Proposal

Project Name: Requiem Registry

Team Member Names

Samantha Cook
Lucas Gamboa
Bricen Hicks

Abstract

The Requiem Registry aims to create a website where people can come to and see information on loved ones who have passed away and give the admin a streamlined way to manage the cemetery. The website will have a page for anyone to access and see basic information on the people buried there. The administrators will have a login to manage the number of plots, information about each plot, and the map features. The overview of the plots from the admin view would have all the information about each grave; the map will be more interactive, allowing the user to click a plot and open up the person view page. The person view page will include each person's first name, last name, birth date, death date, and plot number.

The public side of this website will be a cemetery search engine where anyone can search for the name of a loved one to see if they are in the graveyard. This will act similarly to websites like FindAGrave. There will be an overview of the cemetery, with a search bar to find people. The admin side will require a login; once the admin is logged in, there will be access to many more features. The administrator side will be able to set the size of the cemetery, enter and edit plot information, and access the interactive map.

Description

Requiem Registry will be a website for cemetery management and a search engine to find loved ones who might rest in that cemetery. This website will be divided into two different levels of access. The two different access levels are just for any user vs. an administrator user. The two levels will have about three to four web pages on the user interface.

The user side will be available for anyone with the website URL and will not require login validation. This page will show an overview of the cemetery with a search engine. The search will require the user to enter a first and last name to search the cemetery. If the search returns a match, the website will direct the user to the second view, the deceased page. This deceased page will show a detailed page of all the information the database has on that particular person. The deceased page will include first, middle, and last names, suffixes, date of birth, date of death, and a brief obituary. This concludes everything that the user can interact with.

The administrator page will require a login page. The login page will be done through Google Identity API, meaning any administrator user must have a Google account to log into the system. The Google Identity API will allow us to have a secure login without building it from scratch. Once the administrator user is logged in, there will be a home page for the administrator user. This home page will have buttons for the other pages, such as the cemetery, map, and plots/person.

The cemetery submenu or button will lead the administrator user to a page where they can add and edit things related to the cemetery. This will include the picture and description on the website so that the user can see the size of the cemetery. This page will be editable but on a more general board view regarding the cemetery.

The map page will be an interactive feature with an overview of the cemetery and its plots. If the administrator user clicks on one of the plots, it will bring them to the information page of the deceased buried there.

The plots/person button will allow the administrator user to add and edit information about each plot. The number of plots shown here will depend on what the administrator user selected on the cemetery page. The deceased page here will be the same as what the end user sees but will allow the administrator user to enter and edit the information.

This project's technical aspects will be broken into three categories. There will be a database, a server, and a user interface. The database will use PostgreSQL and be communicated to the server via SQL queries. The database will house all the information that the administrator users enter. There will be a table of plots, which will be the cemetery. There will be plot numbers and people objects, which will have a one-to-one relationship. There will be a personal table with every deceased and seven fields. The fields will include each deceased’s first, middle, and last names, followed by a suffix, birth dates, death dates, and obituaries. The back-end will be a server hosted

on Vercel. The languages used for the back-end will include Node.js, Typescript, and JavaScript. The back-end will communicate with the PostgreSQL database via SQL queries. The front-end of the website will be done using HTML styled with CSS. The front-end will communicate with the back end via HTML POST and GET API calls.

The purpose of the Requiem Registry is to create an environment where managing a cemetery and all pertaining information can be done quickly, effectively, and efficiently. It will also allow regular users to search for names that might be in the cemetery and see an overview. The product idea came from the first-hand experience of team member Samantha Cook, who works with a small, older cemetery that does not have a more effective way to manage its data besides pen and paper. She serves on a committee board tasked with caring for Mallory's Cemetery in Chapmansboro, TN. This is where she saw a need for software. We aim to create a user-friendly program to manage and upkeep small cemeteries. Many small cemeteries are more than likely faced with the same challenges; this software aims to provide a tool that can drastically improve record keeping.

Feature List

A list of features that will be completed by the end of the semester

- A search page will include an overview of the cemetery. The search will require the user to enter a name; if there is a match, it will bring up the deceased's information page.
- An Information Page for the deceased will include name, birth date, death date, and plot number.
- An Admin Page with capabilities to set the size of the cemetery, add and edit plot data, and present an overview of the plots.

A list of features that will be completed if there is time

- Interactive map.

A list of features you would like to implement but cannot be completed this semester

- A means to create and maintain links to the deceased’s obituary from their respective funeral home.
- Include a family tree in the database to track relationships between the deceased.
- A system to track next of kin with their contact information.

Technology

Platform: Chrome

Our target platform is a website; we will use Chrome to test our website as we build it. We will be able to ensure it works smoothly with this web browser, but more than likely, it will work on any other web engine.

IDE: IntelliJ

IntelliJ is an IDE that can handle the build stack we have. Therefore, it will be the easiest option to work out of.

Programming Languages: HTML, CSS, TypeScript, Javascript, SQL, Node.js

The front-end of our website will be built with HTML and CSS. The back-end will use TypeScrip/JavaScript and Node.js. The database will use PostgreSQL to communicate SQL queries to the server. The front-end and back-end will communicate via HTML GET and POST calls.

Third-Party libraries and tools: Google Identity API

Google Identity API will be used for our login page. This will save us time in development as we can use this plugin, and it will be taken care of. The login will only be required for the administrative users.

Server software: Vercel

Vercel will be the hosting service, allowing us to make a public website for all users. It will connect to a PostgreSQL database, making the connection point between our systems seamless.

Communication software: Discord

Discord was chosen as our primary communication software because of the ease with which it can be accessed via mobile and desktop, its capacity to share images and files, and our team's existing familiarity with the software.

Server Information

Vercel is one of the server hosting services recommended for a project of our scope. It was chosen among the other choices because it is compatible with most of the tools and languages we plan on utilizing. This will allow us to use the development stack we have picked out.

Data Sources

The data source for our project will come from Mockaroo; for demonstration purposes, we will use AI-generated data. This will allow us to quickly make many different data sets and test different cases.

Team Members' Background

Lucas Gamboa

I have finished most of the courses needed to get my bachelor's in computer science and am currently taking the last necessary courses. Beyond this, I have not had the opportunity to extensively apply my knowledge outside of the classroom, which may show during this project's lifetime. However, most of what I have learned is in my current recollection or may be quickly relearned using the textbooks I have kept from each semester.

I will be responsible for the Requiem Registry's database. This will include designing and normalizing the table structures to be in third normal form (3NF) to reduce redundant data, implementing stored procedures and triggers, and coordinating with Bricen Hicks to get the database working with the back-end portion of the project. Additionally, I will be responsible for commenting and documenting my code and work throughout this project and assisting my team members to the best of my abilities when needed.

Samantha Cook

I am unfamiliar with most web development, and most of my background is in standalone applications. I am currently taking the database course this semester. I have industry experience from my internship; this experience has also been for standalone apps. I do have some experience in TypeScript, which I plan to leverage for the back-end and front-end portions of the project. I have had experience with Docker and building a website that way; these small instances of web experience should give some background, along with lots of research. The experience and knowledge gaps will be a hurdle to overcome, but with research, we should be able to overcome them easily. I will be responsible for the front-end development of this project.

Bricen Hicks

I have completed server and client-side development classes and a database management course. These classes focused on HTML, CSS, javascript, php, and mysql. I hope my knowledge in similar structures will let me get a grasp on using javascript as a server-side language. I am still a beginner and feel intimidated by the connection between complex systems. The programming of the project seems interesting and straightforward, but the complexity of a connected full-stack application is daunting.

I will be responsible for the communication between the database and the front-end.

Dependencies, Limitations, and Risks

Dependencies

- Having a web browser, such as Google Chrome, as our chosen platform may cause issues if an update for the browser leads to new bugs.
- Vercel may change its policy regarding the tools it allows. A roadblock like this would change our plan and force us to find a workaround. It would also cost us wasted time for the invalid tools we worked on and the new environment we would have to learn.

Limitations

- The team's experience with the tools and languages needed for this project is incomplete, which may mean our more ambitious features will be infeasible with our allotted time this semester.

Risks

- Bugs created by software updates.
- Significant learning curves for the team may lead to progress getting stalled.
- Schedule conflicts may prevent in-person meetings between all team members.
- Using a hosting website and language requirements.
- Learning JavaScript and TypeScript as the server-side language.
- Understanding how to connect the database to the hosting website.
- Implementing a login system for the administration while keeping the base website closed to all other users.
- Complexity between different systems, especially at the beginning of the project.

Timeline

January

- Complete Proposal Documentation.
- Get the GitHub Repository set up and add the proposal document.
- Research the software we plan to use to expand our knowledge on tools.

February

- Install necessary systems and implement the base structure.
- Set up Vercel (hosting website).
- Construct the database.
- Connect a simple test website with the back-end.
- Get all necessary and required documentation into Github.

March

- Front-end development
- Create different appealing HTML pages.
- Implement communications between front-end and back-end.
- Begin testing the product to find possible problems within the software.

April

- Polish server communication and Quality Testing
- Secure the communication between the back-end and the database.
- Test page functionality to determine weak points.
- Fix the remaining bugs to ensure a completed product.
- Complete poster for the project.

May

- Finish source code.
- Work out any last-minute issues that may arise.
- Give presentation.