W E B S C A L E Team Write Up

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

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.

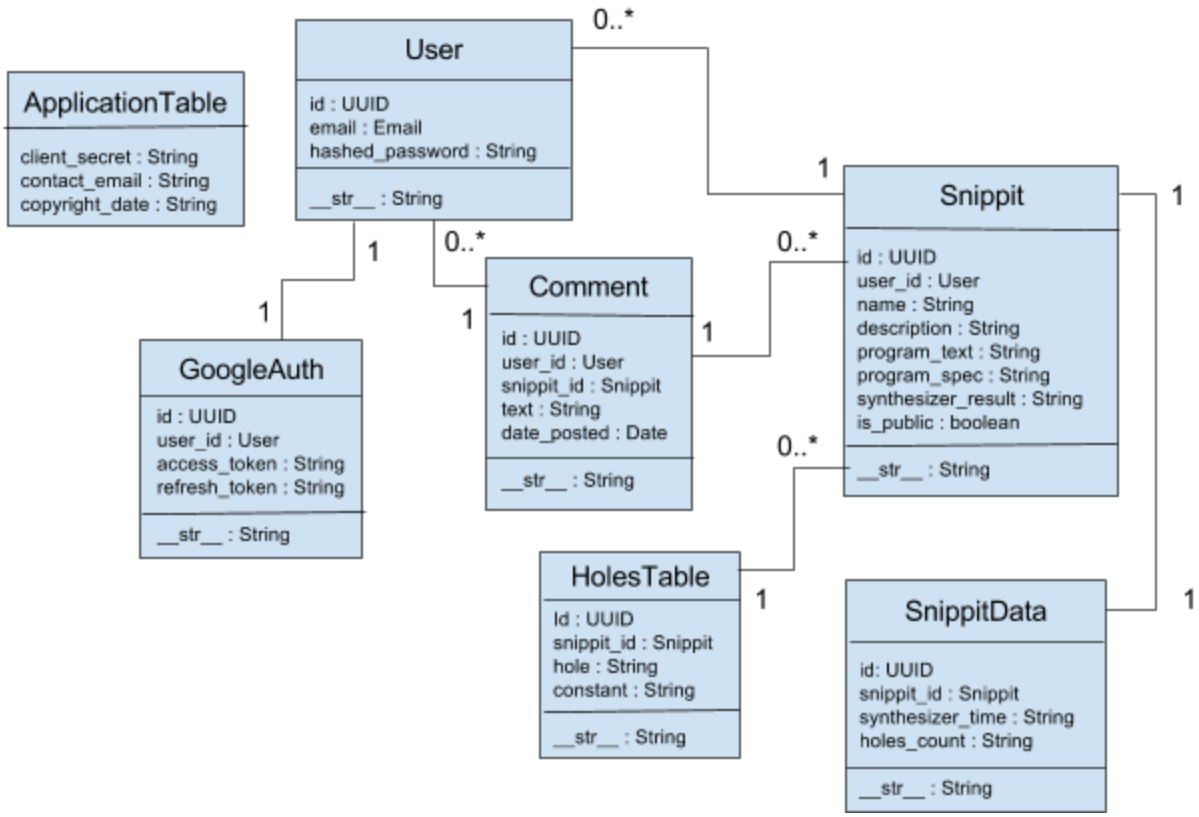
The Synthesizer website contains a main view, an Activity Feed view, a profile view, a discussion view, an About Us view, and a FAQ view. The main view is where the majority of user interaction takes place. Here they can write, load, and save their programs, as well as run them through the synthesizer 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 on the next page.

The url mappings are fairly straightforward. The About, FAQ, and Activity Feed pages are all reachable by '/about', '/faq', and '/feed' respectively. The main page is reachable by the root link. For existing programs, the link is '/program/{program_id}'. It is identical to the main page except that the data from the specified program is filled in and a "Discuss" button is present in the sidebar. If the program is saved a public, users can give the '/program/{program_id}' to others, allowing for easy sharing. The comments page is reachable via '/discussion/{program_id}' since every program has it own associated forum section. Finally, for the profile pages, '/profile' will direct to the profile page of the currently logged in user, '/profile/{user_id}' directs to the profile page of a specific user, and '/profile/edit' directs to the profile editing page of the currently logged in user.

Some troubles we faced early on were on what UI design to go with, what our url mappings / data should look like, and coordinating and communicating on who was working on what. We were able to eventually agree upon a UI design after a few iterations and tweaks. For what the url mappings / data should look like, we tried a few different formats and added / changed some tables since project 1. The solution we ended up going with was a combination of all the different ideas the team presented. For communication, we found that by creating a Slack group, we were able to work much more effectively and added a bot that would post whenever anyone pushed to the repo, letting us be much more aware of the current state of the project.

There are certainly a plethora of features we'd like to implement for the final project. First, a RESTful API used to better facilitate saving and searching data will need to be implemented. Next, we need to use sessions so that users can see specific programs saved to their profile. Another thing to make live activity feed truly "live" is to set up and endpoint and poll intermittently so that data updates don't requiring a page reload. Finally, we'd like to include GDrive integration to allow users to import and export programs to/from their Drive account.

Data model diagram



Individual Write Ups

Individual Write Up: **Steven Borst**

I started the django skeleton, that is, part 0 of the assignment specification sheet. Building on that, I added the models using the attributes from our data model diagram; making educated guesses along the way which django fields would best suit those attributes. I added the admin site shortly after and added a few mock data entries to test the fields.

After learning from homework 7, I integrated our html from project 1 into the templates and static content of our submission. This required the changing of navigational hrefs and figuring out how to add urls/views properly.

I'll put my percentage at the amount of points out of 100 I think I earned us:

Part 1 Data Model (35 Points) **(16/35)**

- 10 points for a completed data model diagram. **//(0/10)**
- 10 points for data model entities implemented in Python/Django. **//(10/10)**
- 5 points for a functioning admin site. **//(5/5)**
- 10 points for the addition of mock data. **//(1/10) I added just a few mock data to test the functionality of the fields**

Part 2 Templates and Views **(11.3/35 Points)**

- 10 points for the implementation of template files. **//(2/10) I added them initially, then they were modified to use a base-and-extension format**

- 15 points for the implementation of view functions/classes. **//(2/15) added definitions for each request, then they were added to**
- 10 points for the implementation of URL mappings. **//(7.3/10) 11 urlpatterns from the urls.py files, I added 8 of them**

Part 3 Team Write Up **(0/15 Points)**
- 15 points for the team writeup. **//I didn't write any of the team writeup**

Part 4 Individual Write Up **(3/15 Points)**
- 15 points for the individual writeup. **//15 points divided by 5 group members**

Contribution: **30.3%**

Individual Write Up: **Jared Holzman**
For this project, I acted as team coordinator and general repair man. I helped to assign different jobs to people and point them in the right direction for how to start on their portions of the project. I bounced around from the frontend and backend where ever I was need to clean up the occasional bug, or help add some necessary features. I helped with the initial templatization of the UI, started the work on changing the urls to accept ID parameters and use them fetch necessary data from the database, and wrote the first draft of the team writeup (as well as edited the final one. Overall, I feel that I contributed roughly 30% to the project.

Individual Write Up: **Rachit Nigam**

For this project, I generated the data for the databases that corresponds to the data generated by our backend. I was responsible for discussing and coming up with a RESTful API for our OCaml based backend with Max (which I did). Additionally, I wired up the views information into the profile page editing and the defaults for all of the individual pages. Since we had initially thought that we needed to complete our backend, I also implemented the ocaml code required to talk to the RESTful service, but since we later realized that this was out of scope, I did not count that as a contribution to the project. Overall, I feel that I contributed roughly 15% to the project.

Individual Write Up: **Maxim Berezin**

For this project, I created the data model diagram for part 1. I was initially responsible for entering the appropriate URL mapping, which required the specification of the data model itself. Discussion of the data model design and the Django ORM implementation was a collaborative effort, and the diagram was used to visualize the pieces that were needed for the Django implementation and for specifying the URL mappings. I also created the 404 error HTML/CSS and mapped it for Django to use in the case of an error (in production). I also fixed various typos. I feel as if I have contributed 15-20% to this particular project.

Individual Write Up: **Sean Barker**

For this project, I worked on converting the existing static html pages into dynamically rendered pages based on the data in the database. Specifically, I worked on making the profile and activity feed pages. Quite a few modifications to the profile page were necessary to support our final data model, including the addition of a brand new page to allow users to edit their profile. Currently the /profile/ endpoint simply loads an arbitrary user, but in the future it will go to the profile of the current session's user. In addition, I helped generally clean up the code base to use more idiomatic django/python. I feel that I have contributed 20% to this project.