Birthday Reminder App

CS405- Software Engineering Project Report

Ramon Lavender, Kristin Barrett, & Roderick Bishop

SC State University, Fall 2021

**Abstract**

Forgetting birthdays are a hurtful but common occurrence in our lives. This problem, though not novel in its solution, was one that “Birthday Reminder App” seeks to abate. “Birthday Reminder App” is a standalone web application that will allow users to add their friends, family, and other connections via the web browser that they use every day in a modern, responsive, and easy way. This project report describes the inception, design, implementation, testing, project outcomes, further steps, and learning statements from creators. The full gamut of the basic software engineering experience was explored in the making of this project. The entailed project was completed by Ramon Lavender, Kristin Barrett, and Roderick Bishop using React.js, HTML5, and CSS3 for the Computer Science 405 course in the Fall semester of 2021.

Table of Contents

[Project Overview 2](#_Toc89557612)

[Problem: 2](#_Toc89557613)

[Solution: 2](#_Toc89557614)

[Tech Stack, Tools, & Platform: 2](#_Toc89557615)

[Project Measures of Success & Expectations: 3](#_Toc89557616)

[Requirements 4](#_Toc89557617)

[Functional Requirements: 4](#_Toc89557618)

[Nonfunctional Requirements: 4](#_Toc89557619)

[Data Requirements: 4](#_Toc89557620)

[Solution In Depth 5](#_Toc89557621)

[UML Diagram 5](#_Toc89557622)

[Use Case Diagram 6](#_Toc89557623)

[System Components & Architecture 7](#_Toc89557624)

[Testing (Scenarios and Methodologies): 7](#_Toc89557625)

[Implementation: 8](#_Toc89557626)

[Data Models: 9](#_Toc89557627)

[Final Project Outcomes 10](#_Toc89557628)

[Project Measures of Success Revisited 10](#_Toc89557629)

[Project mVP Visualized 10](#_Toc89557630)

[Contribution 11](#_Toc89557631)

[Learning Statements 12](#_Toc89557632)

[Future Plans For Project 13](#_Toc89557633)

[Feature Set for v1.0.0: 13](#_Toc89557634)

[Feature Set for v1.0.0+: 13](#_Toc89557635)

[Resources and References 14](#_Toc89557636)

[Project Resources: 14](#_Toc89557637)

# Project Overview

## Problem:

Remembering birthdays is one of the best ways to show a person that you care for them, but too many people forget. Forgetting is something most humans do almost daily, we can’t blame them for forgetting but we also want to remember to celebrate those that we love the most. The world needs a solution to this issue and help others show they care with one of the most accepted acts of love humans can offer.

## Solution:

We propose a standalone React web app that will allow users to add birthdays for people that they know and get notifications when their birthday is arriving.

## Tech Stack, Tools, & Platform:

* Our project uses ReactJS to create a standalone app. We also use HTML for markup and CSS for design.
* For testing, we used JEST.js for our framework to unit test the functionality in our app. The framework gives users robust context as to why our functions don’t work, has a great API and requires zero configuration.
* We utilized Github for versioning, cloud backup & project management. It helped us sync our code across devices and workspaces, manage new changes, tag releases when new versions are available, and keep our “source of truth” or main branch isolated from current work so that we never lose a working version of the application. This use of Github aligns well with the industry standard.
* We used a note-taking platform, Notion, for our Kanban boards. Although Github has a feature for this, we decided to go with the simpler and more customizable option using Notion.

## Project Measures of Success & Expectations:

Our minimally viable product will be a stable version of the app that can:

* Run on localhost
* Loads the page and all assets fully with page load animation
* Allows the user to view all people in the list (hardcoded)

We fully expected to accomplish this mVP at the outset of this project. Our team is satisfied with these requirements given the product planning, design, and implementation that will take place in the learning environment. Our mVP was carefully defined with the intent of giving our group a challenge, as well as with the intent to give us enough breathing room to focus on exploring the full set of tools, processes, and concepts learned in class or required for project completion. This mVP allows us to also display the potential for the app’s development as well.

# Requirements

## Functional Requirements:

* **User can:**
  + Add a Person
  + Add a birthdate for a Person
  + Add a Person’s birthday celebration information
  + CRUD birthdays, celebrations, and people
  + Sort people
  + Set reminders
  + Toggle basic UI elements
* **Application will:**
  + Notify a user when a birthday is near via a configurable value for when to remind a user
  + Show past birthdays as well as upcoming ones

## Nonfunctional Requirements:

* App will hold its data in browser cache or in cloud-based db.
* Application runs and interacts with the user locally

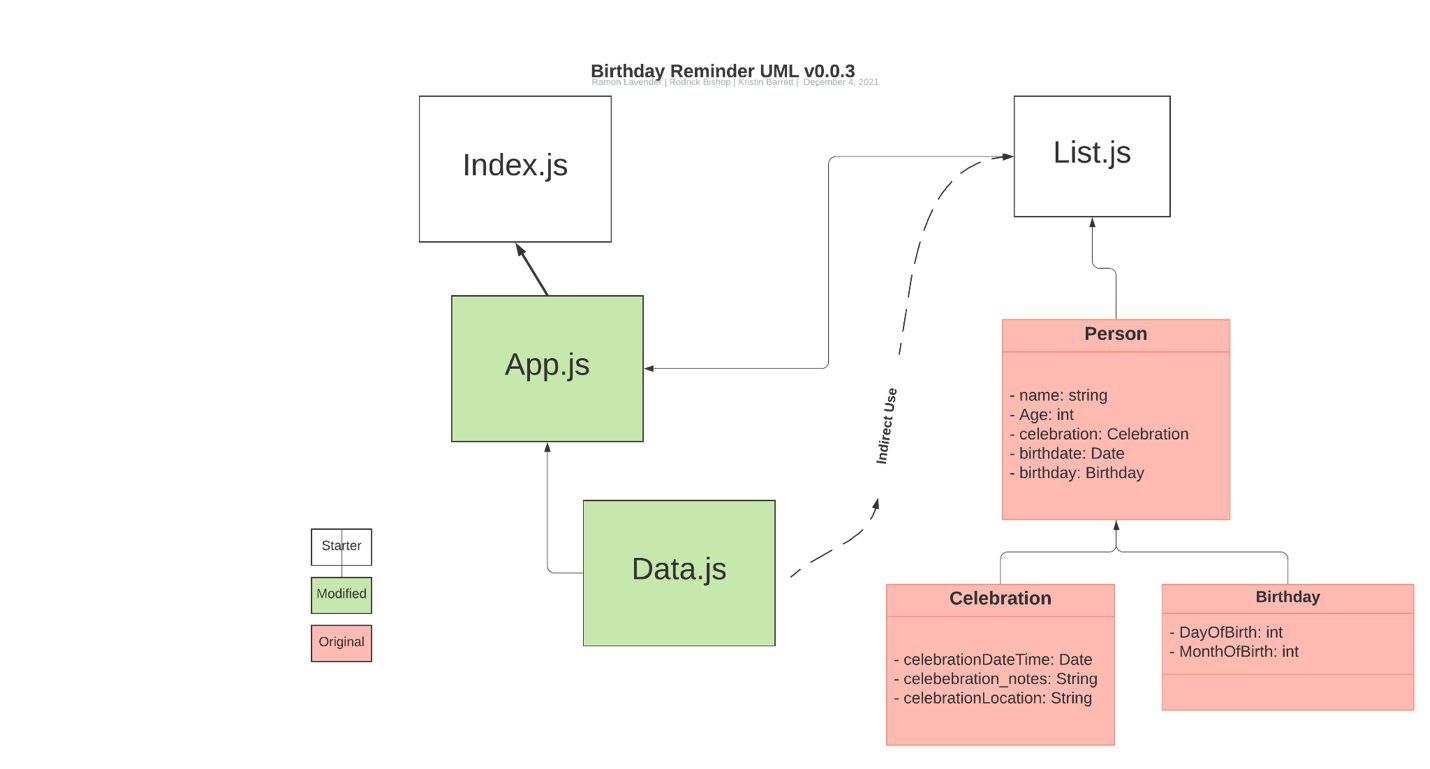
## Data Requirements:

* The user will be able to upload pictures of their contacts for their birthday reminder

# Solution in Depth

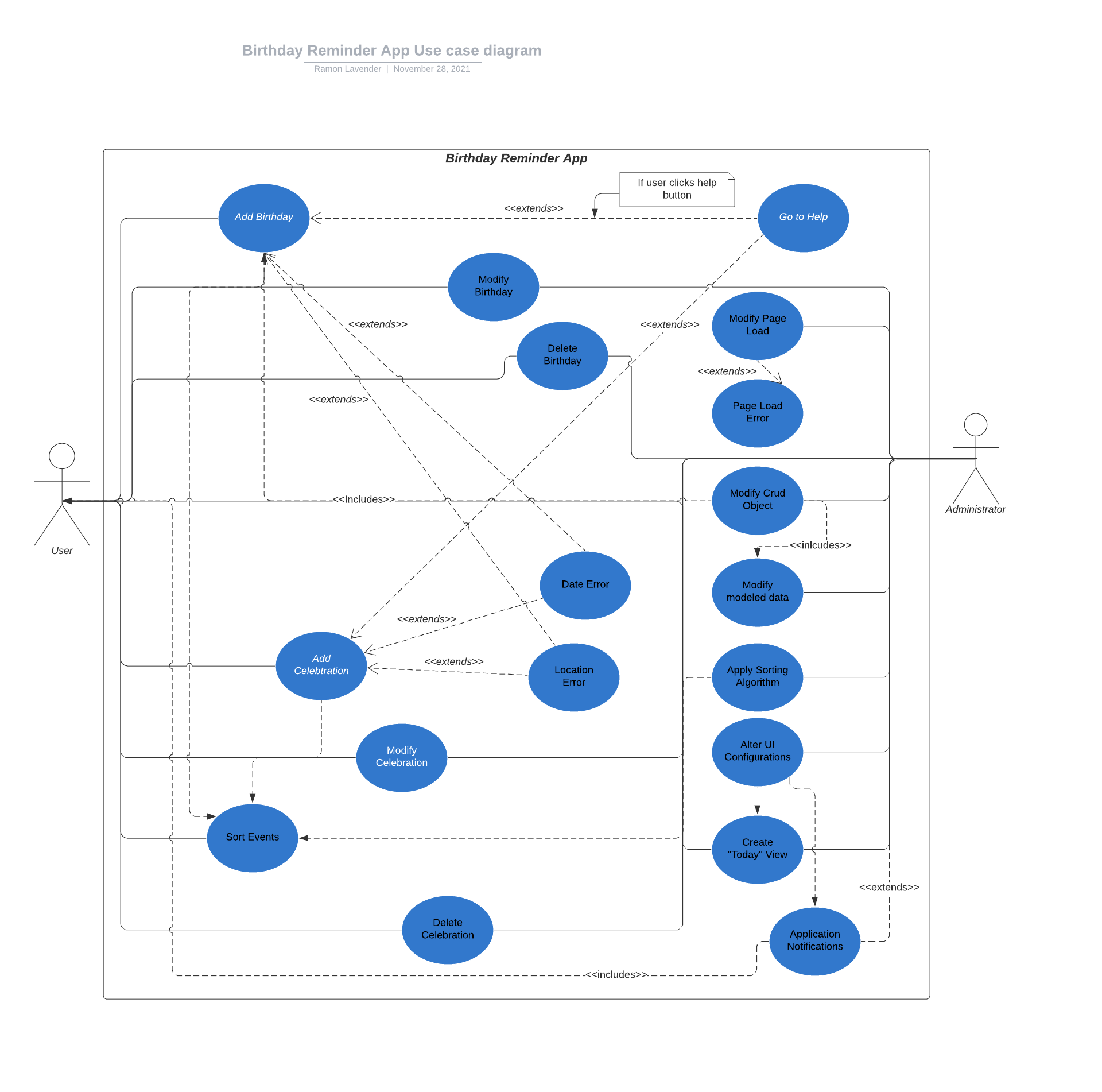
Our final solution code, including all supporting document and diagrams can be found at <https://Github.com/roderick-bishop11/CS405Project>

## UML Diagram

****

The user is able to create a Person object, which in itself creates a birthday object. There is also an option to create a celebration object. Celebration includes a date and time, notes, and location. Birthday contains a day and a month that will be used for sorting and other useful functionality that would rely on this information in a non-Date type.

## Use Case Diagram



The user and administrator are represented in this use case diagram. The user can add, modify, and delete birthdays and celebrations, and also sort events. The admin can do much more such as modify page load, alter UI configurations, and can even do a few of the same user actions, e.g. modify, and delete birthdays and celebrations.

## System Components & Architecture



This three-layered cake represents the three-tier architecture. The users make requests to the browser, which makes requests to the business logic layer. Responses are returned in JavaScript and HTML.

## Testing (Scenarios and Methodologies):

Testing for the Birthday Reminder App is done with Jest.js and through user testing. Jest.js is a framework for testing in JavaScript that requires no configuration and has an extremely simple implementation for basic unit tests. Our group chose Jest.js because of its simplicity and style. Given the nature of our application, live testing methods were common throughout our short testing regime as well. here are not many test scenarios outside of the CRUD operations, and those were not included in our mVP. In v1.0.0, it is expected that we will implement tests as CRUD features come online.

However, for the mVP only functional testing was needed, since the only function that manipulated user data was an onClick “Clear All”.

## Implementation:

Our code is based on a beginner JavaScript project tutorial from freeCodeCamp. The original code, as well as the video tutorial can be found in the references section of this report. Our group did employ the starter code and built off of its base to create the app that was submitted to the professor.

1. **How the app works (starter code):**

Given our group’s use of starter code, it’s important to capture here what the starter code does and how our application works as a whole. Birthday Reminder App works in the following manner:

* 1. Entire application runs from App.js. It is marked up by index.html which then employs index.js that acts as a wrapper for App.js.
  2. All designs and styles are located in index.css which is imported by index.js
  3. Test data is defined and stored in data.js and is used in app.js and list.js.

Future versions of this application will likely have a different implementation.

1. **Code additions to reach v0.0.3:**

Our code for v0.0.3, the mVP, has been largely modified in the following ways:

* 1. Data models that define how our data is organized.
  2. Data.js has been completely overhauled to match our people class definition
  3. A carousel element was added to the UI to view one birthday at a time.
  4. A loading page animation was added to facilitate a nice page load for the user
  5. Design elements added and changed in index.css
  6. App.js was significantly overhauled to show our entire dataset with added UI elements and functionality.

Given the many additions to the starter code, we were able to meet our mVP requirements and deliver a product that is of our own making.

Most notably, with the addition of our data models, we redefined the data in Data.js using JS property syntax. The defined data gets imported as data in App.js and populated via the exported list creation function, people.map()in List.js. As a result, index.js will load the app and display the people in our dataset for the user to clear all people from the new list in v0.0.3.

All other files within the project are related to markup, function, project docs or JS packages and are pretty straightforward.

## Data Models:

The data models, or JS classes, that we created reflect how we view our problem. The Person class is the all-encompassing and organizing data class. It houses our other two support classes in v0.0.3.

Our data models are defined as follows:

1. **Person**- A Person is the top-level class that houses all of the data represented in the UI. A Person has a **string** name, **int** age, **Date** birthdate, **Celebration** celebration and **Birthday** birthday.
2. **Celebration**- A Celebration is used to model all the information that is pertinent to a Person’s birthday celebration plans. Celebration was abstracted out from Person to reduce class size and to simplify maintenance. A Celebration has three class members: **Date** celebrationDateTime, **string** notes, and **string** celebrationLocation.
3. **Birthday-** A Birthday is a rudimentary abstraction for a normal birthday that allows us to store both the month & day of a birthday separately. The intent was for this class to be used for performing logic when sorting the birthdays, but in later versions might very well be deprecated as it is somewhat redundant in its design. Birthdays have two members: **int** date and **int** month.

# Final Project Outcomes

## Project Measures of Success Revisited

Version 0.0.3 of Birthday Reminder App meets and surpasses the mVP requirements set at the inception of the project. Overall, the group is elated with the outcome of this small project and certainly better software engineers as a result.

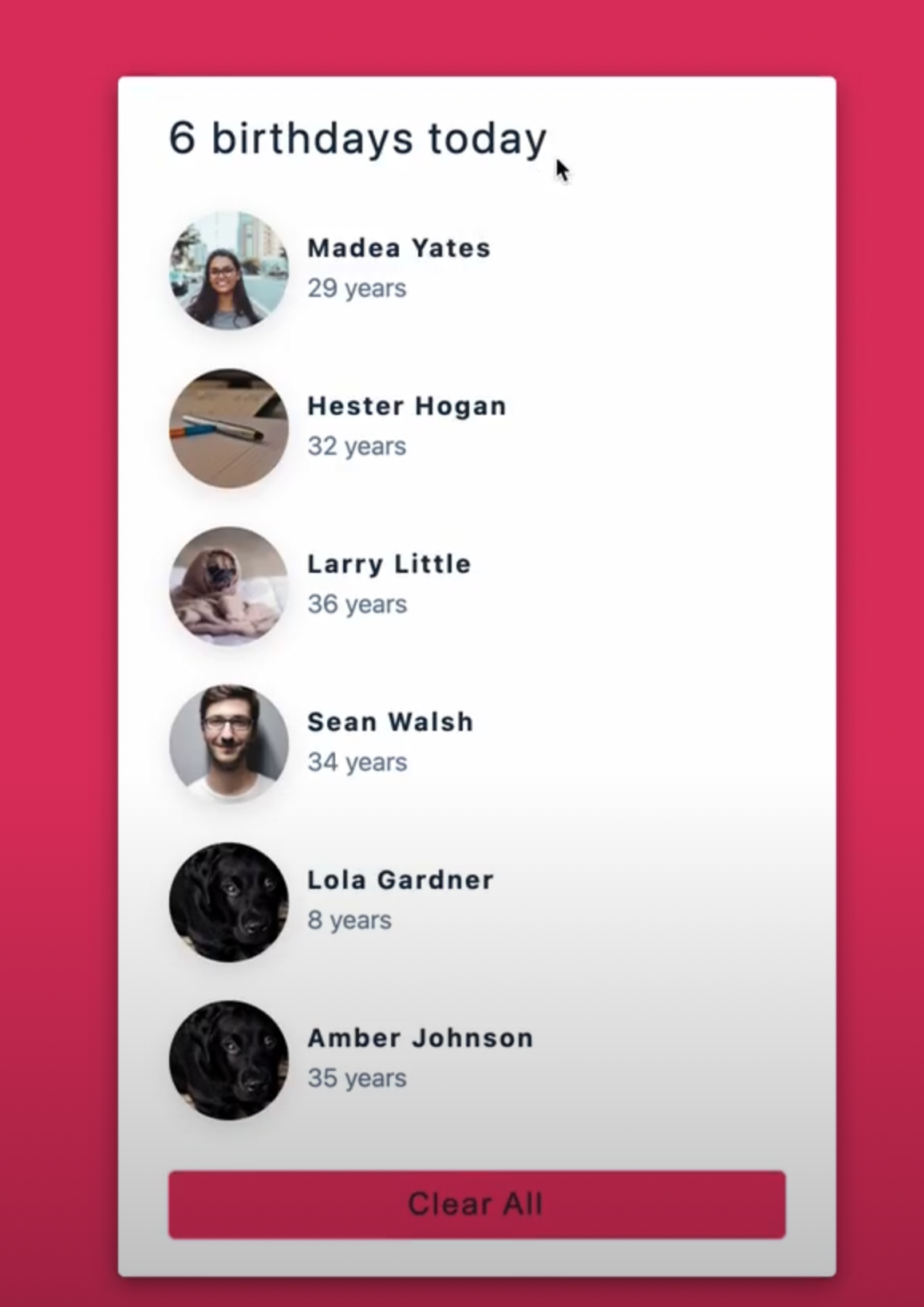
## Project mVP Visualized

“Birthday Reminder App” reached its mVP at v0.0.3 and was demoed to the professor in the same version. This section of the report adds a visualization to that version but will be deprecated in later releases. v0.0.3’s UI features a central list showing all the people with birthdays. At the top of the list, there is a simple text heading for the user to know how many birthdays are in the list. The default heading does not account for birthdays at different dates, however that is a feature that will be improved in v1.0.0. A user can see a photo, age, and name for a Person in the list. A user can see one button that clears all of the people in the list of Persons, and the heading will change to “0 birthdays today” when the onClick function is ran.



*Excerpt of App.js defining onClick() function that returns a blank list when the ‘clear all’ button is pressed.*

For this mVP version, all photos are loaded from API’s as can be seen in Data.js. Each name and age can be seen also.

*Pictured above is a screenshot from v0.0.3 of the project. v0.0.3was actually the first demoed version of the project. A video demo of our mVP can be found in the references portion of this report.* 

## Contribution

As with every group project, each group member must contribute to the group's content. Our group had no issues working together, sharing any and all responsibilities in an equitable fashion. All group members contributed to the development of source code, design elements, as well as the presentation and report.

## Learning Statements

**Roderick-** This project was certainly interesting. We chose it because we wanted to challenge ourselves and take a leap of faith with a new programming language and newer development concepts learned in CS405. Overall, I am elated with our mVP, and excited to see what else we’ll build in the future with JS.

**Ramon-** Due to my expertise being in cybersecurity, creating this project in a language typically for web development was a challenge. I had to go out of my comfort zone and learn new methods, tools, and techniques.

**Kristin**- This project helped me gain a lot of insight on the basic building blocks of most applications. It also introduced me to development tools and the overall process of app development. This creative and informative process was very interesting yet challenging. With learning new concepts, we were forced to expand our knowledge and implement prior experience with new information. This project definitely helped me become more comfortable with certain aspects of software engineering and development.

# Future Plans for Project

## Feature Set for v1.0.0:

* Completed carousel UI
* CRUD
* Toggle night mode
* Save user data to browser cache
* Confetti animation on user’s birthdate
* Notifications with configurable values
* Carousel shows Person with closest birthday relative to current date on page load

## Feature Set for v1.0.0+:

* Allow users to add tags to celebrations
* Tags/celebration (or entire profile) privacy toggle
* Tags and celebrations can be searched/viewed by other users
* “Invites” can be sent to users and non-users via links
* Details view for people in the list.
* Two-factor authentication for users to login and access their saved birthdays from any computer

# Resources and References

All project files including the presentation, diagrams, development versions, tagged releases and this report are available on a public Github repository via this link:

[https://Github.com/roderick-bishop11/CS405Project](https://github.com/roderick-bishop11/CS405Project)

## Project Resources:

* YouTube Video Tutorials:
  + [Image Carousel - HTML, CSS, JavaScript](https://youtu.be/eywZbJ5PVjg)
  + [ReactJS Birthday Notifications App](https://youtu.be/fFAW9JZ18Wk)
* Other web resources:
  + [How to Create a Page Load Animation in React](https://javascript.plainenglish.io/how-to-create-a-page-load-animated-loader-in-react-793d42411fd5)
  + <https://www.w3schools.com/js/js_object_properties.asp>
  + <https://www.w3schools.com/js/js_object_properties.asp>
  + <https://javascript.plainenglish.io/how-to-create-a-page-load-animated-loader-in-react-793d42411fd5>
  + <https://www.w3schools.com/bootstrap/bootstrap_ref_js_carousel.asp>
  + <https://medium.com/allenhwkim/how-to-build-a-carousel-in-pure-javascript-98d758a18811>
  + <https://www.writethedocs.org/guide/writing/beginners-guide-to-docs/>
  + <https://stackoverflow.com/questions/35110505/es6-classes-for-data-models>