Capstone Matching Project Fall 2023 Deployment Instructions

Initial Cloning

1. `git clone git@github.com:Capstone-Matching/capstone-matching.git`

   OR

   `git clone https://github.com/paulinewade/capstone-matching.git`

   NOTE: You need to copy the HTTPS or SSH from the respective branch (e.g., fixes-for-beta, main, etc.)

2. `cd capstone-matching`

Developers only – Local installation - Upon Initial Cloning

Assuming you have Ruby on Rails already installed on your local machine (we use Ruby version 3.2.2 and Rails version 7.0.7.2), to get all of the necessary Gems for this application you can run:

```
$ bundle install
```

To initially set up the local database you need to run this command

```
$ rails db:migrate db:seed
```

If you ever want to clear/reset the database to add new seeds/clear testing data, you can run

```
$ rails db:reset
```

and it will automatically delete, create, run (strictly new) migrations, and seed the database.

If you make changes to an existing migration then you will have to run:

```
$ rails db:drop db:create db:migrate db:seed
```

Google OAuth Setup:

Go to this web page: https://console.cloud.google.com/ and sign in with your Google account

Once you are signed in, create a new project using whatever name you choose.

Click on Admin Dashboard

Click on New Project on the top right

Once you have created the project, navigate to the 'APIs and Services' Page for the project

You are going to have to configure the OAuth consent screen first

Change the project to the one you just created, through the top left drop down, next to

GoogleCloud icon

Choose external for the user type, then fill in your app name, support email (your email), and developer contact information (your email)

You do not need to set up any scopes or test users so just click save and continue on those pages, once you have done that you can click Publish App (or stay in testing and add your email to the Test Users list if you choose)

Then navigate to the 'Credentials Page' and click on 'Create Credentials' and click on create 'OAuth Client ID'

Choose 'Web Application' for the application type

Then click create – take note of the Client ID and Client Secret shown upon successful creation, you will use these later (can download JSON or manually copy to a safe place)

FOR DEVELOPERS ONLY, IF RUNNING LOCALLY, SKIP TO HEROKU SETUP / DEPLOYMENT INSTRUCTIONS

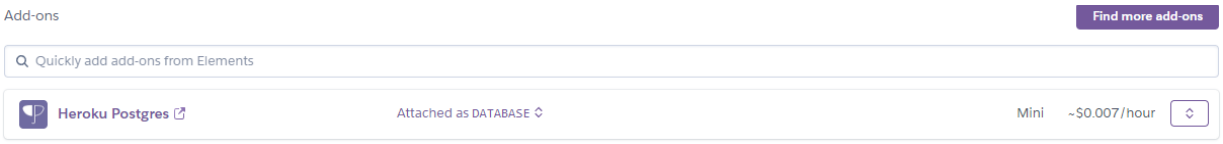
If you are running locally or testing you will set the variables in the .env file in the capstone-matching root directory:

```
GOOGLE_OAUTH_CLIENT_ID=[your client id]
GOOGLE_OAUTH_CLIENT_SECRET=[your client secret]
```

Heroku Setup/Deployment Instructions:

On Heroku create a new app with the title you want for the application.

Once the app is created navigate to resources and add the add-on Heroku Postgres (for now you can use the mini plan, but you may need to upgrade to other plans as more students submit their information)



Navigate to the directory for capstone-matching and on the command line enter the following command:

```
$ heroku login
```

Login to your Heroku account using the prompt (it should have you enter any key and open up your browser for you, but if not ctrl+click on the link that it provides and login)

To “connect” your existing repository (or the repository that you cloned/forked) to your project you can do

```
$  git remote add heroku https://git.heroku.com/[your project name].git
```

Example: `git remote add heroku https://git.heroku.com/tamu-capstone-matching.git`

Each time you want to make changes to the instance on Heroku (or deploy your initial code) you can run the commands - if you have already committed your changes you can just use the last command (the git push command)

```
$ git add *
$ git commit -m “your commit message”
$ git push heroku branch_name:main
```

Example: `git push heroku fixes-for-beta:main`

If you are pushing the main branch you do not need to specify a branch name in the last command and can just run this command: No need if pushing from another branch

```
$ git push heroku main
```

Run the following commands to set the environment variables (using the Client ID and Client Secret you made earlier) on your Heroku instance since it cannot use the .env file

```
$ heroku config:set GOOGLE_OAUTH_CLIENT_ID=[your client id]
$ heroku config:set GOOGLE_OAUTH_CLIENT_SECRET=[your client secret]
```

Once you app is deployed – take note and copy the URL (it will be in terminal) and continue OAuth Setup Pt 2 for the app to work properly

Google OAuth Setup Pt 2:

Now that your app is deployed, navigate back to the APIs and Services page on Google cloud console for your project that you set up earlier.

Navigate to the Credentials page and click edit on the credentials you created earlier (the pencil icon)

Under 'Authorized Redirect URIs' click 'ADD URI' and you are going to put:

- [your heroku app url including https://]/auth/google_oauth2/callback

For developers only: If you want to test locally you can add these 2 as well:

- http://127.0.0.1:3000/auth/google_oauth2/callback
- http://localhost:3000/auth/google_oauth2/callback

Then click 'Save' and if you have followed all of the steps properly you should be able to login with OAuth **through a tamu.edu account** (the application does not give non-tamu accounts access)

Database Setup on Deployed App:

This part is simple because the Postgres add-on in Heroku handles most of this for you, to initially set up the database you can run:

```
$ heroku rake db:migrate
```

And if you want to add the seed data we have provided do this:

```
$ heroku rake db:seed
```

And if you ever want to clear the data in the database (or redo the schema in the migration we have), run this command: then go back to db:migrate and db:seed

```
$ heroku pg:reset --confirm [your project name]
```

Your application should now be fully deployed and work as intended!

For Developers only: Testing and Coverage Instructions:

To test the code you can run either (or both) of these commands

```
$ bundle exec rspec
$ bundle exec cucumber
```

Running either of these commands should provide you with a coverage percentage at the bottom, however you can view a visual of the report if you navigate to the 'coverage' folder and open the index.html file in your browser and it will show you a coverage report of the most recently ran tests. **To get full coverage info, you must run both of these commands.**

```
Finished in 0.41717 seconds (files took 0.99105 seconds to load)
90 examples, 0 failures

Coverage report generated for Cucumber Features, RSpec to /home/hunterzacha/capstone-matching/coverage. 721 / 772 LOC (93.39%) covered.
```

You can see coverage information at the bottom of the tests, after you run both sets of tests you can see the **total coverage information** for all of the tests

All Files (95.39% covered at 3.44 hits/line)

30 files in total.
716 relevant lines, 683 lines covered and 33 lines missed. (95.39%)

Search:

File	% covered	Lines	Relevant Lines	Lines covered	Lines missed	Avg. Hits / Line
app/controllers/professor_preferences_controller.rb	68.42 %	89	19	13	6	2.26
app/controllers/ProfregistrationController.rb	89.29 %	49	28	25	3	2.82
app/controllers/studentform_Controller.rb	91.49 %	339	188	172	16	3.52
app/controllers/adminlanding_controller.rb	91.67 %	23	12	11	1	3.50
app/controllers/manageprof_controller.rb	96.61 %	106	59	57	2	4.32
app/controllers/results_controller.rb	97.20 %	166	107	104	3	1.96
app/controllers/sponsor_preferences_controller.rb	97.37 %	61	38	37	1	2.42
app/controllers/sponsor_restrictions_controller.rb	97.44 %	65	39	38	1	2.41
app/controllers/changeweights_controller.rb	100.00 %	43	27	27	0	4.15
app/controllers/managestudents_controller.rb	100.00 %	70	44	44	0	3.86
app/controllers/projects_controller.rb	100.00 %	88	55	55	0	9.07

The index.html file will look like this and you will be able to click on the files to see which lines are covered/not covered.

Database Management Instructions:

To remove all data and tables from the database run the following command, it will ask you to confirm by entering in your app's name again:

```
$ heroku pg:reset --app [your app name]
```

App name is what is in Heroku dashboard

To import an existing sql file that you have downloaded from our app using the 'Export Database' button (this may take a very long time depending on how much data is in the export you have given):

```
$ heroku pg:psql --app [your app name] < [export/sql file path]
```

Incorporate code changes:

Go to directory where the code is

Determine current branch

```
git status
```

Go to the desired branch

```
git checkout <branchname>
```

Pull changes (force option) from remote repo (make sure my remote is set correctly and pointing to the right branch)

```
git pull -f
```

Push to Heroku

```
git push heroku <branch_name>:main
```

branch name could be 'fixes-for-beta'

Team Contact Information:

If you have any questions that are not answered by this documentation, feel free to contact one (or all) of our team members!

- Hunter Zacha - hunterzacha@tamu.edu
- Aakashdeep Sil - asil@tamu.edu
- Yash Patil - yapatil@tamu.edu
- Shripad Kulkarni - shripad@tamu.edu
- Zachary Laguna - zacharylaguna@tamu.edu; lagunazachary@gmail.com
- Sathvik Kote Revanasiddappa - sathvikkote@tamu.edu