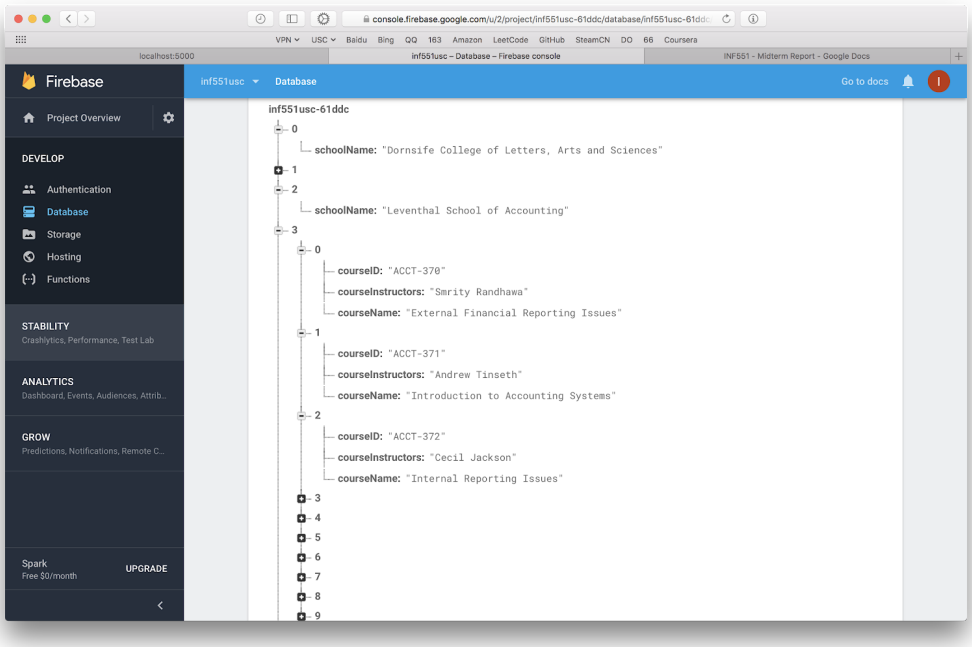
INF-551 Midterm Report

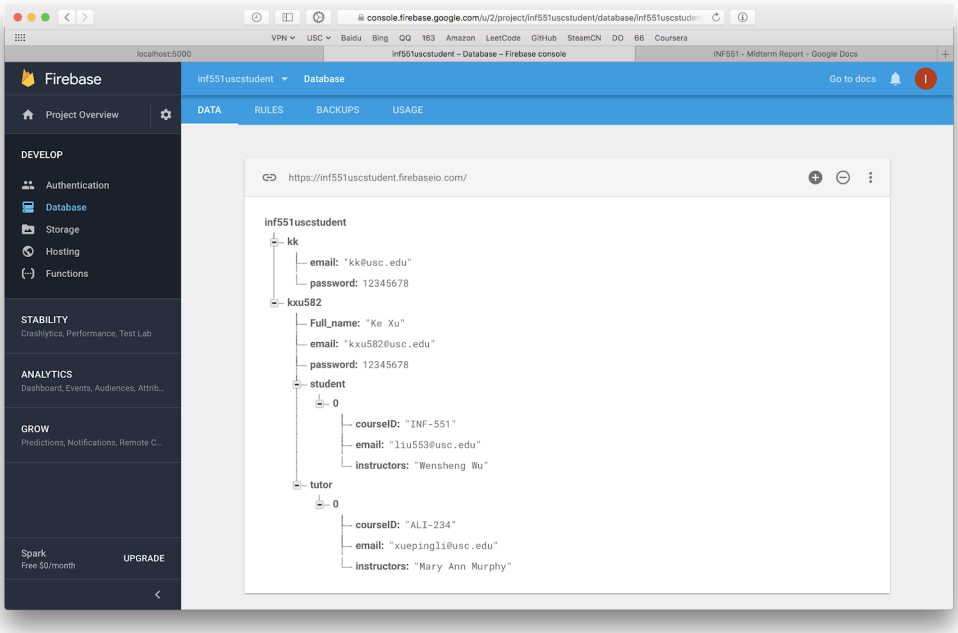
Project Name --- USC Tutor Finder

Ke Xu (kxu582@usc.edu) Xueping Li(xuepingl@usc.edu)

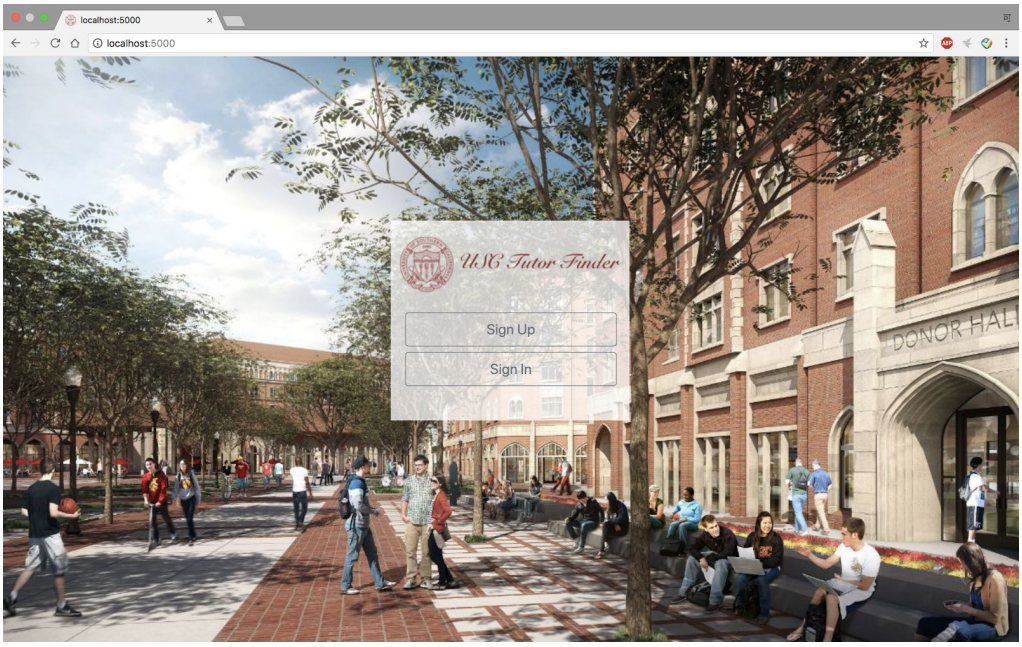
Currently, we have real life data stored on the FireBase and five web pages with a working backend. We use bootstrap with html, css and javascript for the user interface and Flask for backend with Python, jQuery and a bit of javascript implementation. Below is the more detailed implementation of our project.



First of all, we want to use USC classes as our real life data, so we have created a simple web crawler in Python to get course information from https://classes.usc.edu/term-20181/. we get urls step by step, then retrieve information from new urls we've got, and in the end, store them in the format of Json. In the data retrieving part, we haven't used any framework or existing projects as our crawler's base because of the fewer data amounts for this project. For the data structure shown above, we have stored school name, course name, course ID, course instructors and eliminating other useless information on the Google Firebase.

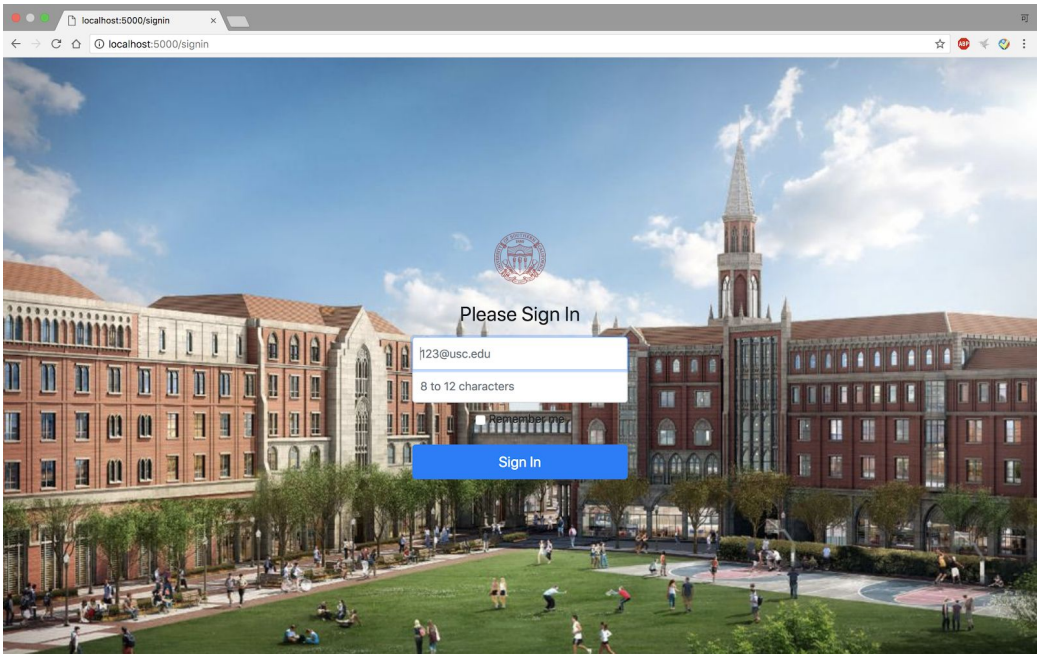


We have devised user information structure like the screenshot shown above. We choose to store user email, password, and his or her full name in the Firebase. This tree-like table also contains the basic information of his or her students and tutor courses.

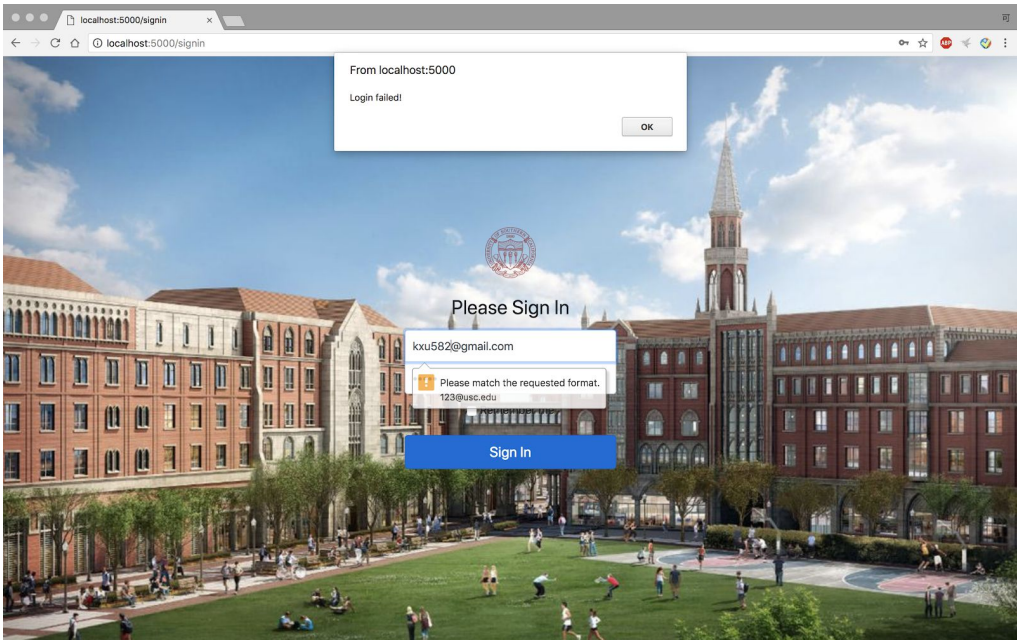


After we have our data ready in the Firebase, we move onto coding the website. This is the current index.html, our home page. On this page, we have a “Sign Up” button and “Sign In” button that will lead to signin.html and signout.html respectively with a single-on-button click.

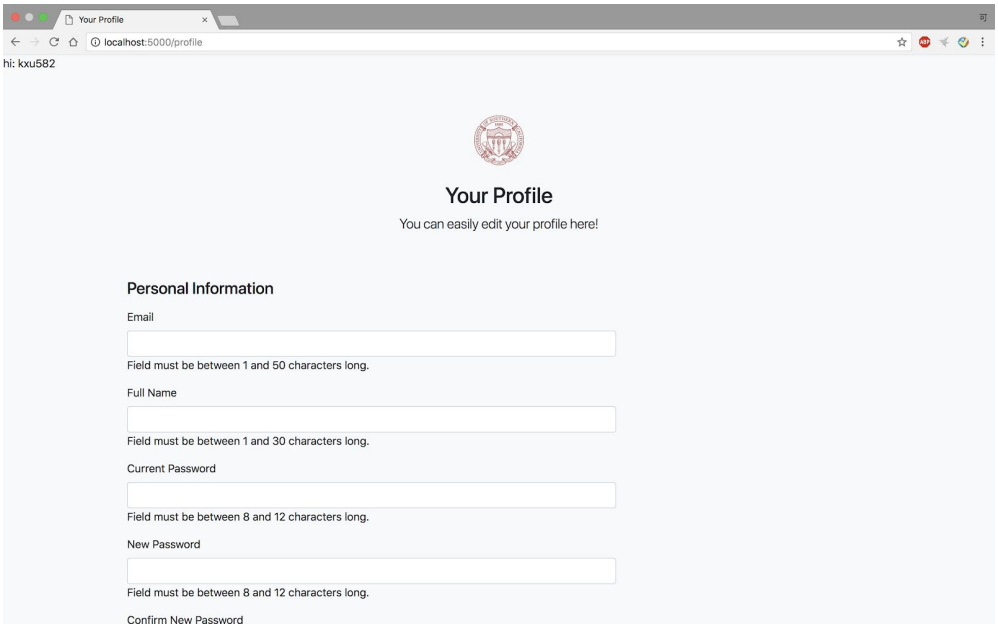
In the “signup” page, we have the same layout as signin.html, except that , in the backend, instead of getting the information from FireBase, we are going to post user typed information onto the FireBase. During the POST action, we will have a check process for the typed information to see if they go against our data format rules. After user successfully sign up, the page will be redirected to sign in page.



After clicking ‘sign in’ in the home page or being directed from the signup page, we will be on the sign in page, which looks like the screenshot above. After user types the usc email, password and hit ‘sign in’, in the backend, typed email and password will be matched with corresponding information on FireBase.



If the match fails, the page will respond with an alert, otherwise, the user will be directed to user profile/dashboard page and his/her signed in status will be maintained until ‘signout’ button is hit.



Once a user gets signed in, the user will be destined to his or her profile page, which is the page that shows all the user’s basic information that has stored in our Firebase. Our design is, the user can edit each sign input box besides the sign in email by clicking edit button next to the text box, which we haven’t put on yet.

From user profile page, the user can be redirected to search.html. For the search part, we can maintain the login status of a user currently. Pages are not fully done yet. We're still working on it. So in the coming days, we will get this page done.

Here is the challenges we face. First of all, we met the problem of connecting with our database in the backend. We started establishing the connection between website and database using Javascript, however, it failed. We found that javascript did not always execute in order. When we sent request to Firebase to retrieve data, Javascript took a longer time to finish the request and concurrently, it would implement latter lines of code while waiting for that to finish, which messed up our logic. This was the reason why we switched to Flask. Later, we had troubles with the communication between different pages, including the maintaining of login status. Also, we encountered problems of UI implementation and interaction.