W E B S C A L E

Synthesizer

Fall 2017

Steven Borst, Jared Holzman, Rachit Nigam, Sean Barker, Max Berezin

https://github.com/326-WEBSCALE/webscale-synthesizer

http://ec2-54-86-24-49.compute-1.amazonaws.com/

Login details: jdoe, password12345

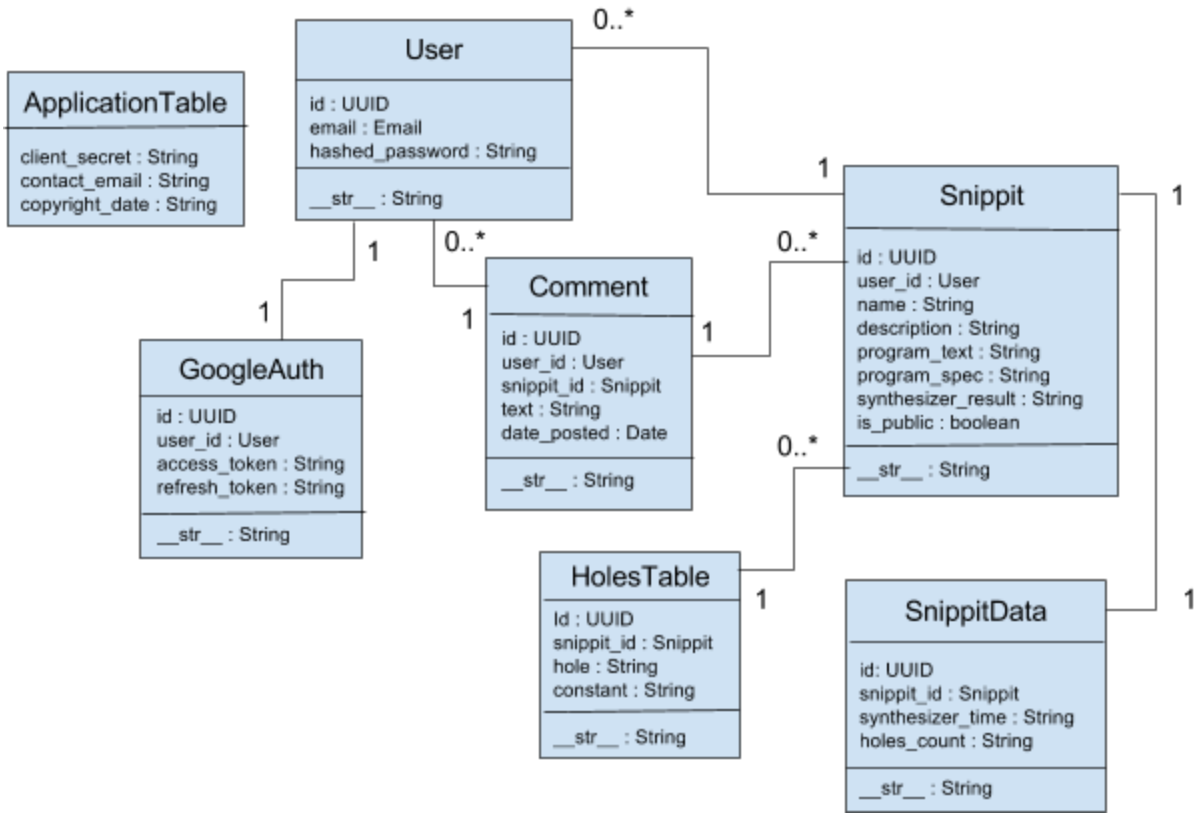
Overview

Synthesizer is a project to bring program synthesis to the masses via an intuitive web UI that allows users to edit programs in the browser, synthesize missing constants in their programs, and collaborate with others. The web application provides a slick interface for synthesizing programs so that is a little less daunting to users, and allows them to easily save and load from Google Drive for speedy development. The web application is innovative because while synthesizing is a hot topic for academic computer science, its applications remain unseen by the everyday programmer. The W E B S C A L E team hopes to open up the power of program synthesis to the masses.

User Interface

The Synthesizer website contains a main view, an Activity Feed view, a profile view, a discussion view, an About Us view, and a FAQ view. The main view is where the majority of user interaction takes place. Here they can write, load, and save their programs, as well as run them through the synthesizer to fill in their missing constants. Users can also search for other user's programs that are public and discuss them in a forum like view. The About Us and FAQ pages serve as a point of reference for new users on what Synthesizer does and how to use it. However, as with most things, examples offer the best look at how to use the website. The activity view allows users to see a feed of public programs that were recently saved so that they can get a sense of and discuss how other users are using Synthesizer. The profile view is where logged in users can find a complete list of their saved programs, toggle each program's public status, delete programs, and change their login and personal information. The data model powering these views is fairly straight forward and best explained by our diagram.

Data Model



URL Routes/Mappings

URL	Description
/	Index, start a blank program
/about	About page
/faq	FAQ page
/feed	Activity Feed
/program/{program id}	View an existing program with the given ID
/discussion/{program id}	View comments/discuss a program
/profile	View your own (currently logged in user) profile
/profile/{user id}	View the profile of an individual user
/profile/edit	Edit your own profile, if you are logged in

Authentication/Authorization

When first visiting the site or after logging out, the user is presented with a “Log in” button in the top-right corner of the page. When they click it they are brought to the login page and after logging in successfully, are brought to the homepage but with the “Log in” button replaced with their name and a drop down menu. The drop down menu includes link to their profile page and the link to logout. If a user elects to log out, they are brought to the logout page. The profile page displays the user’s name and all of their saved programs. If the user is viewing their *own* profile page, they are given a link to view the profile edit page. The profile edit page can be used by users to edit their profile info and the main page supports saving programs for logged in users. Several test users exist, all with the same password, “password12345”. User “jdoe” is a good test user to look at.

An example of a permission that is dependent on the user that is logged in is whether or not a program is public. If it is public, any user can view it. If it is not public, however, only the user who owns the program can view it. In other words, every other user does not have permission to view it.

Team Choice

For our team choice, we brought our of our application’s functionality together into a functional system. First, the actual synthesizer was wired up; that is, we connected our front-end interface for synthesizing programs to our back-end service written in OCaml, which actually computes the program synthesis. Second, we integrated Google Drive functionality with our program to allow users to save and load programs to and from Google Drive. Finally, we now allow users to customize their profile with a profile picture, which will display next to their name in discussions.

Conclusion

First and foremost, working in a group setting is hard. Getting everyone coordinated is not as trivial of a task as it seems. Our group naturally tended towards having Jared as a leader so-to-speak, who delegated tasks for each project to make sure that everyone was doing their parts on time, as well as distributing the workload fairly. We found that different members of the group had different skillsets, and we tried to take advantage of this when deciding who completed what tasks.

Through the design and implementation process, we learned that it’s often very hard to predict everything that your program will do from the start. We had a vague idea of how the application would be laid out, but as we began to program technological complications ended up dictating the structure more than our imaginations. In the end, it comes down to reconciling what we want to accomplish with what is feasible. This often results in changing what we originally imagined, but not necessarily settling for something sub-optimal.

The only prior knowledge that might have helped would be more exposure to the Django framework. Much of the actual work in this project came down to understanding Django’s design choices and opinions, and formatting our ideas into the framework’s idioms.

Individual Write Ups

Individual Write Up: **Steven Borst**

My contributions to the final submission included an attempt to the search function, updating the final presentation slides, creating the single slide, and helping with the team write-up. The search function was difficult to implement in my opinion, I couldn't find a lot of documentation to support me in this effort, whereas with most of the other parts of the assignment we had the Django tutorials from Mozilla to fall back on. The final presentation slides were missing a few points when we presented them, I have since updated them to include all the project specifications. The single slide consists of the main view of our website with the team name in large print below it. My contribution this time is about 15%

Individual Write Up: **Jared Holzman**

My role the project has mostly been team lead. I helped split up and assign tasks to team members. This project I hooked up our OCaml program synthesis tool to Django, allowing the "Synthesize" button on the main page to work. I also added the GDrive integration which allows users to sign into their Google accounts and then export their saved programs to their Drive's. I also deployed the project to AWS. In addition to that, I also did a good about of general code clean up and fixed a good number of bugs. My contribution this time was about 35%.

Individual Write Up: **Rachit Nigam**

My role in the project has been building the synthesizer, implementing program saving and building the slideshow for the presentation. I brought the OCaml program synthesis tool up to date and implemented the interface that Jared used to wire into django. I also fixed a previous bug in program saving that needed me to export the programs using hidden fields on the form. Finally, I built the slideshow and gave the technical talk on how the internals of the synthesis tool works. Overall, my contribution to the project was about 25%.

Individual Write Up: **Maxim Berezin**

My role in the project has been to visualize the data model, and to implement functionality such as the authenticated comments section, some URL mappings, and the 404 page. I contributed 10-15% of the work in this project.

Individual Write Up: **Sean Barker**

My role in this project has been focused on the write-up and prep for submission. I wrote much of the written portions of the final submission write-up. My contribution for this project was 10-15%.