Team: HvZ App

Members: Santiago Rodriguez, Alex Hadley, Matthew Waddell, Kyra Clark

Project 2B.1 — Review Notes

GitHub: [https://github.com/alexhad6/HvZ-app/blob/master/Administrative](https://github.com/alexhad6/HvZ-app/blob/master/Administrative/Phase%202/architecture_2a.pdf)

[/Phase%202/review\_notes\_2b\_sr.docx](https://github.com/alexhad6/HvZ-app/blob/master/Administrative/Phase%202/architecture_2a.pdf)

Primary author: Santiago

# Team Breakout Room 5 Architecture Review

Product Description

* + Streamlined process is ill defined
  + Why make a product if you’re not sure of the feasibility?

Components

* App
  + Preferences – what is the result of the survey attached to? Unfinished sentence.
  + App analysis – if you’re going to mention specifically that you chose the AWS suite over others, you should probably say why. Otherwise there’s no point in saying specifically that you chose it over others
* Card Optimization Algorithm
  + Where do you plan to put this? You say you plan to put it somewhere on the AWS cloud, but where? Is there a server-like thing that you can have this function be on?
  + Will it be secure and not be able to leak your credit card info? Does it have knowledge of credit card information such as number, pin, and expiration date? Or does it abstract the credit cards to card A, card B, card C, etc.?
    - I found this information later
* User Storage
  + You mention a noSQL database, but do you know which type you’d want to choose? (key, graph, document, etc.)
  + Huge issue: The architecture states that you don’t know if plaid provides the functionality you need, but isn’t it essential to the service? Do you have a way to find out if it provides what you need before building your project? Do you have a way around it if it doesn’t provide what you need?
* Card Benefits
  + [aside] I think you could use the same database but just a different section of it to keep the stuff up to date
  + What type of database will you use?
  + What does it mean to normalize different cards?
* Personal Preference Function
  + Shouldn’t this tie in with the optimal card algorithm? It seems like there’s no point to this part if it doesn’t because people already choose the card they want to use based on personal preference.
* Functions in general
  + I think the personal preference and optimal card functions will end up being classes or at least a set of functions if you want to keep it from being disorganized. It may be helpful to plan for that.

Testing

* A sign-up/sign-in service is mentioned. There are many out there that will make this feasible, but I think that must be mentioned in the architecture if you’re planning to implement it.

Architecture Diagram

* Use consistent names in the diagram and across the entire document. The optimization function has been called too many different names. Neither database is referred to as database in the component title.
* Personal preference survey conversion function is listed as a component but is not on the diagram.

Design prototype

* It shows a cool app UI, but there should be some explanation as to what is being shown and what the prototype intends to show as possible

One last overall thing

* It’s not obvious to me why you’d need plaid in your architecture if you don’t have the virtual wallet functionality. You mention that you’d be updating a database with the most popular card benefits, so couldn’t a user just say they had that card rather than inputting their own card info in?