

Application background Color

Off white

Sidebar Color

Primary Accent