Anindya Basu – A09972316
Aishika Kumar – A12493796

stickDex

Available at

lipstickdex.firebaseapp.com

Overview

stickDex is a web application that allows its users to keep track of various lipsticks. The application helps users asses what is already in their collection as well as what they wish to purchase in the future.

Application Overview

- **Frontend:** HTML / CSS / JS
- **Services Used:** Firebase, Google Authentication
- **Previous Tools:** Older iterations of our app used SASS, Picnic.css, and Bootstrap; however, we decided to switch to a vanilla CSS and JS approach because we realized that these were tools that improved our "quality of life" as developers, but ultimately negatively affected our app in terms of page load time as well as total project file size.

Files

CSS/

-login.css -> contains the CSS for login page
-home.css -> contains the CSS for the home page after login

These two CSS pages have some small overlap in CSS however we opted to put them in two different files because we didn't want to flood our HTML files with calls to different CSS files increasing our page load times.

Fonts/

-contains the various fonts used in the website

IMG/

-contains banner image as well as default lipstick image

JS/

-home.js -> contains javascript for the home page

404.html -> contains HTML for error page
database.rules.json -> contains rules used for Firebase database

firebase.json-> configures which files are published on Firebase
home.html -> contains html for home page as well as Javascript, this was done to increase load time of the home page and not have user wait too long for first page of app
index.html -> contains html for login page

Minifying

The following websites were used to minify our various files:
https://cssminifier.com/
http://minifycode.com/html-minifier/
https://jscompress.com/

Comparison to Prototype

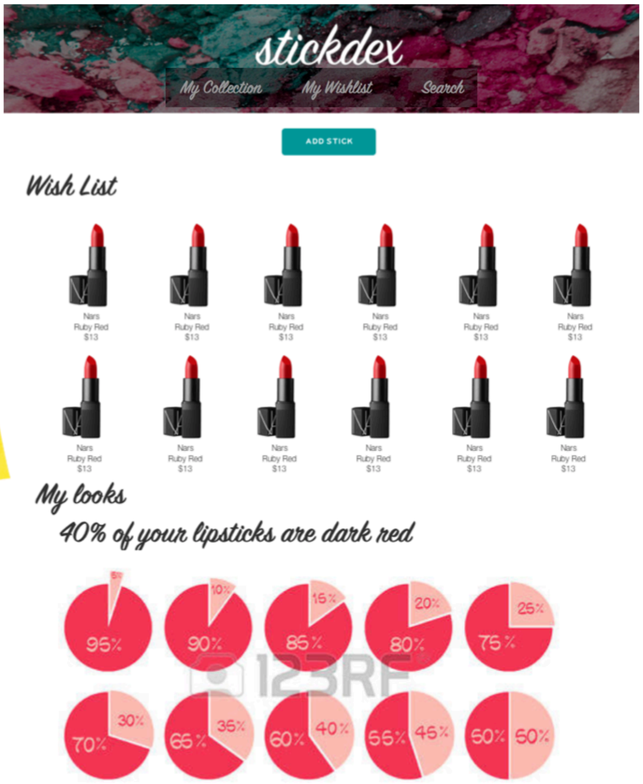
Our original screens were as shown below:

Compared to this, we believe our prototype has come exceedingly close to our original functionality. One thing that is different is the portion of graphs under the "My Looks" category. When designing our prototypes, we knew we wanted to include some sort of useful data here, to help consumers better understand their purchasing habits and also allow them to make more informed decisions in the future. Instead of implementing these pie charts for which we had no real purpose we did the following keeping in mind our two goals:

1. Help consumers better understand their lipstick collection.
2. Help consumers make more informed purchasing decisions in the future.

Our prototype has been upgraded to include the following:
An average amount of money spent on lipsticks so far, as well as an amount showing on average how much a wish listed lipstick costs. In addition to that we show side by side what colors the user already owns, as well as what colors the user wants. This is helpful to users because it shows, at a glance, what colors the user wants and needs, so that even if they can not afford an exact lipstick on their wish list, they can purchase one in a similar color when they are in a store. This tool implements the idea of **recognition over recall** that was discussed in class to empower purchasing decisions.

Prototype

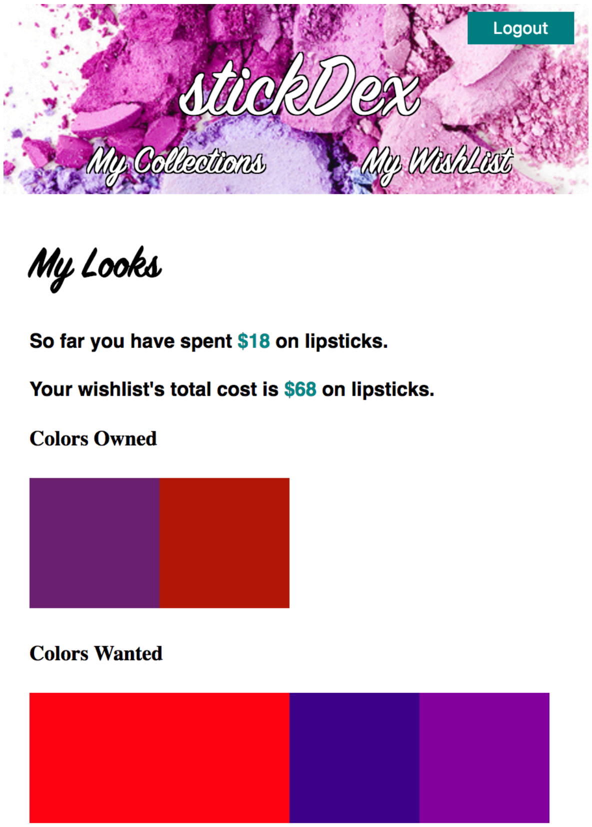


Wish List

My looks

40% of your lipsticks are dark red

Final Version



My Looks

So far you have spent $18 on lipsticks.

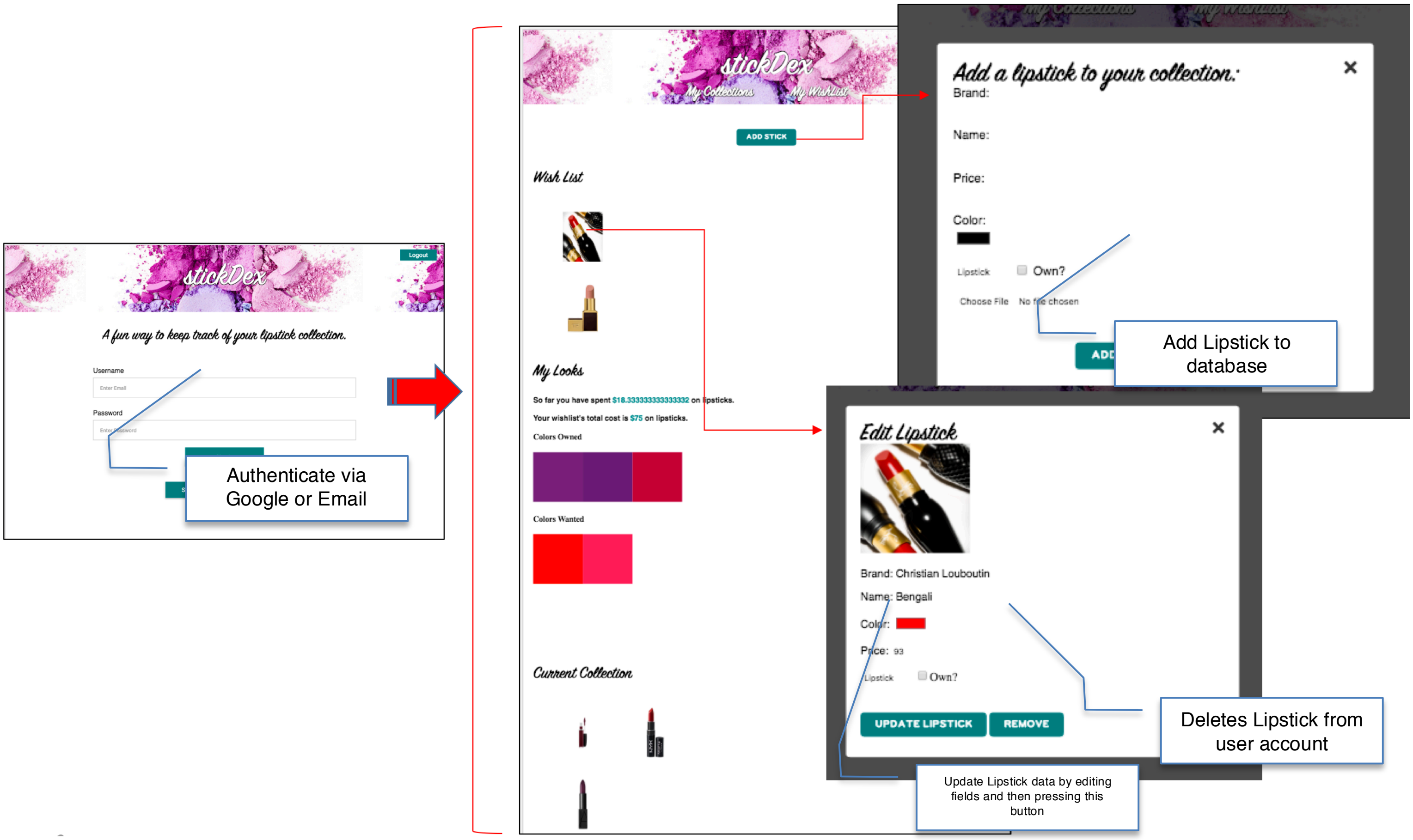
Your wishlist's total cost is $68 on lipsticks.

Colors Owned

Colors Wanted

UI Flow

We designed our app to have a single page layout (not including the log-in screen). We did this to ensure ease of use.



A fun way to keep track of your lipstick collection.

Authenticate via Google or Email

So far you have spent $18.333333333333332 on lipsticks.

Your wishlist's total cost is $75 on lipsticks.

Add Lipstick to database

Brand: Christian Louboutin

Name: Bengali

Price: 93

Deletes Lipstick from user account

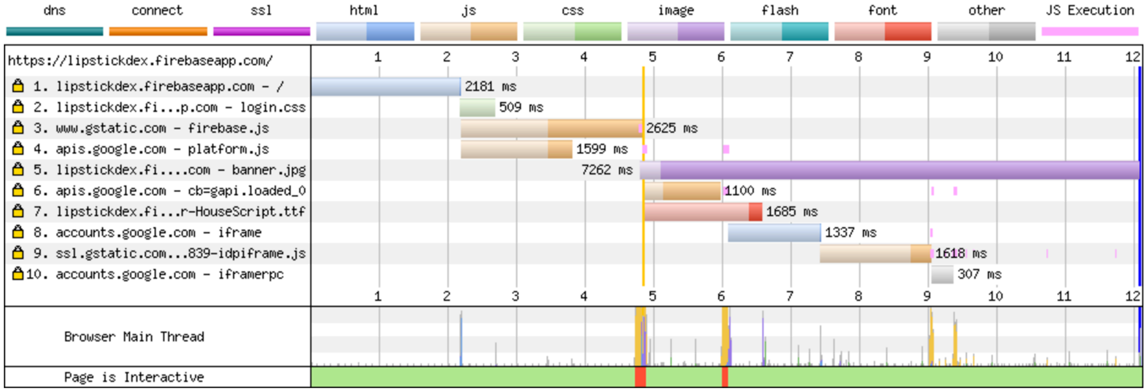
Update Lipstick data by editing fields and then pressing this button

Speed comparisons

Index.html –

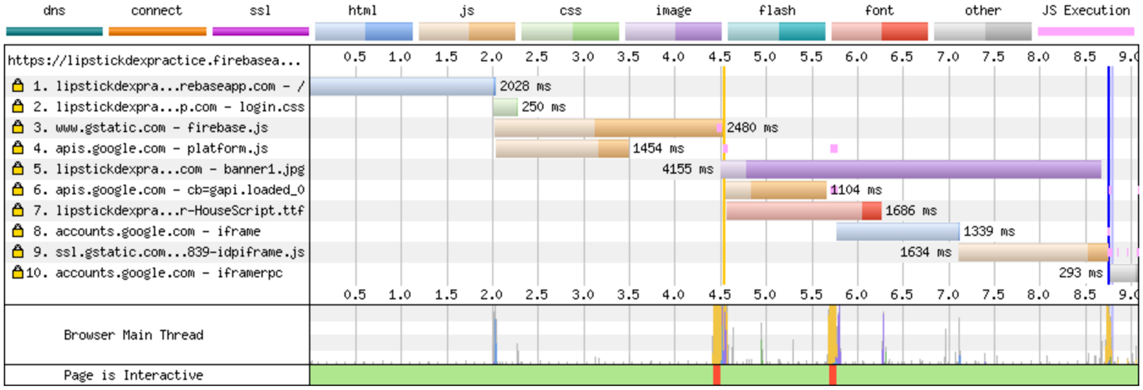
This is the chart when we did a speed test on our app with throttled 3G (slow) on our index page:

First View
(10.207s)



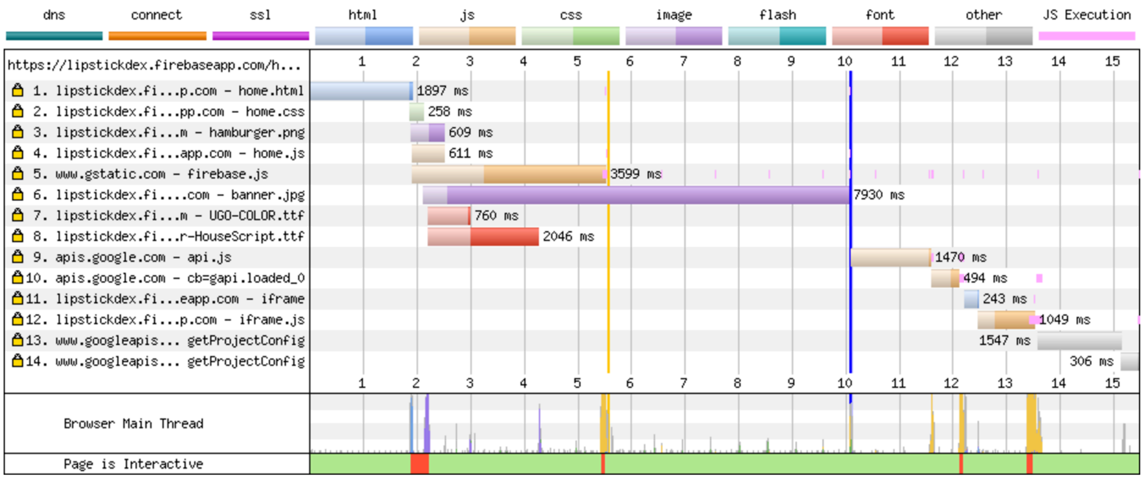
We noticed our banner image was what was giving us the largest time lag so we tried using a much smaller image, this sped up our page by quite a bit:

First View
(8.748s)

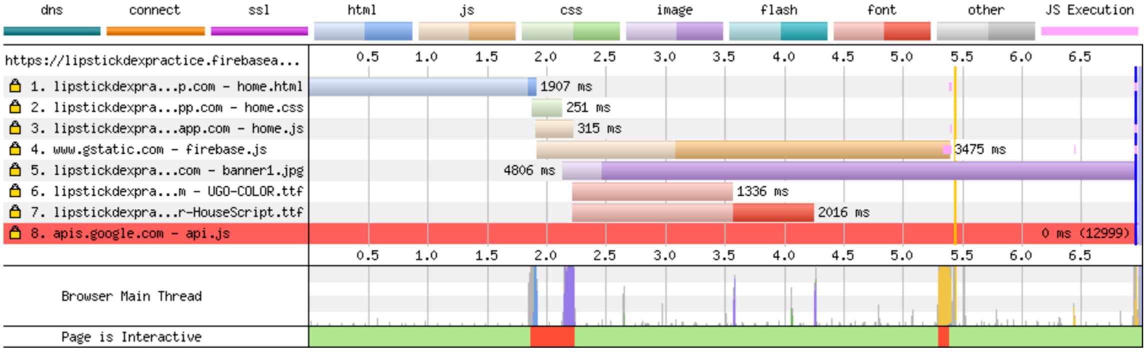


The next page to look at is our home.html page:
Our original page was relatively slow

First View
(10.065s)



Changing the banner image helped this page speed up quite a bit too:
First View
(6.953s)



Known Issues, Bugs, & Limitations:

Here we will discuss limitations, caveats, and to do's.

A minor bug is that we are doing a manual redirect once we do log in – so the page after logging in redirects back to the log in page, and then manually goes to the home page.

Our biggest task that we were not able to accomplish with the time we had was making our app work under the 1 second deadline. We did our best to make small optimizations; however, we definitely have a lot more work to do in order to make our app one that can actually be used by users.

We started to look into encoding our images into strings during storage, and loading all our html, css, and javascript files at the same time in order to reduce page load time – however we did not have the time to implement it completely.

Another option we explored but did not have time to fully implement was image spriting using services like imgix.

Given more time we also would have liked to spend more time covering various factors that could harm our app such as network issues, dependency issues caused by heavily relying on Firebase for functionality, etc.
Another task is that we would have liked to make the lipstick forms a little more user friendly and error proof. As our app stands now, there are small corner cases where it is possible to break the app if certain fields are not filled out. We would like to fix these issues, given more time.

Thank you

- LipstickDex Team