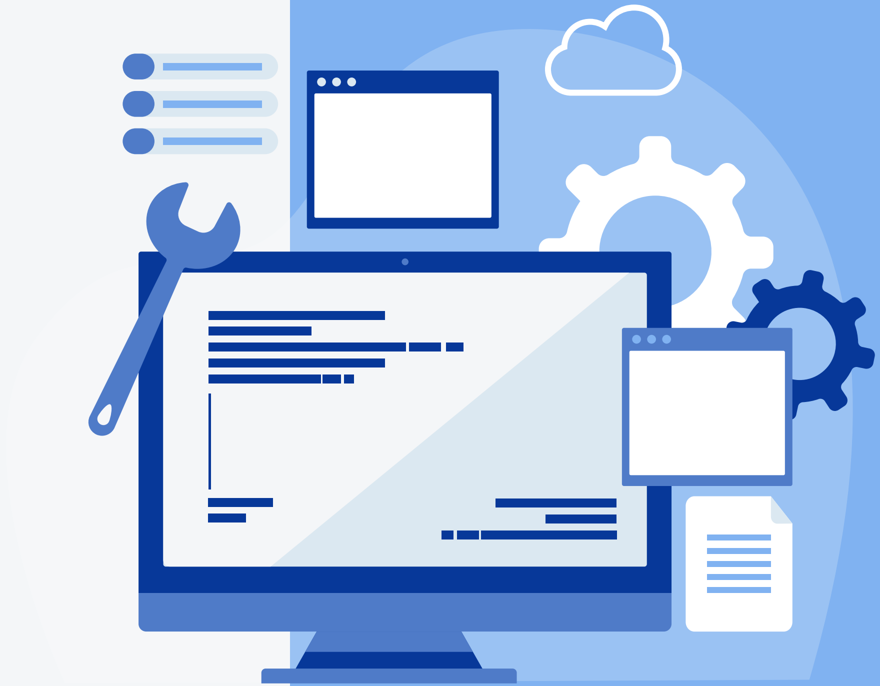
Boolient Internship Project 01

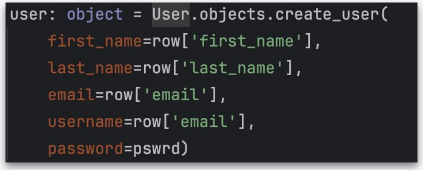


Objectives

- Automatically create an account for new users when given a list and also denies the request for duplicate accounts
- Send a welcome email with temporary login information to new users
- Return list of successfully created users
- Integrate command into webpage

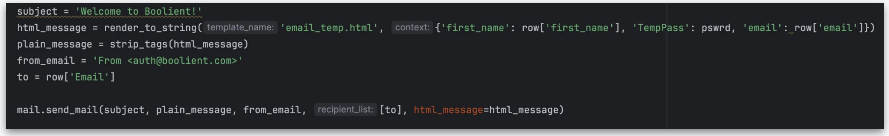
Using Django

- I used the Django framework:
  - To create users



```
user: object = User.objects.create_user(
    first_name=row['first_name'],
    last_name=row['last_name'],
    email=row['email'],
    username=row['email'],
    password=pswrd)
```

  - Send HTML-formatted mail to accepted users



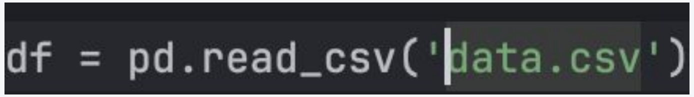
```
subject = 'Welcome to Boolient!'
html_message = render_to_string( template_name: 'email_temp.html',  context: {'first_name': row['first_name'], 'TempPass': pswrd, 'email': row['email']})
plain_message = strip_tags(html_message)
from_email = 'From <auth@boolient.com>'
to = row['Email']

mail.send_mail(subject, plain_message, from_email,  recipient_list: [to], html_message=html_message)
```

  - For their default user/admin interface

Using Pandas

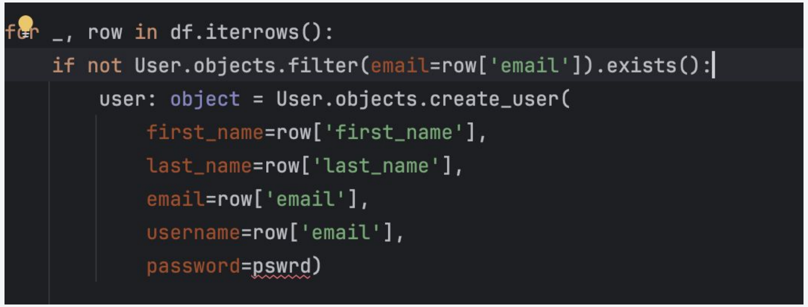
- I used the Pandas framework to:
  - Read comma-separated values



```
df = pd.read_csv('data.csv')
```

  - Iterate through rows of the df

1st Draft



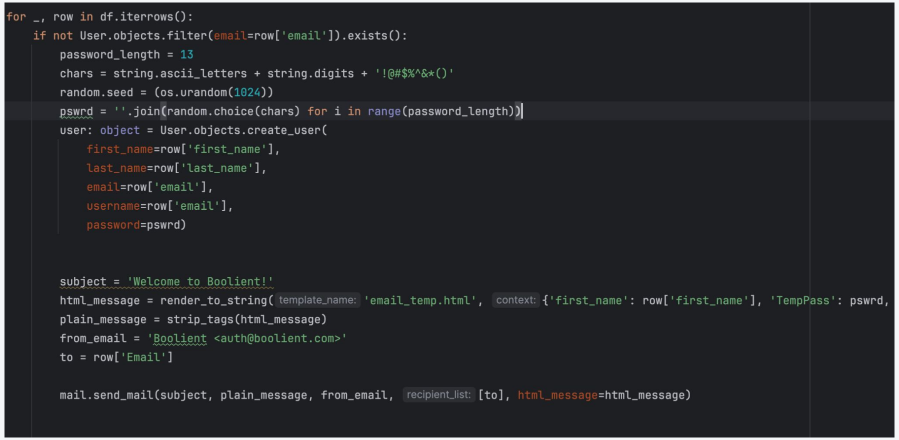
```
for _, row in df.iterrows():
    if not User.objects.filter(email=row['email']).exists():
        user: object = User.objects.create_user(
            first_name=row['first_name'],
            last_name=row['last_name'],
            email=row['email'],
            username=row['email'],
            password=pswrd)
```

- Functional code that could iterate through a CSV dataframe + pick out accounts already in the database
- Could create an account for new, unique users that passed the check

Caveats:

- Could not send email of verification w/ temporary password
- Could not be accessed through the website

2nd Draft



```
for _, row in df.iterrows():
    if not User.objects.filter(email=row['email']).exists():
        password_length = 13
        chars = string.ascii_letters + string.digits + '!@#$%^&*()'
        random.seed = (os.urandom(1024))
        pswrd = ''.join(random.choice(chars) for i in range(password_length))
        user: object = User.objects.create_user(
            first_name=row['first_name'],
            last_name=row['last_name'],
            email=row['email'],
            username=row['email'],
            password=pswrd)


        subject = 'Welcome to Boolient!'
        html_message = render_to_string( template_name: 'email_temp.html',  context: {'first_name': row['first_name'], 'TempPass': pswrd,
        plain_message = strip_tags(html_message)
        from_email = 'Boolient <auth@boolient.com>'
        to = row['Email']

        mail.send_mail(subject, plain_message, from_email,  recipient_list: [to], html_message=html_message)
```

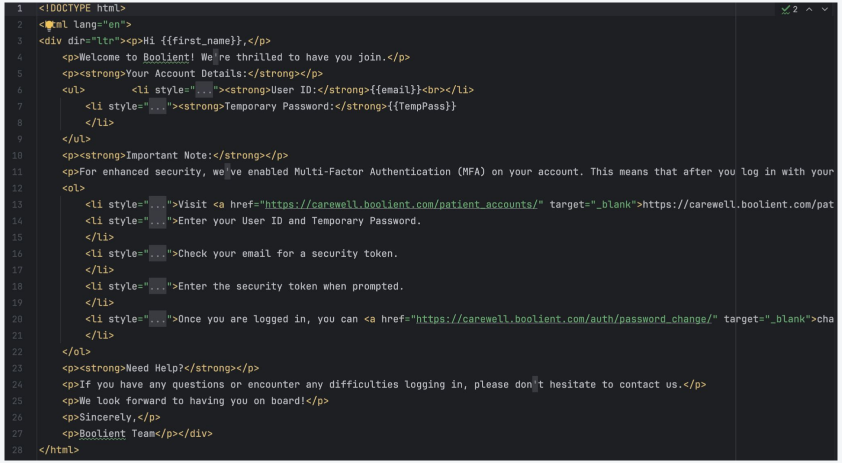
- New user now receives a verification email w/ instructions to set up account

Caveats

- Function not integrated into website
- Struggling to format mail into HTML using the normal Django send_mail function

3rd Draft

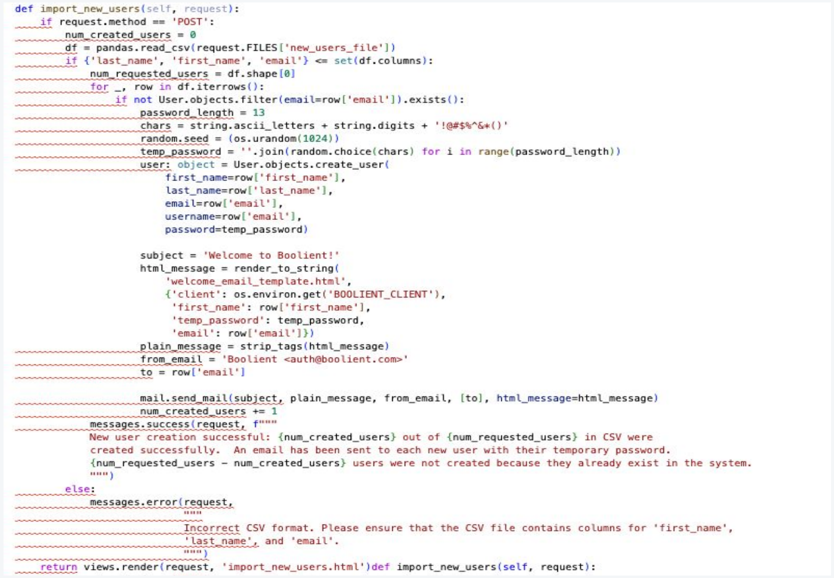
- Create an HTML template for the verification email



```html
<!DOCTYPE html>
<html lang="en">
<div dir="ltr"><p>Hi {{first_name}},</p>
    <p>Welcome to Boolient! We're thrilled to have you join.</p>
    <p><strong>Your Account Details:</strong></p>
    <ul>        <li style="..."><strong>User ID:</strong>{{email}}<br></li>
        <li style="..."><strong>Temporary Password:</strong>{{TempPass}}
        </li>
    </ul>
    <p><strong>Important Note:</strong></p>
    <p>For enhanced security, we've enabled Multi-Factor Authentication (MFA) on your account. This means that after you log in with your
    <ol>
        <li style="...">Visit <a href="https://carewell.boolient.com/patient_accounts/" target="_blank">https://carewell.boolient.com/pat
        <li style="...">Enter your User ID and Temporary Password.
        </li>
        <li style="...">Check your email for a security token.
        </li>
        <li style="...">Enter the security token when prompted.
        </li>
        <li style="...">Once you are logged in, you can <a href="https://carewell.boolient.com/auth/password_change/" target="_blank">cha
        </li>
    </ol>
    <p><strong>Need Help?</strong></p>
    <p>If you have any questions or encounter any difficulties logging in, please don't hesitate to contact us.</p>
    <p>We look forward to having you on board!</p>
    <p>Sincerely,</p>
    <p>Boolient Team</p></div>
</html>
```

FINAL Draft

- Implement error message
- Sends confirmation to admin for processing



```python
def import_new_users(self, request):
    if request.method == 'POST':
        num_created_users = 0
        df = pandas.read_csv(request.FILES['new_users_file'])
        if {'last_name', 'first_name', 'email'} <= set(df.columns):
            num_requested_users = df.shape[0]
            for _, row in df.iterrows():
                if not User.objects.filter(email=row['email']).exists():
                    password_length = 13
                    chars = string.ascii_letters + string.digits + '!@#$%^&*()'
                    random.seed = (os.urandom(1024))
                    temp_password = ''.join(random.choice(chars) for i in range(password_length))
                    user: object = User.objects.create_user(
                        first_name=row['first_name'],
                        last_name=row['last_name'],
                        email=row['email'],
                        username=row['email'],
                        password=temp_password)

                    subject = 'Welcome to Boolient!'
                    html_message = render_to_string(
                        'welcome_email_template.html',
                        {'client': os.environ.get('BOOLIENT_CLIENT'),
                         'first_name': row['first_name'],
                         'temp_password': temp_password,
                         'email': row['email']})
                    plain_message = strip_tags(html_message)
                    from_email = 'Boolient <auth@boolient.com>'
                    to = row['email']

                    mail.send_mail(subject, plain_message, from_email, [to], html_message=html_message)
                    num_created_users += 1
            messages.success(request, f"""
            New user creation successful: {num_created_users} out of {num_requested_users} in CSV were
            created successfully.  An email has been sent to each new user with their temporary password.
            {num_requested_users - num_created_users} users were not created because they already exist in the system.
            """)
        else:
            messages.error(request,
                           """
                           Incorrect CSV format. Please ensure that the CSV file contains columns for 'first_name',
                           'last_name', and 'email'.
                           """)
    return views.render(request, 'import_new_users.html')def import_new_users(self, request):
```