A Million Thanks

University of California, Riverside

Github Repo Link: https://github.com/CS-UCR/final-project-getjeff-d

Authors

Name	Github Profile
Garret Lim	https://github.com/glimreaper
Hongchao Yu	https://github.com/davepie101
Waheed Sharifi	https://github.com/wshar001
Marcos Miranda	https://github.com/mmira014

Last Updated 9/30/2019

Table of Contents

1. Detailed Description of Project

Provide a detailed description of the project / application. Small details may have changed since submitting the project proposal, so be sure to update the project description.

The goal of the project is to help A Million Thanks automate the process of recording the addresses of mail senders into a database. We will capture the address of the sender of received mail using the Google Cloud Vision optical character recognition API. We will create a MySQL database that can hold all the addresses of the senders and support queries of address features including filtering by city, state, and zip code. Our project may allow the user to export address lists to a file as a json dump. Our project may also include a heatmap of where mail is received from, which we will implement using an existing API such as MapBox GL JS.

2. User Stories / Use-Case Diagrams / Requirements

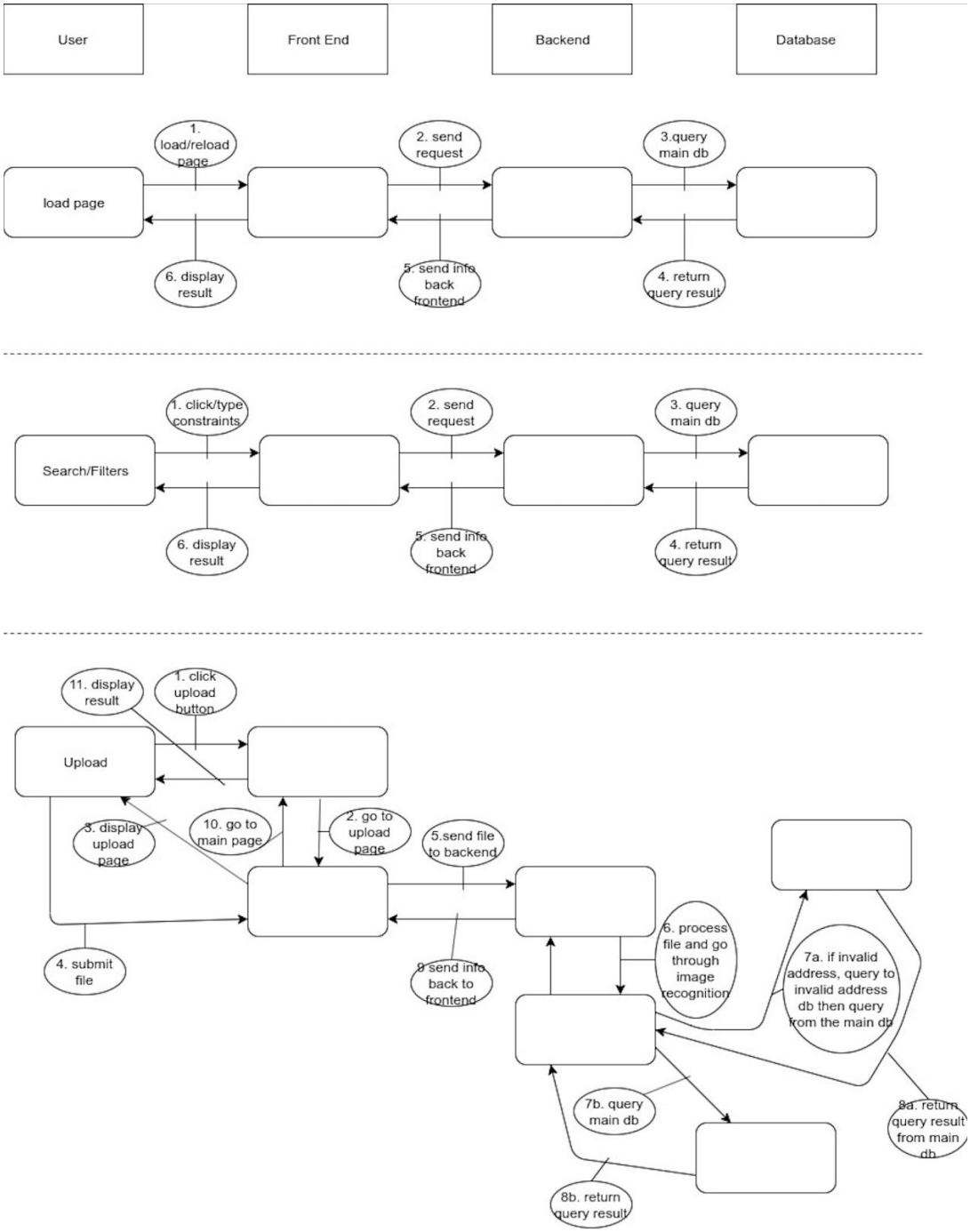
Often times we use “user stories” as a way to capture requirements. User stories are short, simple descriptions of a feature told from the perspective of the person who desires the new capability, usually a user or customer of the system. They typically follow a simple template: As a < type of **user** >, I want < some goal > so that < some reason >. Be sure to think about the user stories from the perspective of different users of the system.

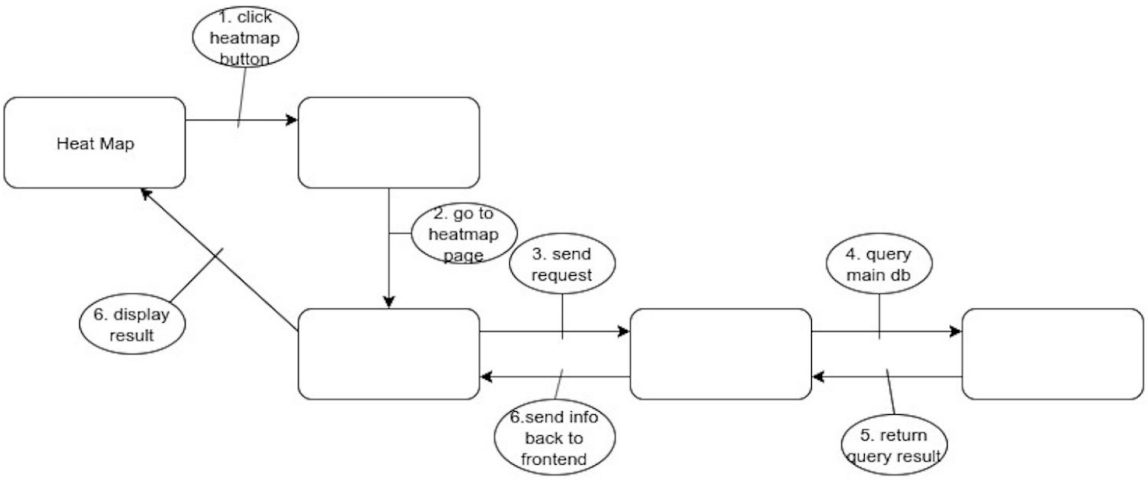
- As a member of A Million Thanks, I want to view address of received mail so that I can send mail back to the senders.
- As a member of A Million Thanks, I want to upload a collection of photos of handwritten addresses so that the project can translate the handwritten text to digital text.
- As a member of A Million Thanks, I want to view addresses filtered by city/state/zipcode so that I can better understand the locational distribution of received mail.
- As a member of A Million Thanks, I want to be able to see the original image corresponding to a database entry so that I can attempt to resolve exceptions such as invalid addresses.
- As a member of A Million Thanks, I want to view a list of invalid addresses received by the organization so that I can attempt to resolve them and send mail back to the sender.

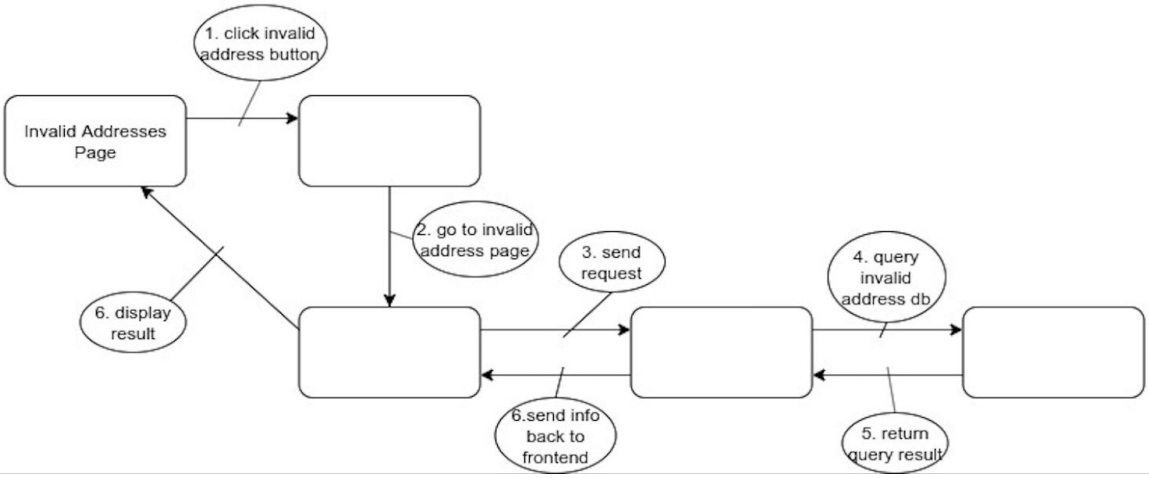
- As a member of A Million Thanks, I want a visual representation such as a heatmap of the distribution of addresses so that I can quickly see the distribution of the locations of senders.
- As a member of A Million Thanks, I want to export results of queries to a file as a json dump so that I can quickly parse addresses within other programs or contexts.

You can also include GUI-Mockups to depict the user-case and flow of your application.

3. Sequence Diagrams







4. Detailed Design

In this section, describe the high-level design and low-level design of your application. This will be a “living document” that will gradually evolve to include additional low-level design details. Be sure to keep updating your design as you understand your application and use-cases in more detail.

a. High-level Design Description

Our One Million Thanks Project will be comprised of several different components. Some of these components will be programmed, while others will be implemented using open-source

APIs. The language the project will be implemented will be a combination of Javascript and SQL. The front-end portion will be comprised of a user interface developed using React and Javascript. The back-end portion will be implemented with React and MySQL. This setup will allow users to have access to an interface capable of providing a list of addresses based on a set of constraints that is defined by the user.

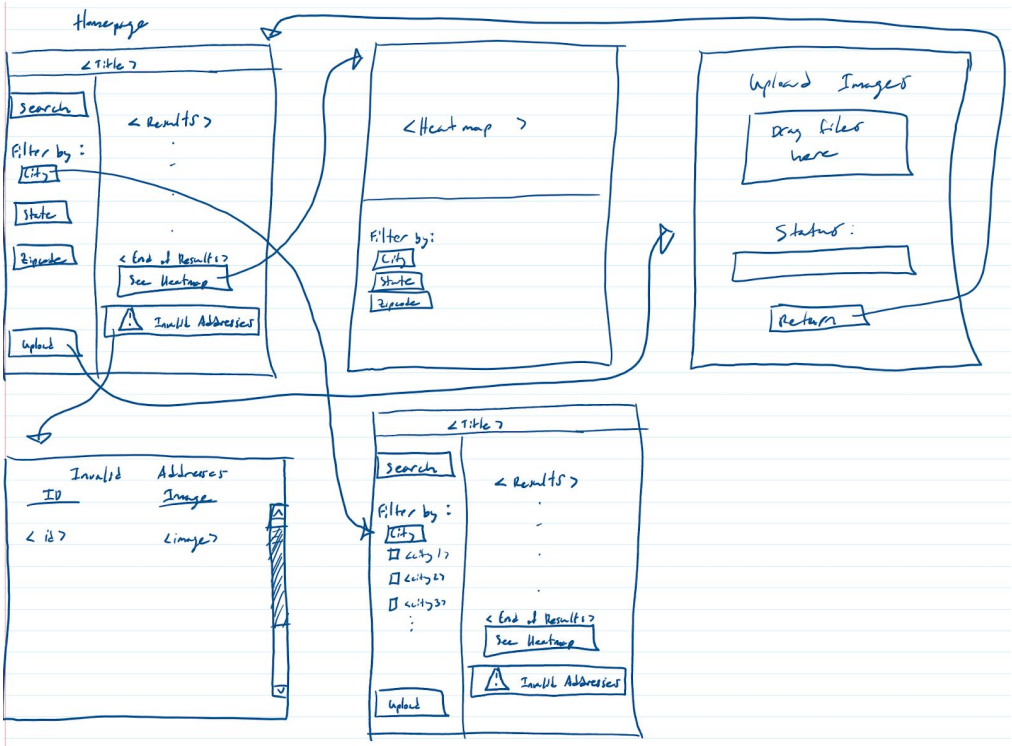
b. Front-End Design

Include a description of the front-end, including mockups, tools you plan to use. Be sure you break the front-end design into features and assign these features to a team member. This needs to be captured on your Github Kanban board.

The front-end design of this project will mainly be consisted of an interactable user interface developed using HTML, CSS, Javascript, and React. Below is a list of key features that will be implemented in our interface.

Features

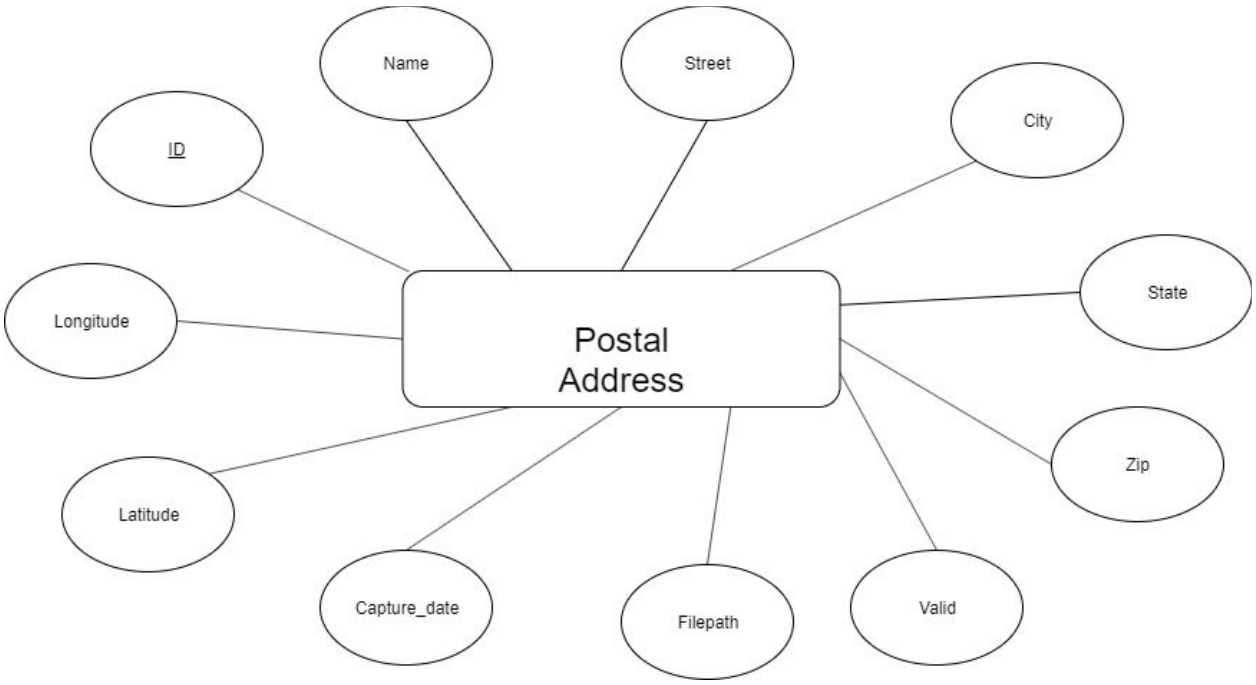
- Users may upload an image file which will then be scanned for a possible address and uploaded into our database
- Users may obtain a list of addresses from our database based on set filters such as zip code, state, city, etc.
- There is a search bar available for users to search up specific addresses
- There is a clickable heat map section that would display a heat map of the distribution of addresses. Reclicking this would go back to the lists of addresses.
- There is a clickable button that would lead to a list of invalid addresses



c. Back-End Design (including Database)

Include a description of the back-end of the application. This includes the server that processes user's requests and the database that maintains the data. Be sure to describe the interaction between the two processes and include a ER-diagram. Describe which tools/language you plan to use.

The back-end portion will be developed with Node.js, Express, and MySQL. The server our interface will be run on will be through Node.js and Express while our database, containing all the addresses and names, will be created with MySQL.



d. Design Constraints

Include any updates to the design constraints

Size Constraint -- In terms of the size of our database, A Million Thanks has approximately 10 million letters sent so our database should be able to accomodate all the letters plus possible future letters.

Robustness Constraint -- Our project should gracefully handle invalid addresses caused by sloppy handwriting or other factors by marking addresses as valid or invalid so that they can be quickly queried and identified as valid or invalid.

e. Engineering Standards

Current standards that your application will utilize:

Description	**Standard**	**Governing body**
Standardized software life	IEEE 12207 - 2017	IEEE

cycle process		
Standardized javascript	ECMA - 262	ECMA
Standardized SQL	ISO/IEC - 9075	ISO

5. Bibliography

List all references, APIs, datasets, code-snippets, etc. that you plan to use in your project.

Google Cloud Vision: Google Vision API allows us to detect handwriting using Optical Character Recognition (OCR) https://github.com/googleapis/nodejs-vision

Google Maps API: Address validation to confirm whether or not an address is legal. Address text parsing and retrieval of longitude and latitude. https://github.com/googlemaps/google-maps-services-js

MySQL https://www.mysql.com/

NodeJS https://nodejs.org/en/

Express JS https://expressjs.com/

ReactJS https://reactjs.org/