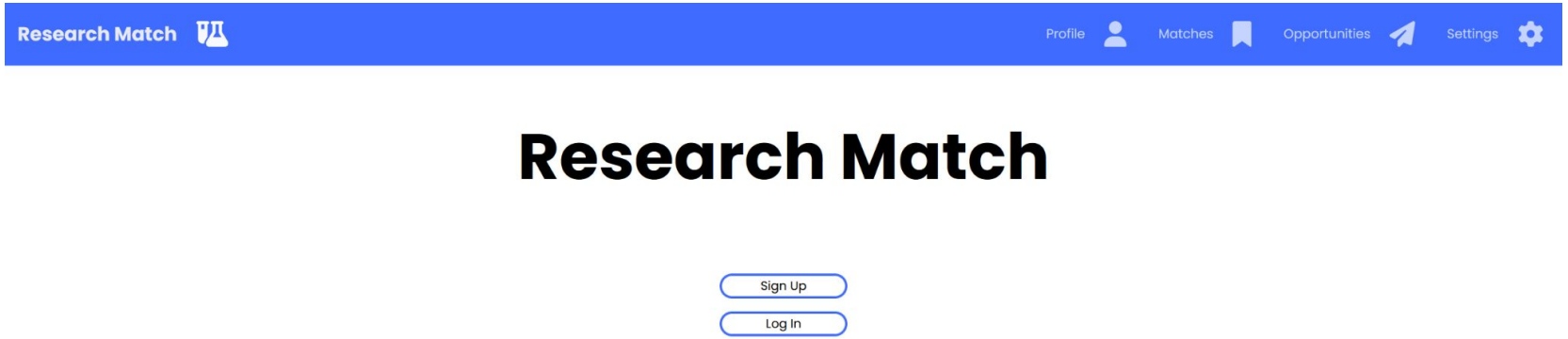
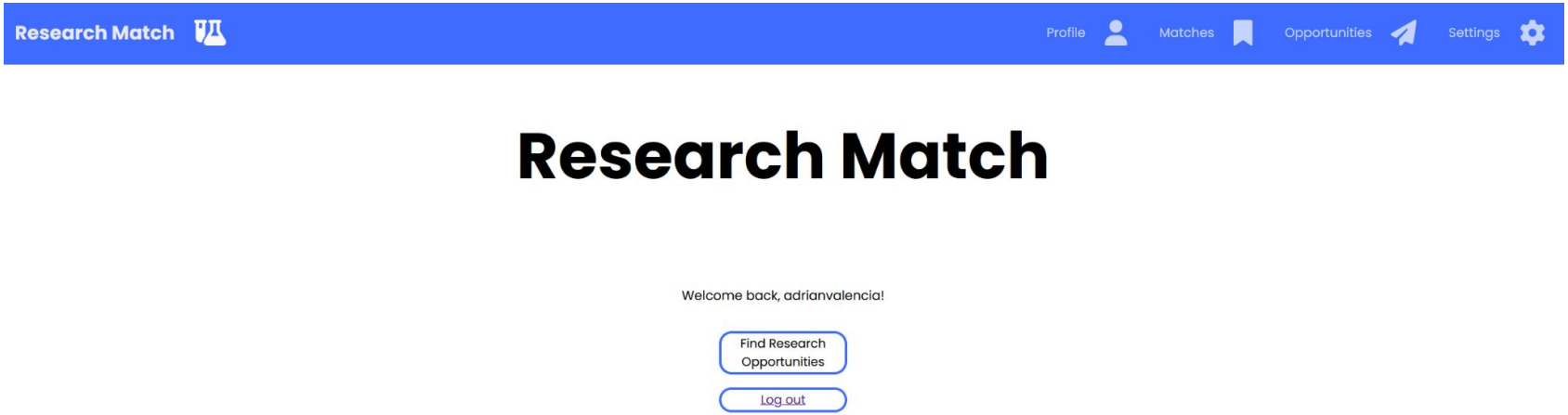
Research Match
Sprint 5

Deadline Tech

Home page

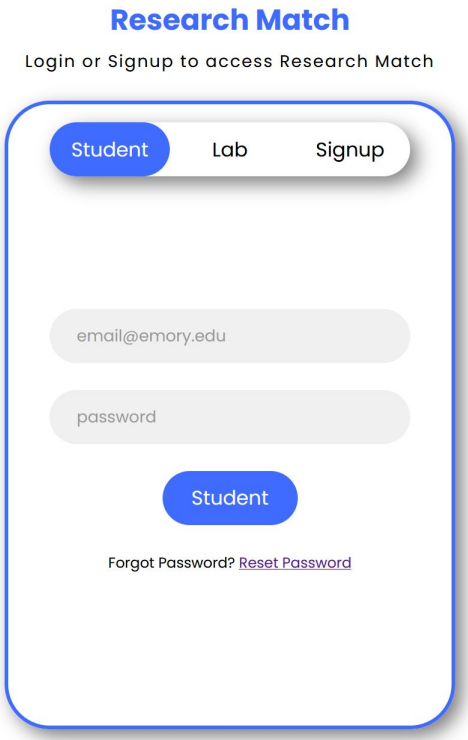

Research Match
Profile
Matches
Opportunities
Settings
Research Match
Sign Up
Log In

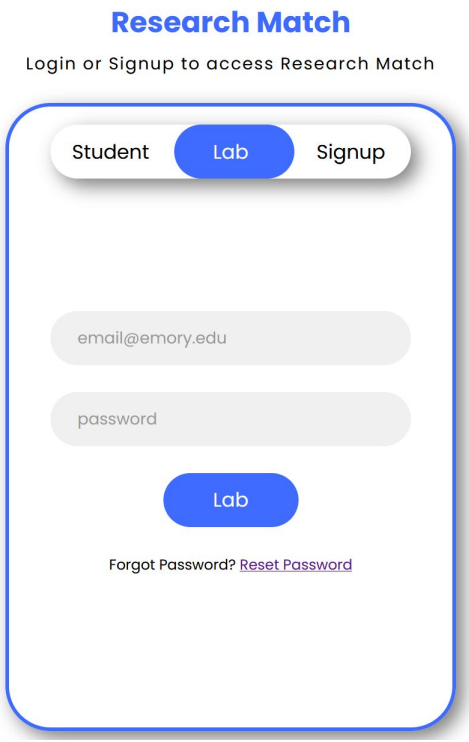
Home page

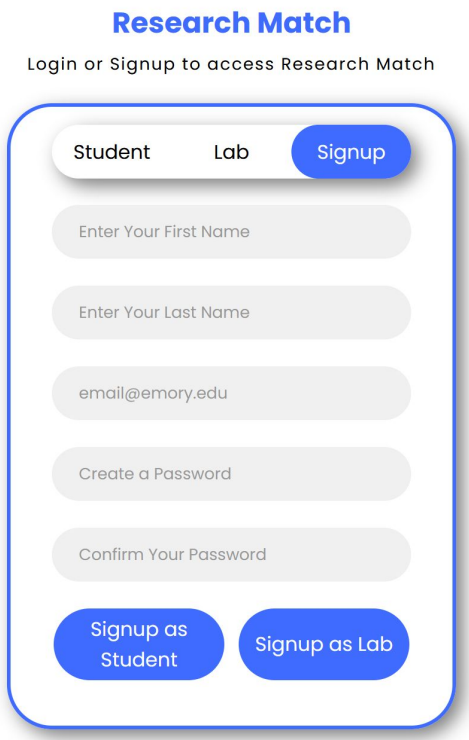



```
<a href="{% url 'studenthomepage' %}" class="nav-link">
{% if user.is_authenticated %} {%else%}
<meta http-equiv="refresh" content="0; /signup" />
{%endif%}
```

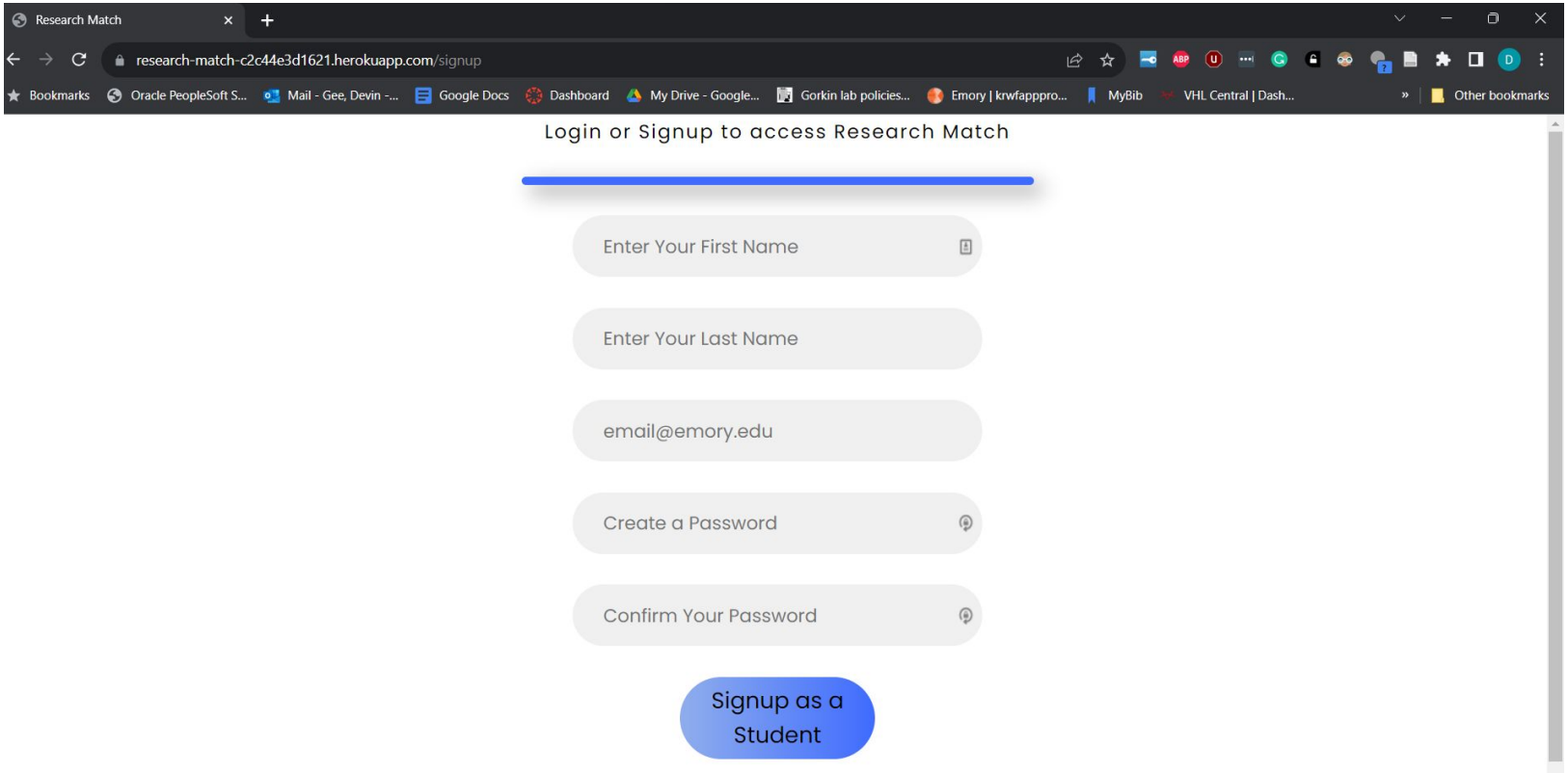
Login page

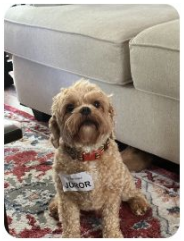

Research Match
Login or Signup to access Research Match
Student
Lab
Signup
email@emory.edu
password
Student
Forgot Password? Reset Password


Research Match
Login or Signup to access Research Match
Student
Lab
Signup
email@emory.edu
password
Lab
Forgot Password? Reset Password


Research Match
Login or Signup to access Research Match
Student
Lab
Signup
Enter Your First Name
Enter Your Last Name
email@emory.edu
Create a Password
Confirm Your Password
Signup as Student
Signup as Lab

Web App and Database Hosting





Skills:

```
# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/4.2/howto/static-files/

STATIC_ROOT = os.path.join(BASE_DIR, 'staticfiles')
STATICFILES_STORAGE = 'whitenoise.storage.CompressedManifestStaticFilesStorage'

STATIC_URL = '/static/'

STATICFILES_DIRS = (
    os.path.join(BASE_DIR, 'static'),
)
#Media files
MEDIA_URL = "/media/"
MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
```

Getting User Input

```python
class User(models.Model):
    username = models.CharField(max_length=70)
    email = models.CharField(max_length=70)
    firstname = models.CharField(max_length=70)
    lastname = models.CharField(max_length=70)
    BACKGROUND = [
        ("S","Student"),
        ("M","Mentor")
    ]
    background = models.CharField(max_length=1, choices=BACKGROUND)
    SUBJECT = [
        ("BIOL", "Biology"),
        ("CHEM", "Chemistry"),
        ("PHYS", "Physics"),
        ("MATH", "Mathematics"),
        ("COMP", "Computer Science"),
        ("PSYC", "Psychology"),
        ("HIST", "History"),
        ("OTHE", "Other")
    ]
    subject =  models.CharField(max_length=4,choices =SUBJECT)



class Student(User):
    gpa = models.IntegerField(default=0)
    documents = models.IntegerField(default=0)
    skill = models.CharField(max_length=255)
    course=models.CharField(max_length=255)
    biography=models.CharField(max_length=500)
    # def __str__(self):
    #     return self.objects

class Mentor(User):
    biography = models.TextField()
    # def __str__(self):
    #     return self.objects
```

Button to edit profile:

```
<div class="editButtons" style="margin-left: 0">
  <form action="{% url 'skill'%}" method="post" enctype="multipart/form-data">
    {% csrf_token %}
    {{form}}
    <input type="submit" id="addSkill" value="Edit My Profile">
```

HTML page for the form:

```
<div class="container">
    <form method="POST">
      <fieldset>
        <legend>Add Skills</legend>
        {% csrf_token %}
        {{ form.as_p }}
        <button type="submit" class="btn btn-primary">Submit</button>
      </fieldset>
    </form>
</div>
```

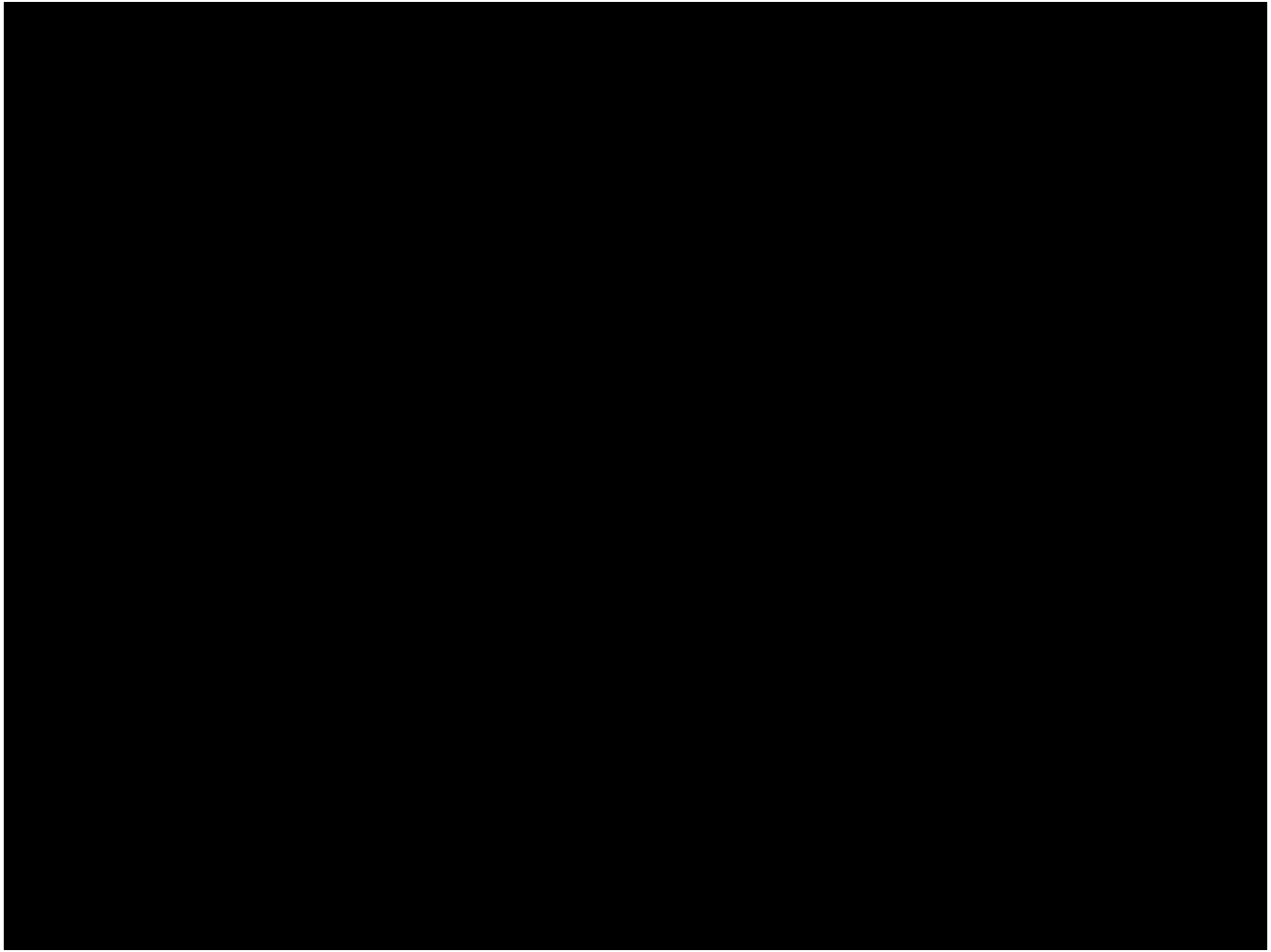
Code for the Form

```python
from django import forms
from .models import Student


class SkillForm(forms.ModelForm):
    # skill = forms.TextInput()
    class Meta:
    # your_skill = forms.CharField(label="Skill_input", max_length=100)
        model = Student
        fields = ['skill','course','biography']

# class SkillForm(forms.Form):
#     skill_input = forms.CharField(max_length=100)

# class SkillForm(forms.Form):
#     Skill_input = forms.CharField(label="Your skill", max_length=100)
```



Connection of Users between models

esearch_match > profilepage > signals.py > ...

```python
from django.db.models.signals import post_save
from django.contrib.auth.models import User
from django.dispatch import receiver
from .models import StudentProfile

# This function creates profile for each new user
@receiver(post_save, sender=User)
def create_profile(sender, instance, created, **kwargs):
    if created:
        StudentProfile.objects.create(user=instance)

#This function saves the profile every time user saves it
@receiver(post_save, sender=User)
def save_profile(sender, instance, **kwargs):
    instance.StudentProfile.save()
```

Update User Input

```
def studentprofile(request):
    if request.method == 'POST':
        u_form = UserUpdateForm(request.POST, instance=request.user)
        p_form = ProfileUpdateForm(request.POST,
                                   request.FILES,
                                   instance=request.user.studentprofile)

        if u_form.is_valid() and p_form.is_valid():
            u_form.save()
            p_form.save()
            messages.success(request, f'Your account has been updated!')
            return redirect('studenthomepage')


    else:
        u_form = UserUpdateForm(instance=request.user)
        p_form = ProfileUpdateForm(instance=request.user.studentprofile)


    context = {
        'u_form': u_form,
        'p_form': p_form
    }
    return render(request, 'profiledemo.html', context)
```

Including Updated User Info

```
<div class="FourBoxLayout">
  {% if user.is_authenticated %}
  <h1 style="padding-left: 2rem; margin-bottom: 0">Welcome Back, {{ user.studentprofile.firstname }} {{user.studentprofile.lastname}}!</h1>
  {% endif %}
  <br />
  <!-- Identity -->
  <div class="ProfileBox">
    <div class="Identityinfo">
      <h2 class="StudentName">{{user.studentprofile.firstname }} {{user.studentprofile.lastname}}</h2>
      <img
        src="{{ user.studentprofile.profile_pic.url }}"
        alt="Eli"
        class="StudentPicture"
      />
    </div>
```

```
<div class="tags">
  <span class="tag"> GPA: {{user.studentprofile.gpa}}</span>
  <div id="Skill_item"></div>
</div>
<div style="justify-self: last">
  <span> {{ user.studentprofile.skill}} </span>
```

```
<form class="Biographyform">
  <!-- <p class="Margins"></p> -->
  <p
    id="bio"
    name="biotext"
    rows="8"
    cols="20"
    placeholder="Hello, I am..."
  >{{ user.studentprofile.biography}}</p>
```

Search Function

```python
from django.shortcuts import render
from django.http import HttpResponse, Http404
from django.db.models import Q
from profilepage.models import User

def search_labs(request):
    if request.htmx:
        result = request.GET.get('search_labs')
        if len(result) > 0:
            # Search by background, first name, or last name
            users = User.objects.filter(
                Q(background__icontains=result) |
                Q(first_name__icontains=result) |
                Q(last_name__icontains=result)
            ).exclude(username=request.user.username)
            return render(request, 'search/list_search.html', {'users': users})
        else:
            return HttpResponse('Please enter a valid search.')
    else:
        raise Http404()
```

Timeline

Future Goals

Allow users to access data across web pages. 11/1

Allow users to interact with other users and other user data. 11/8

Build match system to help suggest labs to users. 11/15

Use test data to demo the website for the final presentation. 11/22

Thank you!