Architecture Documentation

CS 459: Software Engineering Senior Project

Spring 2025

Project Title	HCAR Client Database
Sponsoring Company	Humboldt Community Access and Resource Center
Sponsor(s)	Pennie Lee, Kim Nash, Wes Patterson
Students	1. Orlando Trujillo-Ortiz 2. Carson Gustafson 3. Justin Crittenden 4. Michael Goodwyn

Last Modified: 5/16/2025

ABSTRACT

This document serves as the architecture documentation showing system structure for the HCAR Client Database System.

TABLE OF CONTENTS

1. INTRODUCTION

The Humboldt Community Access and Resource Center (HCAR) Client Database System will be implemented as a web application with a database. The system will be composed of a Presentation Tier, an Application Tier, and a Data Tier. Planning and development of the system will be performed by Orlando Trujillo-Ortiz, Carson Gustafson, Justin Crittenden, and Michael Goodwyn (collectively, "The Team").

The mission of this document is to describe in high-level detail the system architecture for the HCAR Client Database ("The System"). A description of the architectural style will be provided, along with rationale for its usage. Proceeding that will be an architecture diagram depicting the 3-tier architecture. There will also be a description of any technologies involved along with valid rationales for each decision. Lastly, there will be some detail into the project accountability process being used.

2. ARCHITECTURAL STYLE USED

The HCAR Client Database is based on a classic 3-tier Web Application architecture. The first tier is the Presentation Tier, which is the visual subsystem that a user interacts with via the Internet. It will be grounded upon a traditional HTML, CSS, and JavaScript backbone. The Presentation layer is limited to only controlling the Graphical User Interface that a member of HCAR staff would access from their own web browser.

The second tier is the Application Tier, which resides on the HCAR Client Database Cloud Web Server. This subsystem is responsible for interpreting the queries submitted by users of The System and requesting the relevant data from the Cloud. HCAR business rules will be enforced for The System via this subsystem. The backend for the Web Server will be built using PHP language.

The third and final tier is the Data Tier, which comprises the two data sources for The System. With the condition that the appropriate third party can provide The Team with an Application Programming Interface (“API”), The System will utilize HCAR’s current Cloud Storage solution with Nylex. Otherwise, The System shall contract with a different third party Cloud Storage provider.

3. ARCHITECTURAL MODEL

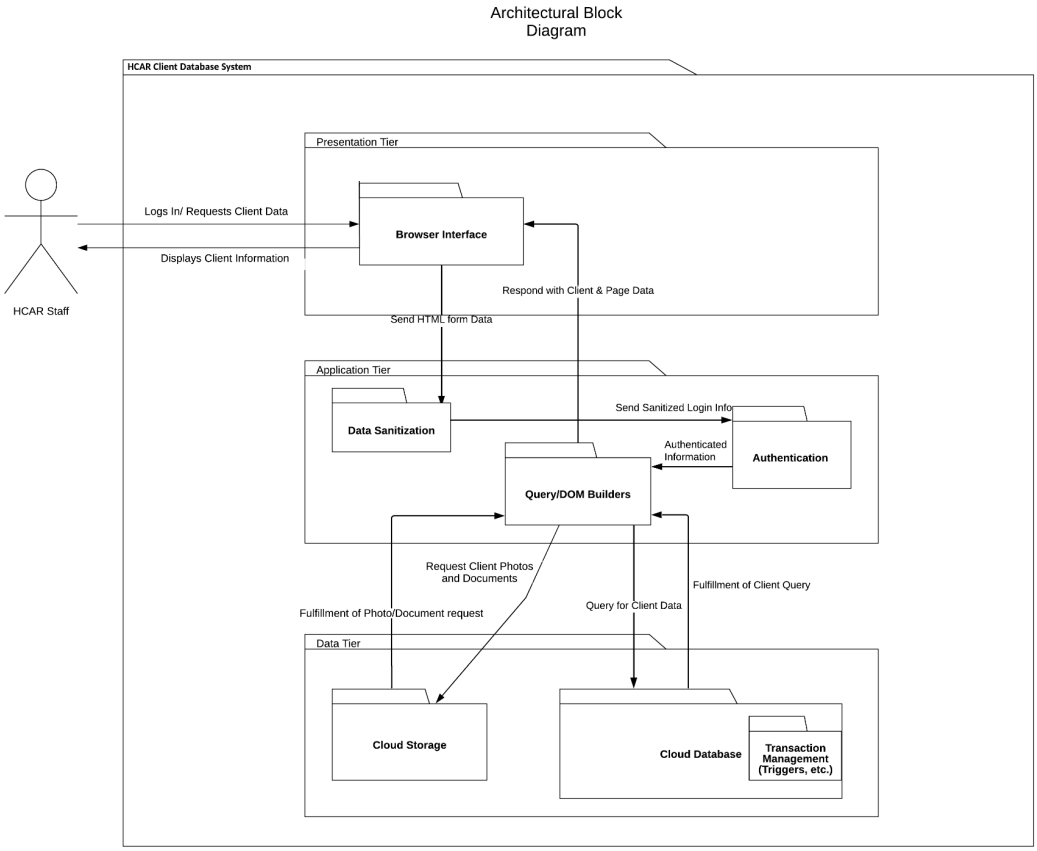

Fig. 1. HCAR Client Database Architectural Block Diagram

4. TECHNOLOGY, SOFTWARE, AND HARDWARE USED

4.1. Web Server

This project uses a Web Server to host the HCAR Client Database System. The web server is maintained by the Google Cloud and does not need any infrastructure management. Said server is located remotely in a facility run by Google. The web server will host The System's public-facing HTML / CSS / JavaScript so that HCAR staff users will be able to access The System's Internet website from any Internet browser that supports modern web-standards.

4.2. Cloud MySQL Database

This project uses a Cloud MySQL Database ("Database") to store Client data. The Database is maintained by Google Cloud, and is located in a secure location run by Google. The Database has a MySQL Relational Database Management System installed that can be interacted with via an API by the Cloud Web Server. When a User sends a request for a Client's records, the Cloud Web Server will receive said request and then subsequently send an appropriately formatted query to the Database

4.3. Google Cloud Integration

This project is integrated with Google Cloud Run. Because the Database is managed by the same provider, this allows communication through APIs between the database and a web server hosted by Google Cloud Run. This is achieved by going from a user's request, to a parsed, sanitized query for the request being sent to the database, sending back the requested data if it exists. Once the data is acquired, the web server can handle the data to display it in a readable format.

4.4. Miscellaneous Software Technologies

- 4.4.1. HTML – HyperText Markup Language
- 4.4.2. CSS – Cascading Style Sheets
- 4.4.3. JavaScript
- 4.4.4. Node.js
- 4.4.5. MySQL
- 4.4.6. Google Cloud / Google Cloud Run

5. RATIONALE

Cloud technologies were used to meet the stakeholder requirements for a client database not located at HCAR facilities. The choice of Google Cloud for the Cloud Computing provider was made after research into current industry leaders that purport adherence to HIPPA regulations. In particular, the choice of Google Cloud SQL was thoroughly documented in the HCAR Client Database System Provider Report, which can be found in the collection of Formal Documentation for this project.

JavaScript running on Node.js was selected as the server-side code platform due to popularity in the industry and the wide number of software libraries available. Along with the Express framework for JavaScript, this software took advantage of the traditional N-tier architecture for web applications.
MySQL was selected as the project's database management system due to its wide popularity, long-term support, and due to previous team familiarity with relational databases.

6. TRACEABILITY

User Story	Subsystem
US1: As a staff member, I want to be able to view client demographics and non-medical information to better serve our clients.	- Client-Side Webpage - Database - Express Routing / APIs
US2: As a staff member, I want to be able to add new clients to the database so that we can securely record client information.	- Client-Side Webpage - Client Data in Database - Express Routing / APIs - Admin Control - Employee Accounts
US3: As a staff member, I'd like to be able to upload my client documents in one location for convenience and so that I can see all my client's information in one place.	- Client-Side Webpage - Cloud Storage - Accounts
US4: As a staff member, I'd like to be able to view my client's documents in one location for convenience and so that I can see all my client's information in one place.	- Client-Side Webpage - Document Export - Database / Cloud Storage - Accounts
US5: As a staff member, I'd like for the purchase of the service date of expiration to notify me (or turn red) in some way when it is getting close to expiring, so I can avoid missing this date and safely obtain another purchase of services in a timely manner.	- Displayed in results.ejs - Client-Side Webpage - Stored in Database
US6: As a staff member, I want to be able to write/upload case notes so that staff assigned to that client can be better informed about said client.	- caseNote.ejs client-side webpage - routing in index.js - Database Storage for case note details - Cloud storage for case notes and documents
US7: As a staff member, I want to be able to review case notes so that myself and others can be better informed about our clients.	- caseNote.ejs client-side webpage - routing in index.js - Database Storage for case note details - Cloud storage for case notes and documents
US8: As a staff member, I want to be able to log into the program so that I can keep client data	- login page at root - redirection to login page if logged out in

securely protected.	index.js - APIs in QueryParser.js to manage privileges - Database to store accounts with sanitized and hashed values
US9: As a staff member, I want the program to automatically log me out of the system if I am inactive for 5 minutes, to prevent other people from using my account while I am away.	- Timeout set to approximately one day in backend - logout routes user back to login page through routing in index.js
US10: As a staff member, I want to filter the client list by Program, Purchase of Service expiration date, and by client demographics such as name, address, and program/service code so I can reach a specific client.	- search.ejs prompts the user to search using these fields - database stores the required fields and clients - API call grabs the clients from the database given the search criteria
US11: As a staff member, I want to be able to recover my account in case I forget my password, so I can continue using my designated account.	- account details stored in database - account recovery not yet implemented outside of google cloud run
US12: As a staff member, I would like my reports to be customizable with a list of checkboxes for each information field, so when I go to export the data as a PDF I am given the requested fields.	- PDF reports are made for the following documents: {mailing list, list of all clients, medication report, list of POS expiration, casenote}
US13: As a Manager, I want to have a higher access level to the system that allows me to manage and control staff/directors in lower access tiers, so some sensitive business information and abilities can be blocked from the other users of the system.	- Admins have privileges to control the system, edit client details
US14: As a Manager, I want to be able to see edit history for a client my colleague recently modified, so that I can hold my colleague accountable for any mistakes they make and reduce the amount of errors in client files.	- Account numbers and dates are tied to all documents produced by an employee - Editing client details is unique to admins