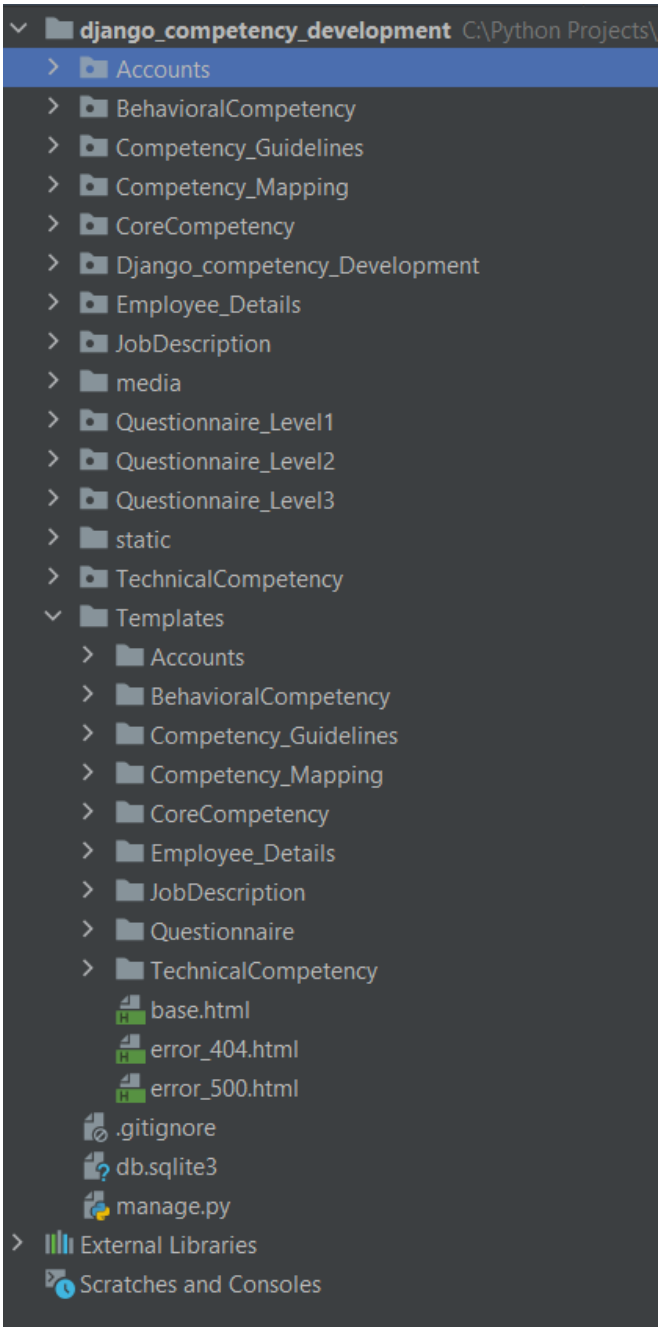
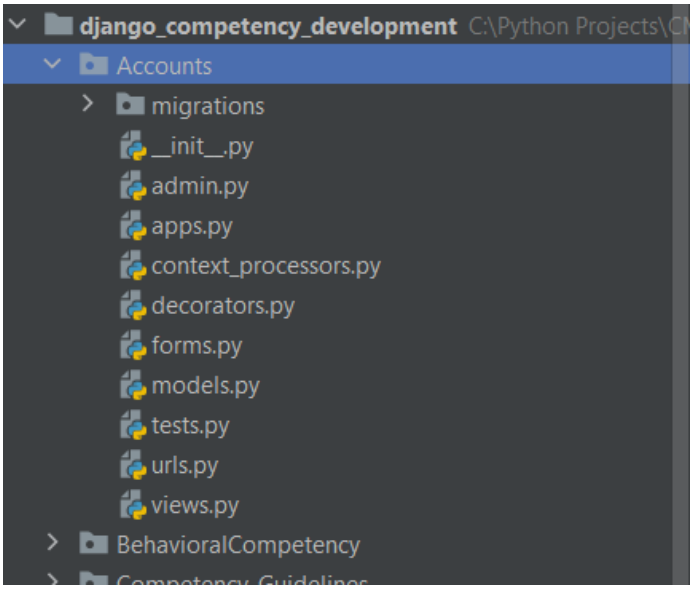
Competency Management Solution

Screenshot

Code structure – Python Django Web Application




django_competency_development C:\Python Projects\C
Accounts
migrations
__init__.py
admin.py
apps.py
context_processors.py
decorators.py
forms.py
models.py
tests.py
urls.py
views.py
BehavioralCompetency

Code Snippets:

Login Code:

```python
from django.shortcuts import render, redirect, HttpResponse
from django.contrib.auth.models import User, auth
from django.contrib import messages
from .forms import Top_Management_SignUp_Form, HOD_SignUp_Form, Individual_SignUp_Form, Competency_Accessor_SignUp_Form
from Accounts.models import User
from Employee_Details.views import Employee_Data
from django.contrib.auth.decorators import login_required
from Employee_Details.models import Setup
# Create your views here.


def login(request):
    if request.method == 'POST':
        username = request.POST['username']
        password1 = request.POST['password1']
        user = auth.authenticate(username=username, password=password1)
        if user is not None:
            auth.login(request, user)
            return redirect("profile")
        else:
            messages.error(request, 'Invalid Credentials')
            return render(request, "Accounts/login.html")
    else:
        return render(request, 'Accounts/login.html')

```

```python
def top_management_signup(request, email_str):
    email = email_str
    form = Top_Management_SignUp_Form(request.POST)
    users = User.objects.values('username')
    row_count = len(users)
    if request.method == 'POST':
        username = request.POST['username']
        email = request.POST['email']
        password1 = request.POST['password1']
        password2 = request.POST['password2']
        for i in range(0, row_count):
            to_list = [users[i]['username']]
            print(to_list)
            if username in to_list:
                messages.error(request, 'User Already Exists')
                return redirect('top_management_signup', email)
            else:
                if password1 == password2 and len(password2 and password1) >= 8 and len(password2 and password1) <= 20:
                    if form.is_valid():
                        form.save()
                        return redirect('login')
                    else:
                        messages.error(request, 'User Not submit pls full fil the Condition')
                        return redirect('top_management_signup', email)
                else:
                    messages.error(request, 'Pls Fill Correct Password credentials')
                    return redirect('top_management_signup', email)
    else:
        form1 = Top_Management_SignUp_Form()
        return render(request, "Accounts/signup_form.html", {'form1': form1, 'email': email})

```

Connection with SQL server and populating in html form

```python
def level1_managerial_edit(request):
    form_data_dict = {}
    comp_detail = Employee_Data.objects.values('comp_name', 'comp_branch').filter(emp_email=request.user).distinct()
    comp = comp_detail.values('comp_name')[0]['comp_name']
    branch = comp_detail.values('comp_branch')[0]['comp_branch']
    level = '1'
    type = 'Managerial'
    # Distinct Competency Name
    question_bank_l1_managerial_comp = QuestionBank.objects.values('competency_name').filter(type=type,
                                                                                             Level=level).distinct()
    # Complete table
    question_bank_l1_managerial = QuestionBank.objects.values('competency_name', 'KSPA', 'statement',
                                                              'comp_description').filter(type=type,
                                                                                         Level=level).distinct()
    if request.method == 'POST':
        form_data_dict = dict(request.POST.lists())
        delete_data = QuestionBank.objects.filter(type=type, Level=level).all()
        delete_data.delete()
        row_count = len(form_data_dict.get('statement'))
        for i in range(0, row_count):
            QuestionBank(comp_name=comp,
                         branch_name=branch,
                         type=type,
                         Level=level,
                         competency_name=form_data_dict['competency_name'][i],
                         KSPA=form_data_dict['KSPA'][i],
                         comp_description=form_data_dict['comp_description'][i],
                         statement=form_data_dict['statement'][i],
                         ).save()
        return redirect('level1_managerial_view')
    parm = {'comp': question_bank_l1_managerial_comp, 'value': question_bank_l1_managerial}
    return render(request, 'Questionnaire/Question_Bank/Managerial_Level1_edit.html', parm)
```

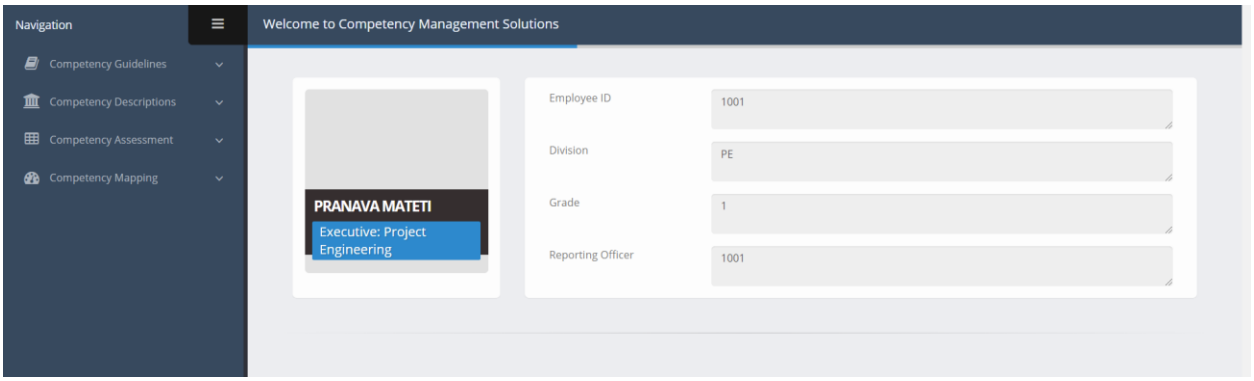
Dashboard population

```python
# Mapping sheet
@login_required(login_url='login')
def emp_competency_mapping(request, emp_id):
    distinctdetails = Employee_Data.objects.values('emp_id', 'designation', 'emp_email', 'emp_name', 'division',
                                                   'grade').filter(emp_id=emp_id).distinct()
    # Core Competencies
    core_data = Response_Table_Ques_Level3.objects.all().filter(emp_id=emp_id, KSPA='Not Applicable')

    # Managerial Competencies
    manage_data = Response_Table_Ques_Level3.objects.all().filter(emp_id=emp_id, task_title='Not Applicable').exclude(
        KSPA='Not Applicable')

    # Functional Competencies
    func_data = Response_Table_Ques_Level3.objects.all().filter(emp_id=emp_id).exclude(task_title='Not Applicable',
                                                                                       KSPA='Not Applicable')

    arg = {'distinctdetails': distinctdetails, 'core_data': core_data, 'manage_data': manage_data,
           'func_data': func_data}
    return render(request, 'Competency_Mapping/Emp_Competency_Mapping.html', arg)
```

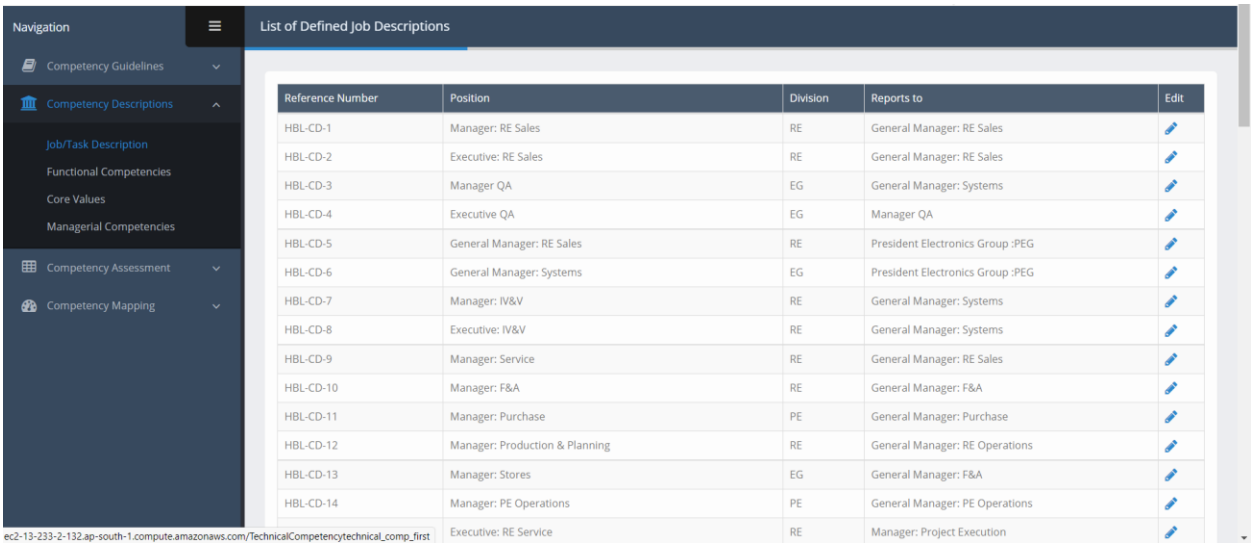
```python
## Functions to calculate dataset values dynamically
def chart_function_behav(designation, division, KSPA, score):
    selected_data_behav = Response_Table_Ques_Level3.objects.values(score).filter(KSPA=KSPA,
                                                                                  designation=designation,
                                                                                  division=division,
                                                                                  task_title='Not Applicable').aggregate(
        Avg(score))
    return selected_data_behav


def chart_function_tech(designation, division, KSPA, score):
    selected_data_tech = Response_Table_Ques_Level3.objects.values_list(score).filter(KSPA=KSPA,
                                                                                      designation=designation,
                                                                                      division=division).exclude(
        task_title='Not Applicable').aggregate(Avg(score))
    return selected_data_tech
```

Front End

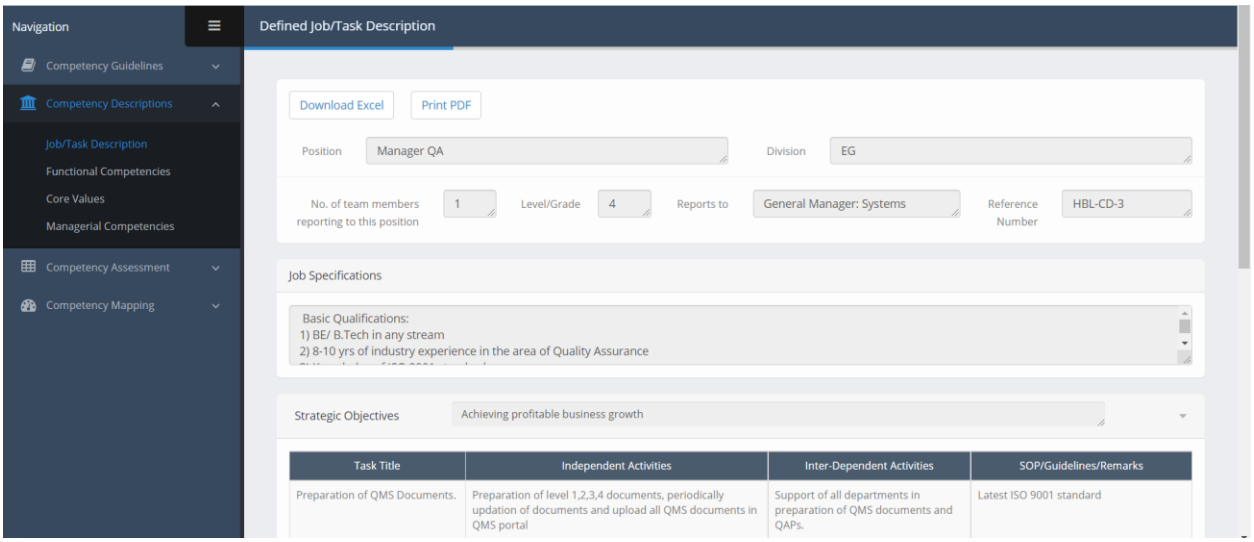
Default Page



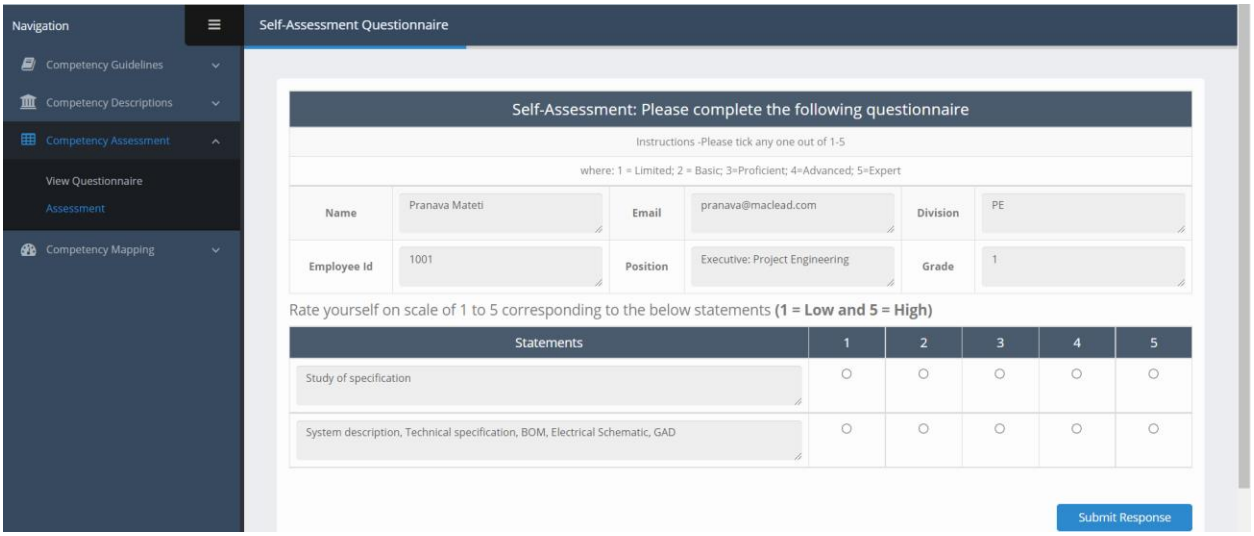
List


Navigation
Competency Guidelines
Competency Descriptions
Job/Task Description
Functional Competencies
Core Values
Managerial Competencies
Competency Assessment
Competency Mapping
List of Defined Job Descriptions
Reference Number
Position
Division
Reports to
Edit
HBL-CD-1 Manager: RE Sales RE General Manager: RE Sales
HBL-CD-2 Executive: RE Sales RE General Manager: RE Sales
HBL-CD-3 Manager QA EG General Manager: Systems
HBL-CD-4 Executive QA EG Manager QA
HBL-CD-5 General Manager: RE Sales RE President Electronics Group :PEG
HBL-CD-6 General Manager: Systems EG President Electronics Group :PEG
HBL-CD-7 Manager: IV&V RE General Manager: Systems
HBL-CD-8 Executive: IV&V RE General Manager: Systems
HBL-CD-9 Manager: Service RE General Manager: RE Sales
HBL-CD-10 Manager: F&A RE General Manager: F&A
HBL-CD-11 Manager: Purchase PE General Manager: Purchase
HBL-CD-12 Manager: Production & Planning RE General Manager: RE Operations
HBL-CD-13 Manager: Stores EG General Manager: F&A
HBL-CD-14 Manager: PE Operations PE General Manager: PE Operations
Executive: RE Service RE Manager: Project Execution
ec2-13-233-2-132.ap-south-1.compute.amazonaws.com/TechnicalCompetencytechnical_comp_first

Form


Navigation
Competency Guidelines
Competency Descriptions
Job/Task Description
Functional Competencies
Core Values
Managerial Competencies
Competency Assessment
Competency Mapping
Defined Job/Task Description
Download Excel
Print PDF
Position
Manager QA
Division
EG
No. of team members reporting to this position
1
Level/Grade
4
Reports to
General Manager: Systems
Reference Number
HBL-CD-3
Job Specifications
Basic Qualifications:
1) BE/ B.Tech in any stream
2) 8-10 yrs of industry experience in the area of Quality Assurance
Strategic Objectives
Achieving profitable business growth
Task Title
Independent Activities
Inter-Dependent Activities
SOP/Guidelines/Remarks
Preparation of QMS Documents.
Preparation of level 1,2,3,4 documents, periodically updation of documents and upload all QMS documents in QMS portal
Support of all departments in preparation of QMS documents and QAPs.
Latest ISO 9001 standard

Questionnaire



Dashboard