Problem:

Background

WePay is a web application used by UPMC and the University of Pittsburgh to administer research studies and the disbursement of funds to subjects associated with these studies. We started implementing WePay over ten years ago and for a period of several years active development became dormant. The application has a solid overall architecture. However, the implementation is dated to the point it does not support modern browsers , User Interface,or technical design patterns.

Goal

We want our application to be easily deployed to a broad subset of modern browsers and in a mobile friendly way. We want to do so in as modular a way as possible; we want to change the implementation parts of our system as our needs change without major changes to the implementation of the user experience.

Requirements:
Open-source, enterprise grade platforms

Backend: Rails
Rails has been growing in popularity in recent years, in large part due to being well-suited for rapid web development. It is also written in Ruby, a well-known and fairly easy to understand language. It is also a platform that all of our team members are familiar with. Given the short time frame of this project, it made sense to use Rails to develop a proof of concept. Rails is also enterprise-ready, an absolute necessity for a system in production.

Why not asp.net, php:
ASP.net - more robust, but not open-source
PHP - Slowest, similar to C, not as “clean”

Risks:
Currently, Ruby is regarded as one of the slower programming languages compared to other languages out there. If scaled improperly, the entire application can slow down easily. Technically, however, it can be scaled just as well as PHP and ASP.net

Frontend: Bootstrap
Bootstrap is a well-known and reliable framework, whose popularity ensures an excellent support community for developers. Most importantly, however, it is one of the more prominent responsive frameworks. Given that one of the main goals of this project was to make WePay compatible across all screens, making sure the application had a responsive design was absolutely critical. The fact that our team had experience with it also meant that we could get our proof of concept off the ground much more quickly and smoothly.

In addition, Bootstrap allows YouPayWePay team to easily change the color scheme and the logo within the html/css files. This allows the team to customize the application and sell it to other clients with ease.

Why Bootstrap over jQuery mobile:
jQuery mobile is another front-end framework that can be used. However, we decided to use Bootstrap over jQuery mobile for couple of reasons:

- More dependent on jQuery, which decreases performance
- Focused on mobile; there are issues when the website is viewed from desktop (i.e. grid not rearranging properly depending on screensize.)
- While Bootstrap is focused for desktop views, it can be easily customzied to look great on both desktop and mobile without creating separate views (i.e. if you shrink the screen, the nav bar changes to fit the smaller screen).

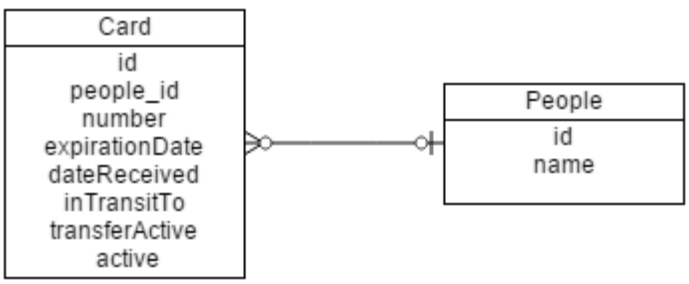
We wanted to stick with one framework: it is time consuming to implement, update, and maintain two frameworks for one web application. Thus, we concluded that Bootstrap was the best option.

Risks:
Bootstrap can make the application look similar, if not same, compared to other websites due to its popularity. While it is possible to customize Bootstrap to look very different form other Bootstrap-implemented websites/application, it is time consuming and somewhat defeats the purpose of front-end framework.

Data: js/jquery
jQuery allows for rapid web development, making it easier to deploy.

An explanation of their choice of stack and other design choices
High-level diagram of the POC and description of the implementation
Documentation of risks and challenges of the approach they chose

ERD:



Frameworks/Screenshot:
Talk about ‘why’ we chose this framework

Suggestions: