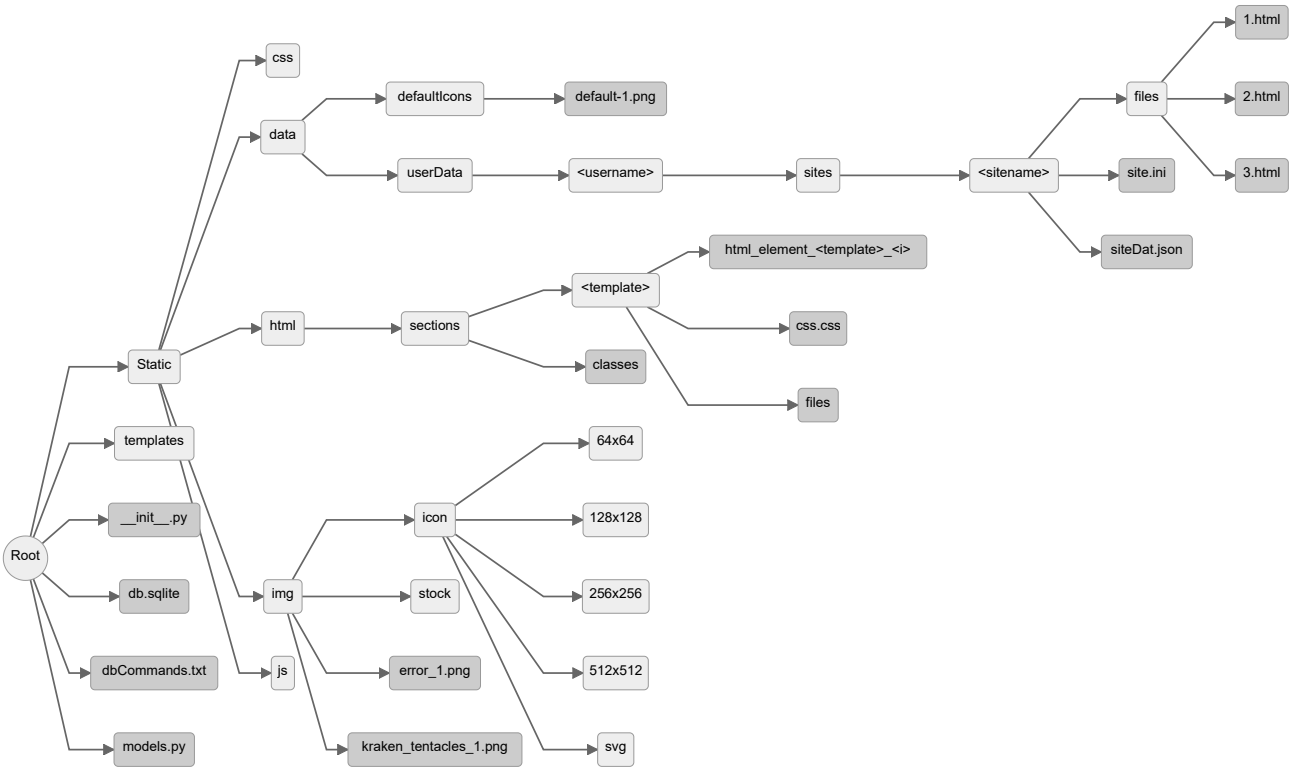
Appendix A - Code

Table of Contents

File Structure Diagram

For clarity, directories which contain large amounts of files, such as `/templates` or `/static/css` do not contain those files in the diagram. Files are tinted slightly darker than directories.



Code

Root Directory/

__init__.py

```
"""
Dependencies:
  flask
  flask-login
  flask-sqlalchemy
  pillow
  configparser
  datetime
  math
"""

# TODO: Fix color picker for site style generation page
# TODO: Add click listener to sections in dropdown
# TODO: make sure you dont render a random site that doesnt exist

from flask import Flask, render_template, Blueprint, redirect, url_for, request, flash, session,
Response
from flask_sqlalchemy import SQLAlchemy
from flask_login import LoginManager, login_user, login_required, current_user, logout_user

# Flask is the application object
# render_template converts a Jinja file to html
# redirect redirects the website to another route function
# url_for fetches the url of a server resource
# request allows the code to handle form inputs
# flash sends messages to the client
# session allows for the usage of client-side variable storage
# Response is used when handling redirects

# SQLAlchemy manages the SQL database

# LoginManager is the object that manages signed in users
# login_user logs in a give user
# login_required makes sure that you have to be logged in to visit said site
# current_user gets the current logged in user
# logout_user logs out a user

from werkzeug.security import generate_password_hash, check_password_hash

# generate_password_hash and check_password_hash are used when generating and authenticating users

from configparser import ConfigParser
import math,json

# ConfigParser is used when read/writing the user and site .ini files
# math.floor is used for calculating file size
# json.loads is used when saving files sent from the javascript

# Create the database object
databaseObject=SQLAlchemy()

# The main class of the application
class Kraken():
  """Initializes the Kraken application and runs it on the given host and port
```

```
    :param host: The host of the application, given in the format ``"<i>.<i>.<i>.<i>"``
    :type host: str
    :param port: The port of the application
    :type port: int

    :returns: Void
    :rtype: None
    """

    # Global reference to database object
    global databaseObject

    def __init__(self,host:str,port:int) -> None:

      # Assign the database object to the local db reference
      self.db=databaseObject

      # import modules and add them to the object
      import os,datetime
      self.os = os
      self.datetime = datetime

      # Initialise the flask application
      self.initFlask()

      # Initialise the SQL database
      self.db.init_app(self.app)

      # Initialise the login manager
      self.loginManager = LoginManager()
      self.loginManager.login_view = "auth_login" # set which function routes to the login page
      self.loginManager.init_app(self.app)

      # Import the User and Site entities from models.py
      from models import User, Site
      self.User = User
      self.Site = Site

      # Fetches a row from the User table in the database
      @self.loginManager.user_loader
      def loadUser(user_id:str):
        return self.User.query.get(user_id)

      # Run the Flask application
      self.app.run(host=host,port=port)

    def initFlask(self) -> None:
      """Initializes the Flask application by setting up the ``Flask`` object, configuring certain
      attributes, and creating the website pages via the function ``self.initPages()``.

      :returns: Void
      :rtype: None
      """

      # Create the Flask application and set a secret key
      self.app = Flask(__name__)
      self.app.config["SECRET_KEY"]="secret-key-goes-here"
      # Set the database file URL to /db.sqlite in the root directory
      self.app.config["SQLALCHEMY_DATABASE_URI"]="sqlite:///db.sqlite"
      self.app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
      # Initialise the website pages
      self.initPages()
```

```
  def initPages(self) -> None:
    """Assigns the ``Flask.app.route`` functions for all of the application pages.

    :returns: Void
    :rtype: None
    """

    # generate all of the website pages from the jinja2 html files in the /templates folder
    # jinja2 is a way of programatically generating html files using variables, iteration,
    # and templates
    self.initPages_main()
    self.initPages_auth()
    self.initPages_site_create()
    self.initPages_settings()

    @self.app.route("/<name>/<site>/")
    @self.app.route("/<name>/<site>/home/")
    @login_required
    def site_edit_home(name:str="",site:str="") -> str:
      """Page for ``/<name>/<site>`` or ``/<name>/<site>/home``.

      The page contains the basic information about the site, and links to perform actions such
      as start editing.

      Flashes ``1``, the username, and the sitename to the page.

      :param name: The username of the owner of a site
      :type name: str, optional, defaults to ``""``
      :param site: The name of the site
      :type site: str, optional, defaults to ``""``

      :returns: The HTML content of the page, generated from the Jinja template syntax
      :rtype: str
      """

      flash(1)
      flash(name)
      flash(site)

      # If the current user is not the owner, render a different version
      if current_user.user_id != name: return "External view of site"

      return render_template("site-edit-home.html")

    @self.app.route("/<name>/<site>/edit/")
    @login_required
    def site_edit_app(name:str="",site:str="") -> str | Response:
      """Page for ``/<name>/<site>/edit``

      The page contains the editor for a given site. Can only be accessed if the user is the owner.

      Flashes ``2``, the username, and the sitename to the page.

      :param name: The username of the owner of a site
      :type name: str, optional, defaults to ``""``
      :param site: The name of the site
      :type site: str, optional, defaults to ``""``

      :returns: The HTML content of the page, generated from the Jinja template syntax OR a
      redirect to the ``site_edit_home`` page if the user does not have the required permissions
      :rtype: str | Response
      """

      flash(2)
```

```
      flash(name)
      flash(site)

      # If the current user is not the owner, redirect to the site home
      if current_user.user_id != name: return redirect(url_for("site_edit_home"))

      return render_template("site-edit.html",name=name,site=site)

    @self.app.route("/<name>/<site>/save/<page>",methods=["post"])
    @login_required
    def site_edit_save(name:str="",site:str="",page:str="") -> str:
      # get the content of the request - the HTML to save
      tosave:str = request.form.get("content")
      # get the path of the file to save from the arguments
      sitePath:str = self.os.path.abspath(f"static/data/userData/{name}/sites/{site}/files/{page}")
      # save to the file
      with open(sitePath,"w") as f: f.write(tosave)
      return "success"

  def initPages_main(self):

    @self.app.route("/")
    def main_index() -> Response:
      """Page for ``/``

      This page redirects to the login page.

      :returns: A redirect to ``auth_login``
      :rtype: Response
      """

      return redirect(url_for("auth_login"))

    @self.app.route("/home/")
    @login_required # User must be logged in to access this page, or it will redirect to the
    # auth_login page (as setup in the constructor)
    def main_home() -> str:
      """Page for ``/home/``.

      This page is the homepage, containing links to all of the current user's sites.

      Requires a user to be logged in.

      Flashes a list of the users sites, in the format ``[<userid>,<sitename>,<isprivate>]``,
      if the user has any sites.

      :returns: The HTML content of the page, generated from the Jinja template syntax
      :rtype: str
      """

      # check to see if user has any sites
      if len(self.Site.query.filter_by(user_id=current_user.user_id).all()) > 0: # and False:
        flash([[x.user_id,x.name,x.private] for x in self.Site.query.filter_by(
          user_id=current_user.user_id).all()])

        # For testing and design purposes, the flash command above was commented out
        # and replaced with this command
        # flash([["user1","Site 1",True],["user1","Epic Webpage",False]])

        return render_template("home-sites.html")
      return render_template("home-nosite.html")

    @self.app.errorhandler(404)
    @self.app.route("/404")
```

```
    def main_404(e=None) -> str: return "Page not found - i.e. you made a mistake"

    @self.app.errorhandler(500)
    @self.app.route("/500")
    def main_500(e=None) -> str: return "Server go boom - i.e. I made a mistake"

    @self.app.route("/help")
    def main_help() -> str: return "This page dont exist yet :("

  def initPages_auth(self):

    @self.app.route("/login/")
    def auth_login() -> str | Response:
      """Page for ``/login/``.

      This page is the login page. It contains the login form.

      Will redirect to ``main_home`` if a user is logged in.

      Flashes an empty list of values (``[False,"","","",""]``) to stop errors when generating
      the Jinja template.

      :returns: The HTML content of the page, generated from the Jinja template syntax OR a
      redirect to the ``main_home`` page if a user is logged in
      :rtype: str | Response
      """

      # Flash an empty list of values to stop errors in the Jinja code
      flash([False,"","","",""])

      # If a user is logged in, redirect to the homepage
      if current_user.is_authenticated: return redirect(url_for("main_home"))
      return render_template("login.html")

    @self.app.route("/login/", methods=["post"])
    def auth_login_post() -> Response:
      """POST method of ``/login/``.

      Fetches the given user information from the form via ``flask.request``, and validates it.
      The user is logged in if the data is valid,
      and shown an error message if it is not.

      Flashes an appropriate error message, if required.

      :returns: A redirect to the ``main_home`` or ``auth_login`` depending on the success of
      the login attempt
      :rtype: Response
      """

      # Get the filled-in items from the login form
      username:str = request.form.get("username")
      password:str = request.form.get("password")
      remember:bool = True if request.form.get('remember') else False

      # Fetch the user from the database. if there's no user it returns none
      user = self.User.query.filter_by(user_id=username).first()

      #  Check for correct password
      if not user or not check_password_hash(user.password, password):
        # Flashes true to signify an error, the error message, the username given, and the
        # remember flag given
        flash(['Please check your login details and try again.',username,remember])
        return redirect(url_for('auth_login'))
```

```
        login_user(user,remember=remember)
        return redirect(url_for("main_home"))

    @self.app.route("/signup/")
    def auth_signup() -> str | Response:
        """Page for ``/signup/``.

        This page is the signup page. It contains the signup form.

        Will redirect to ``main_home`` if a user is logged in.

        Flashes an empty list of values (``[False,"","","","",""]``) to stop errors when generating
        #the Jinja template.

        :returns: The HTML content of the page, generated from the Jinja template syntax OR a
        redirect to the ``main_home`` page if a user is logged in
        :rtype: str | Response
        """

        # Flash an empty list of values to stop errors in the Jinja code
        flash([False,"","","","",""])

        # If a user is logged in, redirect to the homepage
        if current_user.is_authenticated: return redirect(url_for("main_home"))

        return render_template("signup.html")

    @self.app.route("/signup/", methods=["post"])
    def auth_signup_post() -> Response:
        """POST method of ``/signup/``.

        Fetches the given user information from the form via ``flask.request``, and validates it
        via ``self.verifyField``. A new user is created if valid,
        or the user is shown an error message if it is not.

        :returns: A redirect to the ``main_home`` or ``auth_signup`` depending on the success of
        the signup attempt
        :rtype: Response
        """

        # Get the filled-in items from the signup form
        name:str = request.form.get("name")
        email:str = request.form.get("email")
        username:str = request.form.get("username")
        password1:str = request.form.get("password")
        password2:str = request.form.get("password-repeat")

        # the verifyField function returns either an empty string if the field meets the requirements
        # defined by the arguments, or an error message. So, if len(verifyOutput) > 0, that
        # means that the field is invalid

        # Verify the name input and return an error message if invalid
        verifyOutput:str = self.verifyField(name,"Name",canHaveSpace=True,canHaveSpecialChar=True)

        if verifyOutput:
            # Flashes true to signify an error, the error message, the name given,
            # the email given, and the username given
            flash([True,verifyOutput,"",email,username])
            return redirect(url_for("auth_signup"))

        # Verify the email input and return an error message if invalid
        verifyOutput:str = self.verifyField(email,"Email",minLen=0,canHaveSpace=False,
                                            canHaveSpecialChar=True)
```

```
        if verifyOutput:
          # Flash an error message and the filled in values
          flash([True,verifyOutput,name,"",username])
          return redirect(url_for("auth_signup"))

        verifyOutput:str = self.verifyField(username,"Username",canHaveSpecialChar=False)

        # Verify the username input and return an error message if invalid
        if verifyOutput:
          # Flash an error message and the filled in values
          flash([True,verifyOutput,name,email,""])
          return redirect(url_for("auth_signup"))

        # Verify the password input and return an error message if invalid
        verifyOutput:str = self.verifyField(password1,"Password",minLen=8)

        if verifyOutput:
          # Flash an error message and the filled in values
          flash([True,verifyOutput,name,email,username])
          return redirect(url_for("auth_signup"))

        # Return an error message if the passwords do not match
        if password1!=password2:
          # Flash an error message and the filled in values
          flash([True,"Passwords do not match",name,email,username])
          return redirect(url_for("auth_signup"))

        # check whether this email already has an account
        if self.User.query.filter_by(email=email).first():
          flash([True,"That email is already in use",name,"",username])
          return redirect(url_for("auth_signup"))

        # check whether this username already exists
        user = self.User.query.filter_by(user_id=username).first()

        if user:
          flash([True,"That username is already in use",name,email,""])
          return redirect(url_for("auth_signup"))

        # create a new user in the database and server-side storage
        self.createUser(username,email,name,password1)

        # Log in the new user and redirect them to the homepage
        login_user(self.User.query.filter_by(user_id=username).first(),remember=False)
        return redirect(url_for("auth_login"))

      @self.app.route("/logout/")
      @self.app.route("/account/logout/")
      @login_required
      def auth_logout() -> Response:
        """Method for ``/logout/`` or ``/account/logout/``.

        Requires a user to be logged in.

        Logs out the current user.

        :returns: A redirect to ``auth_login``
        :rtype: Response
        """

        logout_user()
        return redirect(url_for("auth_login"))

    def initPages_site_create(self):
```

```
@self.app.route("/home/new/")
@login_required
def site_create() -> str:
  """Page for ``/home/new/``.

  This page is the first site creation page. It contains inputs such as website name,
  description, and privacy setting.

  Requires a user to be logged in.

  Flashes a list of the user's current site names, to be used in client-side validation of
  the inputted website name.

  :returns: The HTML content of the page, generated from the Jinja template syntax
  :rtype: str
  """

  # get all current site names for the logged in user, then flash (send) it to the site, where
  # it is processed by the JavaScript
  out:str = ""
  allusernames:list[str] = [x.name for x in self.Site.query.filter_by(
                            user_id=current_user.user_id).all()]

  for name in allusernames:
    out+=name+","

  flash(out[:-1])

  return render_template("site-create.html")

@self.app.route("/home/new/", methods=["post"])
@login_required
def site_create_post():
  """POST method of ``/home/new/``.

  Fetches the given user information from the form via ``flask.request``, validates it, and
  stores it in ``flask.session``.

  :returns: A redirect to the ``site_create_options_1``
  :rtype: Response
  """

  def listToStr(var:list) -> str:
    """Turns a list into a string

    :param var: The list to merge
    :type var: list

    :returns: The merged list as a string
    :rtype: str
    """

    out:str = ""
    for char in var: out+=char
    return out

  def replaceToDash(var:str) -> str:
    """Replaces any invalid characters in a given string with a dash.
    Used to format the site name correctly so that there arent any errors

    :param var: The string to replace invalid characters
    :type var: string
```

```
        :returns: The formatted string
        :rtype: str
        """

        var:list[str] = list(var)
        for i in range(len(var)):
            if var[i] not in "qwertyuiopasdfghjklzxcvbnm-._1234567890": var[i]="-"
        return listToStr(var)

    def replaceRepeatedDashesRecursion(var:str) -> str:
        """Removes any adjacent dashes in a string via recursion

        :param var: The string to format
        :type var: str

        :returns: The formatted string
        :rtype: str
        """

        var:list[str] = list(var)
        for i in range(len(var)):

            # if this is the last character, return
            if i+1 >= len(var): return listToStr(var)

            # if this and the next character are dashes
            if var[i] == "-" and var[i+1] == "-":
                # remove this character
                del var[i]
                var = list(replaceRepeatedDashesRecursion(var))
                return listToStr(var)

    # get the user inputs
    sitename:str = request.form.get("new_site_name")
    sitedesc:str = request.form.get("new_site_desc")
    isPublic:bool = request.form.get("new_site_privacy")=="public"

    # convert the sitename to lowercase, replace all illegal characters with dashes,
    # and remove adjacent dashes
    sitename:str = replaceRepeatedDashesRecursion(replaceToDash(sitename.lower()))

    # session can carry over variables between functions
    session["new_site_sitename"] = sitename
    session["new_site_sitedesc"] = sitedesc
    session["new_site_isPublic"] = isPublic

    return redirect(url_for("site_create_options_1"))

@self.app.route("/home/new/1")
@login_required
def site_create_options_1() -> str | Response:
    """Page for ``/home/new/1``.

    This page is the second site creation page. It contains the color palette selection system

    Requires a user to be logged in.
    Requires the referrer to be ``site_create``

    :returns: The HTML content of the page, generated from the Jinja template syntax OR a
    redirect to ``site_create`` if ``site_create`` was not the referrer.
    :rtype: str | Response
    """

    # stop people from starting halfway through the form i.e. if they didnt come from the
```

```
    # previous site_create page, send them to the start
    if not request.referrer == url_for("site_create",_external=True):
      return redirect(url_for("site_create"))

    return render_template("site-create-options-1.html")

  @self.app.route("/home/new/1", methods=["post"])
  @login_required
  def site_create_options_1_post() -> Response:
    """POST method of ``/home/new/1``.

    Fetches the given user information from the form via ``flask.request``, validates it,
    and stores it in ``flask.session``.

    :returns: A redirect to the ``site_create_options_2``
    :rtype: Response
    """

    # new_site_color_options_dict is in the format
    # "key1:value,key2:value,key3:value"
    # so splitting by comma gives ["key1:value","key2:value","key3:value"]
    # then iterate through the list, split by colon, and add parts to dictionary

    # split into "k:v" list items
    formOutput:list[str] = request.form.get("new_site_color_options_dict").split(",")

    # create output dictionary
    colorOptions = dict()

    for pair in formOutput: # where pair is in the format "<key>:<value>"
      # split into ["<key>","<value>"]
      colonsplit:list[str] = pair.split(":")

      # append to dictionary - result = {... , "<key>":"<value>"}
      colorOptions[colonsplit[0]] = colonsplit[1]

    # add color options dictionary to session
    session["new_site_colorOptions"] = colorOptions

    # proceed to next page
    return redirect(url_for("site_create_options_2"))

  @self.app.route("/home/new/2")
  @login_required
  def site_create_options_2() -> str | Response:
    """Page for ``/home/new/2``.

    This page is the second third creation page. It contains the typeface selection system.

    Requires a user to be logged in.
    Requires the referrer to be ``site_create_options_1``

    :returns: The HTML content of the page, generated from the Jinja template syntax OR a
    redirect to ``site_create`` if ``site_create_options_1`` was not the referrer.
    :rtype: str | Response
    """

    # stop people from starting halfway through the form i.e. if they didnt come from the
    # previous site_create page, send them to the start
    if not request.referrer == url_for("site_create_options_1",_external=True):
      return redirect(url_for("site_create"))

    return render_template("site-create-options-2.html")
```

```
    @self.app.route("/home/new/2", methods=["post"])
    @login_required
    def site_create_options_2_post() -> Response:
      """POST method of ``/home/new/2``.

      Fetches the given user information from the form via ``flask.request``, validates it,
      and stores it in ``flask.session``.

      :returns: A redirect to the ``site_create_options_3``
      :rtype: Response
      """

      # new_site_font_face_list_active is in the format "<headerfontname>,<paragraphfontname>"
      # so split by comma to get ["<headerfontname>","<paragraphfontname>"]

      # get value from the active form element and split into list
      formOutput:list[str] = request.form.get("new_site_font_face_list_active").split(",")

      # store font selection in session
      session["new_site_fontOptions"]=formOutput

      # proceed to next page
      return redirect(url_for("site_create_options_3"))

    @self.app.route("/home/new/3")
    @login_required
    def site_create_options_3() -> str | Response:
      """Page for ``/home/new/2``.

      This page is the second third creation page. It contains the button styling selection system.

      Requires a user to be logged in.
      Requires the referrer to be ``site_create_options_2``

      :returns: The HTML content of the page, generated from the Jinja template syntax OR a
      redirect to ``site_create`` if ``site_create_options_2`` was not the referrer.
      :rtype: str | Response
      """

      # stop people from starting halfway through the form i.e. if they didnt come from the
      # previous site_create page, send them to the start
      if not request.referrer == url_for("site_create_options_2",_external=True):
        return redirect(url_for("site_create"))

      return render_template("site-create-options-3.html")

    @self.app.route("/home/new/3", methods=["post"])
    @login_required
    def site_create_options_3_post() -> Response:
      """POST method of ``/home/new/2``.

      Fetches the given user information from the form via ``flask.request``, validates it,
      and stores it in ``flask.session``.

      :returns: A redirect to the ``site_create_generate``
      :rtype: Response
      """

      # similarly to the colour options, new_site_font_face_list_active is in the format
      # "key1:value,key2:value,key3:value"
      # so splitting by comma gives ["key1:value","key2:value","key3:value"]
      # then iterate through the list, split by colon, and add parts to dictionary

      # split into "k:v" list items
```

```
    formOutput:list[str] = request.form.get("new_site_style_options_list").split(",")

    # create output dictionary
    styleOptions = {}

    for pair in formOutput: # where pair is in the format "<key>:<value>"
      # split into ["<key>","<value>"]
      colonsplit:list[str] = pair.split(":")

      # Make sure that all variables are in the correct form
      match colonsplit[1]:
          case "false": colonsplit[1] = False
          case "true": colonsplit[1] = True
          case "null": colonsplit[1] = None

      # append to dictionary - result = {... , "<key>":"<value>"}
      styleOptions[colonsplit[0]] = colonsplit[1]

    # store in session
    session["new_site_buttonOptions"] = styleOptions

    # proceed to site generation
    return redirect(url_for("site_create_generate"))

  @self.app.route("/home/new/generate")
  @login_required
  def site_create_generate() -> Response:
    """Method for ``/home/new/generate``.

    This page is the second third creation page. It contains the button styling selection system.

    Requires a user to be logged in.
    Requires the referrer to be ``site_create_options_3``

    :returns: A redirect to the new site homepage (``site_edit_home``) if successful OR a
    redirect to ``site_create`` if ``site_create_options_3`` was not the referrer.
    :rtype: Response
    """

    # stop people from starting halfway through the form i.e. if they didnt come from the
    # previous site_create page, send them to the start
    if not request.referrer == url_for("site_create_options_3",_external=True):
      return redirect(url_for("site_create"))

    # Store session data into a dictionary, along with extra paramaters such as when the site
    # was created and the user id
    siteSettings:dict[str,str | self.datetime] = {
      "name": session["new_site_sitename"],
      "user": str(current_user.user_id),
      "desc": session["new_site_sitedesc"]
        if session["new_site_sitedesc"]!="" else "No Description Set",
      "created": self.datetime.datetime.utcnow(),
      "isPublic": session["new_site_isPublic"],
      "colorOptions": session["new_site_colorOptions"],
      "fontOptions": session["new_site_fontOptions"],
      "buttonOptions": session["new_site_buttonOptions"]
    }

    # Create the site database object and local storage
    self.createSiteStructure(siteSettings)

    # Clear any used session variables
    session["new_site_sitename"] = ""
    session["new_site_sitedesc"] = ""
```

```
        session["new_site_isPublic"] = ""
        session["new_site_colorOptions"] = dict()
        session["new_site_fontOptions"] = list()
        session["new_site_buttonOptions"] = dict()

        # Redirect to the site homepage
        return redirect(url_for("site_edit_home",
                                name=siteSettings["user"],
                                site=siteSettings["name"]))

    def initPages_settings(self):
      @self.app.route("/account/settings/")
      @login_required
      def settings() -> Response:
        """Method for ``/account/settings/``.

        This page redirects to the settings homepage (``settings_profile``)

        :returns: A redirect to ``settings_profile``.
        :rtype: Response
        """

        return redirect(url_for("settings_profile"))

      @self.app.route("/account/settings/profile")
      @login_required
      def settings_profile() -> str:
        """Page for ``/account/settings/profile``.

        This page contains the settings Public Profile page.

        Requires a user to be logged in.

        Flashes ``1`` to inform the local navbar of the current page, and the URL for the
        user's profile picture.

        :returns: The HTML content of the page, generated from the Jinja template syntax.
        :rtype: str
        """

        # Flash the URL for the user's profile picture
        flash(1,self.getUserImage(current_user.user_id))

        return render_template("settings-profile.html")

      @self.app.route("/account/settings/admin")
      @login_required
      def settings_admin() -> str:
        """Page for ``/account/settings/admin``.

        This page contains the settings Account page.

        Requires a user to be logged in.

        Flashes ``2`` to inform the local navbar of the current page.

        :returns: The HTML content of the page, generated from the Jinja template syntax.
        :rtype: str
        """

        flash(2)
        return render_template("settings-admin.html")

      @self.app.route("/account/settings/looks")
```

```
      @login_required
      def settings_looks() -> str:
        """Page for ``/account/settings/looks``.

        This page contains the settings Appearance and Accessibility page.

        Requires a user to be logged in.

        Flashes ``3`` to inform the local navbar of the current page.

        :returns: The HTML content of the page, generated from the Jinja template syntax.
        :rtype: str
        """

        flash(3)
        return render_template("settings-looks.html")

      @self.app.route("/account/settings/sites")
      @login_required
      def settings_sites() -> str:
        """Page for ``/account/settings/sites``.

        This page contains the settings My Sites page.

        Requires a user to be logged in.

        Flashes ``4`` to inform the local navbar of the current page, and a list of all of the
        user's sites in the format ``[userid,sitename,isprivate,sitesize]``.

        :returns: The HTML content of the page, generated from the Jinja template syntax.
        :rtype: str
        """

        flash(4)

        # Flash a list of information about the user's sites to be used in the site table.
        flash([
          [
            x.user_id,
            x.name,
            x.private,
            self.convertByteSize(self.getFolderSize(self.os.path.abspath(x.sitePath)))
          ] for x in self.Site.query.filter_by(user_id=current_user.user_id).all()
        ])

        return render_template("settings-sites.html")

      @self.app.route("/account/settings/code")
      @login_required
      def settings_code() -> str:
        """Page for ``/account/settings/``.

        This page contains the settings My Code page.

        Requires a user to be logged in.

        Flashes ``5`` to inform the local navbar of the current page.

        :returns: The HTML content of the page, generated from the Jinja template syntax.
        :rtype: str
        """

        flash(5)
        return render_template("settings-code.html")
```

```
    @self.app.route("/account/settings/dev")
    @login_required
    def settings_dev() -> str:
      """Page for ``/account/settings/``.

      This page contains the settings Developer Settigns page.

      Requires a user to be logged in.

      Flashes ``7`` to inform the local navbar of the current page.

      :returns: The HTML content of the page, generated from the Jinja template syntax.
      :rtype: str
      """

      flash(7)
      return render_template("settings-dev.html")

def createUser(self,u:str,e:str,n:str,p:str) -> None:
    """Creates a new user in the database and in local storage.

    :param u: The username of the new user
    :type u: str
    :param e: The email of the new user
    :type e: str
    :param n: The name of the new user
    :type e: str
    :param p: The password of the new user
    :type p:

    :returns: Void
    :rtype: None
    """

    # Create a new User object using the varaibles given
    newUser = self.User(
      user_id=u,
      name=n,
      email=e,
      password=generate_password_hash(p,method='sha256'),
      bio="",
      url="",
      archived=False,
      tabpreference=4,
    )

    # Server-side folder generation

    prefix:str = "static/data/userData/"

    # create a list of required directories
    folderStructure:list[str] = [
      self.os.path.abspath(f"{prefix}{u}"),
      self.os.path.abspath(f"{prefix}{u}/sites/")
    ]

    # create the list of directories in the local storage
    self.generateFolderStructure(folderStructure)

    # add and commit the user to the database
    self.db.session.add(newUser)
    self.db.session.commit()
```

```
  def createSiteStructure(self,siteSettings:dict[str,any]) -> None:
    """Creates a new site in the database and in local storage.

    :param siteSettings: a dictionary containing the information about the site, with the
    keys: ``name`` ``user`` ``desc`` ``created`` ``isPublic`` ``colorOptions`` ``fontOptions``
    ``buttonOptions``
    :type siteSettings: dict

    :returns: Void
    :rtype: None
    """

    # get the prefix for the site path
    sitePath:str = self.os.path.abspath(f"static/data/userData/{siteSettings['user']}/sites/{
                                        siteSettings['name']}")

    # get the path for the site config file
    siteConfigFile:str = f"{sitePath}/site.ini"

    # create a list of required directories
    folderStructure:list[str] = [
      f"{sitePath}",
      f"{sitePath}/output",
      f"{sitePath}/files"
    ]

    # create a list of required files
    fileStructure:list[str] = [
      siteConfigFile,
      f"{sitePath}/siteDat.json",
      f"{sitePath}/files/1.html"
    ]

    # create the required directories and files in the local storage
    self.generateFolderStructure(folderStructure)
    self.generateFileStructure(fileStructure)

    # create the JSON file for the site, which contains code locations
    with open(f"{sitePath}/siteDat.json","w") as f:
      f.write("{\"pages\":{\"Home\":\"1.html\"},\"css\":{},\"js\":{}}")

    # create the default webpage for the site
    with open(f"{sitePath}/files/1.html","w") as f:
      f.write(self.defaultHtmlPage(siteSettings["name"],siteSettings["desc"],siteSettings["user"]))

    # Create the config parser for the site.ini file
    with ConfigParser() as cfgContent:

      cfgContent.read(siteConfigFile)

      # create the required sections
      for section in ["settings","color","font","button"]:
        try: cfgContent.add_section(section)
        except: pass

      # Add basic information to the settings section
      cfgContent.set("settings","name",siteSettings["name"])
      cfgContent.set("settings","user",siteSettings["user"])
      cfgContent.set("settings","desc",siteSettings["desc"])

      # Add the color palette
      for key in siteSettings["colorOptions"]:
        cfgContent.set("color",key,siteSettings["colorOptions"][key])
```

```
      # Add the typeface
      cfgContent.set("font","header",siteSettings["fontOptions"][0])
      cfgContent.set("font","body",siteSettings["fontOptions"][1])

      # Add any required button styling settings
      for key in siteSettings["buttonOptions"]:
        cfgContent.set("button",key,str(siteSettings["buttonOptions"][key]).lower())

      # write to the site.ini file
      with open(siteConfigFile,"w") as f:
        cfgContent.write(f)
        f.close()

    # Create a new Site object using the varaibles given
    newSite = self.Site(
      name = siteSettings["name"],
      datecreated = siteSettings["created"],
      private = not siteSettings["isPublic"],
      deleted = False,
      user_id = siteSettings["user"],
      sitePath = sitePath,
    )

    # Add and commit the new site to the database
    self.db.session.add(newSite)
    self.db.session.commit()

  def generateFolderStructure(self,folders:list) -> None:
    """Creates directories in the local storage from the given list, if any of the directories
    do not already exist

    :param folders: A list of directory paths to create
    :type folders: list

    :returns: Void
    :rtype: None
    """

    for folder in folders: # Iterate through given list of directories
      # Ignore if directory already exists
      if self.os.path.isdir(folder): continue
      # Create the directory
      try: self.os.makedirs(folder)
      except OSError as e: raise e

  def generateFileStructure(self,files:list) -> None:
    """Creates empty files in the local storage from the given list, if any of the files
    do not already exist

    :param files: A list of file paths to create
    :type files: list

    :returns: Void
    :rtype: None
    """

    for file in files: # Iterate through given list of files
      # Ignore if file already exists
      if self.os.path.exists(file): continue

      try:
        # Create the empty file
        with open(file,"w") as f:
            if self.os.path.splitext(file)[-1] == ".json": f.write("{\"content\":[]}")
```

```
            f.close()
        except OSError as e: raise e

    def getUserImage(self,u:str) -> str:
        """Returns the path of a given user's profile picture

        :param u: The username to fetch the profile picture of
        :type u: str

        :returns: A local path of the profile picture
        :rtype: str
        """

        return f"/data/userIcons/{u}.png"

    def verifyField(self,field:str,fieldName:str,mustHaveChar:bool=True,minLen:int=3,
                    canHaveSpace:bool=False,canHaveSpecialChar:bool=True) -> str:
        """The validataion system for user inputs implemented in the signup page.

        :param field: The content of the field that is being validated.
        :type field: str
        :param fieldName: The name of the field that is being validated.
        :type fieldName: str
        :param mustHaveChar: Boolean flag determining whether the field must not be empty,
                             defaults to ``True``
        :type mustHaveChar: bool, optional
        :param minLen: The minimum length of the field content, defaults to ``3``
        :type minLen: int, optional
        :param canHaveSpace: Boolean flag determining whether the field can have spaces,
                             defaults to ``False``
        :type canHaveSpace: bool, optional
        :param canHaveSpecialChar: Boolean flag determining whether the field is allowed to contain
                                   any of ``%&{}\\<>*?/$!'\":@+|=``, defaults to ``True``
        :type canHaveSpecialChar: bool, optional

        :returns: An empty string if the field is valid OR an error message if the field is invalid
        :rtype: str
        """

        # List of special characters for the canHaveSpecialChar flag
        specialChar:str = "%&{}\\<>*?/$!'\":@+`|="

        # Make sure that the input given is a string, raise an exception if its not
        if type(field) != str: TypeError("HEY! that's not a string?")

        # Check through all the flags given and throw an appropriate error message if input is invalid

        # If it is empty and it must have content
        if len(field) == 0 and mustHaveChar:
            return f"{fieldName} is not filled out."

        # If it is shorter than the minimum length
        if len(field) < minLen:
            return f"{fieldName} must be greater than {minLen-1} characters."

        # If it has spaces when it shouldn't
        if not canHaveSpace and " " in field:
            return f"{fieldName} cannot contain spaces."

        # If it can't have any special characters, and there are special characters in the field
        if not canHaveSpecialChar:
            for char in specialChar:
                if char in field:
                    return f"{fieldName} cannot contain '{char}'"
```

```
    return "" # Return an empty string if the input is valid

  def getFolderSize(self,path:str) -> int:
    """Gets the size of a given folder path via recursively searching through every directory
    in a depth-first manner.

    :param path: The path of the root folder
    :type path: str

    :returns: The size of the folder, in bytes
    :rtype: int
    """

    # Get the size of the base directory, should return 0
    size:int = self.os.path.getsize(path)

    # for all directories and files under the path
    for sub in self.os.listdir(path):
      subPath=self.os.path.join(path,sub) # get the path of the directory / file

      # get the size if it is a file
      if self.os.path.isfile(subPath): size += self.os.path.getsize(subPath)

      # get the size if it is a directory by calling this function
      elif self.os.path.isdir(subPath): size += self.getFolderSize(subPath)

    # return the size, in bytes
    return size

  def convertByteSize(self,bytes:int) -> str:
    """Converts a given byte value into a human readable version

    :param bytes: The amount of bytes to convert
    :type bytes: int

    :returns: A human readable version of the value
    :rtype: str
    """

    # Zero check
    if bytes==0: return "0B"

    # All possible sizes
    sizes:tuple[str] = ("B","KB","MB","GB","TB","PB","EB","ZB","YB")

    i:int = int(math.floor(math.log(bytes,1024)))
    p:float = math.pow(1024,i)
    s:float = round(bytes/p,2)

    if i==0: s = int(s)

    return f"{s}{sizes[i]}"

  def getSiteCfg(self,siteName:str) -> ConfigParser:
    """Gets the content of the current user's given site's config file.

    :param siteName: the name of the site to check
    :type siteName: str

    :returns: The content of the config file
    :rtype: ConfigParser
    """
```

```
    # Get path of the config file
    cfgPath=self.os.path.abspath(f"static/data/userData/{current_user.user_id}/sites/
                                  {siteName}/site.ini")

    # Open a config parser for said path
    cfgContent=ConfigParser()
    cfgContent.read(cfgPath)

    # Return the config parser
    return cfgContent

  def defaultHtmlPage(self,n:str,d:str,u:str):
    """Returns the default HTML content for a new page

    :param n: The name of the site
    :type n: str
    :param d: The description of the site
    :type d: str
    :param u: The name of the owner
    :type u: str

    :returns: The default HTML content for a new page
    :rtype: str
    """

    return f"""<div class=\"page\" data-content-parentview></div>"""

if __name__ == "__main__":
  # Initialise the application on port 1380
  Kraken("0.0.0.0",1380)
```

dbCommands.txt

```
CREATE TABLE user (
  user_id TEXT PRIMARY KEY,
  name TEXT,
  email TEXT NOT NULL,
  password TEXT NOT NULL,
  bio TEXT,
  url TEXT,
  archived BOOLEAN NOT NULL,
  tabpreference INT NOT NULL
)

CREATE TABLE site (
  site_id INTEGER PRIMARY KEY AUTOINCREMENT,
  name TEXT NOT NULL,
  datecreated DATE,
  private BOOLEAN NOT NULL,
  deleted BOOLEAN NOT NULL,
  user_id TEXT,
  sitePath TEXT,
  CONSTRAINT fk_user_id,
    FOREIGN KEY (user_id) REFERENCES user(user_id)
)
```

models.py

```
from flask_login import UserMixin

# Import the SQLAlchemy database object from the main class
from __init__ import databaseObject as db

# User class to store the user's information in the database
class User(UserMixin,db.Model):
  # Set the name of the table in the database to "user"
  __tablename__="user"

  # Define the columns in the table
  # Primary Key user_id as a string
  user_id = db.Column( db.String, primary_key=True )
  # Name as a string
  name = db.Column( db.String )
  # Email as a string, cannot be null and must be unique
  email = db.Column( db.String, nullable=False, unique=True )
  # Password as a string, cannot be null
  password = db.Column( db.String, nullable=False )
  # Bio as a text field
  bio = db.Column( db.Text )
  # URL as a text field
  url = db.Column( db.Text )
  # Archived flag as a boolean, cannot be null (default False)
  archived = db.Column( db.Boolean, nullable=False )
  # Tab preference as a number, cannot be null (default four)
  tabpreference = db.Column( db.Float, nullable=False )

  # Setup the foreign key relationship
  sites = db.relationship("Site")

  # Function to return the primary key
  def get_id(self) -> str: return self.user_id

# Site class to store the User's sites in the database
class Site(db.Model):
  # Set the name of the table in the database to "site"
  __tablename__="site"

  # Define the columns of the table
  # Primary Key site_id as an integer
  site_id = db.Column( db.Integer, primary_key=True )
  # Foreign Key user_id as a string, referring to user_id in the User table
  user_id = db.Column( db.String, db.ForeignKey("user.user_id") )
  # Name as a string, cannot be null
  name = db.Column( db.String, nullable=False )
  # Datecreated as a datetime format
  datecreated = db.Column( db.DateTime )
  # Private flag as a boolean, cannot be null
  private = db.Column( db.Boolean, nullable=False )
  # Deleted flag as a boolean, cannot be null (default False)
  deleted = db.Column( db.Boolean, nullable=False )
  # Sitepath as a text field
  sitePath = db.Column( db.Text )

  # Function to return the primary key
  def get_id(self) -> int: return self.site_id
```

requirements.txt

```
flask
flask-login
flask-sqlalchemy
pillow
configparser
datetime
```

/static/css/

/static/css/auth.css

```
.application-container {
  width:100vw;
  height:100vh;
  display:flex;
  flex-direction:row;
}

.application-content {
  width:100%;
  height:100%;
  padding: 16px;
}

.application-content .text-header-container {
  margin-bottom:64px;
  display:flex;
  flex-direction:column;
  align-items: center;
}

.application-content .text-header-container .text.one {
  margin-bottom:16px;
}

.application-content .header-option {
  position:relative;
  overflow:visible;
}

.application-content .header-option::after {
  content: "";
  background-color: var(--colors-grey-500);
  width: 70%;
  height: 4px;
  border-radius: 5px;
  position: absolute;
  bottom: -5px;
  right: -8px;
  opacity:1;
  transition: background-color 200ms ease-in-out,width 200ms ease-in-out,height 200ms ease-in-out,
              bottom 200ms ease-in-out,right 200ms ease-in-out,opacity 200ms ease-in-out;
}

.application-content .header-option.active::after {
  background-color: var(--colors-secondary-dark);
}
```

```css
.application-content .header-option:hover::after {
  background-color: var(--colors-grey-300);
  width: 100%;
  height: 100%;
  bottom: 0;
  right: 0;
  opacity: 0.2;
}

.application-content .header-option.active:hover::after {
  background-color: var(--colors-secondary);
}

.application-content .section-header-item:not(:last-child) {
  margin-bottom: 16px
}

.application-content .header-options {
  width:50%;
  display:flex;
  justify-content: space-evenly
}

.application-content .header-option {
  cursor:pointer;
}

.application-content .header-option:active {
  opacity:.8;
}

.application-content .header-option.active {
  color:var(--colors-secondary);
}

.application-content .field-container {
  display: flex;
  flex-direction: column;
  align-items: center;
}

.application-content .field-container .field-submit {
  margin-top:32px;
  align-self: center;
}

.application-content .field-container .field-options {
  width:360px;
  display:flex;
  flex-direction:column;
}

.application-content .field-container .field-option:not(:last-child) {
  margin-bottom:8px
}

.application-content .field-container .field-option {
  display:flex;
  flex-direction: row;
  justify-content: space-between;
}

.application-content .field-container .field-option .field-input-container {
```

```
  display:flex;
  flex-direction: row;
}

.application-content .field-container .field-option .field-input {
  font-family: var(--font-body);
}

.application-content .field-container .field-option .field-input-container .eye-reveal,
.application-content .field-container .field-option .field-input-container .eye-spacer {
  width:19px;
  height:19px;
  display:flex;
  justify-content: center;
  align-items: center;
  margin-left:8px;
}

.application-content .field-container .field-option .field-input-container .eye-reveal:active {
  color:var(--colors-secondary);
}

.application-content .field-container .field-warning {
  color:#e63832;
  margin-bottom:8px;
}
```

/static/css/blank.css

```
/* Remove all animations, transitions and smooth scroll for people that prefer not to see them */
@media (prefers-reduced-motion: reduce) {
  html:focus-within {
    scroll-behavior: auto !important;
  }
  *,
  *::before,
  *::after {
    animation-duration: 0.01ms !important;
    animation-iteration-count: 1 !important;
    transition-duration: 0.01ms !important;
    scroll-behavior: auto !important;
  }
}
*,
*::before,
*::after {
  box-sizing: border-box;
  margin: 0;
  padding: 0;
  color: inherit;
}
html {
  scroll-behavior: smooth;
  -ms-text-size-adjust: 100%;
  -webkit-text-size-adjust: 100%;
}
body {
  min-height: 100vh;
  width: 100vw;
  overflow-x: hidden;
  text-rendering: optimizeSpeed;
```

```
  margin: 0;
  padding: 0;
}
ol,
ul {
  list-style: none;
}
pre,
code,
address,
caption,
th,
figcaption {
  font-size: 1em;
  font-weight: normal;
  font-style: normal
}
fieldset,
iframe {
  border: 0
}
img,
picture,
.image {
  max-width: 100%;
  display: block;
}
caption,
th {
  text-align: left
}
table {
  border-collapse: collapse;
  border-spacing: 0
}
main,
summary,
details {
  display: block
}
audio,
canvas,
video,
progress {
  vertical-align: baseline
}
body,
input,
textarea,
select,
button {
  font-synthesis: none;
  -moz-font-feature-settings: 'kern';
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
  direction: ltr;
  text-align: left
}
```

/static/css/build.css

```
.globalnav {
  background: var(--colors-grey-200);
  position:fixed;
}

.globalnav-floating-options {
  transform:scale(0);
  transform-origin:bottom left;
  opacity:0;
  margin:0;
  transition:transform,opacity,margin,visibility;
  transition-delay:130ms;
  transition-duration: 260ms;
  transition-timing-function: ease-in-out;
  background-color:var(--colors-grey-100);
  position:fixed;
  bottom:0;
  left:96px;
  z-index:9998;
  padding:16px 8px;
  display:flex;
  flex-direction: column;
  align-items: center;
  border-radius: 10px;
  visibility:hidden;
}

.globalnav-floating-options.is-active {
  margin-bottom:16px;
  margin-left:16px;
  transform:scale(1);
  opacity:1;
  visibility:visible;
}

.globalnav-floating-option:not(:first-of-type) {
  margin-top:16px;
}

.globalnav-floating-option-content {

}

.globalnav-floating-options-backdrop {
  z-index:9997;
  position:fixed;
  width:calc(100vw - 96px);
  height:100vh;
  background-color: #000;
  top:0;
  right:0;
  opacity:0;
  transition: opacity 260ms 130ms ease-in-out, visibility 260ms 130ms ease-in-out;
  visibility: hidden;
}

.globalnav-floating-options-backdrop.is-active {
  opacity:0.4;
  visibility: visible;
}
```

```
.application-container {
  width:calc(100vw - 96px);
  height:100vh;
  display:flex;
  flex-direction:row;
  margin-left:96px;
}

.application-content {
  width:100%;
  height:100%;
  padding: 16px;
}

.application-content .text-header-container {
  margin-bottom:64px;
}

.main {
  /*height:calc(100% - 79px - 64px);*/
  width:100%;
  display:flex;
  justify-content: center;
}

.main-content {
  width:100%;
  height:100%;
}

.main-content.thin {
  width:60%
}
```

/static/css/default_content_style.css

`default_content_style.css` has not been included due to its size (3590 SLOC).

/static/css/edit.css

```
.grid-item {
  position: relative;
  left: var(--left);
  z-index: var(--hidden-element-z-index, var(--z-index));
  display: grid;
  grid-area: var(--position);
  width: var(--element-width, 100%);
  height: var(--element-height, 100%);
  text-align: var(--text);
  grid-template-columns: 100%;
}

.grid-item-content {
  grid-area: 1/1/-1/-1;
  align-self: start;
}

.text-box {
  width: 100%;
  padding: 0;
```

```
  margin: 0;
  overflow-wrap: break-word;
  white-space: var(--whiteSpace, break-spaces);
  background-color: var(--backgroundColor);
  outline: none;
}

@media screen and (max-width: 920px) {
  .grid-item {
    grid-area:auto;
    grid-column: 3/-3;
    align-items: var(--m-align, var(--align));
    align-self: var(--m-align-self);
    justify-content: var(--m-justify, var(--justify));
    max-width: 100%;
    text-align: var(--m-text, var(--text))
  }
}

@media screen and (max-width: 920px) {
  .grid-item {
    width:var(--m-width, 100%)
  }
}

.section-grid {
  position: relative;
  z-index: 14;
  display: grid;
  grid-area: 1/1/-1/-1;
  grid-template-rows: repeat(var(--rows),var(--row-size));
  grid-template-columns:
    calc((var(--container-width) - var(--content-width)) / 2)
    1fr repeat(var(--column-count),minmax(0,var(--column-width)))
    1fr calc((var(--container-width) - var(--content-width)) / 2);
  grid-auto-rows: var(--row-size);
  row-gap: var(--row-gap);
  -moz-column-gap: var(--column-gap);
  column-gap: var(--column-gap);
  width: 100%;
  padding: var(--block-padding);
  margin: 0 auto;
  grid-template-rows: repeat(var(--rows),minmax(var(--row-size),auto));
  --column-gap-count: calc(var(--column-count) - 1);
  --column-width: calc( (var(--grid-width)
                        - (var(--column-gap-count)
                        * var(--column-gap)))
                        / var(--column-count) );
  --container-width: 100vw
}

@media screen and (max-width: 920px) {
  .section-grid {
    display:flex;
    flex-direction: column;
    grid-gap: 0;
    padding: var(--m-block-padding)
  }
}

.section-grid-preview {
  position: absolute;
  top:0;
  z-index: 13;
```

```
  display: grid;
  grid-template-rows: repeat(var(--rows),var(--row-size));
  grid-template-columns:
    calc((var(--container-width) - var(--content-width)) / 2)
    1fr repeat(var(--column-count),minmax(0,var(--column-width)))
    1fr calc((var(--container-width) - var(--content-width)) / 2);
  grid-auto-rows: var(--row-size);
  row-gap: var(--row-gap);
  -moz-column-gap: var(--column-gap);
  column-gap: var(--column-gap);
  width: 100%;
  padding: var(--block-padding);
  margin: 0 auto;
  grid-template-rows: repeat(var(--rows),minmax(var(--row-size),auto));
  --column-gap-count: calc(var(--column-count) - 1);
  --column-width: calc( (var(--grid-width)
                        - (var(--column-gap-count)
                        * var(--column-gap)))
                        / var(--column-count) );
  --container-width: 100vw;
  opacity:0;
  transition: opacity 200ms ease-in-out;
}

.section-grid-preview[data-kraken-visible] {
  opacity:1;
}

.section-grid-preview-box {
  box-shadow: inset 0 0 2px rgba(var(--colors-dark-rgb),.1);
  width: var(--column-width);
  height: var(--row-size);
  width: 0px;
  height: 0px;
  position: absolute;
  top: 16px;
  left: 48px;
  border-radius: 0.5em;
  border: 2px solid var(--colors-grey-300);
  opacity:.4;
}
```

/static/css/home.css

```
:root {
  --link-previsited-color:var(--colors-dark);
}

.application-content .text-header-container {
  display:flex;
  align-items: center;
}

.application-content .empty-container {
  display:flex;
  flex-direction: column;
  align-items: center;
  width:100%;
}

.application-content .empty-container .empty-image {
```

```
  background-image: url(../img/kraken_tentacles_one.png);
  width: 65%;
  height: 346px;
  background-size: contain;
  background-repeat: no-repeat;
  background-position: center;
  margin-bottom:32px;
}

.application-content .empty-container .empty-text-container {
  display: flex;
  flex-direction: column;
  align-items: center;
}

.application-content .empty-container .empty-text-container .text.two {
  position: relative;
  overflow:visible
}

.application-content .empty-container .empty-text-container .text.two:hover {
  background-position:right;
  cursor:pointer;
}

.application-content .empty-container .empty-text-container .text.two::after {
  content: "";
  background-color: var(--colors-primary-dark);
  width: 70%;
  height: 4px;
  position: absolute;
  bottom: -5px;
  right: -18px;
  opacity:1;
  border-radius: 5px;

  transition:
    background-color 200ms ease-in-out,
    width 200ms ease-in-out,
    height 200ms ease-in-out,
    bottom 200ms ease-in-out,
    right 200ms ease-in-out,
    opacity 200ms ease-in-out;
}

.application-content .empty-container .empty-text-container .text.two:hover::after {
  background-color: var(--colors-primary);
  width: 100%;
  height: 100%;
  bottom: 0;
  right: 0;
  opacity: 0.2;
}

.site-div {
  width: 320px;
  position: relative;
  height: 180px;
  padding: 16px;
  border-radius: 1em;
  margin-left:32px;
  margin-bottom:32px;
  border: 0px solid rgba(var(--colors-dark-rgb),0);
  transition: 200ms border ease-in-out;
```

```
  cursor:pointer;
}

.site-div:hover {
  border: 5px solid rgba(var(--colors-dark-rgb),1);
}
```

/static/css/index.css

```
.globalnav {
  background: var(--colors-grey-100);
  position:static;
}

.application-container {
  width:100vw;
  height:100vh;
  display:flex;
  flex-direction:row;
}

.application-content {
  width:100%;
  height:100%;
  padding: 16px;
}

.application-content .text-header-container {
  margin-bottom:64px;
  display:flex;
  flex-direction:column;
  align-items: center;
}

.application-content .text-header-container .text.one {
  margin-bottom:16px;
}

.application-content .header-option {
  position:relative;
  overflow:visible;
}

.application-content .header-option::after {
  content: "";
  background-color: var(--colors-grey-500);
  width: 50px;
  height: 4px;
  position: absolute;
  bottom: -5px;
  right: -8px;
  opacity:1;
  transition:background-color 200ms ease-in-out,width 200ms ease-in-out,height 200ms ease-in-out,
  bottom 200ms ease-in-out,right 200ms ease-in-out,opacity 200ms ease-in-out;
}

.application-content .header-option.active::after {
  background-color: var(--colors-secondary-dark);
}

.application-content .header-option:hover::after {
```

```
  background-color: var(--colors-grey-300);
  width: 54px;
  height: 19px;
  bottom: 0;
  right: 0;
  opacity: 0.2;
}

.application-content .header-option.active:hover::after {
  background-color: var(--colors-secondary);
}

.application-content .section-header-item:not(:last-child) {
  margin-bottom: 16px
}

.application-content .header-options {
  width:50%;
  display:flex;
  justify-content: space-evenly
}

.application-content .header-option {
  cursor:pointer;
}

.application-content .header-option:active {
  opacity:.8;
}

.application-content .header-option.active {
  color:var(--colors-secondary);
}

.application-content .field-container {
  display: flex;
  flex-direction: column;
  align-items: center;
}

.application-content .field-container .field-submit {
  margin-top:32px;
  align-self: center;
}

.application-content .field-container .field-options {
  width:360px;
  display:flex;
  flex-direction:column;
}

.application-content .field-container .field-option:not(:last-child) {
  margin-bottom:8px
}

.application-content .field-container .field-option {
  display:flex;
  flex-direction: row;
  justify-content: space-between;
}

.application-content .field-container .field-option .field-input-container {
  display:flex;
  flex-direction: row;
```

```
}

.application-content .field-container .field-option .field-input {
  font-family: var(--font-body);
}

.application-content .field-container .field-option .field-input-container .eye-reveal,
.application-content .field-container .field-option .field-input-container .eye-spacer {
  width:19px;
  height:19px;
  display:flex;
  justify-content: center;
  align-items: center;
  margin-left:8px;
}

.application-content .field-container .field-option .field-input-container .eye-reveal:active {
  color:var(--colors-secondary);
}

.application-content .field-container .field-warning {
  color:#e63832;
  margin-bottom:8px;
}
```

/static/css/main.css

`main.css` has not been included due to its size (3001 SLOC). The syntax that it defines is shown below.

```
Global classes:
    .notextselect
    .nopointerevents
    .visibly-hidden
    .fake
    .box
    .no-inversion

Positional Classes:
    .relative
    .sticky
    .fixed
    .abs
    .static

Text classes:
    .text <light|dark|primary|secondary|accent|grey-100 => grey-800> [italic]
    [bold|thin] [ellipsis] [xl|large|default-size|small] [header|jumbo]
    [lowercase|uppercase] [notextselect] [left|center|right|justify]

Link classes:
    .text .link [classes for text] [disabled] [link-slide] [notformatted]
    (link-slide class requires the css variable --link-slide-width)

Btn classes:
    .btn <light|dark|primary|secondary|accent> <square|rounded|pill> [slide]
    .btn.slide [from-left|from-right]

    If slide is set, the btn must contain a span element with syntax:
        span .btn-content .text (all text classes apply here)
```

```
    which contains the text of the button

    A span like this is recommended even if the .slide class is not present so
    you can format the text inside separately

em classes:
    [classes for text]

/* .section-content.fixed-width will set the width to 1440px, and will set the
width to 100% when the viewport width is less than 1440px */
```

/static/css/preview_override.css

```
/* MAIN.CSS OVERRIDES */

[data-preview] {
  --margin-xs: 6px;
  --padding-xs: 6px;
}

/* Text */
@media {
  [data-preview] .text,[data-preview] .text.default-size {
    font-size: 10.666px;
  }
  [data-preview] .text.large {
    font-size: 16px;
  }
  [data-preview] .text.small,[data-preview] .text.footnote {
    font-size: 8px;
  }
  [data-preview] .text.header,[data-preview] .text.header.default-size {
    font-size: 21.333px;
  }
  [data-preview] .text.header.large {
    font-size: 32px;
  }
  [data-preview] .text.header.small {
    font-size: 18.666px;
  }
  [data-preview] .text.jumbo {
    letter-spacing: 6.666px;
  }
  [data-preview] .text.xl,[data-preview] .text.jumbo {
    font-size: 42.666px;
  }
  [data-preview] .text.header.xl {
    font-size: 64px;
  }
  [data-preview] .text.jumbo.xl {
    font-size: 185.333px;
  }
}

/* Link */
@media {

  [data-preview] .link.link-slide {
    --link-slide-width: calc(100% - 10px);
    padding: 0 5.333px;
    outline-offset: -4.666px;
```

```
  }
  [data-preview] .link.link-slide::before {
    height: 1.333px;
    left: 6px;
  }
  [data-preview] .link.link-slide:hover::before {
    width: 0%;
  }
  [data-preview] .link.underline-box::after {
    bottom: -3.333px;
  }
  [data-preview] .link.underline-box:hover::after {
    background-color: var(--colors-primary-dark);
    width: 70%;
    height: 16%;
    bottom: -3.333px;
    right: -6%;
    opacity: 1;
  }
}

/* Button */
@media {
  [data-preview] .btn:not(.slide):hover:not([disabled]) {
    background-color: var(--colors-dark);
  }
  [data-preview] .btn:not(.slide).light:hover:not([disabled]) {
    background-color: var(--colors-light);
  }
  [data-preview] .btn:not(.slide).dark:hover:not([disabled]) {
    background-color: var(--colors-dark);
  }
  [data-preview] .btn:not(.slide).primary:hover:not([disabled]) {
    background-color: var(--colors-primary);
  }
  [data-preview] .btn:not(.slide).secondary:hover:not([disabled]) {
    background-color: var(--colors-secondary);
  }
  [data-preview] .btn:not(.slide).accent:hover:not([disabled]) {
    background-color: var(--colors-accent);
  }
  [data-preview] .btn:not(.slide).danger:hover:not([disabled]) {
    background-color: var(--colors-danger);
  }
  [data-preview] .btn.slide {
    border-width: 1.333px;
  }
  [data-preview] .btn.slide:hover:not([disabled]) {
    border-color: var(--colors-light);
  }
  [data-preview] .btn.slide.light:hover:not([disabled]) {
    border-color: rgba(var(--colors-light-rgb), .15);
  }
  [data-preview] .btn.slide.dark:hover:not([disabled]) {
    border-color: rgba(var(--colors-dark-rgb), .15);
  }
  [data-preview] .btn.slide.primary:hover:not([disabled]) {
    border-color: rgba(var(--colors-primary-rgb), .15);
  }
  [data-preview] .btn.slide.secondary:hover:not([disabled]) {
    border-color: rgba(var(--colors-secondary-rgb), .15);
  }
  [data-preview] .btn.slide.accent:hover:not([disabled]) {
    border-color: rgba(var(--colors-accent-rgb), .15);
```

```
  }
  [data-preview] .btn.slide.danger:hover:not([disabled]) {
    border-color: rgba(var(--colors-danger-rgb), .15);
  }
  [data-preview] .btn.slide::before {
    content:none;
    left: 100%!important;
  }
  [data-preview] .btn.slide.light::before,
  [data-preview] .btn.slide.dark::before,
  [data-preview] .btn.slide.primary::before,
  [data-preview] .btn.slide.secondary::before,
  [data-preview] .btn.slide.accent::before,
  [data-preview] .btn.slide.danger::before {
    background-color: none;
  }
}

/* Section */
@media {
  [data-preview] {
    --section-separator-height: 5.333px;
  }
  [data-preview] .section .section-content {
    height: calc(100% - 64px);
    padding-left: 42.666px;
    padding-right: 42.666px;
    padding-top: 21.333px;
    padding-bottom: 21.333px;
  }
  [data-preview] .section .section-content.fixed-width {
    width: var(calc(--section-fixed-width * 2 / 3), 960px);
  }
  [data-preview] .section .section-header {
    padding-left: 10.666px;
    padding-right: 10.666px;
  }
}

/* EDIT.CSS OVERRIDES */

[data-preview] .grid-item {
  width: var(calc(--element-width * 2 / 3), 100%);
  height: var(calc(--element-height * 2 / 3), 100%);
}

[data-preview] .section-grid {
  grid-template-rows: repeat(var(--rows),calc(var(--row-size) * 2 / 3));
  grid-template-columns: calc(
    (calc(var(--container-width) * 2 / 3) - calc(var(--content-width) * 2 / 3)) / 2)
    1fr repeat(var(--column-count),minmax(0,calc(var(--column-width) * 2 / 3)))
    1fr calc((calc(var(--container-width) * 2 / 3) - calc(var(--content-width) * 2 / 3)) / 2);
  grid-auto-rows: calc(var(--row-size) * 2 / 3);
  row-gap: calc(var(--row-gap) * 2 / 3);
  -moz-column-gap: calc(var(--column-gap) * 2 / 3);
  column-gap: calc(var(--column-gap) * 2 / 3);
  padding: calc(var(--block-padding) * 2 / 3);
  grid-template-rows: repeat(var(--rows),minmax(calc(var(--row-size) * 2 / 3),auto))
}
```

/static/css/settings.css

```
.settings-container {
  height: calc(100vh - 32px - 188px);
  display: flex;
}

.settings-sidebar {
  width: 400px;
  height: 100%;
  margin-left: 10px;
}

.settings-sidebar-item-icon {
  width:46px;
  font-size:24px;
  display: flex;
  justify-content: center;
}

.settings-sidebar-item {
  display: flex;
  align-items: center;
  justify-content: flex-start;
  border-radius: 5px;
  padding-top: 2px;
  padding-bottom: 2px;
  position:relative;
}

.settings-sidebar-item:not(:last-of-type) {
  margin-bottom: 4px;
}

.settings-sidebar-item.is-active {
  background: var(--colors-grey-200);
}

.settings-sidebar-item.is-active::before {
  content:"";
  background-color:var(--colors-primary);
  height:100%;
  border-radius:5px;
  position:absolute;
  top:0;
  left:-10px;
  width:6px;
}

.settings-sidebar-item:hover {
  background: var(--colors-grey-300);
}

.settings-sidebar-item.is-active .settings-sidebar-item-title {
  font-weight: 700;
}

.settings-sidebar-separator {
  padding-left: 8px;
  padding-top: 2px;
  margin-bottom: 8px;
  margin-top: 24px;
  width: 100%;
```

```
  border-top: 2px solid var(--colors-grey-400);
}

.settings-content {
  width: calc(100% - 400px - 32px);
  margin-left: 32px;
}

.settings-content .settings-content-header {
  border-bottom: 2px solid var(--colors-grey-400);
  width: 100%;
  margin-bottom: 16px;
  padding-bottom: 4px;
}

.settings-content-options {
  padding-bottom:32px;
  padding-left:16px;
}

.settings-content .settings-content-option {
  width: 80%;
  margin-bottom: 24px;
}

.settings-content .settings-content-option-input {
  padding: 5px 12px;
  border-radius: 5px;
  border-style: solid;
  border-width: 2px;
  font-family: var(--font-body);
  width:232px;
}

.settings-content textarea.settings-content-option-input {
  min-width: 190px;
  min-height: 28px;
  max-width: 100%;
  max-height: 192px;
}

.settings-content .settings-content-option-title {
  margin-bottom: 6px;
}

.settings-content .settings-content-option-caption {
  margin-top:6px;
}

.settings-content .settings-content-option-image-upload {
  width:200px;
  height:200px;
  position:relative;
  cursor:pointer;
}

.settings-content .settings-content-option-image-upload-figure {
  width:200px;
  height:200px;
  background-size: 200px 200px;
  background-position: center;
  border-radius:50%
}
```

```css
.settings-content .settings-content-option-image-upload::before {
  content: "Upload Image";
  width: 100%;
  height: 100%;
  background-color: rgba(var(--colors-dark-rgb),.3);
  top: 0;
  left: 0;
  opacity:0;
  position: absolute;
  border-radius: 50%;
  color: var(--colors-dark);
  font-family: var(--font-header);
  font-size: 26px;
  font-weight: 700;
  display: flex;
  justify-content: center;
  align-items: center;
  transition:opacity 100ms ease-in-out;
}

.settings-content .settings-content-option-image-upload:hover::before {
  opacity:1;
}

.settings-content .settings-content-separator {
  width:100%;
  height:0px;
  border-top:2px solid var(--colors-grey-400);
  margin-bottom:24px;
}

.settings-content .settings-content-update .settings-content-option-caption {
  margin-top:0px;
  margin-bottom:8px;
}

.settings-content .settings-content-table {
  border: 2px solid var(--colors-grey-400);
  border-radius: 10px;
  overflow:hidden;
  max-width:780px;
}

.settings-content .settings-content-table-header {
  background-color:var(--colors-grey-100);
}

.settings-content .settings-content-table-row {
  padding: 16px;
}

.settings-content .settings-content-table-row:not(:last-of-type),
.settings-content .settings-content-table-header {
  border-bottom: 2px solid var(--colors-grey-400);
}

.settings-content-table-row-icon,
.settings-content-table-row-title {
  margin-right:6px;
}

.settings-content-table-row-settings {
```

```
  float: right;
}
```

/static/css/site-create.css

```
.new-site-form {
  width:100%;
  margin-bottom: 32px;
}

.new-site-form .form-input-container {
  padding-bottom:8px;
  padding-left:16px;
}

.new-site-form.one .form-input-container.one {
  display:flex;
  flex-wrap:wrap;
}

.new-site-form.one .form-input-container.one .form-input-content-column {
  height:84px;
  display:flex;
  flex-direction:column;
  justify-content:space-between;
}

.new-site-form.one .form-input-container.one .form-input-content-column .text.one {
  padding-top: 8px;
}

.new-site-form.one .form-input-container.one .form-input-content-column * {
  height:32px
}

.new-site-form.one .form-input-container.one [data-form-input-display="inactive"].new-site-input {
  border-color:var(--colors-grey-400);
  box-shadow: none;
}

.new-site-form.one .form-input-container.one [data-form-input-display="success"].new-site-input {
  border-color:var(--colors-success);
  box-shadow: 0px 0px 22px -8px var(--colors-success);
}

.new-site-form.one .form-input-container.one [data-form-input-display="warning"].new-site-input {
  border-color:var(--colors-warning);
  box-shadow: 0px 0px 22px -8px var(--colors-warning);
}

.new-site-form.one .form-input-container.one [data-form-input-display="danger"].new-site-input {
  border-color:var(--colors-danger);
  box-shadow: 0px 0px 22px -8px var(--colors-danger);
}

.new-site-form.one .form-input-container.one .message-container {
  width:100%;
}

.new-site-form.one .form-input-container.three {
  display: flex;
```

```
  justify-content: space-between;
  align-items: center;
}

.input-checkbox {
  margin-top:15px;
  margin-bottom:15px;
  width:100%;
  display:flex;
  align-items: center;
}

.input-checkbox-input {
  float: left;
  vertical-align: middle;
  margin-right:8px;
}

.input-checkbox-icon {
  font-size:26px;
  margin-right:8px;
}

.input-checkbox-text-container {
  display:flex;
  flex-direction:column;
}

.new-site-form.two .light-dark-selector-container {
  width:100%;
  display:flex;
  justify-content:center;
}

.new-site-form.two .light-dark-selector {
  display:flex;
  flex-direction:column;
  width:fit-content;
}

.new-site-form.two .light-dark-selector .button-container {
  width:100%;
  display:flex;
  flex-direction:row;
  justify-content:space-evenly;
}

.new-site-form.two .color-display-container .color-display {
  margin-bottom:var(--margin-m);
}

.new-site-form.two .color-display-container .light-dark-display {
  display:flex;
  justify-content:space-around;
}

.new-site-form.two .color-display-container .grey-display {
  height: 160px;
}

.new-site-form.two .color-display-container .color-single-card {
  display:flex;
  align-items:center;
  justify-content:center;
```

```
  width:160px;
  height:120px;
  border-radius:1em;
  box-shadow: rgb(0 0 0 / 20%) 0px 6px 40px -10px;
}

.new-site-form.two .color-display-container .color-code {
  font-family:var(--font-header);
}

.new-site-form.two .color-display-container .color-columns {
  box-shadow: rgb(0 0 0 / 10%) 2px 6px 40px -10px;
}

.new-site-form.two .color-display-container .color-columns .color-column {
  box-shadow:none;
}

.new-site-form.two .main-color-display {
  display: flex;
  justify-content: space-between;
}

.new-site-form.two .main-color-display .color-triple-card {
  display: flex;
  flex-direction: column;
  width: -webkit-fill-available;
  height: 140px;
  align-items: center;
  justify-content: center;
  border-radius: 1em;
  box-shadow: rgb(0 0 0 / 20%) 0px 6px 40px -10px;
  overflow:hidden;
  position:relative;
}

.new-site-form.two .main-color-display .color-triple-card:not(:last-of-type) {
  margin-right:24px;
}

.new-site-form.two .main-color-display .color-triple-card-main {
  height: 60%;
  width:100%;
  display: flex;
  justify-content: center;
  align-items: center;
}

.new-site-form.two .main-color-display .color-triple-card-sub-container {
  width: 100%;
  height: 40%;
  display: flex;
  flex-direction: row;
}

.new-site-form.two .main-color-display .color-triple-card-sub {
  width: 50%;
  height: 100%;
}

.new-site-form.two .color-card-picker-input {
  position:absolute;
  top:0;
  left:0;
```

```
  -webkit-appearance: none;
  border: none;
  width: 100%;
  height: 100%;
  background: transparent;
  opacity: 0;
  cursor: pointer;
}

.sliding-input {
  -webkit-appearance: none;
  height: 100%;
  background: transparent;
  border:none;
}

.sliding-input::-webkit-slider-thumb {
  -webkit-appearance: none;
  height: 16px;
  width: 16px;
  border-radius: 50%;
  background: var(--colors-light);
  margin-top: -3px;
  box-shadow: 0px 0px 11px -3px rgb(0 0 0 / 50%);
  cursor: pointer;
  border: 1px solid var(--colors-grey-400);
}

.sliding-input::-webkit-slider-runnable-track {
  width: 60%;
  height: 9px;
  background:var(--slider-background)
  border-radius: 3px;
  cursor: pointer;
}

.input-slider-option {
  display: flex;
  flex-direction: row;
  align-items: center;
  justify-content: space-between;
}

.input-slider-option .sliding-input {
  width:40%;
}

.color-options {
  display:flex;
  flex-direction:column;
}

.color-options .input-slider-option:not(:last-of-type) {
  margin-bottom:var(--margin-xs)
}

.input-slider-option
#new_site_colors_light_dark_bounds_slider.sliding-input::-webkit-slider-runnable-track {
  background:linear-gradient(to right, #333333, #000000);
  border-radius:3px;
}

.input-slider-option
#new_site_colors_monochromatic_temperature_slider.sliding-input::-webkit-slider-runnable-track {
```

```
  background:linear-gradient(to right, #3962D2, #ffffff, #F19C38);
  border-radius:3px;
}

.small-number-input {
  padding: 5px;
  width: 45px;
}

.input-slider-and-number {
  display:flex;
  flex-direction:row;
  width:45%;
  align-items: center;
  justify-content: flex-end;
}

.input-slider-and-number .sliding-input {
  width:-webkit-fill-available;
  margin-right:8px;
}

.new-site-form.two .submit-container {
  display:flex;
  justify-content:center;
}

.new-site-form.three .text-options {
  margin-bottom:var(--margin-m);
}

.new-site-form.three .text-option {
  display:flex;
  flex-direction:column;
  width:60%;
  border-radius:0.5em;
  padding:8px;
  background-color:transparent;
  cursor:pointer;
  transition:background-color 100ms ease-in-out,border-color 100ms ease-in-out;
  border-width:2px;
  border-style:solid;
  border-color:transparent;
}

.new-site-form.three .text-option:hover {
  background-color:var(--colors-grey-100);
}

.new-site-form.three .text-option.active {
  border-color:var(--colors-primary)
}

.new-site-form.three .text-option:not(:last-of-type) {
  margin-bottom:16px;
}

.new-site-form.four .button-option {
  display: flex;
  align-items: center;
  justify-content: space-between;
  width: 335px;
  transition: opacity 200ms ease-in-out, visibility 200ms ease-in-out;
}
```

```
.new-site-form.four .button-option-container {
  height:55px;
  margin-bottom: var(--margin-s);
  display: flex;
  align-items: center;
}
```

/static/css/site-edit.css

```
.localnav.one {
  border-radius: 1em;
  margin-right: 32px;
}
.localnav.one .localnav-item.one .localnav-item-collapsible-text,
.localnav.one .localnav-item.three .localnav-item-collapsible-text {
  width: 119px;
}
.localnav.one .localnav-item.two .localnav-item-collapsible-text {
  width: 104px;
}
.localnav.one .localnav-item.four .localnav-item-collapsible-text {
  width: 110px;
}
.localnav.one .localnav-item.five .localnav-item-collapsible-text {
  width: 134px;
}

.lightbox-mask {
  position:absolute;
  top:0;
  left:0;
  right:0;
  bottom:0;
  z-index:9995;
  width:100%;
  height:100%;
  background-color:rgba(0,0,0,.5);
  visibility:hidden;
  opacity:0;
  transition: opacity 200ms ease-out, visibility 200ms ease-out;
  display: none;
}

.lightbox-mask.shown {
  visibility:visible;
  opacity:1;
  cursor:pointer;
  display: block;
}

.section-selector {
  width:80%;
  height:80%;
  background-color: var(--colors-grey-100);
  border-radius: 1em;
  position:relative;
  display:flex;
  flex-direction: row;
  align-items: center;
  z-index: 9996;
```

```
}
.section-selector-container {
  position:absolute;
  width: calc(100% - 208px);
  height: calc(100% - 32px);
  display: flex;
  -webkit-box-align: center;
  -ms-flex-align: center;
  align-items: center;
  -webkit-box-pack: center;
  -ms-flex-pack: center;
  justify-content: center;
  right: 16px;
  opacity:0;
  transform:scale(0.9);
  transition: opacity 200ms ease-in-out, visibility 200ms ease-in-out;
  visibility:hidden;
  z-index: 9996;
  display:none;
}
.section-selector-container.shown {
  opacity:1;
  transform:scale(1);
  visibility:visible;
  display: flex;
}
.section-selector-exit-btn {
  position: absolute;
  top: 16px;
  right: 16px;
  font-size: 24px;
  cursor: pointer;
}

.section-selector-sidebar {
  width:180px;
  height:100%;
}

.section-selector-nav {
  background-color: transparent;
  display: block;
  height: 100%;
  width: 180px;
  max-height: 100%;
  max-width: 180px;
}
.section-selector-nav .section-selector-nav-content {
  margin: 0 auto;
  position: relative;
  height: inherit;
  z-index: 102;
  margin: auto 0;
  display: flex;
  flex-direction: column;
  justify-content: center;
}
.section-selector-nav .section-selector-nav-list {
  cursor: default;
  margin: 0 -8px;
  display: -webkit-box;
  display: -ms-flexbox;
  display: flex;
  -webkit-box-pack: justify;
```

```
  -ms-flex-pack: justify;
  justify-content: flex-start;
  align-items: center;
  height: 85%;
  flex-direction: column;
}

.section-selector-nav-item li {
  margin-bottom:6px;
}

.section-selector-nav-item a.link {
  position: relative;
}

.section-selector-content {
  height: 100%;
  display: flex;
  /*width: -webkit-fill-available;*/
  width: calc(100% - 182px);
  align-items: center;
  justify-content: center;
}
.section-selector-content .section-selector-list {
  max-width: 90%;
  width: 90%;
  height: 85%;
  display: flex;
  flex-direction: column;
  align-items: center;
  overflow-y: scroll;
}
.section-selector-content .section-selector-list-item {
  width: 45%;
  height: fit-content;
  display: flex;
  justify-content: center;
}

.section-selector-content .section-selector-list::-webkit-scrollbar {
  background: transparent;
}
.section-selector-content .section-selector-list::-webkit-scrollbar-thumb {
  background-color:var(--colors-grey-300);
}

.section-selector-404-error {
  font-weight: 700;
  font-family: var(--font-header);
  font-size: 32px;
  text-align: center;
  overflow: hidden;
  text-decoration: none;
  text-transform: none;
  color: var(--colors-danger)!important;
}

.application-content {
  display: flex;
  flex-direction: row;
  justify-content: space-between;
}
.site-builder {
  width:100%;
```

```
  scroll-behavior: auto;
  overflow-y: scroll;
  overscroll-behavior: auto;
  border: 2px solid var(--colors-grey-200);
  border-radius: 1em;
  --scrollbar-display: none;
  position: relative;
}

.site-builder-preview {
  width:100%;
  height:100%;
  position: relative;
}

.site-builder-preview .page {
  width:100%;
}

[data-kraken-section] {
  position: relative;
  display: grid;
  cursor: pointer;
}

@keyframes lock-hash-move {
  from{background-position-x:calc(var(--background-size) * -1)}
  to{background-position-x:var(--background-size)}
}

[data-kraken-section][data-kraken-locked] {
  --hash-stripe-width:20px;
  --hash-opacity:.1;
  --hash-light-color: rgba(var(--colors-light-rgb),var(--hash-opacity));
  --hash-dark-color: rgba(var(--colors-dark-rgb),var(--hash-opacity));
  --background-size: calc(var(--hash-stripe-width) * 16.97);
  background-image: repeating-linear-gradient(
                      45deg,
                      var(--hash-light-color) 0,
                      var(--hash-light-color) var(--hash-stripe-width),
                      var(--hash-dark-color) var(--hash-stripe-width),
                      var(--hash-dark-color) calc(var(--hash-stripe-width) * 2));
  background-size: var(--background-size);
  background-repeat: repeat;
  filter:grayscale(1);
  animation:lock-hash-move 60s linear infinite;
}

[data-kraken-section][data-kraken-section-selected]:not([data-preview]):not([data-kraken-locked])
.controls-ghost-overlay {
  border: 3px solid var(--selected-border-color, #053ea0);
}

[data-kraken-section][data-kraken-section-selected]:not([data-preview])[data-kraken-locked]
.controls-ghost-overlay {
  border: 3px solid var(--selected-border-locked-color, #adb5bd);
}

[data-kraken-section]:not([data-preview]):not([data-kraken-locked]):hover .controls
.controls-ghost-overlay {
  border: 3px solid var(--hover-border-color);
}

.controls {
```

```
  position: absolute;
  top: 0;
  left: 0;
  z-index: 210;
  display: none;
  width: 100%;
  height: 100%;
  pointer-events: none;
  display: grid;
}

.controls .controls-ghost-overlay {
  position: relative;
  z-index: 201;
  grid-area: var(--position);
  pointer-events: none;
  outline-color: initial;
  border: 0px solid transparent;
  transition: border 80ms ease-in-out;
  --position: 0/0/0/0;
  --outline-color: var(--accent-two);
  --hover-border-color: var(--colors-primary-light);
  --selected-border-color: var(--colors-primary);
  --selected-border-locked-color: var(--colors-grey-300);
}

.controls .controls-resize-handle-container {
  position: absolute;
  z-index: 203;
  align-self: end;
  justify-self: center;
  width: 100%;
  max-width: 1200px;
}

[data-kraken-section][data-kraken-section-selected]:not([data-preview]):not([data-kraken-locked])
.controls .resize-handle,.section-wrapper:hover:not([data-preview]) .controls .resize-handle {
  display:flex
}

.controls .resize-handle {
  display: flex;
  flex-direction: column;
  padding: 4px 20px;
  color: #fff;
  cursor: row-resize;
  background: #357df9;
  border-radius: 3px;
  transition: background .2s cubic-bezier(.4,0,.2,1),transform .1s ease-out;
  transform: translateY(var(--resize-movement-length));
  position: absolute;
  right: 0;
  z-index: 203;
  display: none;
}

.controls .controls-resize-handle-container .trigger {
  display: flex;
  align-items: center;
  justify-content: center;
  position: relative;
}

.grid-item .grid-item-content,
```

```css
.grid-item .grid-item-content * {
  width:100%;
  height:100%;
}

.section-edit-actions {
  position: absolute;
  inset: 0px auto auto 0px;
  margin: 0px;
  top:0;
  left:100%;
  transform:translateX(calc(-100% - 8px));
  z-index: 104;
  opacity: 1;
  transition: opacity .15s cubic-bezier(.4,0,.2,1), top .12s ease-in-out;
  margin-top:16px;
  margin-right:8px;
}

.section-edit-actions .actions-list {
  display: none;
  justify-content: center;
}

.section-edit-actions.shown .actions-list {
  display:flex;
}

.section-edit-actions .edit-action {
  border-radius: 1em;
  background: var(--colors-light);
  width: 32px;
  height: 32px;
  display: flex;
  justify-content: center;
  align-items: center;
  font-size: 16px;
  box-shadow: 0 4px 9px 0 rgba(0,0,0,.15);
  cursor:pointer;
  margin-right:8px;
  transition-property: background-color,box-shadow;
  transition-duration: 115ms;
  transition-timing-function: ease-in-out;
}

.section-edit-actions .edit-action:active {
  --pressed-color: var(--colors-grey-200);
  --inset-shadow-opacity: .15;
  background-color: var(--pressed-color);
  box-shadow: 0 4px 9px 0 rgba(0,0,0,.15), inset 0 0 8px rgba(0,0,0,var(--inset-shadow-opacity));
}

.section-edit-actions .edit-action.hilight-danger:active {
  --pressed-color: var(--colors-danger);
  --inset-shadow-opacity: .35;
}

.element-resize-container {
  display:grid;
  grid-area:1/1/-1/-1;
  visibility:hidden;
  opacity:0;
  transition: opacity 80ms ease-in-out, visibility 80ms ease-in-out, display 80ms ease-in-out;
}
```

```css
[data-kraken-section]:not([data-kraken-locked]) .grid-item[data-kraken-element-selected]
.element-resize-container,
[data-kraken-section]:not([data-kraken-locked]) .grid-item:hover .element-resize-container {
  visibility:visible;
  opacity:1;
}

.element-resize-container .resize-corners {
  position: relative;
  z-index: 1;
  display: grid;
  pointer-events: none;
}

.element-resize-container .resize-corner {
  position: absolute;
  grid-area: 1/1/-1/-1;
  padding: 48px;
  pointer-events: all;
}

.element-resize-container .resize-corner-visual-element{
  width: 10px;
  height: 10px;
  background: white;
  outline: 2px solid var(--colors-secondary);
}

.element-resize-container .resize-corner-top-left {
  align-self: start;
  justify-self: start;
  cursor: nwse-resize;
  transform: translate(-50%,-50%);
}

.element-resize-container .resize-corner-top-right {
  align-self: start;
  justify-self: end;
  cursor: nesw-resize;
  transform: translate(50%,-50%);
}

.element-resize-container .resize-corner-bottom-left {
  align-self: end;
  justify-self: start;
  cursor: nesw-resize;
  transform: translate(-50%,50%);
}

.element-resize-container .resize-corner-bottom-right {
  align-self: end;
  justify-self: end;
  cursor: nwse-resize;
  transform: translate(50%,50%);
}

.element-resize-container .resize-corner-horizontal-left {
  align-self: center;
  justify-self: start;
  cursor: ew-resize;
  transform: translateX(-50%);
}
```

```css
.element-resize-container .resize-corner-horizontal-right {
  align-self: center;
  justify-self: end;
  cursor: ew-resize;
  transform: translateX(50%);
}

.element-resize-container .element-outline {
  --outline-style: solid;
  --outline-color: var(--colors-secondary);

  position: absolute;
  top: 0;
  left: 0;
  width: 100%;
  height: 100%;
  pointer-events: none;
  outline: 2px var(--outline-style) var(--outline-color);
}

.section-selector-list [data-preview] section {
  border: 2px solid rgba(var(--colors-primary-light-rgb),0.25);
  border-radius: 0.25em;
  transition-property: transform border;
  transition-duration: 120ms;
  transition-timing-function: ease-in-out;
}

.section-selector-list [data-preview] section:hover {
  border: 2px solid rgba(var(--colors-primary-light-rgb),0.75);
}

.section-selector-nav-list .link {
  opacity:.5;
}

.section-selector-nav-list .link:hoverd {
  opacity:.85;
}

.localnav.localnav-collapsible .localnav-item.four {
  --save-animation-duration:2s;
}

.localnav.localnav-collapsible .localnav-item.four .saveloader,
.localnav.localnav-collapsible .localnav-item.four .savetick,
.localnav.localnav-collapsible .localnav-item.four .savecross {
  position: absolute;
  top:0;
  opacity:0;
}

.localnav.localnav-collapsible .localnav-item.four .saveloader .faicon {
  animation: saveloader_rotate infinite 3s linear;
}

.localnav.localnav-collapsible .localnav-item.four .localnav-item-collapsible-icon .fa-save {
  transition: opacity 500ms ease-in-out;
}

.localnav.localnav-collapsible .localnav-item.four .saveloader {
  transition: opacity 500ms 200ms ease-in-out;
}
```

```
.localnav.localnav-collapsible .localnav-item.four .savetick,
.localnav.localnav-collapsible .localnav-item.four .savecross {
  transition: opacity 500ms ease-in-out;
}

/*ALL OF THE BELOW STATEMENTS BEGIN WITH ".localnav.localnav-collapsible" which has been removed*/

... .localnav-item.four[data-kraken-savestate="saving"] .localnav-item-collapsible-icon .fa-save,
... .localnav-item.four[data-kraken-savestate="saving"] .savetick,
... .localnav-item.four[data-kraken-savestate="success"] .localnav-item-collapsible-icon .fa-save,
... .localnav-item.four[data-kraken-savestate="success"] .saveloader,
... .localnav-item.four[data-kraken-savestate="error"] .localnav-item-collapsible-icon .fa-save,
... .localnav-item.four[data-kraken-savestate="error"] .saveloader,
... .localnav-item.four:not([data-kraken-savestate]) .saveloader,
... .localnav-item.four:not([data-kraken-savestate]) .savetick,
... .localnav-item.four:not([data-kraken-savestate]) .savecross {
  opacity:0;
}

... .localnav-item.four[data-kraken-savestate="saving"] .saveloader,
... .localnav-item.four[data-kraken-savestate="success"] .savetick,
... .localnav-item.four[data-kraken-savestate="error"] .savecross,
... .localnav-item.four:not([data-kraken-savestate]) .localnav-item-collapsible-icon .fa-save {
  opacity:1;
}

.localnav.localnav-collapsible .localnav-item.four .savetick {
  color:var(--colors-success);
}

.localnav.localnav-collapsible .localnav-item.four .savecross {
  color:red;
}

@keyframes saveloader_rotate {
  from {
    transform:rotate(0deg)
  } to {
    transform:rotate(360deg)
  }
}
```

/static/html/sections/

The following code has been reduced down to one example headline file, and the necessary structure required. In reality, there are a lot more files that have not been included. The ellipses denotes where code has been removed.

/static/html/sections/classes

```
headline
...
```

/static/html/sections/headline/

/static/html/sections/headline/css.css

```
[data-preview] .--headline {
  padding-left:16px;
  padding-right:16px;
  padding-top:32px;
  padding-bottom:32px;
  -webkit-user-select: none;
  -moz-user-select: none;
  -ms-user-select: none;
  user-select: none;
  transition: transform 120ms ease-in-out;
  margin-bottom: 32px;
  cursor:pointer;
}

[data-preview] .--headline.--type-4,
[data-preview] .--headline.--type-7,
[data-preview] .--headline.--type-10 {
  min-height: 216px;
}

[data-preview] .--headline.--type-6,
[data-preview] .--headline.--type-11 {
  min-height: 164px;
}

[data-preview] .--headline:hover {
  transform: translateY(-6px);
}

[data-preview] .--headline:first-of-type {
  margin-top: 6px;
}
```

/static/html/sections/headline/files

```
html_element_headline_1.html
...
```

/static/html/sections/headline/html_element_headline_1.html

```
<div class="section-wrapper" data-kraken-section data-preview>

  <section class="--headline --type-1"

    style="--content-width: 1600px;
    --grid-width: 1224px;
    --column-count: 12;
    --content-padding: 16px;
    --cols: 12;
    --rows: 6;
    --width: 1224px;
    --m-rows: 1;
```

```
    --col-gap: 24px;
    --row-gap: 16px;
    --row-size: 48px;
    --column-gap: 24px;
    --block-padding-top: 16px;
    --block-padding: 16px 0 16px 0;
    --block-padding-right: 0;
    --block-padding-bottom: 16px;
    --block-padding-left: 0;
    --m-block-padding: 100px 16px 56px 16px;
    --oldContentWidth: 1400px;
    --grid-gap-history: 16px 24px;
    --half-column-count: 6;">

    <div class="section-background"></div>

    <div class="section-grid">

      <div class="grid-item" data-kraken-element data-kraken-resizable data-kraken-draggable
      data-kraken-editable-style data-kraken-editable-text
        style="--text: center;
        --align: flex-start;
        --justify: center;
        --position: 2/5/2/13;
        --m-element-margin: 0 0 24px 0;
        --element-z-index: 1;
        --element-height: 0px;">

          <div class="text-box grid-item-content">
            <h1 style class="text-content text dark bold large header center">Headline Title</h1>
          </div>

      </div>

      <div class="grid-item" data-kraken-element data-kraken-resizable data-kraken-draggable
      data-kraken-editable-style data-kraken-editable-text
        style="--text: center;
        --align: flex-start;
        --justify: center;
        --position: 4/5/4/13;
        --m-element-margin: 0 0 24px 0;
        --element-z-index: 1;
        --element-height: 0px;">

        <div class="text-box grid-item-content">
          <h2 style class="text-content text primary header">Subtitle text that says things</h2>
        </div>
      </div>

    </div>

  </section>

  <div class="controls">
    <div class="controls-ghost-overlay"></div>
    <!--<div class="controls-resize-handle-container">
      <div class="trigger">
        <div class="resize-handle">
          <i class="fa-solid fa-chevron-up"></i>
          <i class="fa-solid fa-chevron-down"></i>
        </div>
      </div>
    </div>-->
  </div>
```

```
</div>
```

/static/js/

/static/js/auth.js

```
// Function called for each field to make sure it is in the correct format,
// takes a few arguments as flags for what makes it valid
function verifyField(field,fieldName,mustHaveChar=true,minLen=3,canHaveSpace=false,
                     canHaveSpecialChar=true,isPassword=false) {
  // List of special characters for the canHaveSpecialChar flag
  specialChar="%&{}\\<>*?/$!'\":@+`|="

  // Make sure that the input given is a string
  if (typeof field != "string") {throw new Error("HEY! that's not a string?")}

  // Check through all the flags given and throw an appropriate error message if input is invalid
  if (field.length == 0 && mustHaveChar) {return `${fieldName} is not filled out.`}
  if (field.length < minLen) {return `${fieldName} must be greater than ${minLen-1} characters.`}
  if (!canHaveSpace && field.includes(" ")) {return `${fieldName} cannot contain spaces.`}
  if (!canHaveSpecialChar) {
    // Iterate through each character in specialChar to see if its in the input
    // I didn't use regex for this as I wanted to be able to tell the user which char wasn't allowed
    var char;
    for (var i=0;i<specialChar.length;i++) {
      char=specialChar[i]
      if (field.includes(char)) {
        return `${fieldName} cannot contain '${char}'`
      }
    }
  }
  // If the given input is a password
  if (isPassword) {
    // If it doesnt match the given regular expression for password checks
    if (!field.match(
        /(?=.*?[A-Z])(?=.*?[a-z])(?=.*?[0-9])(?=.*?[#?!@$%^&*-_%&{}\\<>*?\/$!'\":@+`|=]).{8,}/)) {
      return `${fieldName} must contain at least 1 of each: uppercase character,
              lowercase character, number, and special character`
    }
  }

  /*
  Regex pattern breakdown
    (?=.*?[A-Z]) = contains an uppercase character
    (?=.*?[a-z]) = contains a lowercase character
    (?=.*?[0-9]) = contains a digit
    (?=.*?[#?!@$%^&*-_%&{}\\<>*?\/$!'\":@+`|=]) = contains a special character
    .{8,} = has a minimum length of 8 and no upper limit
  */

  return ""
}

// Initialise the code for the all seeing eyes to enable viewing the password
function initAllSeeingEye(element,reveal) {
  // Add onclick event to given element (the eye element)
  element.addEventListener("click", e=> {
    // toggle input type of given input between password and text
    reveal.setAttribute('type',reveal.getAttribute('type') === 'password' ? 'text' : 'password');
```

```
    // toggle the fa-eye-slash class for the eye (this sets the icon displayed)
    element.classList.toggle('fa-eye-slash');
  })
}

// Function takes a string and returns a boolean determining whether it is in a valid email format
// using regex
function isEmail(email) {
  return email.match(/^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/)
}

// fetch warning element and disable submit bottom
warningSpan = document.querySelector(".field-container .field-warning")
document.querySelector(".field-submit").disabled = true
```

/static/js/colorConversion.js

```
// Force a number between 0 and 1
function clamp01(val) {
  return Math.min(1, Math.max(0, val));
}

// Force a hex value to have 2 characters
function pad2(c) { return c.length == 1 ? '0' + c : '' + c }

// Assumes: r, g, and b are contained in [0, 255]
// Returns: { h, s, l } in [0,1]
function rgbToHsl(r, g, b) {
  r /= 255, g /= 255, b /= 255;
  var max = Math.max(r, g, b), min = Math.min(r, g, b);
  var h, s, l = (max + min) / 2;

  if (max == min) {
    h = s = 0; // achromatic
  } else {
    var d = max - min;
    s = l > 0.5 ? d / (2 - max - min) : d / (max + min);
    switch (max) {
      case r: h = (g - b) / d + (g < b ? 6 : 0); break;
      case g: h = (b - r) / d + 2; break;
      case b: h = (r - g) / d + 4; break;
    }
    h /= 6;
  }

  return { h: h, s: s, l: l };
}

// Assumes: h, s, and l are contained in the set [0, 1]
// Returns: { r, g, b } in the set [0, 1]
function hslToRgb(h, s, l) {
  var r, g, b;

  if (s == 0) {
    r = g = b = l; // achromatic
  } else {
    var hue2rgb = function hue2rgb(p, q, t) {
      if (t < 0) t += 1;
      if (t > 1) t -= 1;
      if (t < 1 / 6) return p + (q - p) * 6 * t;
      if (t < 1 / 2) return q;
```

```
      if (t < 2 / 3) return p + (q - p) * (2 / 3 - t) * 6;
      return p;
    }

    var q = l < 0.5 ? l * (1 + s) : l + s - l * s;
    var p = 2 * l - q;
    r = hue2rgb(p, q, h + 1 / 3);
    g = hue2rgb(p, q, h);
    b = hue2rgb(p, q, h - 1 / 3);
  }

  return [Math.round(r * 255), Math.round(g * 255), Math.round(b * 255)];
}

// Assumes r, g, and b are contained in the set [0, 255]
// Returns a 3 or 6 character hex
function rgbToHex(r, g, b) {
  var hex = [
    pad2(Math.round(r).toString(16)),
    pad2(Math.round(g).toString(16)),
    pad2(Math.round(b).toString(16))];
  return hex.join("");
}

// Assumes hex is a string of six alphanumeric characters.
// It can have a hashtag at the start as well
// Returns: { r, g, b } in the set [0, 255]
function hexToRgb(hex) {
  if (hex[0] == "#") { hex = hex.slice(1) }
  const r = parseInt(hex.slice(0, 2), 16);
  const g = parseInt(hex.slice(2, 4), 16);
  const b = parseInt(hex.slice(4, 6), 16);
  return { r: r, g: g, b: b };
}

// Assumes color is in a { h, s, l } format.
// Amount is a percentage for how much you decrease it from it's current value
function desaturate(color, amount) {
  amount = (amount === 0) ? 0 : (amount || 10);
  color.s -= ((amount) / 100) * color.s;
  return color
}

// Assumes color is in a { h, s, l } format.
// Amount is a percentage for how much you increase it from it's current value
function saturate(color, amount) {
  amount = (amount === 0) ? 0 : (amount || 10);
  color.s += (amount / 100) * (100 - color.s);
  return color
}

// Assumes color is in a { h, s, l } format.
function greyscale(color) { return desaturate(color, 100) }

// Assumes color is in a { h, s, l } format.
// Amount is a percentage for how much you increase it from it's current value
function lighten(color, amount) {
  amount = (amount === 0) ? 0 : (amount || 10);
  color.l += (amount / 100) * (100 - color.l);
  return color;
}

// Assumes color is in a { h, s, l } format.
// Amount is a percentage for how much you decrease it from it's current value
```

```
function darken(color, amount) {
  amount = (amount === 0) ? 0 : (amount || 10);
  color.l -= ((amount) / 100) * color.l;
  return color
}

// r, g, and b are contained in [0, 255],
// Amount is a percentage for how much you decrease it from it's current value
function brighten(r, g, b, amount) {
  amount = (amount === 0) ? 0 : (amount || 10);
  r = Math.max(0, Math.min(255, r - Math.round(255 * - (amount / 100))));
  g = Math.max(0, Math.min(255, g - Math.round(255 * - (amount / 100))));
  b = Math.max(0, Math.min(255, b - Math.round(255 * - (amount / 100))));
  return { r: r, g: g, b: b }
}
```

/static/js/globalnav-floating-options.js

```
document.getElementById("globalnav-hamburger").addEventListener("click",()=>{
  document.getElementById("globalnav-hamburger").classList.toggle('is-active');

  document.querySelectorAll(".globalnav-floating-options").forEach((e)=>{
    e.classList.toggle("is-active")
  });

  document.querySelectorAll(".globalnav-floating-options-backdrop").forEach((e)=>{
    e.classList.toggle("is-active")
  });
});

document.querySelectorAll(".globalnav-floating-options-backdrop").forEach((e)=>{
  e.addEventListener("click",()=>{
    document.getElementById("globalnav-hamburger").classList.remove('is-active');

    document.querySelectorAll(".globalnav-floating-options").forEach((e)=>{
      e.classList.remove("is-active")
    });

    document.querySelectorAll(".globalnav-floating-options-backdrop").forEach((e)=>{
      e.classList.remove("is-active")
    });
  })
});
```

/static/js/login.js

```
// function called when an input is changed, to check whether all inputs are valid
function verifyAllFields() {
  if (fields["Username"].value.length < 1) {
    warningSpan.innerText = "Username is not filled out"
    document.querySelector(".field-submit").disabled = true
    return
  }

  if (fields["Password"].value.length < 1) {
    warningSpan.innerText = "Password is not filled out"
    document.querySelector(".field-submit").disabled = true
    return
  }
```

```
  // If no errors are called, then enable the button and clear the warning message
  warningSpan.innerText = ""
  document.querySelector(".field-submit").disabled = false
}

// Dictionary of all fields in the form
fields={
  "Username":document.querySelector(".field-option-username .field-input"),
  "Password":document.querySelector(".field-option-password .field-input")
}

initAllSeeingEye(
  document.querySelector(".field-option-password .eye-reveal i"),
  document.querySelector(".field-option-password .field-input"))

document.querySelectorAll(".field-input").forEach(field=>{
  console.log(field)
  field.addEventListener("change",verifyAllFields)
})
```

/static/js/main.js

```
function letterWrapper(query) {
  // takes a html element query and wraps every letter inside with a span
  // <span class="js-wrapped-letter js" style="display:inline-block"></span>
  document.querySelectorAll(query).forEach((item) => {
    item.innerHTML=item.textContent.replace(/\w/g,
    "<span class='js-wrapped-letter js' style='display:inline-block'>$&</span>");
  });
}

function wordWrapper(query) {
  // takes a html element query and wraps every word inside with a span
  // <span class="js-wrapped-word js" style="display:inline-block"></span>
  document.querySelectorAll(query).forEach((item) => {
    item.innerHTML=item.textContent.replace(/\w+/g,
    "<span class='js-wrapped-word js' style='display:inline-block'>$&</span>");
  });
}

function lineWrapper(query) {
  // takes a html element query and wraps every line inside with a span
  // <span class="js-wrapped-line js" style="display:inline-block"></span>
  document.querySelectorAll(query).forEach((item) => {
    item.innerHTML=item.textContent.replace(/.+$/gm,
    "<span class='js-wrapped-line js' style='display:inline-block'>$&</span>");
  });
}

function jsQ(query) {
  // takes a html element query and appends the class "js" to it.
  // Used to recognise which elements have been edited by javascript
  try {
    document.querySelectorAll(query).forEach((item) => {item.classList.add("js")});
  } catch(e) {
    console.log(e);
  }
}

function jsE(el) {
```

```
  // takes a html element object and appends the class "js" to it.
  // Used to recognise which elements have been edited by javascript
  try {
    el.classList.add("js");
  } catch(e) {
    console.log(e);
  }
}

function removeClass(e,a) { e.classList.remove(a) }
function removeClasses(e,a) { a.forEach((x)=>{ e.classList.remove(x) }) }

function addClass(e,a) { e.classList.add(a) }
function addClasses(e,a) { a.forEach((x)=>{ e.classList.add(x) }) }

$(document).ready(function () {
  // reveal the page when it has been loaded
  document.body.classList.remove("visibly-hidden");jsE(document.body);
});
```

/static/js/signup.js

```
// function called when an input is changed, to check whether all inputs are valid
function verifyAllFields() {
  verifyOutput=verifyField(fields["Name"].value,"Name",true,3,true,false)

  if (verifyOutput.length > 0) {
    warningSpan.innerText = verifyOutput
    document.querySelector(".field-submit").disabled = true
    return
  }

  verifyOutput=verifyField(fields["Email"].value,"Email",true,0)

  if (verifyOutput.length > 0) {
    warningSpan.innerText = verifyOutput
    document.querySelector(".field-submit").disabled = true
    return
  }

  // check for email in the correct format
  if (!isEmail(fields["Email"].value)) {
    warningSpan.innerText = "Email is not a valid email address"
    document.querySelector(".field-submit").disabled = true
    return
  }

  verifyOutput=verifyField(fields["Username"].value,"Username",true,3,false,false)

  if (verifyOutput.length > 0) {
    warningSpan.innerText = verifyOutput
    document.querySelector(".field-submit").disabled = true
    return
  }

  console.log("Password")
  verifyOutput=verifyField(fields["Password"].value,"Password",true,8,false,true,true)

  if (verifyOutput.length > 0) {
    warningSpan.innerText = verifyOutput
    document.querySelector(".field-submit").disabled = true
```

```
    return
  }

  // Make sure passwords match
  if (fields["Password"].value!=fields["Repeat Password"].value) {
    warningSpan.innerText = "Passwords do not match"
    document.querySelector(".field-submit").disabled = true
    return
  }

  // If no errors are called, then enable the button and clear the warning message
  warningSpan.innerText = ""
  document.querySelector(".field-submit").disabled = false
}

// Dictionary of all fields in the form
fields={
  "Name":document.querySelector(".field-option-name .field-input"),
  "Email":document.querySelector(".field-option-email .field-input"),
  "Username":document.querySelector(".field-option-username .field-input"),
  "Password":document.querySelector(".field-option-password .field-input"),
  "Repeat Password":document.querySelector(".field-option-password-repeat .field-input")
}

initAllSeeingEye(#
  document.querySelector(".field-option-password .eye-reveal i"),
  document.querySelector(".field-option-password .field-input"))

initAllSeeingEye(
  document.querySelector(".field-option-password-repeat .eye-reveal i"),
  document.querySelector(".field-option-password-repeat .field-input"))

document.querySelectorAll(".field-input").forEach(field=>{
  console.log(field)
  field.addEventListener("change",verifyAllFields)
})
```

/static/js/site-create-options-1.js

```
function toggleOptions() {
  lightOptions.classList.add("visibly-hidden");darkOptions.classList.add("visibly-hidden");
  if (lightModeSelected) { lightOptions.classList.remove("visibly-hidden") }
  else { darkOptions.classList.remove("visibly-hidden") }
}

function setColor(k,v) { colors[k]=v }

function setDarkMode() {
  setColor["default-background", colors["dark"]]; setColor["default-text", colors["light"]]
}

function setLightMode() {
  setColor["default-background", colors["light"]]; setColor["default-text", colors["dark"]]
}

var btnLight=document.getElementById("new_site_lightModeToggle");
var btnDark=document.getElementById("new_site_darkModeToggle");
var lightModeSelected=true;
var lightOptions=document.querySelector(".new-site-form.two .light-color-options");
var darkOptions=document.querySelector(".new-site-form.two .dark-color-options");
var stored=document.getElementById("color-output")
```

```
var primaryColorPicker=document.getElementById("new_site_colors_primary_picker");
var secondaryColorPicker=document.getElementById("new_site_colors_secondary_picker");
var accentColorPicker=document.getElementById("new_site_colors_accent_picker");

// the primary color should be quite visible against the selected theme. Dark against light etc
// the secondary color should be slightly less so, and a rotation around the color wheel.
// Red against purple etc
// the accent color should be lighter than both the primary and secondary colors,
// but not very overpowering

// there will be an option to change the starting and ending lightness of the grey colors
// and also to tint them warm or cold slightly
// you wont be able to edit the specific colors of the greys though

// the main colors will have light and dark versions that are generated by the code
// their lightness can be editied but the dark one cant be lighter than 10% darker than
// the original etc
// ie there will be bounds on how light and dark the colors can go

// the default background and text colors are defined by whether the document is set to light or
// dark mode
// in the ui design, the defaults can change based on a parent element having
// .style-light or .style-dark

// the light and dark colors can be changed, but must be kept within bounds
// (eg #303030 and # f0f0f0) which havent been set
// the darkest and lightest greys cannot go darker or lighter than these colors

function updateStored() {
  var out="";var keys=Object.keys(colors);
  for (var i=0; i<keys.length; i++) { out=out+keys[i]+":"+colors[keys[i]]+"," }
  out=out.slice(0,out.length-1)
  stored.value=out
}

function updateLightDarkVariables() {
  var val = 100-lightDarkBoundsSlider.value;

  var newColor = darken({h:0,s:0,l:100},val/12);
  newColor = hslToRgb(newColor.h,newColor.s,newColor.l);
  newColor = rgbToHex(newColor.r,newColor.b,newColor.g);
  setColor("light","#"+newColor);

  var newColor = lighten({h:0,s:0,l:0},val/8);
  newColor = hslToRgb(newColor.h,newColor.s,newColor.l);
  newColor = rgbToHex(newColor.r,newColor.b,newColor.g);
  setColor("dark","#"+newColor);

  updateStored()
}

function updateLightDarkDisplay() {
  colorDisplay["light"][0].style.backgroundColor=colors["light"]
  colorDisplay["light"][1].innerHTML=colors["light"]
  colorDisplay["dark"][0].style.backgroundColor=colors["dark"]
  colorDisplay["dark"][1].innerHTML=colors["dark"]

  updateStored()
}

function updateColorVariables() {
  var changePercent = 20
```

```
  for (var color of ["primary","secondary","accent"]) {
    console.log(color)

    var newColor = hexToRgb(colors[color])

    // console.log("Color as RGB:")
    // console.log(newColor)

    newColor = rgbToHsl(newColor.r,newColor.g,newColor.g)

    // console.log("Color as HSL:")
    // console.log(newColor)

    newColor = darken(newColor,changePercent)

    // console.log("Darker Color as HSL:")
    // console.log(newColor)

    newColor = hslToRgb(newColor.h,newColor.s,newColor.l);

    // console.log("Darker Color as RGB:")
    // console.log(newColor)

    newColor = rgbToHex(newColor[0],newColor[1],newColor[2]);

    // console.log("Darker Color as Hex:")
    // console.log(newColor)

    setColor(color+"-dark","#"+newColor)

    newColor = hexToRgb(colors[color])
    newColor = rgbToHsl(newColor.r,newColor.g,newColor.g)

    // console.log("Color as HSL:")
    // console.log(newColor)

    newColor = lighten(newColor,changePercent)

    // console.log("Lighter Color as HSL:")
    // console.log(newColor)

    newColor = hslToRgb(newColor.h,newColor.s,newColor.l);

    // console.log("Lighter Color as RGB:")
    // console.log(newColor)

    newColor = rgbToHex(newColor[0]/255,newColor[1]/255,newColor[2]/255);

    // console.log("Lighter Color as Hex:")
    // console.log(newColor)


    setColor(color+"-light","#"+newColor)

    console.log(colors[color+"-dark"])
    console.log(colors[color+"-light"])

  }

  updateStored()

}

function updateColorDisplays() {
```

```
  colorDisplay["primary"][0].style.backgroundColor=colors["primary"]
  colorDisplay["primary"][1].innerHTML=colors["primary"]
  colorDisplay["primary-dark"][0].style.backgroundColor=colors["primary-dark"]
  colorDisplay["primary-light"][0].style.backgroundColor=colors["primary-light"]

  colorDisplay["secondary"][0].style.backgroundColor=colors["secondary"]
  colorDisplay["secondary"][1].innerHTML=colors["secondary"]
  colorDisplay["secondary-dark"][0].style.backgroundColor=colors["secondary-dark"]
  colorDisplay["secondary-light"][0].style.backgroundColor=colors["secondary-light"]

  colorDisplay["accent"][0].style.backgroundColor=colors["accent"]
  colorDisplay["accent"][1].innerHTML=colors["accent"]
  colorDisplay["accent-dark"][0].style.backgroundColor=colors["accent-dark"]
  colorDisplay["accent-light"][0].style.backgroundColor=colors["accent-light"]

  updateStored()
}

function updateDisplays() {
  updateStored()
  updateColorDisplays()
  updateLightDarkDisplay()
}

function updatePrimaryColorDiv() {
  setColor("primary",primaryColorPicker.value)

  updateColorVariables()
  updateColorDisplays()
}

function setLightMode() {
  document.body.removeAttribute("data-kraken-darkmode")
  updateLightDarkDisplay()
}

function setDarkMode() {
  document.body.setAttribute("data-kraken-darkmode","")
  updateLightDarkDisplay()
}

var options = [
  "light","dark",
  "primary","primary-dark","primary-light",
  "secondary","secondary-dark","secondary-light",
  "accent","accent-dark","accent-light",
  "grey-100","grey-200","grey-300","grey-400",
  "grey-500","grey-600","grey-700","grey-800",
  "grey-900"
]

var defaultColors = {
  "light":"#ffffff", "dark":"#000000",
  "primary":"#e63946", "primary-dark":"", "primary-light":"",
  "secondary":"#457b9d","secondary-dark":"","secondary-light":"",
  "accent":"#a8dadc","accent-dark":"","accent-light":"",
  "grey-100":"#303030", "grey-200":"#474747", "grey-300":"#5e5e5e", "grey-400":"#757575",
  "grey-500":"#8c8c8c", "grey-600":"#a3a3a3", "grey-700":"#bababa", "grey-800":"#d1d1d1",
  "grey-900":"#e8e8e8"
}

var colors = defaultColors

var prefix = ".new-site-form.two .color-display-container"
```

```
var prefix2 = " .main-color-display "

var colorDisplay = {
  "light":[
    document.querySelector(prefix+" .light-dark-display .light-color"),
    document.querySelector(prefix+" .light-dark-display .light-color .color-code")
  ],

  "dark":[
    document.querySelector(prefix+" .light-dark-display .dark-color")
    document.querySelector(prefix+" .light-dark-display .dark-color .color-code")
  ],

  "primary":[
    document.querySelector(prefix+prefix2+".primary-color .color-triple-card-main")
    document.querySelector(prefix+prefix2+".primary-color .color-triple-card-main .color-code")
  ],

  "primary-dark":[
    document.querySelector(prefix+prefix2+".primary-color .color-triple-card-sub.one")
  ],

  "primary-light":[
    document.querySelector(prefix+prefix2+".primary-color .color-triple-card-sub.two")
  ],

  "secondary":[
    document.querySelector(prefix+prefix2+".secondary-color .color-triple-card-main")
    document.querySelector(prefix+prefix2+".secondary-color .color-triple-card-main .color-code")
  ],

  "secondary-dark":[
    document.querySelector(prefix+prefix2+".secondary-color .color-triple-card-sub.one")
  ],

  "secondary-light":[
    document.querySelector(prefix+prefix2+".secondary-color .color-triple-card-sub.two")
  ],

  "accent":[
    document.querySelector(prefix+prefix2+".accent-color .color-triple-card-main")
    document.querySelector(prefix+prefix2+".accent-color .color-triple-card-main .color-code")
  ],

  "accent-dark":[
    document.querySelector(prefix+prefix2+".accent-color .color-triple-card-sub.one")
  ],

  "accent-light":[
    document.querySelector(prefix+prefix2+".accent-color .color-triple-card-sub.two")
  ],

  "grey-100":[
    document.querySelector(prefix+" .grey-display .color-column.g100")
    document.querySelector(prefix+" .grey-display .color-column.g100 .color-code")
  ],

  "grey-200":[
    document.querySelector(prefix+" .grey-display .color-column.g200")
    document.querySelector(prefix+" .grey-display .color-column.g200 .color-code")
  ],

  "grey-300":[
    document.querySelector(prefix+" .grey-display .color-column.g300")
```

```
    document.querySelector(prefix+" .grey-display .color-column.g300 .color-code")
  ],

  "grey-400":[
    document.querySelector(prefix+" .grey-display .color-column.g400")
    document.querySelector(prefix+" .grey-display .color-column.g400 .color-code")
  ],

  "grey-500":[
    document.querySelector(prefix+" .grey-display .color-column.g500")
    document.querySelector(prefix+" .grey-display .color-column.g500 .color-code")
  ],

  "grey-600":[
    document.querySelector(prefix+" .grey-display .color-column.g600")
    document.querySelector(prefix+" .grey-display .color-column.g600 .color-code")
  ],

  "grey-700":[
    document.querySelector(prefix+" .grey-display .color-column.g700")
    document.querySelector(prefix+" .grey-display .color-column.g700 .color-code")
  ],

  "grey-800":[
    document.querySelector(prefix+" .grey-display .color-column.g800")
    document.querySelector(prefix+" .grey-display .color-column.g800 .color-code")
  ],

  "grey-900":[
    document.querySelector(prefix+" .grey-display .color-column.g900")
    document.querySelector(prefix+" .grey-display .color-column.g900 .color-code")
  ],
}

var lightDarkBoundsSlider = document.getElementById("new_site_colors_light_dark_bounds_slider");

lightDarkBoundsSlider.addEventListener("mouseup",()=>{updateLightDarkVariables();updateDisplays()})
primaryColorPicker.addEventListener("change",()=>{updatePrimaryColorDiv();updateStored()})

updateStored()
```

/static/js/site-create-options-2.js

```
var textOptions = document.querySelectorAll(".new-site-form.three .text-option");

textOptions.forEach((e)=>{
  e.addEventListener("click",()=>{
    textOptions.forEach((f)=>{
      f.classList.remove("active")
      f.querySelector(".text-option-list").name="new_site_font_face_list_inactive"
    });
    e.classList.add("active")
    e.querySelector(".text-option-list").name="new_site_font_face_list_active"
  });
});
```

/static/js/site-create-options-3.js

```
function revealOption(e) {
  e.style.opacity=1;e.style.visibility="visible"
}

function hideFromLeft() {
  fromLeftOptions.style.opacity=0;fromLeftOptions.style.visibility="hidden"
}

function makeSlide() {
  document.querySelectorAll(buttonsPrefix+":not(.slide-or-static) .btn").forEach((e)=>{
    e.classList.add("slide")
  })
}

function makeStatic() {
  document.querySelectorAll(buttonsPrefix+":not(.slide-or-static) .btn").forEach((e)=>{
    e.classList.remove("slide")
  })
}

function makeSquare() {
  document.querySelectorAll(buttonsPrefix+":not(.corner-options) .btn").forEach((e)=>{
    removeClasses(e,["rounded","pill"])
    addClass(e,"square")
  })
}

function makeRounded() {
  document.querySelectorAll(buttonsPrefix+":not(.corner-options) .btn").forEach((e)=>{
    removeClasses(e,["square","pill"])
    addClass(e,"rounded")
  })
}

function makePill() {
  document.querySelectorAll(buttonsPrefix+":not(.corner-options) .btn").forEach((e)=>{
    removeClasses(e,["rounded","square"])
    addClass(e,"pill")
  })
}

function makeThin() {
  document.querySelectorAll(buttonsPrefix+".left-or-right .btn").forEach((e)=>{
    e.classList.add("thin")
  })
}

function makeThick() {
  document.querySelectorAll(buttonsPrefix+".left-or-right .btn").forEach((e)=>{
    e.classList.remove("thin")
  })
}

function enableSubmit() {
  document.querySelector(".new-site-form.four .field-submit").removeAttribute("disabled")
}

function disableSubmit() {
  document.querySelector(".new-site-form.four .field-submit").setAttribute("disabled","")
}
```

```
function updateStored() {
  var out = "sliding:" + slidingPreference
            + ",cornerType:" + squaredPreference
            + ",thin:" + thinPreference
            + ",fromLeft:" + fromLeftPreference

  stored.value = out
}

var slidingPreference = null;
var squaredPreference = null
var thinPreference = null;
var fromLeftPreference = null;
var canOpenFromLeft = false;
var hasOpenFromLeft = false;

var slidingOptions = document.querySelector(".new-site-form.four .button-option.slide-or-static")
var squaredOptions = document.querySelector(".new-site-form.four .button-option.corner-options")
var thinOptions = document.querySelector(".new-site-form.four .button-option.thin-or-large")
var fromLeftOptions = document.querySelector(".new-site-form.four .button-option.left-or-right")
var buttonsPrefix = ".new-site-form.four .button-option"
var stored=document.getElementById("style-option-output")

slidingOptions.querySelector(".option-true").addEventListener("click",(e)=>{
  slidingPreference=true;
  revealOption(squaredOptions);
  makeSlide();
  canOpenFromLeft=true;
  if(hasOpenFromLeft){revealOption(fromLeftOptions)}
  if(thinPreference!=null){revealOption(fromLeftOptions)}
  if(fromLeftPreference==null){disableSubmit()}
  updateStored();
})
slidingOptions.querySelector(".option-false").addEventListener("click",(e)=>{
  slidingPreference=false;
  revealOption(squaredOptions);
  makeStatic();
  canOpenFromLeft=false;
  hideFromLeft();
  updateStored();
})

squaredOptions.querySelector(".square").addEventListener("click",(e)=>{
  squaredPreference="square";
  revealOption(thinOptions);
  makeSquare();
  updateStored();
})
squaredOptions.querySelector(".rounded").addEventListener("click",(e)=>{
  squaredPreference="rounded";
  revealOption(thinOptions);
  makeRounded();
  updateStored();
})
squaredOptions.querySelector(".pill").addEventListener("click",(e)=>{
  squaredPreference="pill";
  revealOption(thinOptions);
  makePill();
  updateStored();
})

thinOptions.querySelector(".option-true").addEventListener("click",(e)=>{
  thinPreference=true;
  if(canOpenFromLeft){revealOption(fromLeftOptions);hasOpenFromLeft=true};
```

```
  makeThin();
  if(!(slidingPreference)){enableSubmit()}
  updateStored();
})
thinOptions.querySelector(".option-false").addEventListener("click",(e)=>{
  thinPreference=false;
  if(canOpenFromLeft){revealOption(fromLeftOptions);hasOpenFromLeft=true};
  makeThick();
  if(!(slidingPreference)){enableSubmit()}
  updateStored();
})

fromLeftOptions.querySelector(".option-true").addEventListener("click",(e)=>{
  fromLeftPreference=true;
  enableSubmit();
  updateStored();
})
fromLeftOptions.querySelector(".option-false").addEventListener("click",(e)=>{
  fromLeftPreference=false;
  enableSubmit();
  updateStored();
})
```

/static/js/site-create.js

```
String.prototype.replaceAt = function(index, replacement) {
  return this.substr(0, index) + replacement + this.substr(index + replacement.length);
}

function checkFormSubmitButton() {
  if (!(formName.getAttribute("data-form-input-display") == "success" ||
        formName.getAttribute("data-form-input-display") == "warning")) {

          formSubmit.setAttribute("disabled","")
          return
  }

  if (!(formPrivacy1.checked) && !(formPrivacy2.checked)) {
    formSubmit.setAttribute("disabled","")
    return
  }

  formSubmit.removeAttribute("disabled")
  return
}

function editFormMessageSiteNameWarning(val) {
  messageContainer.classList.remove("visibly-hidden")
  newInner=val.toLowerCase()
  for (var i=0; i<newInner.length; i++) {
    var letter=newInner[i];
    if (!(allowedChars.includes(letter))) {
      newInner=newInner.replaceAt(i,"-")
    }
  }
  if (hasRepeatedDashes(newInner)) { newInner=replaceRepeatedDashes(newInner) }
  messageSpan.innerHTML=newInner
}

function hideFormMessage() {
  messageContainer.classList.add("visibly-hidden")
```

```
  messageSpan.innerHTML=""
}

function hasRepeatedDashes(val) {
  for (var i=0;i<val.length;i++) {
    if (val[i] == "-" && val[i+1] == "-") {
      return true
    }
  }
  return false
}

function replaceRepeatedDashes(val) {
  return listToStr(replaceRepeatedDashesRecursion(val.split("")))
}

function listToStr(lst) {
  var out=""
  for (var i=0;i<lst.length;i++) {
    out=out+lst[i]
  }
  return out
}

function replaceRepeatedDashesRecursion(val) {
  for (var i=0;i<val.length;i++) {
    if (val[i] == "-" && val[i+1] == "-") {
      val.splice(i+1,1)
      val=replaceRepeatedDashesRecursion(val)
    }
  }
  return val
}

function verifyNameField() {
  var nameInput = document.getElementById("new_site_name");
  var val = nameInput.value;

  hideFormMessage()

  if (val.length < 1) { return "inactive" }
  if (val.length < 4) { return "danger" }

  var check=true
  for (var i=0;i<val.length;i++) {
      var letter = val[i]
      if (requiredChars.includes(letter)) { check=false }
  }

  if (check) { return "danger" }

  var sitenames = ["helloworld"]

  if (flashedSiteNames.includes(val)) {
    editFormMessage("A site with this name already exists!")
    return "danger"
  }

  for (var i=0;i<val.length;i++) {
      var letter = val[i]
      if (!(allowedChars.includes(letter))) {
        editFormMessageSiteNameWarning(val)
        return "warning"
      }
```

```
  }

  if (hasRepeatedDashes(val)) {
    editFormMessageSiteNameWarning(replaceRepeatedDashes(val))
    return "warning"
  }

  hideFormMessage()
  return "success"
}

var requiredChars = "qwertyuiopasdfghjklzxcvbnm1234567890"
var allowedChars = "qwertyuiopasdfghjklzxcvbnm-._1234567890";
var formSubmit = document.getElementById("new_site_form_submit");
var formName = document.getElementById("new_site_name");
var formDesc = document.getElementById("new_site_desc");
var formPrivacy1 = document.getElementById("new_site_privacy_visible");
var formPrivacy2 = document.getElementById("new_site_privacy_hidden");

var messageContainer = document.querySelector(
  ".new-site-form .form-input-container.one .message-container");

var messageSpan = document.querySelector(
  ".new-site-form .form-input-container.one .message-container .message-container-jsedit");

formName.addEventListener("keyup",(event) => {
  formName.setAttribute("data-form-input-display",verifyNameField())
  checkFormSubmitButton();
})

formPrivacy1.addEventListener("click",checkFormSubmitButton)
formPrivacy2.addEventListener("click",checkFormSubmitButton)

formSubmit.addEventListener("click",(event)=>{
  if (!(formSubmit.getAttribute("disabled"))) {
    formSubmit.children[0].innerHTML = "CREATING SITE..."
  }
})
```

/static/js/site-edit.js

Due to the size of the file (1228 SLOC), `#region` and `#endregion` are used to divide blocks of code so that the file can be easily collapsed into labelled sections.

```
/* --------------- CODE FROM OTHER SOURCES --------------- */
// #region

// source https://stackoverflow.com/questions/14766951/
// /transform-numbers-to-words-in-lakh-crore-system Juan Gaitan

var num = `zero one two three four five six seven eight nine ten eleven twelve thirteen fourteen
          fifteen sixteen seventeen eighteen nineteen`.split(" ");
var tens = "twenty thirty forty fifty sixty seventy eighty ninety".split(" ");

function number2words(n){
  if (n < 20) return num[n];
  var digit = n%10;
  if (n < 100) return tens[~~(n/10)-2] + (digit? "-" + num[digit]: "");
  if (n < 1000) return num[~~(n/100)] +" hundred" + (n%100 == 0? "": " and " + number2words(n%100));
  return number2words(~~(n/1000)) + " thousand" + (n%1000 != 0? " " + number2words(n%1000): "");
```

```
}

// source https://www.codegrepper.com/profile/code-grepper

function capitalizeWords(string) {
  return string.replace(/(?:^|\s)\S/g, function(a) { return a.toUpperCase() })
};

// source https://www.grepper.com/answers/17868/javascript+sleep

const sleep = (milliseconds) => {
  return new Promise(resolve => setTimeout(resolve, milliseconds))
}

// endsource

// #endregion

/* --------------- VARIABLES AND CONSTANTS --------------- */
// #region

var builder = document.getElementById("contains_site")
var saveSiteBtn = document.getElementById("localnav_save")
var currentsitename = "Home"

// add section modal variables

var addSectionBtn = document.getElementById("localnav_add_section_btn");
var addSectionModal_selectedNavItem="one";
var addSectionModal_selectedNavItemInt=1;
var addSectionModal_container=
  document.querySelector(".application-content .section-selector-container");

var addSectionModal_list=document.getElementById("section_selector_list");
var addSectionModal_styleElement=document.getElementById("section_selector_style")
var addSectionModal_nav=document.getElementById("section_selector_nav");
var addSectionModal_navItem=
  `<li class="section-selector-nav-item [i]">
     <a class="link unformatted" id="section_selector_nav_[n1]">
       <span class="text bold">[n2]</span>
     </a>
   </li>`;

var addSectionModal_data = {}
var addSectionModal_headers = []

var addSectionModal_getAllTemplates = function() { return addSectionModal_list.childNodes }

// section edit actions

var sectionEditActions = document.querySelector(".section-edit-actions")

// abstract variables

var user
var site
var siteinfo
var currentpage
var selectedSection
var selectedElement

// query prefixes

queryprefix_sitecontainer = "#contains_site"
```

```
queryprefix_notlocked = `:not([data-kraken-locked])`
queryprefix_locked = `[data-kraken-locked]`

queryprefix_section = `${queryprefix_sitecontainer} [data-kraken-section]:not([data-preview])`
queryprefix_sectionselected = `${queryprefix_section}[data-kraken-section-selected]`

queryprefix_element = `${queryprefix_section} [data-kraken-element]`
queryprefix_elementselected = `${queryprefix_element}[data-kraken-element-selected]`
queryprefix_elementresizable=`${queryprefix_element}${queryprefix_notlocked}[data-kraken-resizable]`
queryprefix_elementdraggable=`${queryprefix_element}${queryprefix_notlocked}[data-kraken-draggable]`

// #endregion

/* ------------------- QUERY FUNCTIONS ------------------- */
// #region

function siteroot() { return document.querySelector("[data-content-parentview]")}
function resizable() { return document.querySelectorAll(queryprefix_elementresizable) }
function draggable() { return document.querySelectorAll(queryprefix_elementdraggable) }
function sections() { return document.querySelectorAll(queryprefix_section) }
function elements() { return document.querySelectorAll(queryprefix_element) }

// #endregion

/* ------------ ADDING ELEMENTS AND LISTENERS ------------ */
// #region

// appends a section to the end of the editor
function addSection(html) {

  try {siteroot().insertAdjacentHTML("beforeend",html)}
  catch {siteroot().innerHTML += html}

  sectionEventListeners()
  elementEventListeners()

  createResizeBoxes()
}

// onclick function for section event listeners
function sectionOnClick(e) {
  var x = e.currentTarget
  //while (!(x.hasAttribute("data-kraken-section"))) x = x.parentElement

  clearSelectedSections()
  x.setAttribute("data-kraken-section-selected","")
  sectionEditActions.classList.add("shown")

  sectionEditActions.style.top =
    `${x.getBoundingClientRect().top-siteroot().getBoundingClientRect().top}px`

  if (x.hasAttribute("data-kraken-locked")) sectionEditActions_setlock_locked()
  else sectionEditActions_setlock_unlocked()

  selectedSection=x;
}

// onclick function for element event listener
function elementOnClick(e) {
  clearSelectedElements()

  setSelectedElement(e.currentTarget)
  createResizeCornerEventListeners()
```

```
  createDragEventListeners()
}

// creates the event listeners for all the sections
function sectionEventListeners() {
  clearSelectedSections()

  for (var section of sections()) {
    section.removeEventListener("click",sectionOnClick)
    section.addEventListener("click",sectionOnClick)
  }
}

// creates the event listeners for every element
function elementEventListeners() {
  clearSelectedElements()

  for (var element of elements()) {
    element.removeEventListener("click",elementOnClick)
    element.addEventListener("click",elementOnClick)
  }
}

// #endregion

/* ------------ SELECTED ELEMENTS AND SECTIONS ----------- */
// #region

// removes the data-kraken-section-selected tag from every section
function clearSelectedSections() {
  for (var section of sections()) section.removeAttribute("data-kraken-section-selected")
  sectionEditActions.classList.remove("shown")
}

// removes the data-kraken-element-selected tag from every element
function clearSelectedElements() {
  for (var element of elements()) element.removeAttribute("data-kraken-element-selected")
}

// #endregion

/* ------------------ ADD SECTION MODAL ------------------ */
// #region

// removes all sections from the add section modal
function addSectionModal_clearCurrentPreviews() {
  // remove each template section if it is not the style element
  addSectionModal_getAllTemplates().forEach((e)=>{
    if (!(e == addSectionModal_styleElement)) e.remove()
  })
}

// populates the add section modal with templates based on a dictionary of data
function addSectionModal_populate(data) {
  addSectionModal_styleElement.innerHTML = data.css

  addSectionModal_clearCurrentPreviews()

  var i=1

  for (var section of data.sections) {
    //addSectionModal_list.innerHTML += section
    addSectionModal_list.insertAdjacentHTML("beforeend",section)
```

```
    var e = document.querySelector(".--headline.--type-"+i)

    e.parentElement.addEventListener("click",(e) => {
      addSection(stripPreviewTags(e.currentTarget.outerHTML))
      addSectionModal_hide()
    })

    i++
  }
}

// add a title to the local add section modal navigation bar
function addSectionModal_setNavbar(text) {
  // format the string based on the content of the string
  out=addSectionModal_navItem
      .replace("[i]",number2words(1)) // TODO work out the correct number
      .replace("[n1]",text.replace(" ",""))
      .replace("[n2]",capitalizeWords(text))

  // append to the local navbar
  addSectionModal_nav.querySelector("ul.section-selector-nav-list").innerHTML+=out;
}

// displays the add section modal
function addSectionModal_show() {
  addSectionModal_container.classList.add("shown");
  document.querySelector(".lightbox-mask").classList.add("shown")
}

// hides the add section modal
function addSectionModal_hide() {
  document.querySelector(".application-content .section-selector-container")
    .classList.remove("shown")

  document.querySelector(".lightbox-mask").classList.remove("shown")

  addSectionModal_clearCurrentPreviews()
}

// loads all section templates from the /static/html/ directory and stores them to the
// dictionary addSectionModal_data, along with the types of templates being stored in
// the list addSectionModal_headers
// The function is asynced to allow the use of await functions to fetch the content of the file
async function addSectionModal_loadsections() {

  // load the classes file, which contains all of the template types
  // and store them in the headers list
  await fetch("../../../static/html/sections/classes")
    // throw an error if there is an issue with the response
    .then( response => {
      if (!response.ok) { throw new Error(`HTTP error: ${response.status}`) }
      return response.text();
    })
    .then( text => {
      for (line of text.split("\n")) {
        addSectionModal_setNavbar(line)
        addSectionModal_headers.push(line)
      }
    })

  var path
  var files

  // iterate through all the section types
```

```
  for (var header of addSectionModal_headers) {

    // add the section type to the data
    addSectionModal_data[header] = {css:"",sections:[]}

    // set the root URL for the section type
    path=`../../../static/html/sections/${header}`

    // fetch the custom CSS file for this type of headers
    await fetch(path+"/css.css")
      // throw an error if there is an issue with the response
      .then( response => {
        if (!response.ok) { throw new Error(`HTTP error: ${response.status}`) }
        return response.text();
      })
      // add to the data dictionary
      .then( text => addSectionModal_data[header].css=text )

    // fetch the file list
    await fetch(path+"/files")
      // throw an error if there is an issue with the response
      .then( response => {
        if (!response.ok) { throw new Error(`HTTP error: ${response.status}`) }
        return response.text();
      })
      // split the file into lines and filter out any empty lines
      .then( text => files=text.split(/[\r\n]+/g)
                               .filter(function(value, index, arr){ return value != "" }) )

    // iterate through all given files
    for (var file of files) {

      // fetch the template HTML
      await fetch(path+"/"+file)
        // throw an error if there is an issue with the response
        .then( response => {
          if (!response.ok) { throw new Error(`HTTP error: ${response.status}`) }
          return response.text();
        })
        // add the template to the sections list
        .then( text => addSectionModal_data[header].sections.push(text))

    }
  }
}

// removes all data-preview tags from a string
function stripPreviewTags(html) {
  return html.replaceAll("data-preview"," ")
}

// event listener for the add section button, uses addSectionModal_show &
addSectionBtn.addEventListener("click",() => {
  // if the add section modal is not already shown
  if (!(addSectionModal_container.classList.contains("shown"))) {
    addSectionModal_show()

    addSectionModal_selectedNavItem="one";
    addSectionModal_selectedNavItemInt=1;

    addSectionModal_nav.querySelector(
      `ul.section-selector-nav-list li a.link`)
      .style.opacity = 0.75;
```

```
    addSectionModal_nav.querySelector(
      `ul.section-selector-nav-list li.${addSectionModal_selectedNavItem} a.link`)
      .style.opacity = 1;

    addSectionModal_clearCurrentPreviews()

    addSectionModal_populate(
      addSectionModal_data[addSectionModal_headers[addSectionModal_selectedNavItemInt-1]]
    )

  }
});

// event listener for the close x in the add section modal
document.querySelector(".application-content .section-selector-exit-btn")
  .addEventListener("click",addSectionModal_hide);

// event listener for the dark backround of the modals
document.querySelector(".lightbox-mask").addEventListener("click",addSectionModal_hide);

// #endregion

/* ---------------- SECTION EDIT ACTIONS ----------------- */
// #region

// set the locked icon in the lock button
function sectionEditActions_setlock_locked() {
  document.querySelector("#section_edit_action_lock i").classList.add("fa-unlock")
}

// set the unlocked icon in the lock button
function sectionEditActions_setlock_unlocked() {
  document.querySelector("#section_edit_action_lock i").classList.remove("fa-unlock")
}

// check whether there is a section selected - if not, hide the actions
function sectionEditActions_checkVisibility() {
  try {
    if (!(document.querySelectorAll("[data-kraken-section-selected]").length > 0)) {
      sectionEditActions.classList.remove("shown")
    }
  } catch (e) {
    sectionEditActions.classList.remove("shown")
  }
}

// section action event listeners
document.getElementById("section_edit_action_settings").addEventListener("click",e=>{
  console.log("settings")
})

document.getElementById("section_edit_action_duplicate").addEventListener("click",e=>{
  addSection(selectedSection.outerHTML)
})

document.getElementById("section_edit_action_delete").addEventListener("click",e=>{
  selectedSection.parentElement.removeChild(selectedSection)
})

// lock event listener requires changing of the symbol
// depending on whether or not the section is locked
document.getElementById("section_edit_action_lock").addEventListener("click",e=>{
  console.log(selectedSection)
  if (selectedSection.hasAttribute("data-kraken-locked")) {
```

```
    selectedSection.removeAttribute("data-kraken-locked","")
    sectionEditActions_setlock_unlocked()
  } else {
    selectedSection.setAttribute("data-kraken-locked","")
    sectionEditActions_setlock_locked()
  }
})

// #endregion

/* --------- STARTING THE EDITOR AND LOADING HTML -------- */
// #region

// call on the page being loaded
// loads HTML from the server, starts all required event listeners,
// creates any required resize boxes, and starts the autosave timer
async function run() {
  // get the username and sitename from the site url
  user = window.location.pathname.split("/")[1]
  site = window.location.pathname.split("/")[2]

  // get the siteinfo dictionary from the server
  siteinfo = await fetch(`../../../static/data/userData/${user}/sites/${site}/siteDat.json`)
                       .then(response => { return response.json(); })

  // get the current site page, defaults to "Home" if nothing is set
  currentpage = siteinfo["pages"][currentsitename]

  // load the current page
  await loadcurrentpage(user,site,currentpage)

  // create any required event listeners
  sectionEventListeners()
  elementEventListeners()

  // create any required resize boxes for elements and grid previews for sections
  createResizeBoxes()

  // start the autosave process
  saveTimeout()

  addSectionModal_loadsections()

  // event listener for the add section button, uses addSectionModal_show &
  saveSiteBtn.addEventListener("click",saveContent);

  // constant checks for stuff
  document.querySelector(".site-builder").addEventListener("click",() => {
    sectionEditActions_checkVisibility()
    createGridPreviews()
  })
}

// loads the current page into the editor from a set criteria
async function loadcurrentpage(user,site,currentpage) {
  return sethtml(
    await loadhtmlfile(
      `../../../static/data/userData/${user}/sites/${site}/files/${currentpage}`
    )
  )
}

// loads a file from a path
async function loadhtmlfile(path) {
```

```
  return await fetch(path)
    .then( response => {
      if (!response.ok) { throw new Error(`HTTP error: ${response.status}`) }
      return response.text();
    })
    .then( text => { return text })
}

// sets the content of the editor
function sethtml(html) {
  builder.innerHTML=html;
}

// #endregion

/* ----------------- SAVING USER CONTENT ----------------- */
// #region

// await function used to ensure that multiple saves are not occuring at the same time
async function waitForNotSaving() {
  var _ = document.querySelector(".localnav.one .localnav-item.four")
  while (_.hasAttribute("data-kraken-savestate")) { await sleep(100) }
  return true
}

// sends a request to the server to save the content of the editor to the server file storage
async function saveContent() {
  // remove all existing timeouts to reduce conflicts
  clearTimeout(saveContent)

  // wait for the current save to finish, if running
  await waitForNotSaving()

  console.log("Saving...")

  // start the save animation
  showSaveLoader()

  var url = `../../../
            ${window.location.pathname.split("/")[1]}/
            ${window.location.pathname.split("/")[2]}/save/
            ${siteinfo["pages"][currentsitename]}`,

  req = $.ajax({
    type:"post",
    url:url,
    data:{content:siteroot().outerHTML},
    datatype:"json",
    error: function(xhr,ajaxOptions,thrownError) { hideSaveLoader(false) },
    success: function(response) { hideSaveLoader(true) },
  })

  // create the new timeout
  saveTimeout()
}

// starts the timeout for the saveContent function
function saveTimeout() {
  clearTimeout(saveContent) // clear any previous timeouts
  setTimeout(saveContent,60000) // run the savecontent function every minute
}

// starts the save loading animation
function showSaveLoader() {
```

```
  document.querySelector(".localnav.one .localnav-item.four")
    .setAttribute("data-kraken-savestate","saving")
}

// finishes the save loading animation and shows a success/fail icon, then resets to normal
async function hideSaveLoader(success) {
  document.querySelector(".localnav.one .localnav-item.four")
    .setAttribute("data-kraken-savestate",success?"success":"error")

  await sleep(1500)

  document.querySelector(".localnav.one .localnav-item.four")
    .removeAttribute("data-kraken-savestate")
}

// #endregion

/* -------------------- RESIZE BOXES ------------------- */
// #region

// generates resize boxes for every resizable element
function createResizeBoxes() {
  var r = resizable() // get all resizable elements
  for (var element of r) {
    // remove the current resize box
    element.querySelectorAll(".element-resize-container").forEach((e) => {
      e.parentElement.removeChild(e)
    })

    // set the resize type based on the element tags
    type = "corners"
    if (element.hasAttribute("data-kraken-editable-text")) type = "horizontal"

    // create the resize box for resizable and draggable elements
    element.appendChild(createResizeBoxContainerElement(type,element,true))
  }

  // iterate through all draggable elements
  for (var element in draggable()) {
    // if the element is not resizable
    if (!(element in r)) {
      // create the resize box for non-resizable elements
      element.appendChild(createResizeBoxContainerElement("",element,false))
    }
  }
}

// returns a resize box container element with all required things inside
function createResizeBoxContainerElement(type,parent,corners) {
  // create the container element and add required classes
  container = document.createElement("div")
  container.classList.add("element-resize-container")

  if (corners) { // if it requires corners, append a corner container child
    if (type == "corners") container.appendChild(createResizeBoxFourCornersElement())
    if (type == "horizontal") container.appendChild(createResizeBoxHorizontalCornersElement())
  }

  // append an outline child
  container.appendChild(createResizeBoxEdgesElement(parent))

  return container
}
```

```
// returns the resize box edges element
function createResizeBoxEdgesElement(parent) {
  // create the outline element and add required classes
  outline = document.createElement("div")
  outline.classList.add("element-outline")

  // set the correct height
  outline.style.height=parent.querySelector("div").offsetHeight+"px"

  return outline
}

// returns an element containing four resize corners
function createResizeBoxFourCornersElement() {
  // create the container element and add required classes
  root = document.createElement("div")
  root.classList.add("resize-corners")

  // append each required corner to the element
  root.appendChild(createResizeBoxCornerElement("top-left"))
  root.appendChild(createResizeBoxCornerElement("top-right"))
  root.appendChild(createResizeBoxCornerElement("bottom-left"))
  root.appendChild(createResizeBoxCornerElement("bottom-right"))

  return root
}

// returns an element containing two horizontal resize corners
function createResizeBoxHorizontalCornersElement() {
  // create the container element and add required classes
  root = document.createElement("div")
  root.classList.add("resize-corners")

  // append each required corner to the element
  root.appendChild(createResizeBoxCornerElement("horizontal-left"))
  root.appendChild(createResizeBoxCornerElement("horizontal-right"))
  return root
}

// reurns a resize corner
function createResizeBoxCornerElement(loc) {
  // create the corner element and add required classes
  corner = document.createElement("div")
  corner.classList.add("resize-corner")
  corner.classList.add(`resize-corner-${loc}`)

  // create the visual elemet and add required classes
  trigger = document.createElement("div")
  trigger.classList.add("resize-corner-visual-element")

  // add the visual element as a child of the corner element
  corner.appendChild(trigger)

  return corner
}

// #endregion

/* ------------ DRAG AND DROP EVENT LISTENERS ------------ */
// #region

function closestValInArray(num, arr) {
  prev = arr[0]
  for (val of arr) {
```

```
      if (val > num) return prev
      prev = val
  }
}

function closestValInArray(num, arr) {
  prev = arr[0]
  diff = Math.abs(num - prev)
  for (var val of arr) {
    var newdiff = Math.abs (num - val)
    if (newdiff < diff) {
      diff = newdiff
      prev = val
    }
  }
  return prev
}

// event listener function for element resizing
function elementResize(e2) {
  // get the corner that has been dragged
  current = e2.currentTarget

  // set the selected element and section to the parents of the corner

  parent = current.parentElement
  while (!(parent.hasAttribute("data-kraken-element"))) {
    parent = parent.parentElement
  }

  setSelectedElement(parent)

  while (!(parent.hasAttribute("data-kraken-section"))) {
    parent = parent.parentElement
  }

  setSelectedSection(parent)

  // get the current section's grid and current element
  currentsection = document.querySelector(queryprefix_sectionselected+" section .section-grid")
  currentelement = document.querySelector(queryprefix_elementselected)

  // get the computed styles of said elements and any required parameters
  currentsection_style = getComputedStyle(currentsection)
  currentelement_style = getComputedStyle(currentelement)
  currentsection_columngap = parseInt(currentsection_style.columnGap.replace("px",""))
  currentsection_gridtemplatecolumns = currentsection_style.gridTemplateColumns
  currentsection_startx = currentsection.getBoundingClientRect().left
  currentelement_position = currentelement_style.getPropertyValue("--position").split("/")

  columnlist = []
  cumulative = 0
  snappoints = []

  //console.log(currentelement)
  //console.log(currentsection)

  //console.log(currentsection_columngap)

  // get a list of the template columns as integers
  // eg [0,0,4,4,4,4,4,4,0,0]
  for (var col of currentsection_gridtemplatecolumns.split(" "))
    columnlist.push(parseInt(col.replace("px","")))
```

```
  // get the cumulative values of the column positions
  // eg [0,0,4,8,12,16,24,32,32,32] when the columngap is 0
  // this gets the offset from the start of the grid, in pixels, of every column
  for (var col of columnlist) {
    snappoints.push(cumulative)
    cumulative += col + currentsection_columngap
  }

  //console.log(snappoints)

  type = ""

  // get the type of resize based on the classes of the handle
  if (classlist.contains("resize-corner-top-left") |
      classlist.contains("resize-corner-top-right") |
      classlist.contains("resize-corner-bottom-left") |
      classlist.contains("resize-corner-bottom-right"))
        type="xy"

  if (classlist.contains("resize-corner-horizontal-left") |
      classlist.contains("resize-corner-horizontal-right"))
        type="x"

  // if its a horizontal resize
  if (type=="x") {

    //console.log("Old position variable: ",currentelement_position)

    // get the final x position (from the reference of the start of the viewport)
    endx = e2.clientX
    //console.log("End x position: ",endx)

    // get the final x position (from the reference of the start of the section grid)
    endxoffset = endx-currentsection_startx
    //console.log("End x offset from parent: ", endxoffset)
    //console.log("Column offsets: ", snappoints)

    // get the closest column to the final value
    closestsnappoint = closestValInArray(endxoffset,snappoints)
    //console.log("Closest column offset to end x offset: ", closestsnappoint)

    // get the index of the closest column, ie the column number - 1
    index = snappoints.indexOf(closestsnappoint)
    //console.log("Index of closest column: ", index)

    // set the column number for the position variable
    columnnumber = String(index+1)
    //console.log("Closest column number: ",columnnumber)

    // assign the new position based on the postoken
    currentelement_position[postoken] = columnnumber
    //console.log("New position variable: ",currentelement_position)

    // assign the new position property to the selected element
    currentelement.style.setProperty("--position",currentelement_position.join("/"))

  }
}

// event listener function for element drag-and-drop drop
function elementDrop(e2,endpos) {

  // get the dragged element
  current = e2.currentTarget
```

```
// get the current section, and any required style paramaters
currentsection = document.querySelector("[data-kraken-section-selected] section .section-grid")
currentsection_style = getComputedStyle(currentsection)
currentsection_columngap = parseInt(currentsection_style.columnGap.replace("px",""))
currentsection_rowgap = parseInt(currentsection_style.rowGap.replace("px",""))
currentsection_gridtemplatecolumns = currentsection_style.gridTemplateColumns
currentsection_gridtemplaterows = currentsection_style.gridTemplateRows
currentsection_startx = currentsection.getBoundingClientRect().left
currentsection_starty = currentsection.getBoundingClientRect().top

// get the elemenets style and position values
current_style = getComputedStyle(current)
current_position = current_style.getPropertyValue("--position").split("/")

//console.log("Position: ",current_position)

columnlist = []
cumulative = 0
columnsnappoints = []

// get a list of the template columns as integers
for (var col of currentsection_gridtemplatecolumns.split(" "))
  columnlist.push(parseInt(col.replace("px","")))

// get the cumulative values of the column positions
// this gets the offset from the start of the grid, in pixels, of every column
for (var col of columnlist) {
  columnsnappoints.push(cumulative)
  cumulative += col + currentsection_columngap
}

// repeat for rows

rowlist = []
cumulative = 0
rowsnappoints = []

// get a list of the template rows as integers
for (var row of currentsection_gridtemplaterows.split(" "))
  rowlist.push(parseInt(row.replace("px","")))

// get the cumulative values of the row positions
// this gets the offset from the top of the grid, in pixels, of every row
for (var row of rowlist) {
  rowsnappoints.push(cumulative)
  cumulative += row + currentsection_rowgap
}

//console.log("Col snap points: ",columnsnappoints)
//console.log("Row snap points: ",rowsnappoints)

// get the end x and y positions
// (endpos is the bounding box of the element when it was dropped)
endx = endpos.left
endy = endpos.top

// get the end x and y positions in relation to the section
endxoffset = endx-currentsection_startx
endyoffset = endy-currentsection_starty

//console.log("End X Offset: ",endxoffset)
//console.log("End Y Offset: ",endyoffset)
```

```
  // get the closest column/row offset point to the end offsets
  closestcolumnsnappoint = closestValInArray(endxoffset,columnsnappoints)
  closestrowsnappoint = closestValInArray(endyoffset,rowsnappoints)

  //console.log("Closest Col snap point ",closestcolumnsnappoint)
  //console.log("Closest Row snap point ",closestrowsnappoint)

  // get the numbers of the column/row
  columnindex = columnsnappoints.indexOf(closestcolumnsnappoint)
  rowindex = rowsnappoints.indexOf(closestrowsnappoint)

  //console.log("Col index: ",columnindex)
  //console.log("Row index: ",rowindex)

  // set the column and row numbers
  columnnumber = String(columnindex+1)
  rownumber = String(rowindex+1)

  //console.log("Col number: ",columnnumber)
  //console.log("Row number: ",rownumber)

  //console.log("Position: ",current_position)

  // 0 = start row
  // 1 = start column
  // 2 = end row
  // 3 = end column

  // Set the new end positions based on the difference between the original and new position
  current_position[2] = String(
                          parseInt(current_position[2])
                          + parseInt(rownumber)
                          - parseInt(current_position[0]))

  current_position[3] = String(
                          parseInt(current_position[3])
                          + parseInt(columnnumber)
                          - parseInt(current_position[1]))

  // set the new start positions
  current_position[1] = columnnumber
  current_position[0] = rownumber

  console.log("Position: ",current_position)

  // set the --position style variable
  current.style.setProperty("--position",current_position.join("/"))

}

// creates event listeners for resize corners on currently selected elements
function createResizeCornerEventListeners() {
  // get all corners of every resizable element
  for (corner of document.querySelectorAll(
       queryprefix_elementresizable+" .element-resize-container .resize-corner")) {

    // make a copy of the corner - this removes any existing event listeners so that there are
    // not multiple resize events being triggered
    newcorner = corner.cloneNode(true)
    corner.parentNode.replaceChild(newcorner,corner)

    corner=newcorner

    // mouse down event listener
```

```
    corner.addEventListener("mousedown",(e1) => {

      // get the clicked element
      current = e1.currentTarget

      // get the parent section by iterating through every parent element
      parent = current.parentElement
      while (!(parent.hasAttribute("data-kraken-section"))) parent = parent.parentElement

      // render the grid preview element
      parent.querySelectorAll(".section-grid-preview").forEach((e)=>{
        e.setAttribute("data-kraken-visible","")
      })

      type = ""
      classlist = current.classList
      postoken = 0

      // work out whether it is a horizontal resize or a 2D resizes by looking at the
      // classes of the elemenet
      if (classlist.contains("resize-corner-top-left") |
          classlist.contains("resize-corner-top-right") |
          classlist.contains("resize-corner-bottom-left") |
          classlist.contains("resize-corner-bottom-right"))
            type="xy"

      if (classlist.contains("resize-corner-horizontal-left") |
          classlist.contains("resize-corner-horizontal-right"))
            type="x"

      // postoken defines which corner is being dragged, and how the code should change the
      // attributes based on that
      if (classlist.contains("resize-corner-horizontal-left")) postoken=1
      if (classlist.contains("resize-corner-horizontal-right")) postoken=3

      // get the start position of the mouse
      startpos = [e1.clientX,e1.clientY]

      // set the mouse move event listener
      current.addEventListener("mousemove",(e2) => {

        // get the clicked element
        current = e2.currentTarget

        // get the new position of the mouse
        newpos = [e2.clientX,e2.clientY]

        // get the full style of the current element
        current_style = getComputedStyle(current)

        // get the current transformation property by converting
        // matrix(a,b,c,d,e,f)
        // into
        // ["a","b","c","d","e","f"]
        current_transform = current_style.transform.replace("matrix(","")
                            .replace(")","").replaceAll(" ","").split(",")

        //console.log(e2.clientX,startpos[0],e2.clientX - startpos[0])

        // if it is a horizontal resize
        if (type=="x") {
          // get the difference in position since the last render
          diff = newpos[0] - startpos[0]
```

```
            // set the horizontal translation property of the matrix
            current_transform[4] = String(parseInt(current_transform[4])+diff)

            // set the new matrix as the transform property
            current.style.transform = "matrix(" + current_transform.join(",") + ")"

            // get the outline element of the resize box
            outline = current.parentNode.parentNode.querySelector(".element-outline")
            outline_style = getComputedStyle(outline)
            outline_width = outline_style.width
            outline_width = parseInt(outline_width.replace("px",""))

            outline_left = outline_style.left
            outline_left = parseInt(outline_left.replace("px",""))

            outline_right = outline_style.right
            outline_right = parseInt(outline_right.replace("px",""))

            // if the element is being resized from the left handle
            if (postoken == 1) {
              // set the style properties to reflect the transformation
              outline.style.left = String(outline_left + diff) + "px"
              outline.style.width = String(outline_width - diff) + "px"
            }

            // if the element is being resized from the right handle
            if (postoken == 3) {
              // set the style properties to reflect the transformation
              outline.style.right = String(outline_right + diff) + "px"
              outline.style.width = String(outline_width + diff) + "px"
            }
          }

          // set the start position so that the next render resize works properly
          startpos = newpos

        })

        // set the mouse up event listener
        current.addEventListener("mouseup",(e2) => {

          // set the new properties for the element
          elementResize(e2)

          // get the parent section by iterating through every parent element
          parent = current.parentElement
          while (!(parent.hasAttribute("data-kraken-section"))) parent = parent.parentElement

          // hide the grid preview element
          parent.querySelectorAll(".section-grid-preview").forEach((e)=>{
            e.removeAttribute("data-kraken-visible")
          })

          // recreate the resize boxes and event listeners to remove any visual errors and clear
          // the mousemove and mouseup event listeners
          createResizeBoxes()
          createResizeCornerEventListeners()

        })
      })
    }
}

// creates event listeners for all draggable elements
```

```
function createDragEventListeners() {
  // get every draggable element
  for (element of draggable()) {

    // create the mouse down eevent listener
    element.addEventListener("mousedown",(e1) => {

      // end the listener if a resize corner is being clicked, because otherwise the drag and
      // resize functions will be happening simultaneously
      if (e1.target.classList.contains("resize-corner") |
          e1.target.classList.contains("resize-corner-visual-element"))
            return

      // get the current element, set it as being dragged, and fetch the content
      current = e1.currentTarget
      current.setAttribute("data-kraken-dragging","")
      content = current.querySelector("div")

      // get the starting cursor position
      startpos = [e1.clientX,e1.clientY]

      // get the current --position value
      storedPosition = current.style.getPropertyValue("--position")

      // set the box shadow and background color for the element
      content.style.boxShadow = "0px 0px 8px 3px rgba(0,0,0,0.25"
      content.style.backgroundColor = "var(--colors-light)"

      // get the parent section by iterating through every parent element
      parent = current.parentElement
      while (!(parent.hasAttribute("data-kraken-section"))) parent = parent.parentElement

      // render the grid preview element
      parent.querySelectorAll(".section-grid-preview").forEach((e)=>{
        e.setAttribute("data-kraken-visible","")
      })

      // create the mouse move event listener
      current.addEventListener("mousemove",(e2) => {

        // get the selected element
        current = e2.currentTarget

        // store the new cursor position
        newpos = [e2.clientX,e2.clientY]

        // fetch the style of the selected element
        current_style = getComputedStyle(current)

        // get the differences in position
        xdiff = newpos[0] - startpos[0]
        ydiff = newpos[1] - startpos[1]

        // get the bounding boxes of the section and element
        sectionbox = selectedSection.getBoundingClientRect()
        currentbox = current.getBoundingClientRect()

        // get the element's width and height in pixels
        currentwidth = current.offsetWidth
        currentheight = current.offsetHeight

        // get the top and left positions relative to the section boundary
        relativetop = currentbox.top - sectionbox.top + ydiff
        relativeleft = currentbox.left - sectionbox.left + xdiff
```

```
        // set the element style so that it follows the cursor
        current.style.position = "absolute"
        current.style.setProperty("--position","")
        current.style.top = String(relativetop) + "px"
        current.style.left = String(relativeleft) + "px"
        current.style.width = String(currentwidth) + "px"
        current.style.height = String(currentheight) + "px"
        current.style.zIndex = "999999"

        // set the start position so that the next render works properly
        startpos = [e2.clientX,e2.clientY]

      })

      // set the mouse up event listener
      current.addEventListener("mouseup",(e2) => {

        // get the dragged element, remove the dragging tag, and get its content
        current = e2.currentTarget
        current.removeAttribute("data-kraken-dragging")
        content = current.querySelector("div")

        // get the dragged element's final position
        endpos = current.getBoundingClientRect()

        // reset the styling used in the dragging animation
        content.style.boxShadow = ""
        content.style.backgroundColor = ""

        current.style.removeProperty("position")
        current.style.removeProperty("top")
        current.style.removeProperty("left")
        current.style.removeProperty("width")
        current.style.removeProperty("height")
        current.style.removeProperty("z-index")

        // set the element position back to what it was while the new one is being calculated
        current.style.setProperty("--position",storedPosition)

        // calculate and set the new position property for the element
        elementDrop(e2,endpos)

        // get the parent section by iterating through every parent element
        parent = current.parentElement
        while (!(parent.hasAttribute("data-kraken-section"))) parent = parent.parentElement

        // hide the grid preview element
        parent.querySelectorAll(".section-grid-preview").forEach((e)=>{
          e.removeAttribute("data-kraken-visible")
        })

        // duplicate the element to remove all used event listeners
        newcurrent = current.cloneNode(true)
        current.parentNode.replaceChild(newcurrent,current)
        current = newcurrent

        // recreate the element event listeners
        elementEventListeners()

      })
    })

}
```

```
}

// #endregion

/* ------------ SELECTED ELEMENTS AND SECTIONS ----------- */
// #region

function setSelectedElement(element) {
  document.querySelectorAll("[data-kraken-element-selected]").forEach((e)=>{
    e.removeAttribute("[data-kraken-element-selected]")
  })

  element.setAttribute("data-kraken-element-selected","")
}

function setSelectedSection(section) {
  document.querySelectorAll("[data-kraken-section-selected]").forEach((e)=>{
    e.removeAttribute("[data-kraken-section-selected]")
  })

  section.setAttribute("data-kraken-section-selected","")
}

// #endregion

/* ------------------ GRID PREVIEW BOX ------------------- */
// #region

function createGridPreviews() {
  // for every section
  for (var section of document.querySelectorAll(queryprefix_section + " section")) {

    // remove any existing grid previews in the section
    for (var existing of section.querySelectorAll(".section-grid-preview")) {
      existing.parentNode.removeChild(existing)
    }

    // create the grid container element and add required classes
    container = document.createElement("div")
    container.classList.add("section-grid-preview")

    // get the style of the section grid and any required paramaters
    gridstyle = getComputedStyle(section.querySelector(".section-grid"))
    gridstyle_rows = gridstyle.gridTemplateRows.split(" ")
    gridstyle_columns = gridstyle.gridTemplateColumns.split(" ")
    gridstyle_rowgap = parseInt(gridstyle.rowGap.replace("px",""))
    gridstyle_columngap = parseInt(gridstyle.columnGap.replace("px",""))

    columnoffset = parseInt(gridstyle.getPropertyValue("--block-padding-left"))
    rowoffset = parseInt(gridstyle.getPropertyValue("--block-padding-top"))

    //console.log(gridstyle.gridTemplateColumns)

    // for every column in the grid
    for (var columnwidth of gridstyle_columns) {

      // set the rowoffset back to the starting value
      rowoffset = parseInt(gridstyle.getPropertyValue("--block-padding-top"))

      // for every row in the column
      for (var rowheight of gridstyle_rows) {
        //console.log(columnoffset,rowoffset)

        // add a child box with the correct positions
```

```
        container.appendChild(createGridPreviewBox(columnwidth,rowheight,rowoffset,columnoffset))

        //console.log(parseInt(rowheight.replace("px","")))

        // increase the row offset by the width of the row and the row gap
        rowoffset = rowoffset + parseInt(rowheight.replace("px","")) + gridstyle_rowgap
      }

      // increase the column offset by the width of the column and the column gap
      columnoffset = columnoffset + parseInt(columnwidth.replace("px","")) + gridstyle_columngap

    }

    // add the grid previeew
    section.appendChild(container)
  }
}

// creates a single box for the grid previews
function createGridPreviewBox(width,height,top,left) {
  // create the box element and add any required classes
  box = document.createElement("div")
  box.classList.add("section-grid-preview-box")

  // set the style to match the arguments
  box.style.width = width
  box.style.height = height
  box.style.top = top+"px"
  box.style.left = left+"px"

  // hide the box if one of the dimensions is 0
  if (width == "0px" || height == "0px") box.style.opacity = "0"

  return box
}

// #endregion

/* ------------------------- RUN ------------------------- */

run()
```

/templates/

/templates/base.html

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-type" content="text/html; charset=utf-8">
    <meta http-equiv="Content-type" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Kraken</title>

    <!-- Site Meta -->
    <meta name="title" content="Kraken">
    <meta name="description" content="SiteDescription">
    <meta name="robots" content="index, follow">
    <meta name="language" content="English">
```

```
    <meta name="author" content="Tom_gxz">
    <meta name="keywords" content="keywords">

    <!-- navbarLogoColor is a jinja variable that is defined in files that extend from this one.
    It defines what colour the logo should be - primary, secondary, or gradient -->
    <link rel="apple-touch-icon" sizes="512x512" href="{{url_for('static',
    filename='img/icon/512-512/kraken-icon-png-'+navbarLogoColor+'-128-128.png')}}">

    <link rel="icon" type="image/png" sizes="128x128" href="{{url_for('static',
    filename='img/icon/128-128/kraken-icon-png-'+navbarLogoColor+'-128-128.png')}}">

    <link rel="mask-icon" href="{{url_for('static',filename='img/icon/mask-icon/mask-icon.svg')}}">

    <meta name="apple-mobile-web-app-title" content="Kraken">
    <meta name="application-name" content="SiteName">
    <meta name="theme-color" content="#SiteColor">
    <link rel="canonical" href="CanonicalUrl">

    <!-- Open Graph Protocol -->
    <meta property="og:site_name" content="Kraken">
    <meta property="og:title" content="Kraken">
    <meta property="og:description" content="SiteDescription">
    <meta property="og:image:type" content="image/png">
    <meta property="og:image:width" content="512">
    <meta property="og:image:height" content="512">
    <meta property="og:image" content="{{url_for('static',
    filename='img/icon/tab-icon/tab-icon-512-512.png')}}">
    <meta property="og:type" content="website">
    <meta property="og:url" content="PageUrl">

    <!-- Twitter Embed -->
    <meta property="twitter:card" content="summary">
    <meta property="twitter:site" content="@TwitterHandle">
    <meta property="twitter:title" content="Kraken">
    <meta property="twitter:description" content="SiteDescription">
    <meta property="twitter:image" content="{{url_for('static',
    filename='img/icon/tab-icon/tab-icon-512-512.png')}}">
    <meta property="twitter:url" content="PageUrl">

    <!-- Font Awesome Imports -->
    <script src="https://kit.fontawesome.com/73a2cc1270.js"></script>
    <link rel="stylesheet" href="https://pro.fontawesome.com/releases/v6.0.0-beta3/css/all.css">

    <!-- Internal Stylesheet Imports -->
    <link href="{{url_for('static',filename='css/main.css')}}" rel="stylesheet" type="text/css" />
    <link href="{{url_for('static',filename='css/build.css')}}" rel="stylesheet" type="text/css" />

  </head>
  <body>

    <div class="page">
      <div class="application-container">

        <nav class="globalnav globalnav-vertical">
          <div class="globalnav-content">
            <div class="globalnav-list">

              <div class="globalnav-logo">
                <a class="globalnav-link globalnav-link-home link unformatted"
                href="{{ url_for('main_home') }}">

                <img class="globalnav-logo-image" alt="Kraken" src="{{url_for('static',
                filename='img/icon/512-512/kraken-icon-png-'+navbarLogoColor+'-512-512.png')}}"
                preserveAspectRatio>
```

```
          <span class="globalnav-link-hidden-text visibly-hidden">Kraken</span>
          </a>
        </div>

        {% if navbarOptionsEnabled %}

          <ul class="globalnav-list">

            <li class="globalnav-item one fake" role="button"></li>

            <li class="globalnav-item two" role="button">
              <div class="hamburger hamburger--collapse js-hamburger"
              id="globalnav-hamburger">

                <div class="hamburger-box">
                  <div class="hamburger-inner"></div>
                </div>
              </div>
            </li>

          </ul>

        {% endif %}

      </div>
    </div>
  </nav>

  {% if navbarOptionsEnabled %}

    <div class="globalnav-floating-options">

      <a class="globalnav-floating-option one" href="{{ url_for('main_home') }}">
        <span class="globalnav-floating-option-content text header small dark">
          My Sites
        </span>
      </a>

      <a class="globalnav-floating-option two" href="{{ url_for('settings') }}">
        <span class="globalnav-floating-option-content text header small dark">
          Settings
        </span>
      </a>

      <a class="globalnav-floating-option three" href="{{ url_for('auth_logout') }}">
        <span class="globalnav-floating-option-content text header small dark">
          Logout
        </span>
      </a>

    </div>

    <div class="globalnav-floating-options-backdrop"></div>

    <script src="{{url_for('static', filename='js/globalnav-floating-options.js')}}">
    </script>

  {% endif %}

  <!-- External Script Imports -->
  <script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>
  <script src="https://code.jquery.com/jquery-3.5.1.min.js" crossorigin="anonymous"></script>
```

```
        <!-- Internal Script Imports -->
        <script src="{{url_for('static', filename='js/main.js')}}"></script>

        {% block content %}
        {% endblock %}

      </div>
    </div>

  </body>

</html>
```

/templates/home-nosite.html

```
{% extends "base.html" %}

{% set navbarLogoColor = "primary" %}
{% set navbarOptionsEnabled = True %}

{% block content %}

<link href="{{url_for('static', filename='css/home.css')}}" rel="stylesheet" type="text/css" />

<div class="application-content">

  <div class="text-header-container">
    <h2 class="text header large dark one">Welcome, {{current_user.name}}</h2>
  </div>

  <div class="empty-container">
    <div class="empty-image"></div>
    <div class="empty-text-container">

      <h4 class="text header dark one">Looks Pretty Empty Here...</h4>

      <a class="text header two link primary" href="{{url_for('site_create')}}">
        Maybe you should create a new site?
      </a>

    </div>
  </div>

</div>

{% endblock %}
```

/templates/home-sites.html

```
{% extends "base.html" %}

{% set navbarLogoColor = "primary" %}
{% set navbarOptionsEnabled = True %}
{% set sites = get_flashed_messages()[0] %}

{% block content %}

<link href="{{url_for('static', filename='css/home.css')}}" rel="stylesheet" type="text/css" />
```

```
<div class="application-content">

  <div class="text-header-container">
    <h2 class="text header large dark one">Welcome, {{current_user.name}}</h2>
  </div>

  <div class="site-divs-container" style="display:flex;flex-wrap:wrap">

    {% for site in sites %}

      <a class="site-div link notformatted" href="{{url_for('site_edit_home',
      name=site[0],site=site[1])}}" style="background-color:
      {%if site[3]%}{{site[3]+'bb'}}{%else%}var(--colors-primary){%endif%}">

        {% if site[2] %}

          <div class="site-div-private-watermark" style="position: absolute;opacity: 0.5;
          font-size: 136px;right: 16px;top: 16px;">

            <i class="faicon fa-regular fa-lock"></i>
          </div>

        {% else %}

          <div class="site-div-public-watermark" style="position: absolute;opacity: 0.5;
          font-size: 136px;right: 16px;top: 16px;">

            <i class="faicon fa-regular fa-book-bookmark"></i>
          </div>

        {% endif %}

        <h5 class="site-div-title text large one">{{ site[1] }}</h5>
      </a>

    {% endfor %}

    <a class="site-div link notformatted" style="background-color:var(--colors-primary-light);
    display:flex;align-items:center;justify-content:center" href="{{url_for('site_create')}}">

      <h5 class="site-div-title text header jumbo small one center">Create New Site</h5>
    </a>

  </div>

</div>

{% endblock %}
```

/templates/login.html

```
{% extends "base.html" %}

{% set navbarLogoColor = "secondary" %}
{% set navbarOptionsEnabled = False %}
{% set messages = get_flashed_messages()[0] %}

{% block content %}

<link href="{{url_for('static', filename='css/auth.css')}}" rel="stylesheet" type="text/css" />
```

```
<div class="application-content">

  <div class="text-header-container">
    <h2 class="text header xl dark one">Kraken - Login</h2>

    <ul class="header-options">

      <li class="header-option header-option-login active notextselect">
        <h4 class="text header bold">Login</h4>
      </li>

      <li class="header-option header-option-signup notextselect"
      /onclick="window.location.href=`{{ url_for('auth_signup') }}`">

        <h4 class="text header bold">Signup</h4>
      </li>

    </ul>

  </div>

  <div class="field-container active">

    <span class="field-warning text italic">
      {% if messages[0] %}
        {{ messages[1] }}
      {% endif %}
    </span>

    <form class="field-options" method="post" action="/login/">

      {% set formItems = [
      ["Username","Username","text","username",false,messages[2]],
      ["Password","Password","password","password",true,""]
      ]
      %}

      {% for item in formItems %}

        <div class="field-option field-option-{{item[3]}}">

          <h4 class="text italic">{{item[0]}}</h4>

          <div class="field-input-container">
            <input class="field-input" placeholder="{{item[1]}}" type="{{item[2]}}"
            name="{{item[3]}}" value="{{item[5]}}">

            {% if item[4] %}

              <span class="eye-reveal">
                <i class="fa-solid fa-eye"></i>
              </span>

            {% else %}

              <span class="eye-spacer"></span>

            {% endif %}

          </div>
        </div>

      {% endfor %}
```

```
      <div class="field-option field-option-remember">

        <h4 class="text italic">Remember Me</h4>

        <div class="field-input-container">
          <input class="field-input" type="checkbox" name="remember" value={{messages[3]}}>
          <span class="eye-spacer"></span>
        </div>

      </div>

      <button class="field-submit btn secondary rounded slide" type="submit">
        <span class="btn-content text uppercase secondary">Submit</span>
      </button>

    </form>

  </div>

</div>

<script src="../static/js/auth.js"></script>
<script src="../static/js/login.js"></script>

{% endblock %}
```

/templates/settings-admin.html

```
{% extends "settings-base.html" %}

{% set navbarLogoColor = "primary" %}
{% set navbarOptionsEnabled = True %}
{% set settingsSidebarActivated = 2 %}

{% block settings_content %}

<h3 class="settings-content-header text header dark">Account</h3>

<form class="settings-content-options">

  <div class="settings-content-option one">
    <h4 class="settings-content-option-title text dark bold">Change Username</h4>

    <p class="settings-content-option-caption text small danger dark">
      <span class="text small danger bold">WARNING</span>.
      Changing your username can
      <a class="text small link danger bold">create issues</a>.
    </p>

    <div class="btn primary thin rounded slide m-xs-t">
      <span class="btn-content text uppercase primary">Dew it</span>
    </div>

  </div>

  <div class="settings-content-option two">
    <h4 class="settings-content-option-title text dark bold">Change Email</h4>

    <input class="settings-content-option-input" type="text"
    name="settings_admin_email" placeholder="New Email"><br>
```

```
    <input class="settings-content-option-input m-xs-t" type="password"
    name="settings_admin_email_password" placeholder="Password">

    <p class="settings-content-option-caption text small dark">
      Your old and new email addresses will be sent confirmation codes in order to change them.
    </p>

    <div class="btn primary thin rounded slide m-xs-t">
      <span class="btn-content text uppercase primary">Dew it</span>
    </div>

  </div>

  <div class="settings-content-separator one"></div>

  <div class="settings-content-option three">
    <h4 class="settings-content-option-title text dark bold">Export Account Data</h4>

    <p class="settings-content-option-caption text small dark">
      Export all metadata, websites, and other stored information for your account.
      They will be available to download here.
    </p>

    <div class="btn primary thin rounded slide m-xs-t">
      <span class="btn-content text uppercase primary">Export Metadata</span>
    </div>

    <div class="btn primary thin rounded slide m-xs-t">
      <span class="btn-content text uppercase primary">Export Websites</span>
    </div>

    <div class="btn primary thin rounded slide m-xs-t">
      <span class="btn-content text uppercase primary">Download All</span>
    </div>

  </div>

  <div class="settings-content-separator two"></div>

  <div class="settings-content-option four">
    <h4 class="settings-content-option-title text dark bold danger">Archive Account</h4>

    <p class="settings-content-option-caption text small dark">
      This will disable your account until you wish to unlock it.
    </p>

    <div class="btn danger thin rounded slide m-xs-t">
      <span class="btn-content text uppercase danger">Archive your account</span>
    </div>

  </div>

  <div class="settings-content-option five">
    <h4 class="settings-content-option-title text dark bold danger">Reset Account</h4>

    <p class="settings-content-option-caption text small dark">
      This will remove all of your websites, custom code, and non-essential settings.
      The only remaining settings will be your username, name, email, and password.
      It is recommended that you export your account data before doing this.
    </p>

    <div class="btn danger thin rounded slide m-xs-t">
      <span class="btn-content text uppercase danger">Reset your account</span>
    </div>
```

```
  </div>

  <div class="settings-content-option six">
    <h4 class="settings-content-option-title text dark bold danger">Delete Account</h4>

    <p class="settings-content-option-caption text small dark">
      This will remove all trace of your account from our servers, and is an irreversable action.
      It is recommended that you export your account data before doing this.
    </p>

    <div class="btn danger thin rounded slide m-xs-t">
      <span class="btn-content text uppercase danger">Delete your account</span>
    </div>

  </div>

</form>

{% endblock %}
```

/templates/settings-base.html

```
{% extends "base.html" %}

{% set navbarLogoColor = "primary" %}
{% set navbarOptionsEnabled = True %}
{% set settingsSidebarActivated = get_flashed_messages()[0] %}

{% block content %}

<link href="{{url_for('static', filename='css/settings.css')}}" rel="stylesheet" type="text/css" />

<div class="application-content">

  <div class="text-header-container">
    <h2 class="text header large dark one">Settings</h2>
  </div>

  <div class="settings-container">

    <div class="settings-sidebar">

      <a class="settings-sidebar-item one{% if settingsSidebarActivated==1 %} is-active{%endif%}
      link notformatted" {% if not settingsSidebarActivated==1 %}
      href="{{ url_for('settings_profile') }}"{% endif %}>

        <i class="settings-sidebar-item-icon fa-regular fa-user"></i>
        <span class="settings-sidebar-item-title text large">Public Profile</span>
      </a>

      <a class="settings-sidebar-item two{% if settingsSidebarActivated==2 %} is-active{%endif%}
      link notformatted" {% if not settingsSidebarActivated==2 %}
      href="{{ url_for('settings_admin') }}"{% endif %}>

        <i class="settings-sidebar-item-icon fa-regular fa-gear"></i>
        <span class="settings-sidebar-item-title text large">Account</span>
      </a>

      <a class="settings-sidebar-item three{% if settingsSidebarActivated==3 %} is-active{%endif%}
      link notformatted" {% if not settingsSidebarActivated==3 %}
```

```
      href="{{ url_for('settings_looks') }}"{% endif %}>

        <i class="settings-sidebar-item-icon fa-regular fa-paintbrush"></i>
        <span class="settings-sidebar-item-title text large">Appearance & Accessibility</span>
      </a>

      <div class="settings-sidebar-separator text one">Code and websites</div>
      <a class="settings-sidebar-item four{% if settingsSidebarActivated==4 %} is-active{%endif%}
      link notformatted" {% if not settingsSidebarActivated==4 %}
      href="{{ url_for('settings_sites') }}"{% endif %}>

        <i class="settings-sidebar-item-icon fa-regular fa-browser"></i>
        <span class="settings-sidebar-item-title text large">My Websites</span>
      </a>

      <a class="settings-sidebar-item five{% if settingsSidebarActivated==5 %} is-active{%endif%}
      link notformatted" {% if not settingsSidebarActivated==5 %}
      href="{{ url_for('settings_code') }}"{% endif %}>

        <i class="settings-sidebar-item-icon fa-regular fa-list-timeline"></i>
        <span class="settings-sidebar-item-title text large">Custom Code & Elements</span>
      </a>

      <div class="settings-sidebar-separator text two"> </div>
      <a class="settings-sidebar-item six{% if settingsSidebarActivated==6 %} is-active{%endif%}
      link notformatted" {% if not settingsSidebarActivated==6 %}
      href="{{ url_for('main_help') }}"{% endif %}>

        <i class="settings-sidebar-item-icon fa-regular fa-book-blank"></i>
        <span class="settings-sidebar-item-title text large">Help & Documentation</span>
      </a>

      <a class="settings-sidebar-item seven{% if settingsSidebarActivated==7 %} is-active{%endif%}
      link notformatted" {% if not settingsSidebarActivated==7 %}
      href="{{ url_for('settings_dev') }}"{% endif %}>

        <i class="settings-sidebar-item-icon fa-regular fa-code-simple"></i>
        <span class="settings-sidebar-item-title text large">Developer settings</span>
      </a>

    </div>

    <div class="settings-content">

      {% block settings_content %}
      {% endblock %}

    </div>
  </div>
</div>

{% endblock %}
```

/templates/settings-code.html

```
{% extends "settings-base.html" %}

{% set navbarLogoColor = "primary" %}
{% set navbarOptionsEnabled = True %}

{% set settingsSidebarActivated = 5 %}
```

```
{% block settings_content %}

ToDo:

My custom code
  Displays name, type of code, size of code, and settings for it

My custom elements
  Displays name, description, and settings for it

{% endblock %}
```

/templates/settings-dev.html

```
{% extends "settings-base.html" %}

{% set navbarLogoColor = "primary" %}
{% set navbarOptionsEnabled = True %}

{% set settingsSidebarActivated = 7 %}

{% block settings_content %}

To do:

warning at the start

{% endblock %}
```

/templates/settings-looks.html

```
{% extends "settings-base.html" %}

{% set navbarLogoColor = "primary" %}
{% set navbarOptionsEnabled = True %}

{% set settingsSidebarActivated = 3 %}

{% block settings_content %}

<h3 class="settings-content-header text header dark">Appearance and Accessibility</h3>

<form class="settings-content-options">

  <div class="settings-content-option one">
    <h4 class="settings-content-option-title text dark bold">Tab Preference</h4>

    <select class="settings-content-option-input" name="settings_looks_tab_preference">
      <option value="1">1</option>
      <option value="2">2</option>
      <option value="3">3</option>
      <option value="4" selected="selected">4 (Default)</option>
      <option value="5">5</option>
      <option value="6">6</option>
      <option value="8">8</option>
      <option value="10">10</option>
      <option value="12">12</option>
    </select>
```

```
    <p class="settings-content-option-caption text small dark">
      When editing and rendering code, this determines how many spaces represent one tab.
      (Doesn't do anything yet)
    </p>

  </div>

</form>

{% endblock %}
```

/templates/settings-profile.html

```
{% extends "settings-base.html" %}

{% set navbarLogoColor = "primary" %}
{% set navbarOptionsEnabled = True %}
{% set settingsSidebarActivated = 1 %}
{% set avatarImagePath = get_flashed_messages()[1] %}

{% block settings_content %}

<h3 class="settings-content-header text header dark">Profile</h3>

<form class="settings-content-options">

  <div class="settings-content-option one">
    <h4 class="settings-content-option-title text dark bold">Name</h4>

    <input class="settings-content-option-input" type="text" name="settings_profile_name"
    placeholder="{{current_user.name}}">

    <p class="settings-content-option-caption text small dark">
      Your name probably isn't used much yet, but may appear in reference to websites that you
      have created. Your current name is set to "{{current_user.name}}".
    </p>
  </div>

  <div class="settings-content-option two">
    <h4 class="settings-content-option-title text dark bold">Profile Picture</h4>

    <div class="settings-content-option-image-upload">
      <figure class="settings-content-option-image-upload-figure" style="background-image:url(
      {{ url_for('static',filename='data/userIcons/'+current_user.user_id+'.png') }})"></figure>
    </div>

    <p class="settings-content-option-caption text small dark">
      The image must be a minimum of 200x200 pixels, and in a 1:1 ratio.
      This is not operational yet, and probably won't be for a while.
    </p>
  </div>

  <div class="settings-content-separator one"></div>

  <div class="settings-content-option three">
    <h4 class="settings-content-option-title text dark bold">Bio</h4>

    <textarea class="settings-content-option-input" type="text" name="settings_profile_bio"
    maxlength=240 placeholder="Tell us about yourself"></textarea>
  </div>
```

```
  <div class="settings-content-option four">
    <h4 class="settings-content-option-title text dark bold">Url</h4>

    <input class="settings-content-option-input" type="text" name="settings_profile_url">
  </div>

  <div class="settings-content-separator two"></div>

  <div class="settings-content-option settings-content-update">

    <p class="settings-content-option-caption text small dark">
      All of the above fields are optional and can be left blank. By filling them out,
      you agree that this information can be displayed publically and stored in our servers.
      We don't have a privacy statement, but we probably should.
    </p>

    <button class="field-submit btn primary thin rounded slide" type="submit">
      <span class="btn-content text uppercase primary">Update Profile</span>
    </button>

  </div>

</form>

{% endblock %}
```

/templates/settings-sites.html

```
{% extends "settings-base.html" %}

{% set navbarLogoColor = "primary" %}
{% set navbarOptionsEnabled = True %}

{% set settingsSidebarActivated = 4 %}
{% set flashedSiteNames = get_flashed_messages()[1] %}

{% block settings_content %}

<h3 class="settings-content-header text header dark">My Websites</h3>

<form class="settings-content-options">

  <div class="settings-content-table">
    <div class="settings-content-table-header">
      <div class="settings-content-table-row">
        <span class="text header dark">Websites</span>
      </div>
      <div class="settings-content-table-row">
        <span class="text header small dark">@{{current_user.user_id}}</span>
      </div>
    </div>

    <div class="settings-content-table-content">

      {% for site in flashedSiteNames %}

        <div class="settings-content-table-row
        {% if site==flashedSiteNames[-1] %}last-row{% endif %}">

          <span class="settings-content-table-row-icon text dark"><i class="fa-regular
```

```
          {% if site[2] %}fa-lock{% else %}fa-book-bookmark{%endif%}"></i></span>

          <span class="settings-content-table-row-title text dark">
            <a href='{{url_for("site_edit_home",name=site[0],site=site[1])}}'
            class="text link dark notformatted">@{{site[0]}}/{{site[1]}}</a>
          </span>

          <span class="settings-content-table-row-size text dark">{{site[3]}}</span>

          <span class="settings-content-table-row-settings text dark">
            <a href='{{url_for("site_edit_home",name=site[0],site=site[1])}}'
            class="text link primary notformatted">Website Settings</a>
          </span>
        </div>

      {% endfor %}

    </div>

  </div>

</form>

{% endblock %}
```

/templates/signup.html

```
{% extends "base.html" %}

{% set navbarLogoColor = "secondary" %}
{% set navbarOptionsEnabled = False %}
{% set messages = get_flashed_messages()[0] %}

{% block content %}

<link href="{{url_for('static', filename='css/auth.css')}}" rel="stylesheet" type="text/css" />

<div class="application-content">

  <div class="text-header-container">

    <h2 class="text header xl dark one">Kraken - Signup</h2>

    <ul class="header-options">
      <div class="header-option header-option-login notextselect"
      onclick="window.location.href=`{{ url_for('auth_login') }}`">

        <h4 class="text header bold">Login</h4>
      </div>
      <div class="header-option header-option-signup active notextselect">
        <h4 class="text header bold">Signup</h4>
      </div>
    </ul>

  </div>

  <div class="field-container active">

    <span class="field-warning text italic">

      {% if messages[0] %}
```

```
        {{ messages[1] }}
      {% endif %}

    </span>

    <form class="field-options" method="post" action="/signup/">

      {% set formItems = [
        ["Name","Name","text","name",false,messages[2]],
        ["Email","name@domain.com","email","email",false,messages[3]],
        ["Username","Username","text","username",false,messages[4]],
        ["Password","Password","password","password",true,""],
        ["Repeat Password","Again :/","password","password-repeat",""]
      ]
      %}

      {% for item in formItems %}

        <div class="field-option field-option-{{item[3]}}">
          <h4 class="text italic">{{item[0]}}</h4>
          <div class="field-input-container">
            <input class="field-input" placeholder="{{item[1]}}" type="{{item[2]}}"
            name="{{item[3]}}" value="{{item[5]}}">

            {% if item[4] %}

              <span class="eye-reveal">
              <i class="fa-solid fa-eye"></i>
              </span>

            {% else %}

              <span class="eye-spacer"></span>

            {% endif %}
          </div>
        </div>

      {% endfor %}

      <button class="field-submit btn secondary rounded slide" type="submit">
        <span class="btn-content text uppercase secondary">Submit</span>
      </button>

    </form>

  </div>

</div>

<script src="../static/js/auth.js"></script>
<script src="../static/js/signup.js"></script>

{% endblock %}
```

/templates/site-create-base.html

```
{% extends "base.html" %}

{% set navbarLogoColor = "primary" %}
{% set navbarOptionsEnabled = True %}
```

```
{% block content %}

<link href="{{url_for('static', filename='css/site-create.css')}}" rel="stylesheet"
type="text/css" />

<div class="application-content">

  <div class="text-header-container">
    <h2 class="text header large dark one">Create a new site</h2>
  </div>

  <div class="main">
    <div class="main-content thin">

      {%block site_create_base%}
      {%endblock%}

    </div>
  </div>

</div>
{% endblock %}
```

/templates/site-create-options-1.html

```
{% extends "site_create_base.html" %}

{% set navbarLogoColor = "primary" %}
{% set navbarOptionsEnabled = True %}

{% block site_create_base %}

<p class="text dark">Choose a color scheme!</p>

<div class="horizontal-separator one m-s-v"></div>

<form class="new-site-form two" method="post">

  <div class="light-dark-selector-container">
    <div class="light-dark-selector">

      <h2 class="text large center one">Light or dark mode?</h2>

      <div class="button-container">
        <div class="btn primary thin rounded slide from-left m-xs-t"
        id="new_site_lightModeToggle">

          <span class="btn-content text uppercase primary notextselect">Light</span>
        </div>

        <div class="btn secondary thin rounded slide from-right m-xs-t"
        id="new_site_darkModeToggle">

          <span class="btn-content text uppercase secondary notextselect">Dark</span>
        </div>
      </div>

    </div>
  </div>
```

```
<div class="horizontal-separator one m-s-v"></div>

<div class="color-options">

  <div class="input-slider-option one">

    <span class="text dark bold">Light & Dark Bounds</span>

    <div class="input-slider-and-number">

      <input class="input sliding-input" type="range" min="0" max="100" value="100"
      name="new_site_colors_light_dark_bounds" id="new_site_colors_light_dark_bounds_slider">

      <input class="input small-number-input" type="text" pattern="[-+]?d*" min="-100"
      max="100" id="new_site_colors_light_dark_bounds_number">

    </div>

  </div>

  <div class="input-slider-option two">

    <span class="text dark bold">Monochromatic Temperature</span>

    <div class="input-slider-and-number">

      <input class="input sliding-input" type="range" name="new_site_colors_monochromatic_tint"
      min="-100" max="100" value="0" id="new_site_colors_monochromatic_temperature_slider">

      <input class="input small-number-input" type="text" pattern="[-+]?d*" min="-100"
      max="100" id="new_site_colors_monochromatic_temperature_number">

    </div>

  </div>
</div>

<div class="horizontal-separator three m-s-v"></div>

<div class="color-display-container">

  <div class="color-display light-dark-display">

    <div class="color-single-card light-color" style="background-color:#ffffff;color:#000000">
      <span class="color-code text uppercase center">#ffffff</span>
    </div>

    <div class="color-single-card dark-color" style="background-color:#000000;color:#ffffff">
      <span class="color-code text uppercase center">#000000</span>
    </div>

  </div>

  <div class="color-display main-color-display">

    <div class="color-triple-card primary-color">
      <input type="color" class="color-card-picker-input" id="new_site_colors_primary_picker"
      value="#e63946">

      <div class="color-triple-card-main" style="background-color:#e63946;color:#000">
        <span class="color-code text uppercase center">#e63946</span>
      </div>

      <div class="color-triple-card-sub-container">
```

```
        <div class="color-triple-card-sub one" style="background-color:#ba2b36;color:#000"></div>
        <div class="color-triple-card-sub two" style="background-color:#f2414f;color:#000"></div>
      </div>
    </div>

    <div class="color-triple-card secondary-color">
      <input type="color" class="color-card-picker-input" id="new_site_colors_secondary_picker"
      value="#457b9d">

      <div class="color-triple-card-main" style="background-color:#457b9d;color:#000">
        <span class="color-code text uppercase center">#457b9d</span>
      </div>

      <div class="color-triple-card-sub-container">
        <div class="color-triple-card-sub one" style="background-color:#3f708f;color:#000"></div>
        <div class="color-triple-card-sub two" style="background-color:#508eb5;color:#000"></div>
      </div>
    </div>

    <div class="color-triple-card accent-color">
      <input type="color" class="color-card-picker-input" id="new_site_colors_accent_picker"
      value="#a8dadc">

      <div class="color-triple-card-main" style="background-color:#a8dadc;color:#000">
        <span class="color-code text uppercase center">#a8dadc</span>
      </div>

      <div class="color-triple-card-sub-container">
        <div class="color-triple-card-sub one" style="background-color:#93c5c7;color:#000"></div>
        <div class="color-triple-card-sub two" style="background-color:#b4ebed;color:#000"></div>
      </div>
    </div>

  </div>

  <div class="color-display grey-display">
    <div class="color-columns grey-colors expanding-columns-container">
      <div class="color-column expanding-column g900" style="background-color:#303030;color:#fff">
        <span class="color-code expanding-text text uppercase center">#303030</span>
      </div>
      <div class="color-column expanding-column g800" style="background-color:#474747;color:#fff">
        <span class="color-code expanding-text text uppercase center">#474747</span>
      </div>
      <div class="color-column expanding-column g700" style="background-color:#5e5e5e;color:#fff">
        <span class="color-code expanding-text text uppercase center">#5e5e5e</span>
      </div>
      <div class="color-column expanding-column g600" style="background-color:#757575;color:#fff">
        <span class="color-code expanding-text text uppercase center">#757575</span>
      </div>
      <div class="color-column expanding-column g500" style="background-color:#8c8c8c;color:#000">
        <span class="color-code expanding-text text uppercase center">#8c8c8c</span>
      </div>
      <div class="color-column expanding-column g400" style="background-color:#a3a3a3;color:#000">
        <span class="color-code expanding-text text uppercase center">#a3a3a3</span>
      </div>
      <div class="color-column expanding-column g300" style="background-color:#bababa;color:#000">
        <span class="color-code expanding-text text uppercase center">#bababa</span>
      </div>
      <div class="color-column expanding-column g200" style="background-color:#d1d1d1;color:#000">
        <span class="color-code expanding-text text uppercase center">#d1d1d1</span>
      </div>
      <div class="color-column expanding-column g100" style="background-color:#e8e8e8;color:#000">
        <span class="color-code expanding-text text uppercase center">#e8e8e8</span>
      </div>
```

```
      </div>
    </div>

  </div>

  <div class="submit-container">
    <button class="field-submit btn primary thin rounded slide" type="submit">
      <span class="btn-content text uppercase primary">Continue</span>
    </button>
  </div>

  <input id="color-output" class="visibly-hidden" type="text" value="."
  name="new_site_color_options_dict">

</form>

<script src="{{url_for('static', filename='js/colorConversion.js')}}"></script>
<script src="{{url_for('static', filename='js/site-create-options-1.js')}}"></script>

{% endblock %}
```

/templates/site-create-options-2.html

```
{% extends "site_create_base.html" %}

{% set navbarLogoColor = "primary" %}
{% set navbarOptionsEnabled = True %}

{% set fontsList = [
  ["Lexend","Roboto",True,True],
  ["Prata","Lato",True,True],
  ["DM Sans","Catamaran",True,True],
  ["Titillium Web","Raleway",True,True],
  ["Caudex","PT Mono",True,True],
  ["Noto Serif Display","Lora",True,True],
  ["Staatliches","Syne Mono",True,True],
]%}

{% block site_create_base %}

<link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>

<p class="text dark">Choose font family - individual elements can be customised</p>

<div class="horizontal-separator one m-s-v"></div>

<form class="new-site-form three" method="post">

  <div class="text-options">

    {%for fontList in fontsList%}

      {%set counter=fontsList.index(fontList)+1%}
      {%set headerFont=fontList[0]%}
      {%set paraFont=fontList[1]%}

      <div class="text-option {{counter}}{%if counter==1%} active{%endif%}">

        {%if fontList[2]%}
```

```
          <link href="https://fonts.googleapis.com/css2?family={{headerFont}}&display=swap"
          rel="stylesheet">

        {%endif%}

        {%if fontList[3]%}

          <link href="https://fonts.googleapis.com/css2?family={{paraFont}}&display=swap"
          rel="stylesheet">

        {%endif%}

        <div class="text-option-header text header dark one" style="font-family:'{{headerFont}}'">
          {{headerFont}}
        </div>

        <div class="text-option-paragraph text two" style="font-family:'{{paraFont}}'">
          Paragraph text - {{paraFont}}
        </div>

        <input class="visibly-hidden text-option-list" value="{{headerFont}},{{paraFont}}"
        name="new_site_font_face_list_{%if counter==1%}active{%else%}inactive{%endif%}">
      </div>

    {%endfor%}

  </div>

  <div class="submit-container">
    <button class="field-submit btn primary thin rounded slide" type="submit">
      <span class="btn-content text uppercase primary">Continue</span>
    </button>
  </div>

</form>

<script src="{{url_for('static', filename='js/site-create-options-2.js')}}"></script>

{% endblock %}
```

/templates/site-create-options-3.html

```
{% extends "site_create_base.html" %}

{% set navbarLogoColor = "primary" %}
{% set navbarOptionsEnabled = True %}

{% block site_create_base %}

<p class="text dark">Choose default styles</p>

<div class="horizontal-separator one m-s-v"></div>

<form class="new-site-form four" method="post">

  <div class="button-options">
    <div class="button-option-container">

      <div class="button-option slide-or-static">

        <div class="btn secondary thin rounded slide option-true">
```

```
          <span class="btn-content text uppercase">Sliding Button</span>
        </div>

        <div class="btn secondary thin rounded option-false">
          <span class="btn-content text uppercase">Solid Button</span>
        </div>

      </div>

    </div>

    <div class="button-option-container">

      <div class="button-option corner-options" style="opacity:0;visibility:hidden;">

        <div class="btn secondary thin square slide option-true">
          <span class="btn-content text uppercase">Squared</span>
        </div>

        <div class="btn secondary thin rounded option-false">
          <span class="btn-content text uppercase">Rounded</span>
        </div>

        <div class="btn secondary thin pill option-false">
          <span class="btn-content text uppercase">Pill</span>
        </div>

      </div>
    </div>

    <div class="button-option-container">

      <div class="button-option thin-or-large" style="opacity:0;visibility:hidden;">

        <div class="btn secondary thin rounded slide option-true">
          <span class="btn-content text uppercase">Thin</span>
        </div>

        <div class="btn secondary rounded slide option-false">
          <span class="btn-content text uppercase">Large</span>
        </div>

      </div>

    </div>

    <div class="button-option-container">

      <div class="button-option left-or-right" style="opacity:0;visibility:hidden;">

        <div class="btn secondary thin rounded slide from-left option-true">
          <span class="btn-content text uppercase">From Left</span>
        </div>

        <div class="btn secondary thin rounded slide from-right option-false">
          <span class="btn-content text uppercase">From Right</span>
        </div>

      </div>

    </div>

  </div>
```

```
  <div class="horizontal-separator two m-s-v"></div>

  <div class="submit-container">
    <button class="field-submit btn primary thin rounded slide" type="submit" disabled>
      <span class="btn-content text uppercase primary">Continue</span>
    </button>
  </div>

  <input id="style-option-output" class="visibly-hidden" type="text" value="."
  name="new_site_style_options_list">

</form>

<script src="{{url_for('static', filename='js/colorConversion.js')}}"></script>
<script src="{{url_for('static', filename='js/site-create-options-3.js')}}"></script>

{% endblock %}
```

/templates/site-create.html

```
{% extends "site_create_base.html" %}

{% set navbarLogoColor = "primary" %}
{% set navbarOptionsEnabled = True %}

{% block site_create_base %}

<p class="text dark">
  Want to import an exported site?
  <a class="link text primary">Import a website.</a>
</p>

<div class="horizontal-separator one m-s-v"></div>

<form class="new-site-form one" method="post">

  <div class="form-input-container one">

    <div class="form-input-content-column">
      <span class="text large dark one">Owner</span>
      <span class="text large dark two">
        <span>@{{current_user.user_id}}</span>
        <span class="m-m-l m-s-r">/</span>
      </span>
    </div>

    <div class="form-input-content-column">
      <span class="text large dark one">Website Name <sup class="text large danger">*</sup></span>
      <input id="new_site_name" class="new-site-input text dark two input small-text-input"
      data-form-input-display="inactive" type="text" name="new_site_name">
    </div>

    <div class="message-container m-s-t text small one visibly-hidden">
      Your site name will look like: <span class="message-container-jsedit"></span>
    </div>

    <p class="text dark m-s-t two">The name must be at least 4 characters long, and contain only
    lowercase alphanumeric characters, dashes, underscores and periods. Any illegal characters
    will be converted into dashes. It must also be unique! If you need inspiration for a name,
    you ain't gonna get any from me :)</p>
```

```
  </div>

  <div class="horizontal-separator two m-s-v"></div>

  <div class="form-input-container three">
    <span class="text large dark one">Description (Optional)</span>

    <input id="new_site_desc" class="new-site-input text dark two input small-text-input"
    type="text" name="new_site_desc">

  </div>

  <div class="horizontal-separator three m-s-v"></div>

  <div class="form-input-container two">

    <div class="input-checkbox">
      <input class="input-checkbox-input" name="new_site_privacy" id="new_site_privacy_visible"
      type="radio" value="public">

      <span class="input-checkbox-icon">
        <i class="faicon fa-regular fa-book-bookmark"></i>
      </span>

      <div class="input-checkbox-text-container">
        <span class="input-checkbox-title text bold one">Public</span>

        <span class="input-checkbox-caption text small two">
          Anyone online can see this website. Only you can edit it.
        </span>
      </div>
    </div>

    <div class="input-checkbox">
      <input class="input-checkbox-input" name="new_site_privacy" id="new_site_privacy_hidden"
      type="radio" value="private">

      <span class="input-checkbox-icon">
        <i class="faicon fa-regular fa-lock" style="color: var(--colors-warning)"></i>
      </span>

      <div class="input-checkbox-text-container">
        <span class="input-checkbox-title text bold one">Private</span>
        <span class="input-checkbox-caption text small two">
          Only people who you give the link can view the website.
        </span>
      </div>
    </div>

    <div class="horizontal-separator three m-s-v"></div>

    <button class="field-submit btn primary thin rounded slide" type="submit" disabled=true
    id="new_site_form_submit">

      <span class="btn-content text uppercase primary">Create Site</span>
    </button>

  </div>

</form>

<script>
  var flashedSiteNames = "{{get_flashed_messages()[0]}}".split(",")
</script>
```

```
<script src="{{url_for('static', filename='js/site-create.js')}}"></script>

{% endblock %}
```

/templates/site-edit-home.html

```
{% extends "base.html" %}

{% set navbarLogoColor = "gradient" %}
{% set navbarOptionsEnabled = True %}
{% set settingsSidebarActivated = get_flashed_messages()[0] %}
{% set currentName = get_flashed_messages()[1] %}
{% set currentSite = get_flashed_messages()[2] %}

{% block content %}

<link href="{{url_for('static', filename='css/settings.css')}}" rel="stylesheet" type="text/css" />

<div class="application-content">

  <div class="text-header-container">
    <h2 class="text header large dark one">{{currentSite}}</h2>
  </div>

  <div class="settings-container">

    <div class="settings-sidebar">

      <a class="settings-sidebar-item one{% if settingsSidebarActivated==1 %} is-active{% endif %}
      link notformatted" {% if not settingsSidebarActivated==1 %}
      href="{{ url_for('site_edit_home',name=currentName,site=currentSite) }}"{% endif %}>

        <i class="settings-sidebar-item-icon fa-regular fa-browser"></i>
        <span class="settings-sidebar-item-title text large">Home</span>
      </a>

      <a class="settings-sidebar-item two link notformatted" href="{{ url_for('site_edit_app',
      name=currentName,site=currentSite) }}">

        <i class="settings-sidebar-item-icon fa-regular fa-pen-to-square"></i>
        <span class="settings-sidebar-item-title text large">Edit Site</span>
      </a>

    </div>

    <div class="settings-content">
      {% block settings_content %}
      {% endblock %}
    </div>

  </div>
</div>
{% endblock %}
```

/templates/site-edit.html

```
{% extends "base.html" %}
```

```
{% set navbarLogoColor = "gradient" %}
{% set navbarOptionsEnabled = True %}
{% set settingsSidebarActivated = get_flashed_messages()[0] %}
{% set currentName = get_flashed_messages()[1] %}
{% set currentSite = get_flashed_messages()[2] %}

{% block content %}

<link href="{{url_for('static', filename='css/site-edit.css')}}" rel="stylesheet" type="text/css"/>
<link href="{{url_for('static', filename='css/edit.css')}}" rel="stylesheet" type="text/css" />

<link href="{{url_for('static', filename='css/preview_override.css')}}" rel="stylesheet"
type="text/css" />

<script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.0/jquery.min.js"></script>

<div class="lightbox-mask"></div>

<div class="application-content">

  <div class="localnav one localnav-vertical localnav-collapsible">
    <div class="localnav-content">
      <ul class="localnav-list">

        <ul class="localnav-list" style="height:30%">

          <li class="localnav-item one">
            <a class="link unformatted">
              <div class="localnav-item-collapsible-icon">
                <i class="faicon fa-solid fa-layer-group"></i>
              </div>
              <div class="localnav-item-collapsible-text text bold primary">Website Pages</div>
            </a>
          </li>

          <li class="localnav-item two">
            <a class="link unformatted" id="localnav_add_section_btn">
              <div class="localnav-item-collapsible-icon">
                <i class="faicon fa-solid fa-circle-plus"></i>
              </div>
              <div class="localnav-item-collapsible-text text bold primary">Add Section</div>
            </a>
          </li>

          <li class="localnav-item three">
            <a class="link unformatted">
              <div class="localnav-item-collapsible-icon">
                <i class="faicon fa-solid fa-paintbrush"></i>
              </div>
              <div class="localnav-item-collapsible-text text bold primary">Website Styles</div>
            </a>
          </li>

        </ul>

        <ul class="localnav-list" style="height:fit-content">

          <li class="localnav-item four">
            <a class="link unformatted" id="localnav_save">

              <div class="localnav-item-collapsible-icon">
                <i class="faicon fa-solid fa-save"></i>
              </div>
```

```
            <div class="localnav-item-collapsible-text text bold primary">Save Website</div>

            <div class="localnav-item-collapsible-icon saveloader">
              <i class="faicon fa-solid fa-loader"></i>
            </div>

            <div class="localnav-item-collapsible-icon savetick">
              <i class="faicon fa-solid fa-check"></i>
            </div>

            <div class="localnav-item-collapsible-icon savecross">
              <i class="faicon fa-solid fa-xmark"></i>
            </div>
          </a>
        </li>

        <li class="localnav-item five">
          <a class="link unformatted">
            <div class="localnav-item-collapsible-icon">
              <i class="faicon fa-solid fa-gear"></i>
            </div>

            <div class="localnav-item-collapsible-text text bold primary">Website Settings</div>
          </a>
        </li>

      </ul>
    </ul>

  </div>

</div>

<div class="section-selector-container">

  <div class="section-selector">

    <div class="section-selector-exit-btn"><i class="faicon fa-solid fa-xmark"></i></div>

    <div class="section-selector-sidebar">
      <div class="section-selector-nav" id="section_selector_nav">
        <div class="section-selector-nav-content">
          <ul class="section-selector-nav-list">
          </ul>
          </ul>
        </div>
      </div>
    </div>

    <div class="vertical-separator" style="height:85%"></div>

    <div class="section-selector-content">
      <div class="section-selector-list" id="section_selector_list">
        <style id="section_selector_style"></style>
      </div>
    </div>

  </div>

</div>

<div class="site-builder">

  <div class="site-builder-preview" id="contains_site">
```

```
    </div>

    <div class="section-edit-actions">
      <div class="actions-list">
        <div class="edit-action" id="section_edit_action_settings">
          <i class="fa-regular fa-gear"></i>
        </div>

        <div class="edit-action" id="section_edit_action_duplicate">
          <i class="fa-regular fa-clone"></i>
        </div>

        <div class="edit-action" id="section_edit_action_lock">
          <i class="fa-regular fa-lock"></i>
        </div>

        <div class="edit-action hilight-danger" id="section_edit_action_delete">
          <i class="fa-regular fa-trash"></i>
        </div>
      </div>
    </div>

  </div>

</div>

<script src="{{url_for('static', filename='js/site-edit.js')}}"></script>

{% endblock %}
```