Introduction

Now a days performance and ability is must require in corporate industry. They choose a perfect employee whose can do their work on time and accurate, there are lot of employee for in the organization, but this is very tedious task for organization to keep track each of employee's productivity, accuracy and performance and manage their task.

TeamWorks is the task management system which can track task(work), productivity and performance of each employee in the organization, and make report for their task pendency and total work flow for team this is also a system of team collaboration. Task management is the process of managing tasks through a life cycle. This is like a discipline as a ball in game of football where a ball pass from player to player for goal. This is this system where user can create an task for their task and assign to their team member for completion then team member check their pending task and change the status of task after the complete task. Every task could have status, start date, created by, assigned to, task, comments, and attached files, it is also support dependency, priority, maximum time to complete,

Objective

- The main objective to make this project team collaboration and manage their task and productivity
- This project is reduce manual work of track each employee's task day by day
- This will lead to increase the transparency of each employee.
- This project make user friendly as possible so that anyone can use has little knowledge of computer.

Purpose

The main purpose to make this project is analyzing all the projects of organization and divided into subtask then distribute to all team member this make a track to particular project and its task running or complete status.

The second main purpose is to keep track of all employee working such as what they do, how much work is pending to his hand, in how much time is take by him to complete a particular task.

Another purpose is though this project organization has complete and summaries report of all the employee performance and total work pending to any specific team in organization.

Survey of Technology

Python

Python is a widely used high-level, general-purpose, interpreted, dynamic programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than would be possible in languages such as C++ or Java. The language provides constructs intended to enable clear programs on both a small and large scale.

Python supports multiple programming paradigms, including object-oriented, imperative and functional programming or procedural styles. It features a dynamic type system and automatic memory management and has a large and comprehensive standard library.

Python is a multi-paradigm programming language: object-oriented programming and structured programming are fully supported, and there are a number of language features which support functional programming and aspect-oriented programming (including by meta programming and by magic methods). Many other paradigms are supported using extensions, including design by contract and logic programming.

Django

Django is a high-level Python **Web framework** that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so we can focus on writing your app without needing to reinvent the wheel. It’s free and open source.

Django's primary goal is to ease the creation of complex, database-driven websites. Django emphasizes reusability and "pluggability" of components, rapid development, and the principle of don't repeat yourself. Python is used throughout, even for settings, files, and data models. Django also provides an optional administrative create, read, update and delete interface that is generated dynamically through introspection and configured via models.

MVC architecture

Model–view–controller (MVC) is a software architectural pattern mostly (but not exclusively) for implementing user interfaces on computers. It divides a given software application into three interconnected parts, so as to separate internal representations of information from the ways that information is

presented to or accepted from the user.

- **Model** : The model directly manages the data, logic and rules of the application.
- **View** :A view can be any output representation of information, such as a chart or a diagram. Multiple views of the same information are possible, such as a bar chart for management and a tabular view for accountants.
- **Controller** : The third part, the controller, accepts input and converts it to commands for the model or view.

RESTful API

RESTful web services are one way of providing interoperability between computer systems on the internet. REST-compliant web services allow requesting systems to access and manipulate textual representations of web resources using a uniform and predefined set of stateless operations. Other forms of web service exist, which expose their own arbitrary sets of operations such as via WSDL and SOAP. 'Web resources' were first defined on the World Wide Web as documents or files identified by their URLs, but today they have a much more generic and abstract definition encompassing every 'thing' or entity

that can be identified, named, addressed or handled, in any way whatsoever, in the web. In a REST web service, requests made to a resource's URI will elicit a response that may be in XML, HTML, JSON or some other defined format. The response may confirm that some alteration has been made to the stored resource, and it may provide hypertext links to other related resources or collections of resources. Using HTTP, as is most common, the kind of operations available include those predefined by the HTTP verbs GET, POST, PUT, DELETE and so on.

Backbonejs

Backbone.js is a JavaScript framework with a RESTful JSON interface and is based on the model–view–presenter (MVP) application design paradigm. Backbone is known for being lightweight, as its only hard dependency is on one JavaScript library, Underscore.js, plus jQuery for use of the full library. It is designed for developing single-page web applications, and for keeping various parts of web applications (e.g. multiple clients and the server) synchronized.Backbone was created by Jeremy Ashkenas, who is also known for CoffeeScript and Underscore.js

Problem definition

SRS(Software Requirement specification)

Introduction :- TeamWorks is the a web application is which is built for an organization or company to manage their work and task. It is also platform of team collaboration. With this application organization can track each work and employee.

Purpose: - The main purpose to make this project is team collaboration and tracking task within the organization. It is difficult to manage task with excel the other system. This is user-friendly system to accomplished all requirement to track the task and employee.

Scope:- On this platform manager can create task and project which can edit or update by employee as processing their task. it will save their time to manage their employee and work.

Product perspective:- TeamWorks is aimed towards a organization who want to track their work, employee and team collaboration. it is help to organization to save time and make the process automatic and smooth.

Function:- In this application there are different type of function such as task or

project creation, deletion and modification with associate and manager and for admin has user or group creation, deletion and updation. It also has permission module to give particular permission to manager or associate.

User Characteristics:-there are three user of this project

- Admin : admin has full rights to make changes in application, user and groups.
- Manager: manager has rights to create, delete and modify the Task and assign to associate.
- Associate : associate only has rights to update/edit the task and assigning to another to this.
- Department Head : It has rights to create/delete projects.

Functional requirements:-

- Login
- Create/Delete/Update the tasks/project
- Create/delete/update Users/Groups by admin
- Compose/received/Trash email wihtin organization.
- Search the tasks/Email
- View Reports/Export report in CSV

Performance requirements:- It should run in Intel P4 or higher with 512 MB RAM. And every page is response within 2 seconds.

Requirement Specification

Existing System

- Existing system is an MS-Excel based system,
- Project team members had to maintain the status of their assigned task on their individual desks.
- On demand reports are prepared by collecting status information from individuals, and printed or softcopy is submitted to relevant authority.

Drawbacks

- Spread sheet files are maintained on individual desktop, that create lot of redundancy and discrepancy
- Report generations on ad-hoc basis is not possible, even to know status of the project, required to look in hundreds files
- It is difficult to check that which work has been done and which are the pendency.
- It will take enough time to collect full status of each task of employee this process also exhaust the productivity of employee.

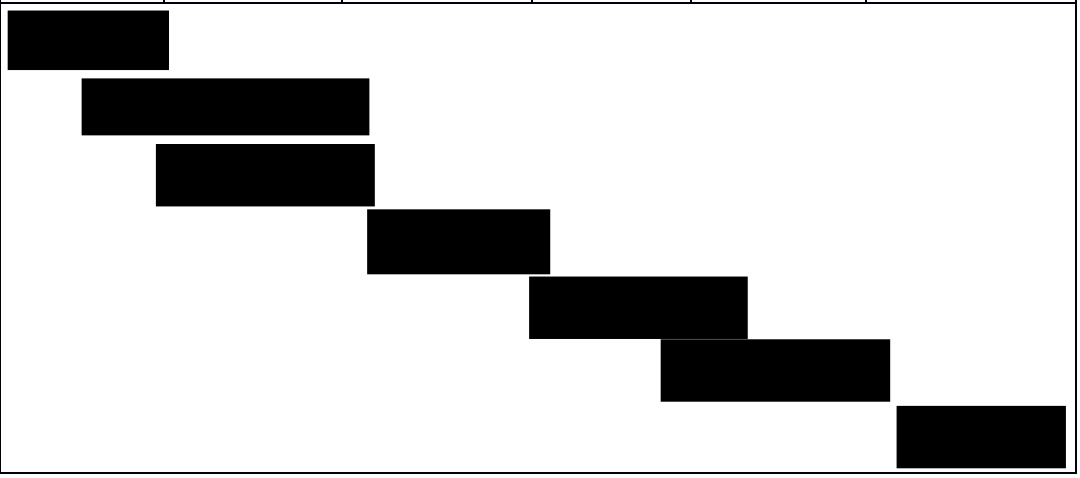
Proposed System

- Proposed system is web based project or task management system.
- It control all the projects and its task(task) with their current status
- Reflect accurate status each of all project and its task at any instance of time
- Ease the work of project leader or manager to check their which work has been done and how much is pending
- It enhance the communication among the team members using assigning the work to each other until complete the task.
- This project also control the each employee productivity and accuracy of work on particular project or task.
- It will save the time of manager or project leader to manage their team member.
- It prevent the redundancy of task.

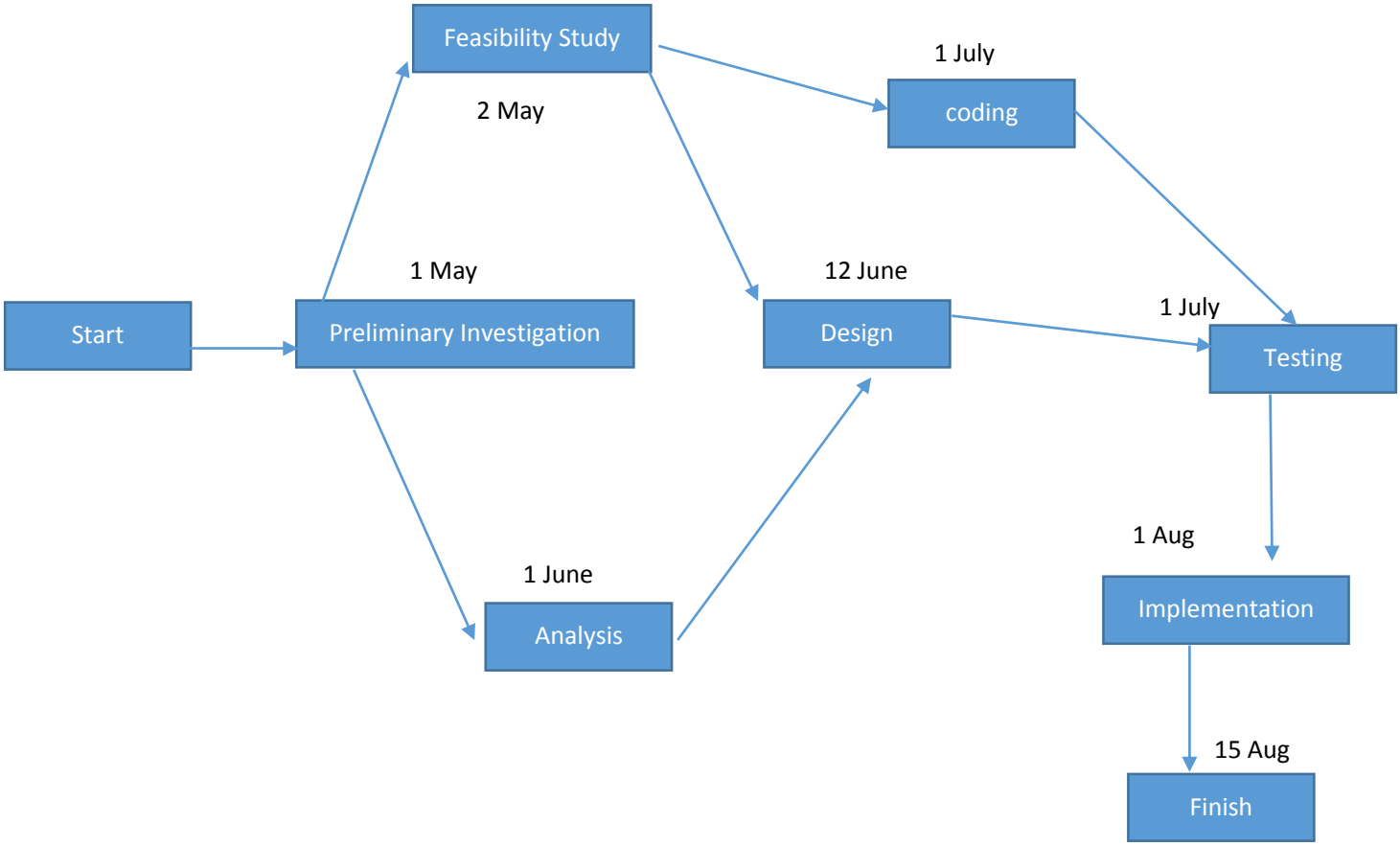
Planning and Scheduling

Gant Chart

Task	01 May –6 May	1 June – 10 June	12 June –25 June	1 July -20 July	25 July - 30 July	1 Aug - 15 Aug
Preliminary Investigation						
Feasibility Study						
Requirement Analysis						
Design						
Coding						
Testing						
Implement						



Pert Chart


Feasibility Study
1 July
coding
2 May
1 May
12 June
Start
Preliminary Investigation
Design
1 July
Testing
1 Aug
1 June
Analysis
Implementation
15 Aug
Finish

Software and Hardware Requirements

Software Interface

Server :

Frontend : Python

Framework : Django, Bootstrap

Backend : Sqlite3

Operating System : Linux

Client :

Operating System : Windows/Linux/Mac OSX

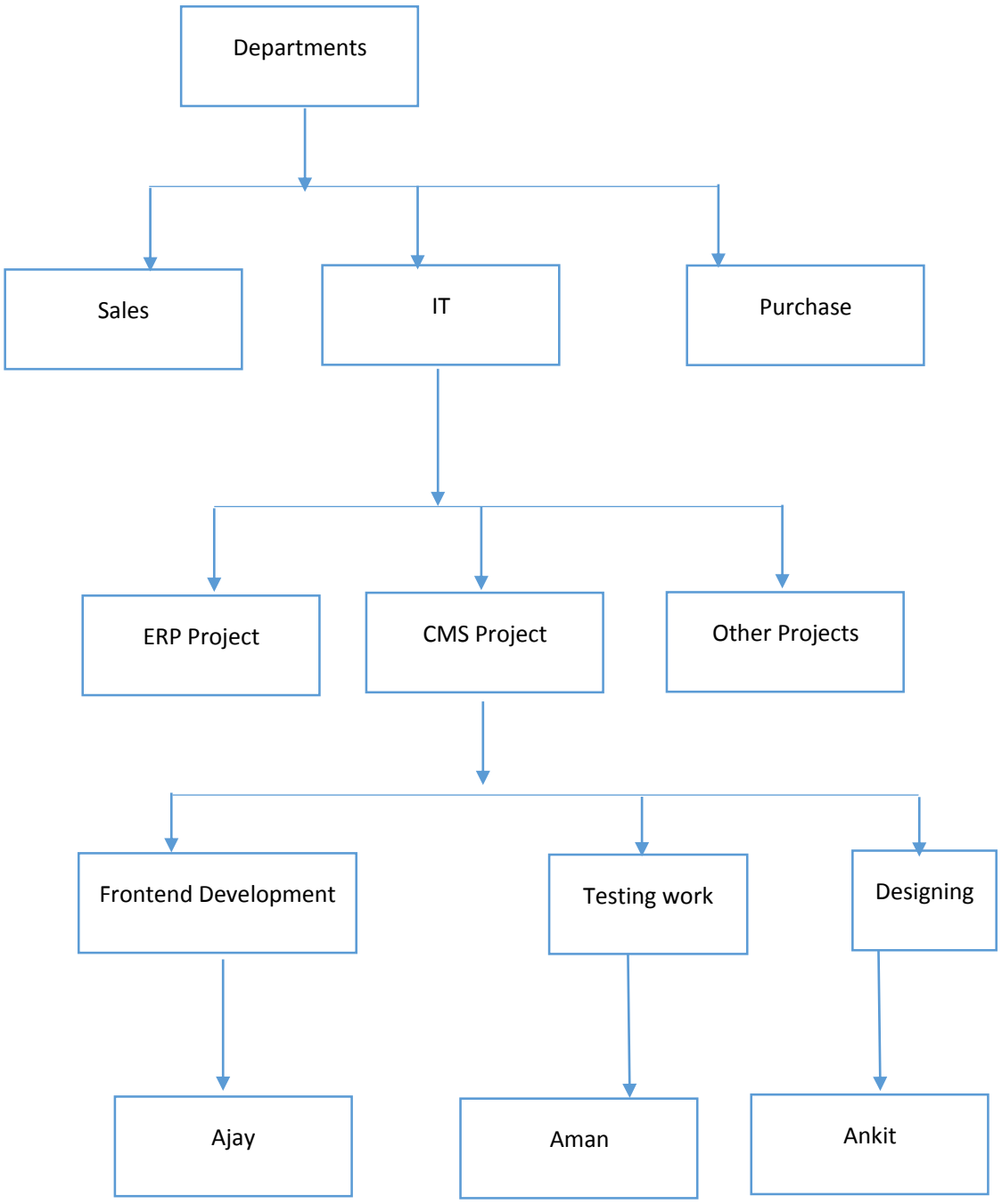
Web Browser : Chrome/Internet Explorer/Mozilla FireFox

Hardware Interface

Server : Pentium Series 2 GB RAM

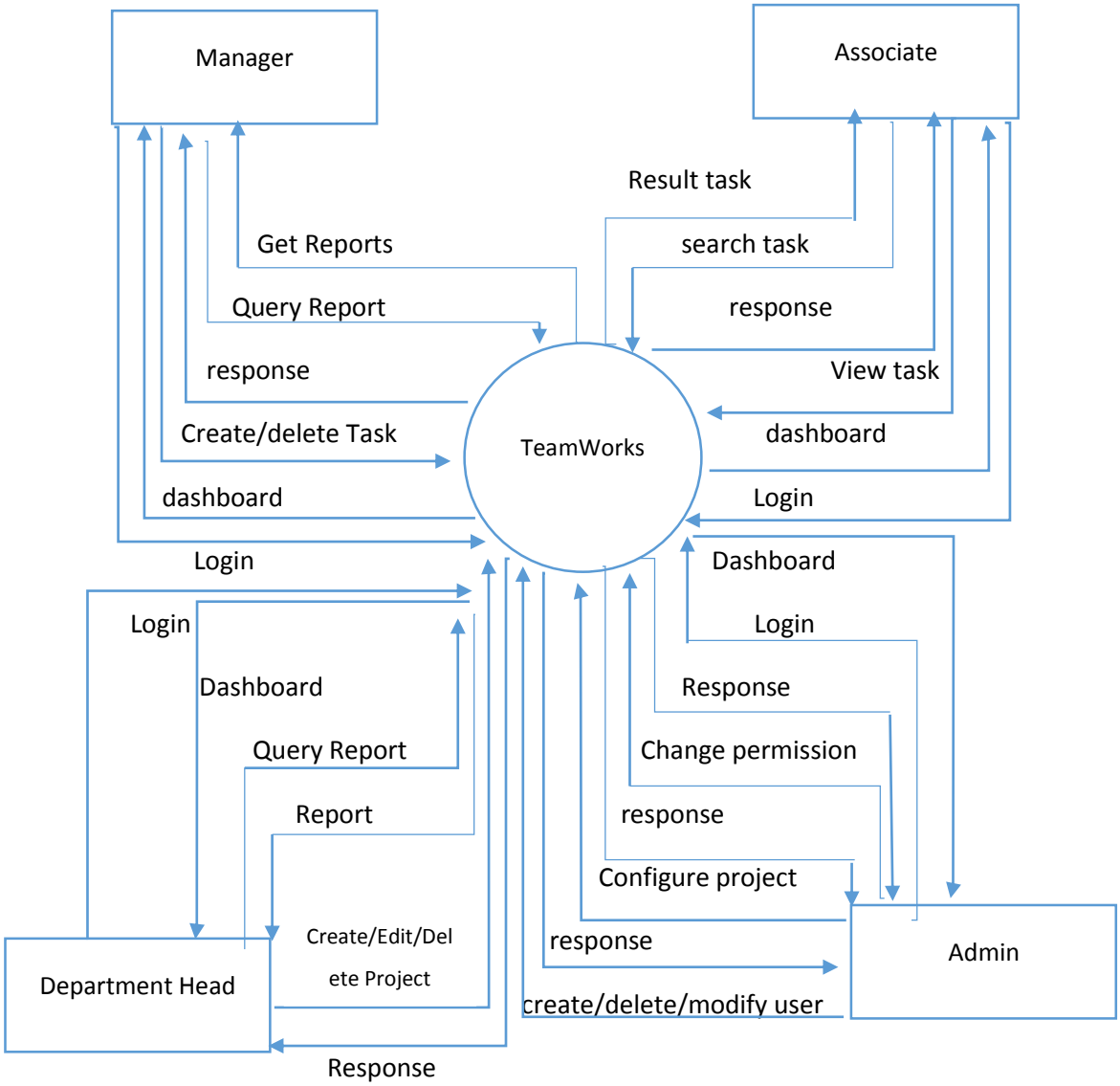
Client : Pentium Series 1 GB RAM

Conceptual Model

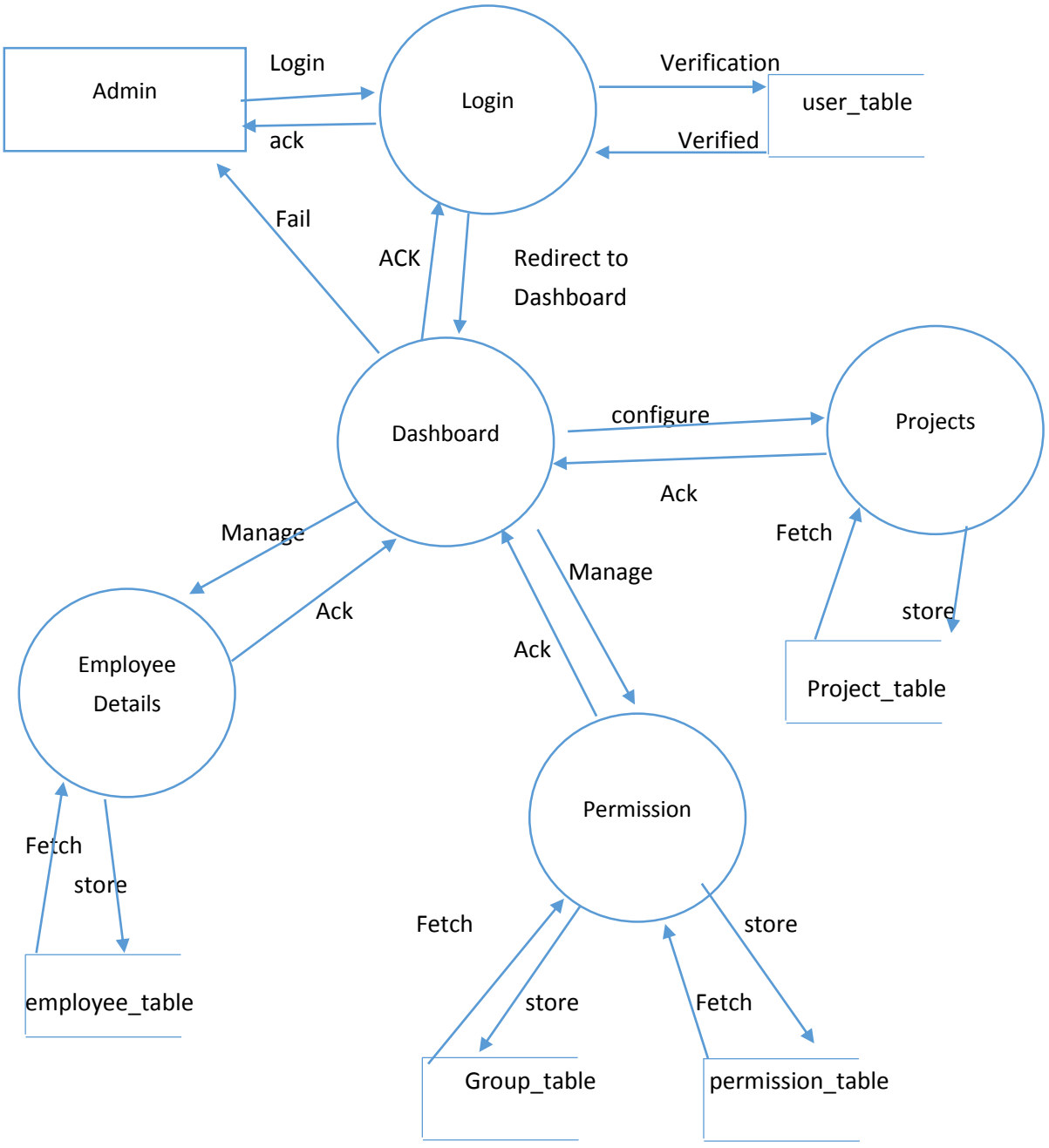


Data Flow Diagram

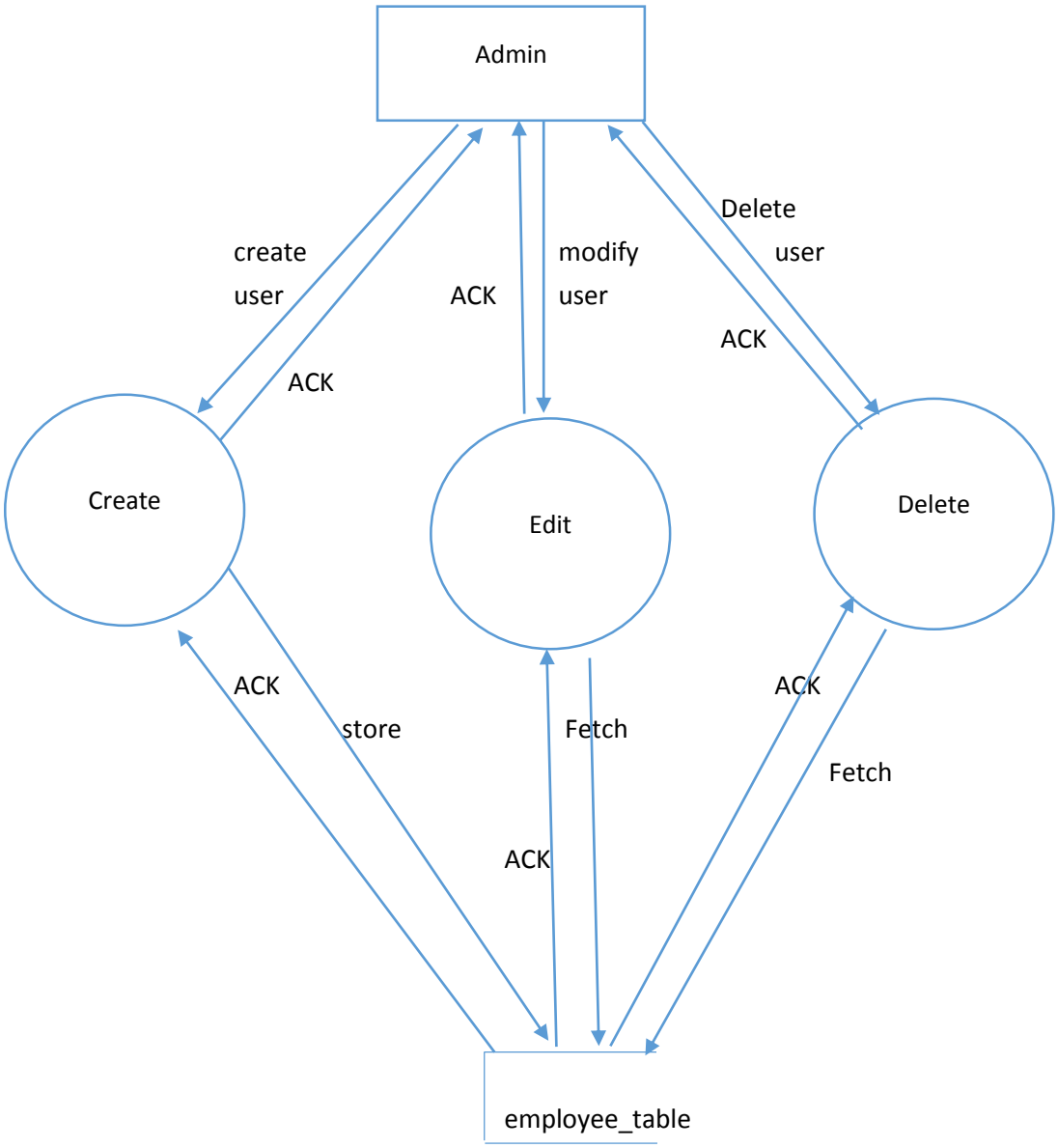
Zero Level DFD



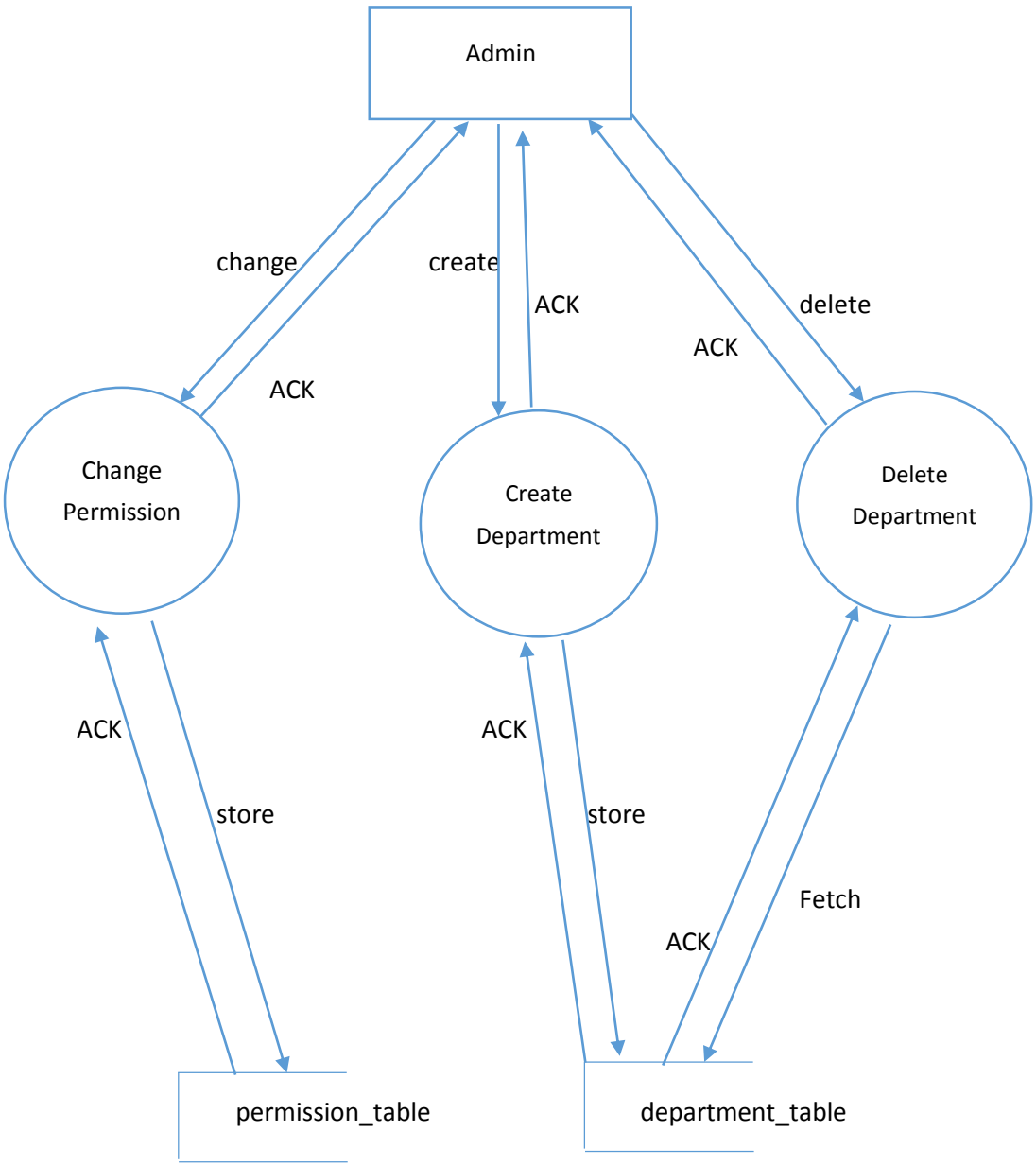
First Level DFD Admin



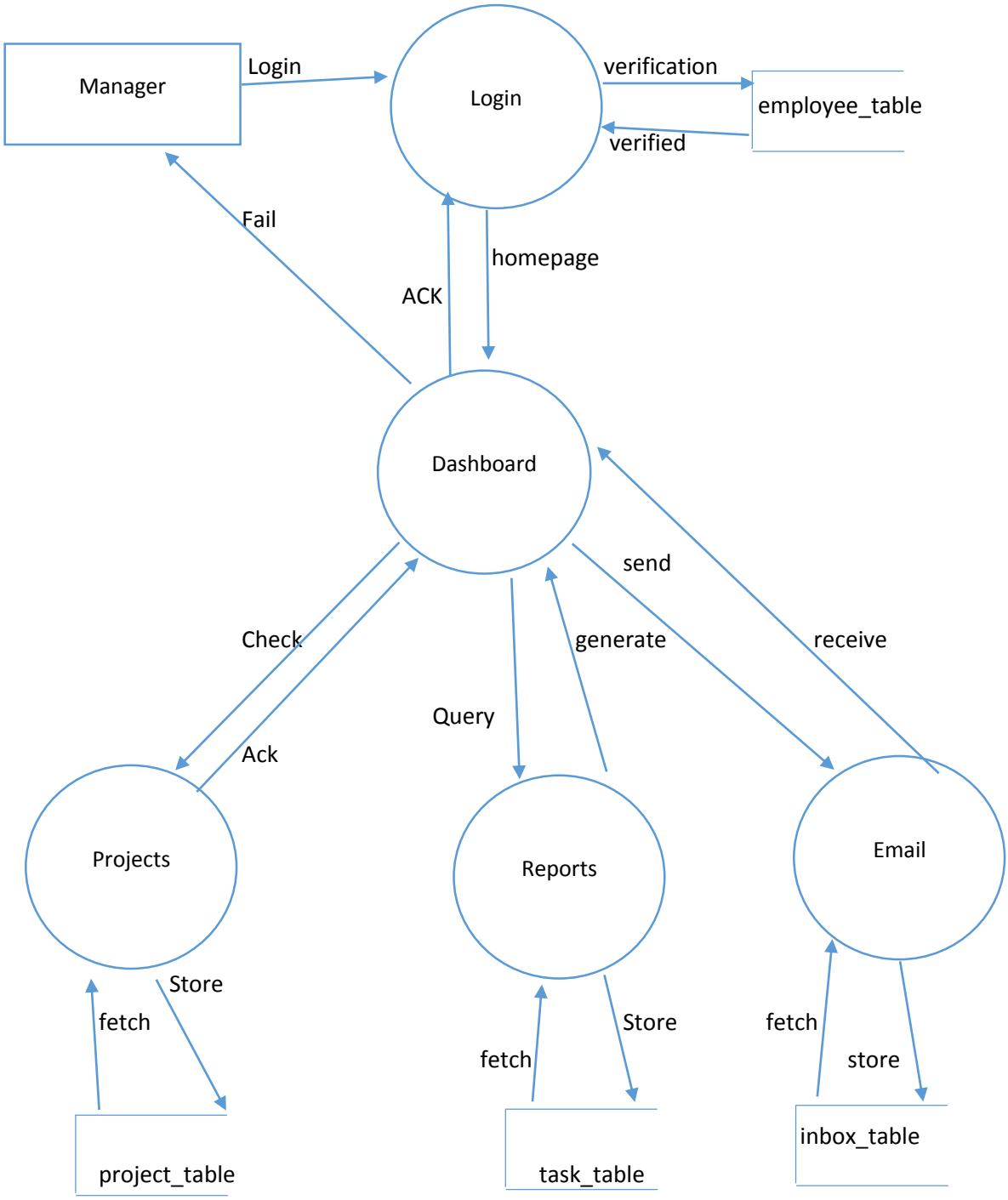
2nd level Employee Detail: ADMIN



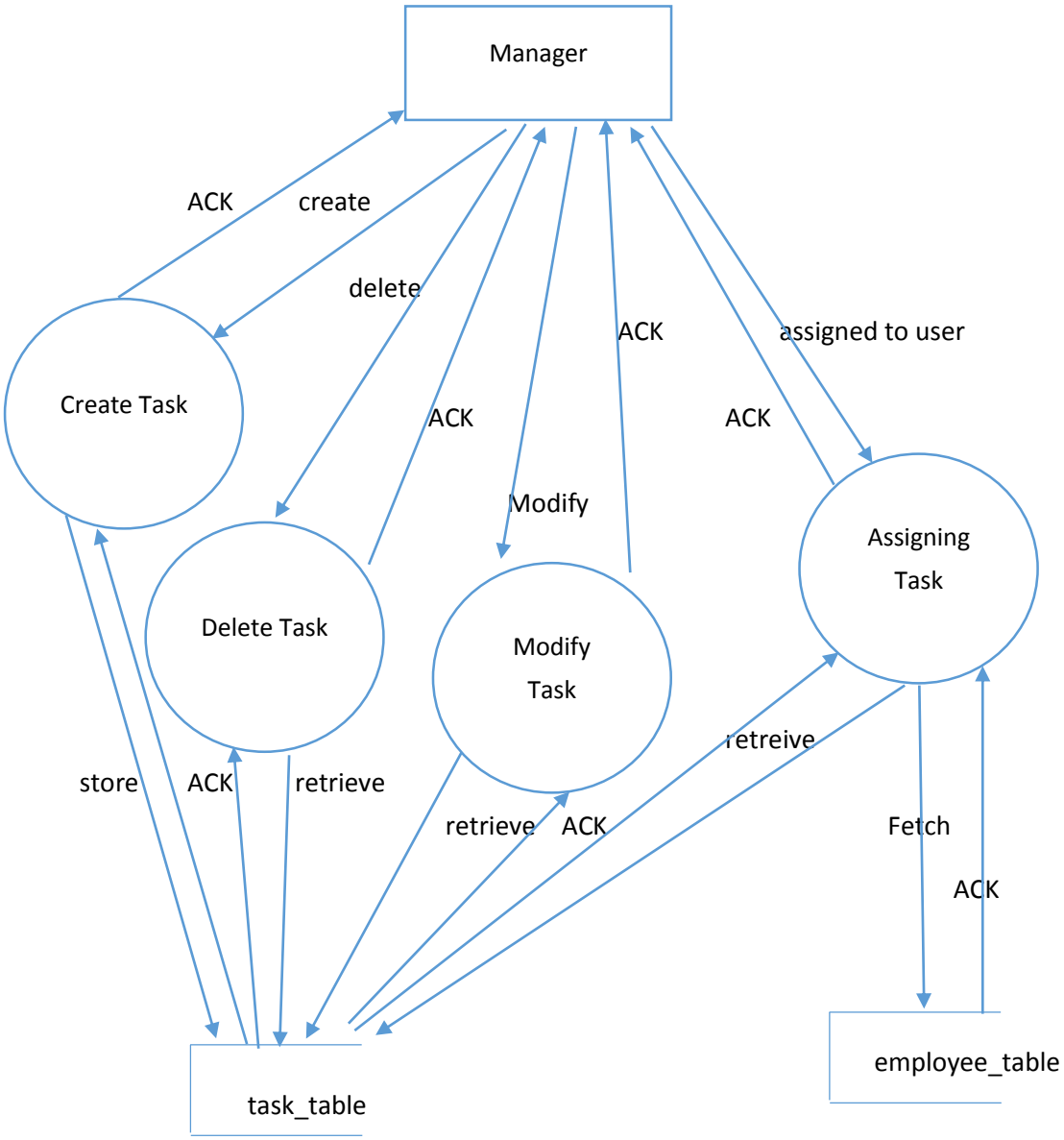
2nd level Permission : ADMIN



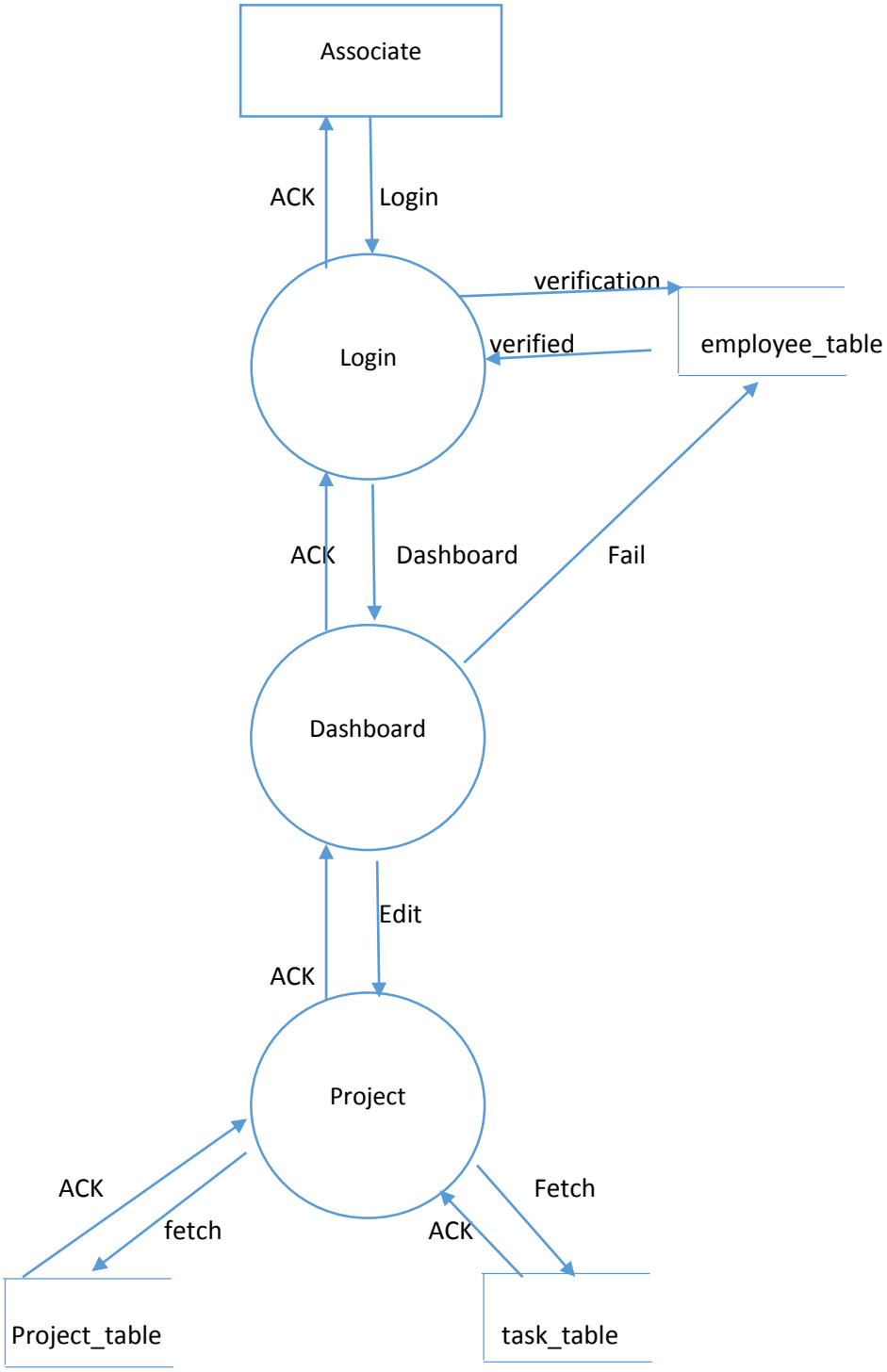
1st Level DFD : Manager



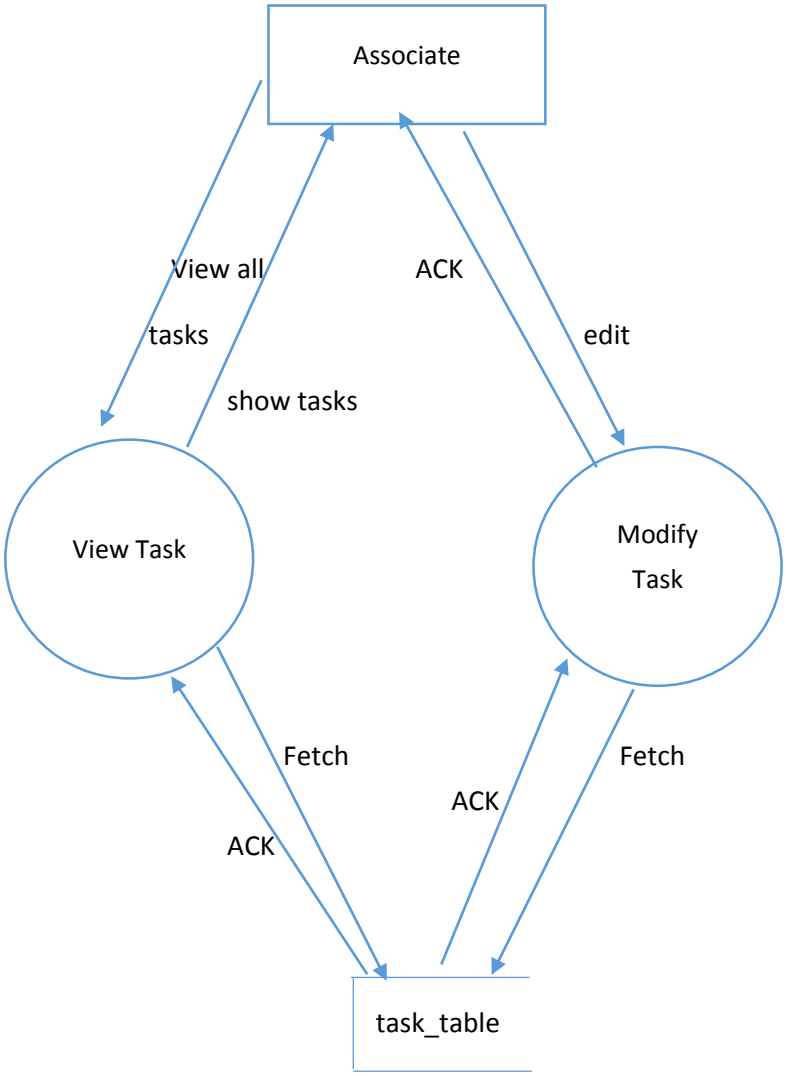
2nd Level Projects: Manager



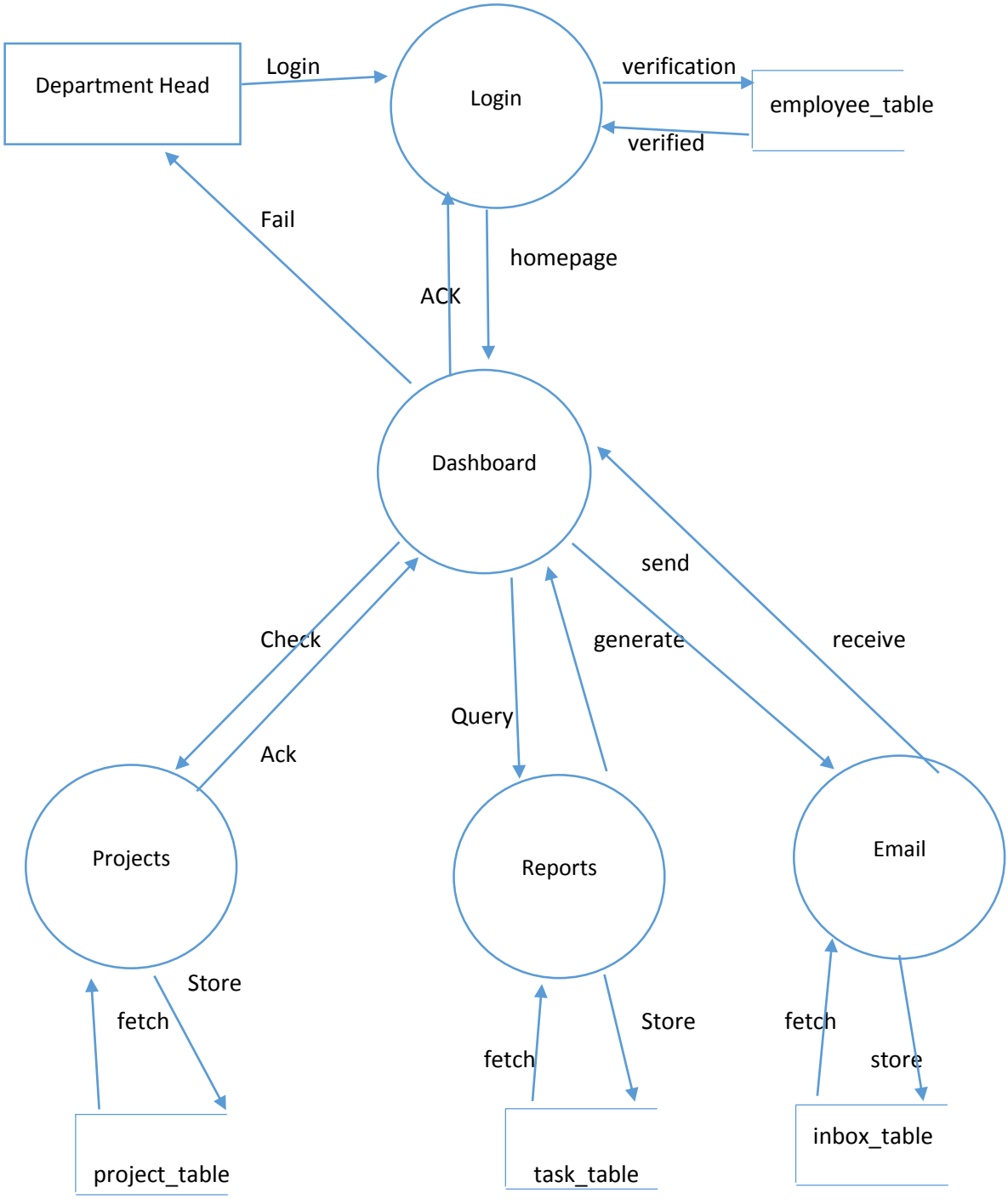
1st Level DFD : Associate



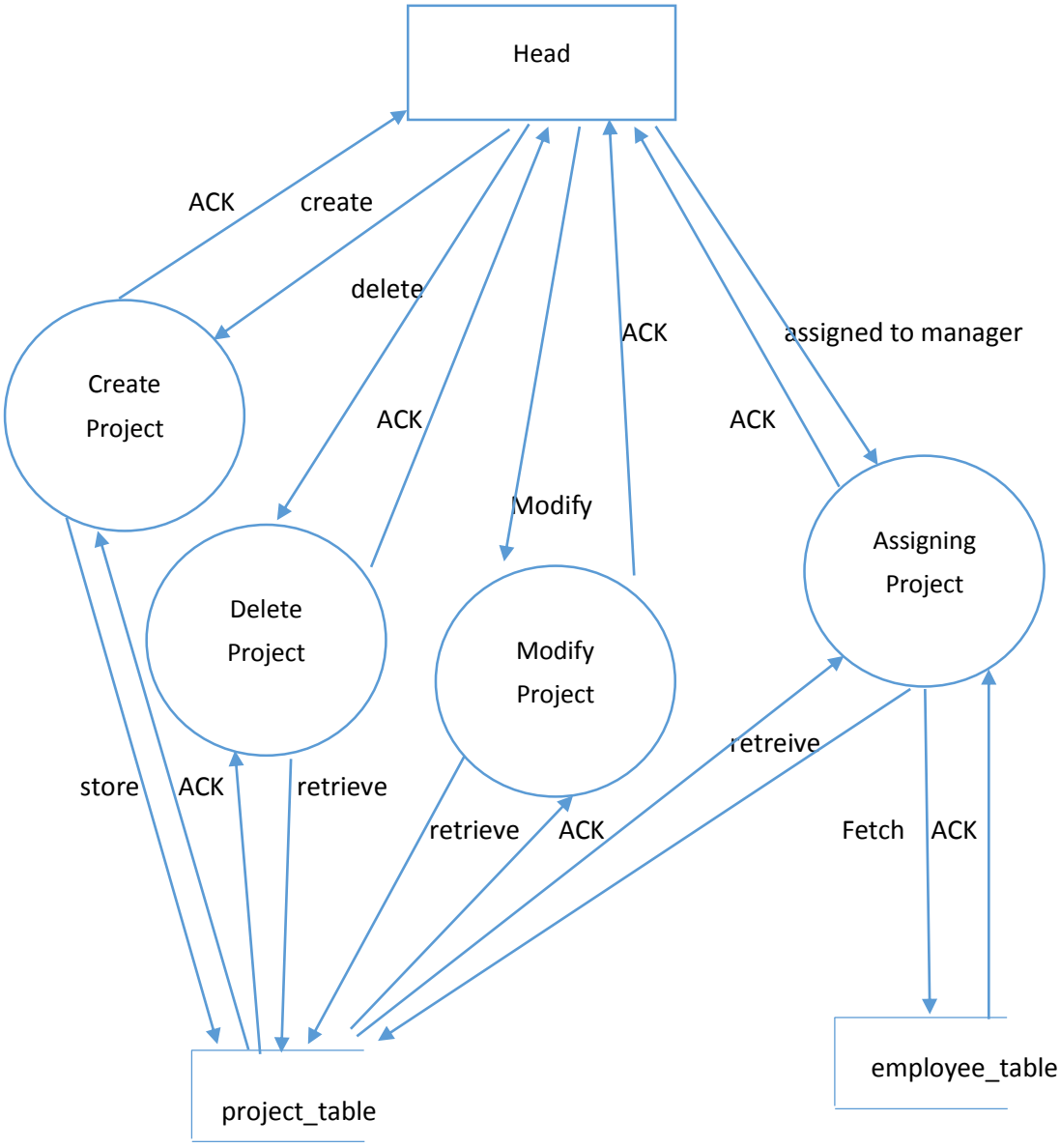
2nd level project : Associate



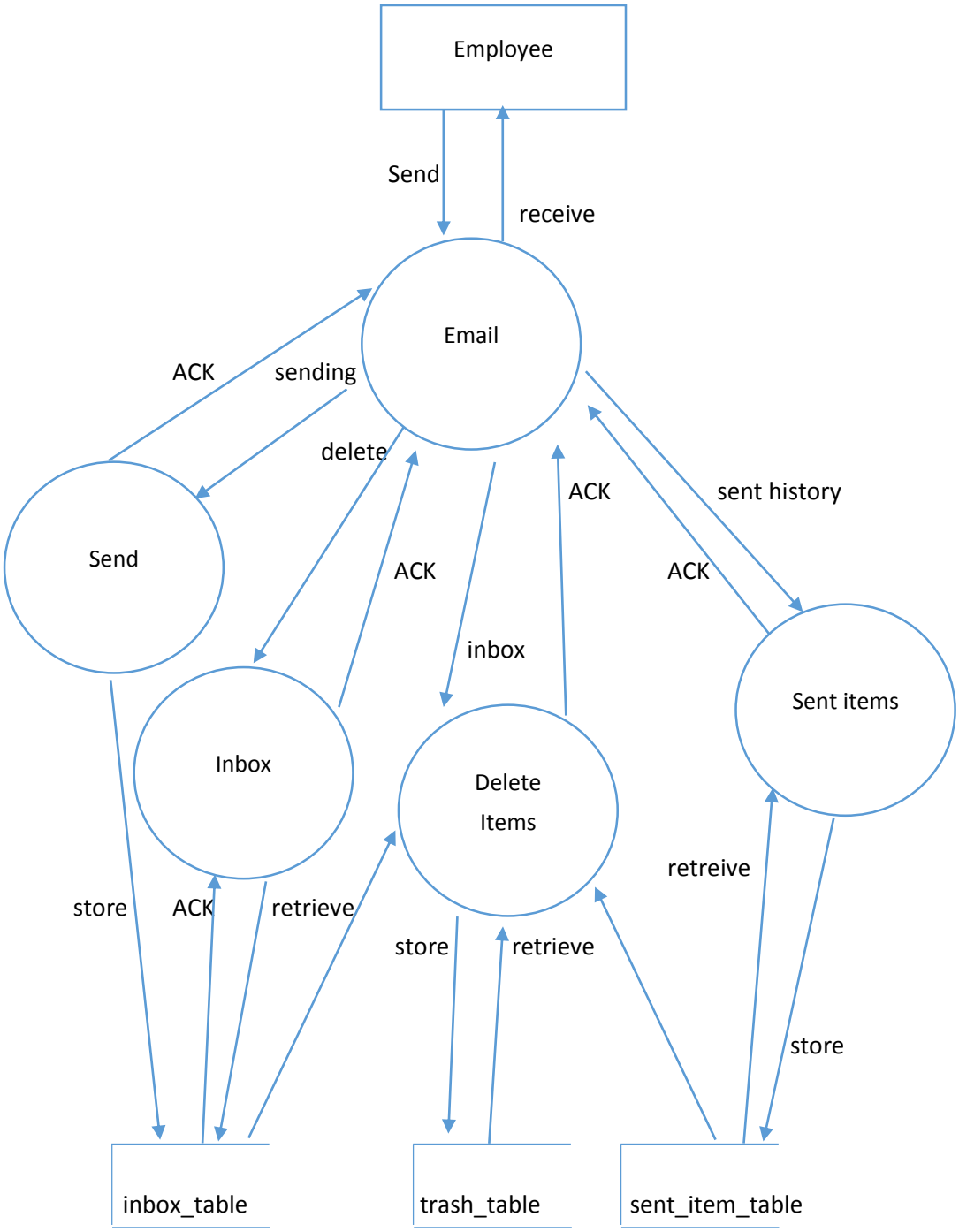
1st Level DFD : Department Head



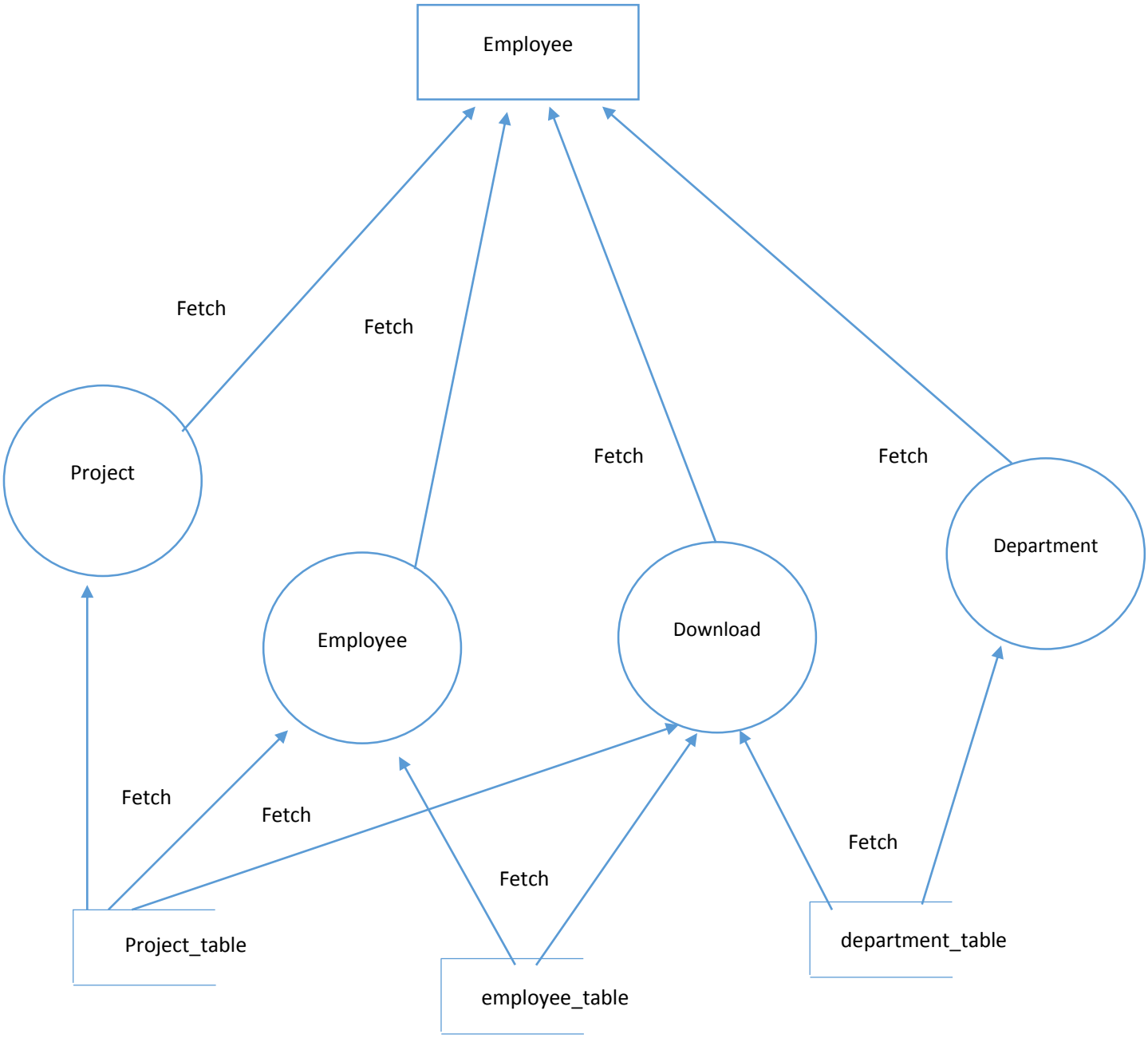
2nd Level Projects: Department Head



2nd Level DFD : Email

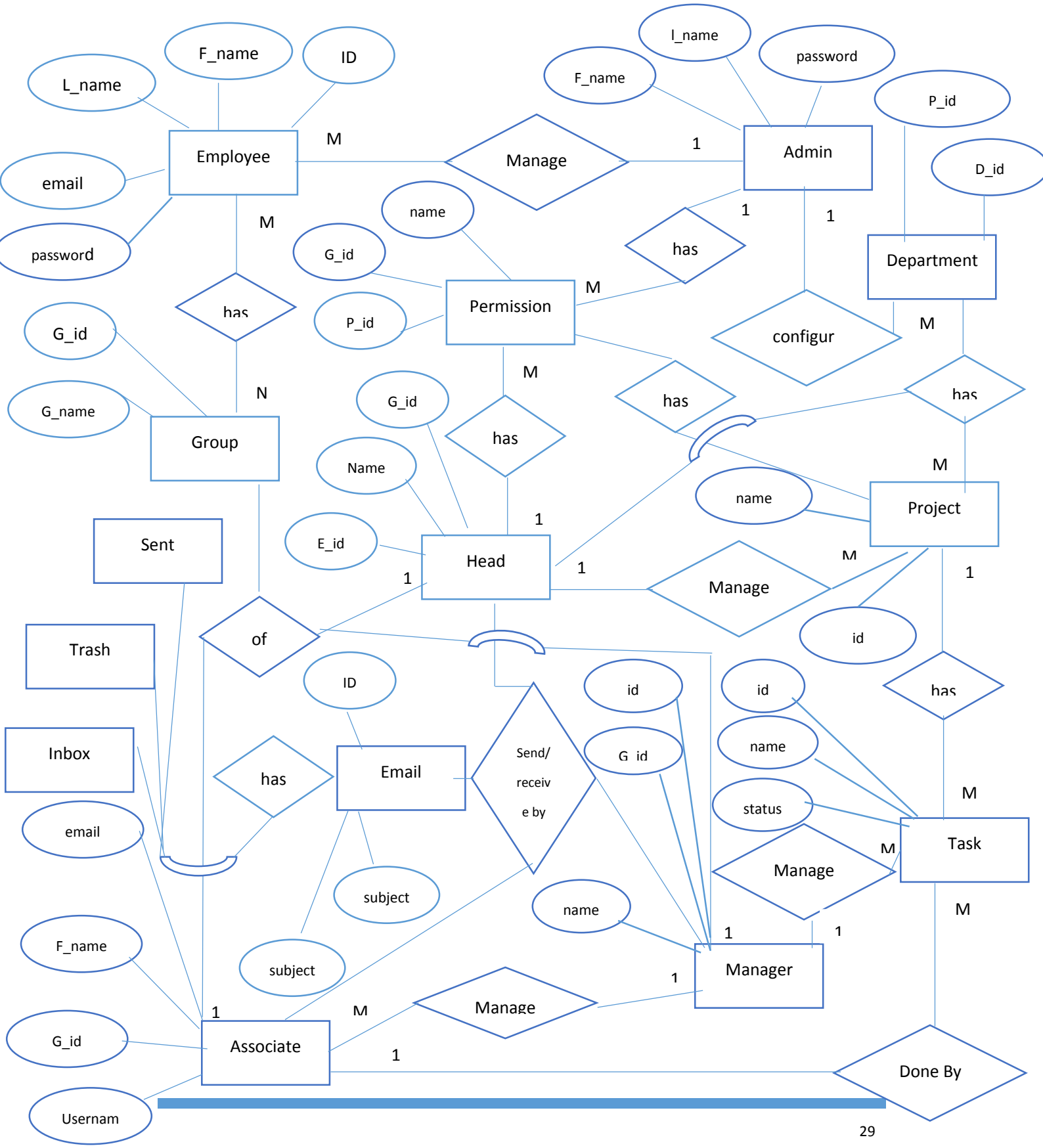


2nd Level DFD : Reports



Restful Api DFD

ER Diagram


F_name
ID
L_name
Employee
M
Manage
1
Admin
l_name
password
F_name
P_id
D_id
email
M
name
G_id
password
has
1
1
Department
has
Permission
M
G_id
P_id
configur
M
G_id
M
G_name
N
Group
G_id
has
has
has
Name
M
name
Project
Sent
E_id
Head
1
1
1
Manage
M
1
Trash
of
id
ID
id
id
has
Inbox
has
Email
Send/
receiv
e by
G_id
name
email
status
M
Task
Manage
M
subject
name
M
F_name
subject
1
1
Manager
1
G_id
1
M
Manage
Associate
1
Done By
Usernam
29

Database Design

Table Name: employee_table

Attributes	Field Type	Field Size	Constraints	Description
E_id	Numeric	10	Primary Key	Employee ID
Password	Password	10	Not Null	Password
F_name	Char	20	Not Null	Name
L_name	Char	20	Not Null	Email ID
Email	Varchar	10	Not Null	Contact Number
date_joined	DateTime	10	Not Null	Joining Date
Is_staff	Boolean	1	Not Null	Check staff emp or not

Table Name: Group_table

Attributes	Field Type	Field Size	Constraints	Description
G_id	Numeric	10	Primary key	Group ID
G_name	Varchar	20	Not Null	Group Name
P_id	Numeric	10	Foreign Key(permission)	Permission ID
E_id	Numeric	10	Foreign Key(Employee)	Employee id

Table Name: permission_table

Attributes	Field Type	Field Size	Constraints	Description
P_id	Numeric	5	Primary key	Permission ID
p_name	char	30	Not Null	Permission Name
Codename	Varchar	20	Not Null	Permission code

Table Name: task_table

Attributes	Field Type	Field Size	Constraints	Description
task_id	Numeric	10	Primary key	Task ID
Task_name	Char	10	Foreign key(product)	Task name
Created_by	Char	20	Foreign Key(Employee)	Created by employee
Modify_by	Char	20	Foreign Key(Employee)	Modify by employee
comments	Varchar	100	Not Null	Comments
Status	Char	20	Not Null	Current status of task
Assigned To	Char	20	Foreign Key(Employee)	Assigning to employee
TAT	Numeric	20	Not Null	Time of project
Priority	Numeric	10		Priority of task
Created_Date	Date		Not Null	Date of Creation

Table Name: Project_table

Attributes	Field Type	Field Size	Constraints	Description
Prj_id	Numeric	10	Primary key	Project ID
Name	Char	10	Not Null	Name
Description	Varchar	100	Not Null	Description
Task_id	Numeric	10	Foreign Key(task)	Task id

Table Name: department_table

Attributes	Field Type	Field Size	Constraints	Description
D_id	Numeric	10	Primary key	Department ID
Name	Char	20	Not Null	Name
Head	Varchar	10	Foreign Key(Group)	Department Head
Project	Varchar	10	Foreign Key(project)	project id

Table Name: inbox_table

Attribute	Field Type	Field Size	Constraints	Description
msg_id	Numeric	10	Primary Key	Message ID
Subject	Varchar	50	Foreign Key(msg_table)	
From	Varchar	20	Foreign Key(Employee)	Sender Name
Is_read	Boolean	1	Not Null	0 : Not Read 1 : Read

Table Name: sent_item_table

Attributes	Field Type	Field Size	Constraints	Description
msg_id	Numeric	10	Primary key	Message ID
sender_id	Numeric	10	Foreign Key(Employee)	Sender Emp ID
receiver_id	Numeric	10	Foreign Key(Employee)	Receiver Emp ID
Subject	Varchar	50	Not Null	Subject
content	Varchar	100	Not Null	Description
Date	DateTime	10	Not Null	Date

Table Name : Trash_item_table

Attributes	Field Type	Field Size	Constraints	Description
msg_id	Numeric	10	Primary key	Message ID
sender_id	Numeric	10	Foreign Key(Employee)	Sender Emp ID
receiver_id	Numeric	10	Foreign Key(Employee)	Receiver Emp ID
Subject	Varchar	50	Not Null	Subject
Content	Varchar	100	Not Null	Description
Date	DateTime	10	Not Null	Date

Table Name : token_authentication

Attributes	Field Type	Field Size	Constraints	Description
Key	Varchar	50	Auto	Token for user
Employee	Varchar	20	Foreign Key(Employee)	Employee name
Created	DateTime	20	Not Null	Created date

Number of Module and their structure

1. Employee

Description: - This module keeps the records of its all Employee in organization i.e their names, their identities, address of residence.

2. Searching

Description:- This module is related to search any task or emails on TeamWorks.

3. Task

Description:-This module is meant for maintaining all task and their creation deletion, modification and assigned to another employee.

4. Project

Description:-This module is meant for maintaining all project of organization only department head/admin can be create or delete projects.

5. Department

Description:-This module is meant for maintaining all Departments such as Information technolgy, finance etc. Project are parts of department module.

6. Group

Description:-This module of the project contain all the department and group of organization such as manager, associate, HR etc.

7. Permission

Description: - This module is contains portal permission of each user and groups. For example to delete and add task or project it must be permission of it.

8. Email

Description: - This module is responsible for mailing system within the organization, There are main three function of email

I. **Inbox :** This module contain inbox mail

II. **Sent Items :** This module contains sent mail of user

III. **Trash:** This module just like recycle bin in our computer mean when we delete mails it automatically sent to trash which can be restore, and delete permanently.

9. Report

Description: - This module is responsible to generate report behalf of task and projects where we keep track each project and every employee work status.

10. API Point

Description: - This module is responsible for whole site api, api is very useful in single page application and webservices, this module response all data of site in JSON form.

11. Token Authentication

Description: - This module is responsible for authentication in api, token are generate automatically when new user is create.

Project Directory Structure

- **Root**
  - **Template**
    - **Registration**
      - Login.html
      - Logout.html
    - **Rest_framework**
      - Base.html
    - Base.html
    - Ck.html
    - Css.html
    - Email.html
    - Home.html
    - Js.html
    - Nav.html
    - Reports.html
    - Search_result.html
  - **Application**
    - **Static**
      - Jquery.min.js
      - Style.css
      - **Backbone**
        - Backbone.js
        - Underscore.js
      - **Bootstrap**
        - Bootstrap.min.css
        - Bootstrap.min.js
    - __init__.py
    - Admin.py
    - Form.py
    - Models.py
    - Serializer.py
    - Views.py
  - **TeamWorks**
    - __init__.py
    - Settings.py
    - Urls.py
    - Wsgi.py
  - Manage.py

Code

Models.py

```
from django.db import models
from datetime import datetime
from django.contrib.auth.models import User
from ckeditor.fields import RichTextField
#from django.contrib.auth.models import Use
# Create your models here.

from django.conf import settings
from django.db.models.signals import post_save
from django.dispatch import receiver
from rest_framework.authtoken.models import Token

@receiver(post_save, sender=settings.AUTH_USER_MODEL)
def create_auth_token(sender, instance=None, created=False, **kwargs):
    if created:
        Token.objects.create(user=instance)



class Department(models.Model):
    id = models.AutoField(primary_key=True)
    name = models.CharField(max_length=30)
    head = models.ForeignKey(User,on_delete=models.CASCADE,null=True)
    #project =
models.ForeignKey(Project,on_delete=models.CASCADE,null=True)
    def __str__(self):
        return self.name


class Project(models.Model):
    id = models.AutoField(primary_key=True)
    name=models.CharField(max_length=20)
    description=models.TextField()
    head =
```

```
models.ForeignKey(Department,on_delete=models.CASCADE,null=True)
    def __str__(self):
        return self.name


class Task(models.Model):
    nothing = '----'
    new ='New'
    inprocess='In Process'
    complete='Complete'
    choices=(
        (new , 'New'),
        (inprocess,'In Process'),
        (complete,'Complete'),
    )

    high='High'
    low='Low'
    normal='Normal'

    p_choice=(
        (high,'High'),
        (low,'Low'),
        (normal,'Normal')
    )


    id=models.AutoField(primary_key=True)
    name=models.CharField(max_length=50)
    created_date=models.DateTimeField(default=datetime.now,null=True)
    created_by=models.CharField(max_length=20,blank=True,null=True)

status=models.CharField(max_length=10,choices=choices,default=nothing)
    task_description=models.TextField()
    comments=models.TextField(blank=True)
    modify_by=models.CharField(max_length=20,blank=True)

priority=models.CharField(max_length=10,choices=p_choice,default=nothin
g)
```

```
    tat=models.IntegerField()

#file=models.ImageField(upload_to='photos/%d/',blank=True,null=True)
    #file=models.ImageField(upload_to=upload_path_handler,
blank=True,null=True)
    assign=models.ForeignKey(User,on_delete=models.CASCADE,null=True)

project_id=models.ForeignKey(Project,on_delete=models.CASCADE,null=True
)
    def __str__(self):
        return str(self.id)+' '+self.name



class Email_msg(models.Model):
    id = models.AutoField(primary_key=True)
    sender_id = models.CharField(max_length=20)
    receiver_id = models.ForeignKey(User, on_delete=models.CASCADE,
null=True)
    subject = models.TextField(blank=True)
    content = models.TextField(blank=True)
    date = models.DateTimeField(default=datetime.now,blank=True)
    def __str__(self):
        return self.subject

class Inbox(models.Model):
    id = models.AutoField(primary_key=True)
    sender_id = models.CharField(max_length=20,null=True)
    subject = models.CharField(max_length=150,null=True)

    #content = models.TextField()
    content = RichTextField()
    receiver_id =
models.ForeignKey(User,on_delete=models.CASCADE,null=True)
    date = models.DateTimeField(default=datetime.now,blank=True)
    is_read = models.BooleanField(default=False)
    def __str__(self):
        return self.subject
```

```
class Sent_item(models.Model):
    id = models.AutoField(primary_key=True)
    sender_id = models.CharField(max_length=20)
    receiver_id = models.ForeignKey(User,
on_delete=models.CASCADE,null=True)
    subject = models.TextField(blank=True)
    content = RichTextField()
    is_read = models.BooleanField(default=True)
    date = models.DateTimeField(default=datetime.now,blank=True)
    def __str__(self):
        return self.subject
```

Serializer.py

```
from rest_framework import serializers
from Workflow.models import *
from django.contrib.auth.models import User, Group,
Permission,ContentType
from rest_framework.reverse import  reverse
from django.contrib.auth.models import User
from .form import UserName,UserFullName,Alluser
from django import forms
```

```
class UserSerializer(serializers.HyperlinkedModelSerializer):
    #user_permissions = PermissionSerializer()
    class Meta:
        model = User
        fields =
('id','username','first_name','last_name','email','inbox_set','user_per
missions')


import json
class Taskserializer(serializers.HyperlinkedModelSerializer):
    assign_name =
serializers.CharField(source='assign.get_full_name',required=False)
    #assign_id =
serializers.PrimaryKeyRelatedField(queryset=Task.objects.all(),
write_only=True)
    class Meta:
        model = Task
        fields =
('id','name','created_date','created_by','status','task_description','c
omments','modify_by','priority','tat','assign_name','assign')



class ProjectSeralizer(serializers.HyperlinkedModelSerializer):
#    task_set = Taskserializer(many=True)
    task_set = Taskserializer(required=False,many=True)
    class Meta:
        model = Project
        fields=('id','name','description','head','task_set')

class SentItemSerializer(serializers.HyperlinkedModelSerializer):
    class Meta:
        model = Sent_item
        fields =
('id','subject','content','sender_id','receiver_id','date','is_read')
```

```
class DepartmentSerializer(serializers.HyperlinkedModelSerializer):
    project_set = ProjectSeralizer(required=False,many=True)
    class Meta:
        model = Department
        fields = ('id','name','head','project_set')


class InboxSerializer(serializers.HyperlinkedModelSerializer):
    class Meta:
        model = Inbox
        fields =
('id','subject','content','sender_id','receiver_id','is_read','date')

class ContentTypeSerialzier(serializers.HyperlinkedModelSerializer):
    class Meta:
        model = ContentType
        fields = ('id','app_label','model')

class PermissionSerializer(serializers.HyperlinkedModelSerializer):
    #content_type = ContentTypeSerialzier(many=True,)
    class Meta:
        model = Permission
        fields = ('id','content_type','codename','name')

class GroupSerializer(serializers.HyperlinkedModelSerializer):
    class Meta:
        model = Group
        fields = ('id','name','user_set')
```

admin.py

```
from django.contrib import admin
from .models import *
# Register your models here.
admin.site.register(Project)
admin.site.register(Task)
admin.site.register(Department)
admin.site.register(Email_msg)
admin.site.register(Inbox)
admin.site.register(Sent_item)
admin.site.register(Alert)
```

forms.py

```
from Workflow.models import *
from django.contrib.auth.forms import
AuthenticationForm,UserCreationForm
from django.contrib.auth import authenticate
from django.contrib.auth.models import User
from django import forms
from django.contrib.auth.models import User
from ckeditor.widgets import CKEditorWidget
from django.forms import Textarea,CharField,TextInput,ChoiceField


class LoginForm(AuthenticationForm):
    username = forms.CharField(label="Username", max_length=30,
widget=forms.TextInput(attrs={'class': 'form-control', 'name':
'username'}))
    password = forms.CharField(label="Password", max_length=30,
widget=forms.TextInput(attrs={'class': 'form-control', 'name':
'password','type':'password'}))

    def clean(self,*args,**kwargs):
        username =self.cleaned_data.get('username')
```

```
        password = self.cleaned_data.get('password')
        if username and password:
            user = authenticate(username=username,password=password)
            if not user:
                raise forms.ValidationError("this user doest not
exist")
        return super(LoginForm,self).clean(*args,**kwargs)


class UserMail(User):
    class Meta:
        proxy = True
    def __str__(self):
        return self.first_name+'<'+self.email+'>'


class UserFullName(User):
    class Meta:
        proxy = True

    def __str__(self):
        return self.get_full_name()


class UserName(User):
    class Meta:
        proxy = True
    def __str__(self):
        return self.first_name+' '+self.last_name

import json
def Alluser(User):
    dic={}
    user=User.objects.all()
    for i in user:
        dic.update({i.get_full_name():str(i.id)})
        #dict.update({'name':i.get_full_name})
    return json.loads(json.dumps(dic))
```

```
class ProjectForm(forms.ModelForm):
    class Meta:
        model = Project
        head = forms.ModelChoiceField(queryset=UserName.objects.all())
        fields = ('name','description','head')
        widgets = {
            'name':TextInput(attrs={'class':'form-ctr'}),
            'description':Textarea(attrs={'class':'form-control'}),
            'head':forms.Select(attrs={'class':'form-ctr'})
        }


class TaskEditForm(forms.ModelForm):

assign=forms.ModelChoiceField(UserName.objects.all(),widget=forms.Selec
t(attrs={'class': 'form-control'}))
    class Meta:
        model = Task
        fields =
['name','status','created_date','created_by','task_description','commen
ts','priority','tat','assign']
        widgets ={
              control','disabled':'true'}),
            'created_by' : TextInput(attrs={'class':'form-control'}),
            'created_date' : TextInput(attrs={'class':'form-control'}),

            'name' : TextInput(attrs={'class':'form-control'}),
            'task_description':Textarea(attrs={'class':'form-
control','style':'height: 3cm !important;overflow: hidden;max-
width:100%;min-width:100%'}),#,style="height: 30px !important;overflow:
hidden;resize: none;"
            'comments':Textarea(attrs={'class':'form-
control','style':'height: 3cm !important;overflow: hidden;max-
width:100%;min-width:100%'}),
            'priority':forms.Select(attrs={'class':'form-control'}),
            'tat':forms.NumberInput(attrs={'class':'form-control'}),
            'status':forms.Select(attrs={'class':'form-control'}),
            'assign':forms.Select(attrs={'class':'form-control'}),
```

```
            #'assign':CharField(attrs={'class':'form-control'}),
            #,widget=forms.Select(attrs={'class':'hidden'})
        }

class InboxForm(forms.ModelForm):
    content = Inbox
    content = forms.CharField(widget=CKEditorWidget())
    receiver_id=forms.ModelChoiceField(queryset=UserMail.objects.all())
    #subject = CharField(widget={'class':'subject'})
    class Meta:
        model= Inbox
        fields=['subject','content','receiver_id']
        widgets={
            'content' : Textarea(attrs={'class':'texarea'}),
            'subject' : TextInput(attrs={'class':'subject'})
        }

class SentForm(forms.ModelForm):
    #content = Sent_item
    receiver_id=forms.ModelChoiceField(queryset=UserMail.objects.all())
    class Meta:
        model = Sent_item
        fields = ['subject','content','receiver_id']
```

Views.py

```
from django.shortcuts import
render,render_to_response,get_object_or_404
from .models import *
from .form import *
from django.views.generic import TemplateView
from rest_framework import viewsets
from django.core.context_processors import csrf
from django.http.response import HttpResponseRedirect,JsonResponse
from django.contrib.auth.decorators import login_required
from django.contrib import auth
from django.contrib.auth.models import
Group,User,Permission,ContentType
from django.contrib.auth.views import logout
from django.http import HttpResponse
import json
from rest_framework.views import APIView
from rest_framework.response import Response
from rest_framework import status
from .serializer import *
from django.contrib.auth.decorators import login_required
import re
from django.core import serializers

# Create your views here.

class TaskSet(viewsets.ModelViewSet):
 #   queryset = Task.objects.all()
    queryset = Task.objects.all()
    serializer_class = Taskserializer



class ProjectSet(viewsets.ModelViewSet):
    queryset = Project.objects.all()
    serializer_class = ProjectSeralizer
    def post(self, request,format=None):
        serializer = Taskserializer(data=request.data)
        if serializer.is_valid():
            serializer.save()
```

```
            return Response(serializer.data,
status=status.HTTP_201_CREATED)
        return Response(serializer.errors,
status=status.HTTP_400_BAD_REQUEST)

class ContentTypeSet(viewsets.ModelViewSet):
    queryset = ContentType.objects.all()
    serializer_class = ContentTypeSerialzier


class UserSet(viewsets.ModelViewSet):
    queryset = User.objects.all()
    serializer_class = UserSerializer

    def get(self,request,pk):
        queryset = User.objects.all()
        serializer = UserSerializer(queryset, many=True)
        return Response(serializer.data)



class DepartmentSet(viewsets.ModelViewSet):
    queryset = Department.objects.all()
    serializer_class = DepartmentSerializer


class InboxSet(viewsets.ModelViewSet):
    queryset = Inbox.objects.all()
    serializer_class = InboxSerializer


class GroupSet(viewsets.ModelViewSet):
    queryset = Group.objects.all()
    serializer_class = GroupSerializer


class PermissionSet(viewsets.ModelViewSet):
    queryset = Permission.objects.all()
    serializer_class = PermissionSerializer

class SentItemset(viewsets.ModelViewSet):
    queryset = Sent_item.objects.all()
    serializer_class = SentItemSerializer
```

```
@login_required(login_url='/accounts/login')
def Logout(request):
    logout(request)
    return render_to_response('registration/logout.html')
#    request.user.logout()


def prj_by_dpt(request,name):
    #option = [i.name for i in l]
    d=Department.objects.get(name=name)
    project_name=[i.name  for i in d.project_set.all()]
    dataset={}
    for i in project_name:
        #data=serializers.serialize('json',d.project_set.all())

data=serializers.serialize('json',Project.objects.get(name=i).task_set.
all())
        dataset.update({i:json.loads(data)})
    js={
        'option':project_name,
        'label':dataset
    }
    return JsonResponse(js)
    #return render_to_response('home.html')


def Alert(request):
    return render_to_response('alert.html')


def task_by_emp(request,name):
    option = [i[0] for i in Task.objects.all()[0].choices]
    dataset={}
    for i in option:

l=Task.objects.filter(assign_id=User.objects.get(username=name.lower())
.id,status=i)
        dataset.update({i:json.loads(serializers.serialize('json',l))})
    js={
        'option':option,
        'label':dataset
    }
    return JsonResponse(js)
```

```
@login_required(login_url='/accounts/login')
def Single(request,project):
    f=Task.objects.filter()
    option = [i[0] for i in Task.objects.all()[0].choices]
    dataset={}
    for i in option:

l=Task.objects.filter(project_id=Project.objects.get(name=project).id,s
tatus=i)
        dataset.update({i:json.loads(serializers.serialize('json',l))})
    js={
        'option':option,
        'label':dataset
    }

    return JsonResponse(js)

@login_required(login_url='/accounts/login')
def profile(request):
    email=request.user.email
    nn=request.user.date_joined
    argus={'user':request.user,
           'email':email,
           'join':nn,
           }
    return render_to_response('registration/profile.html',argus)


@login_required(login_url='/accounts/login')
def Home_rest(request):
    p=Project.objects.all()
    #t=Task.objects.all()
    user = request.user
    t=Task.objects.filter(assign_id=user.id)
    #t=Task.objects.all()
    pform=ProjectForm()
    form = TaskEditForm()
    return
render(request,'rest_home.html',{'pform':pform,'taskform':form,'project
':p,'task':t,'pid':'home','user':user})
```

```
def Home(request):
    p=Project.objects.all()
    #t=Task.objects.all()
    user = request.user
    t=Task.objects.filter(assign_id=user.id)
    #t=Task.objects.all()
    pform=ProjectForm()
    form = TaskEditForm()
    return
render(request,'rest_home.html',{'pform':pform,'taskform':form,'project
':p,'task':t,'pid':'home','user':user})


@login_required(login_url='/accounts/login')
def TaskDetail(request,pk):
    p = Project.objects.all()
    #p = Project.objects.get(id=pid)
    t=Task.objects.get(id=pk)
    user = request.user
    return
render_to_response('home.html',{'task':t,'project':p,'work':'edit','use
r':user})


@login_required(login_url='/accounts/login')
def prj_data(request,what):
    model = globals()[what]
    pp=model._meta.related_objects[0].name
    d=model.objects.all()
    p=serializers.serialize('json',d)
    c=[]
    sets=(str(pp)+'_set')

    if(what=='Department'):
        for i in d:
            l=i.serializable_value(sets).values()
            c.append(l.count())


        js={
            'label':json.loads(p),
```

```
            'count':c
        }
    else:
        s_m=globals()[pp.capitalize()]
        option=[i[0] for i in s_m.objects.all()[0].choices]
        status=[]
        for i in d:
            l=i.serializable_value(sets).values()
            c.append(l.count())
            #[ i['status'] for i in range(l)]
            temp=[]
            for j in option:
                cn=[k['status'] for k in l]
                #cn.count(j)
                temp.append(cn.count(j))
            status.append(temp)

            js={
            'label':json.loads(p),
            'count':c,
            'option':option,
            'status':status
            }
    return JsonResponse(js)

@login_required(login_url='/accounts/login')
def emp_data(request):
    user=User.objects.all()
    related=User._meta.get_all_related_objects()[1:]
    c=[]
    p=serializers.serialize('json',user[1:])
    for i in user[1:]:
        c.append(i.serializable_value('task_set').values().count())

    js={
        'label':json.loads(p),
        'count':c
    }
    return JsonResponse(js)

    if what == 'priority':
        l_count=Task.objects.filter(priority='Low').count()
```

```
        h_count=Task.objects.filter(priority='High').count()
        n_count=Task.objects.filter(priority='Normal').count()

        label=['Hight','Low','Normal']
        count=[h_count,l_count,n_count]
        data={
            'label':label,
            'count':count
        }
        return JsonResponse(data)

    elif what == 'department':
        t=Department.objects.all()
        count=[i.project_set.all().count() for i in t]
        label=[i.name for i in t]
        data={
        'label' : label,
        'count' : count
        }
        return JsonResponse(data)



    elif what == 'status':
        c_count=Task.objects.filter(status='Complete').count()
        i_count=Task.objects.filter(status='In Process').count()
        n_count=Task.objects.filter(status='New').count()
        label=['Complete','In Process','New']
        count=[c_count,i_count,n_count]
        data={
            'label':label,
            'count':count
        }
        return JsonResponse(data)

    #p=Project.objects.filter()
@login_required(login_url='/accounts/login')
def project_data(request):
    p=Project.objects.all()
    project=[i.name for i in p]
    print(project)
```

```
count=[p[0].task_set.count(),p[1].task_set.count(),p[2].task_set.count(
)]
    data={
        'project':project,
        'count':count
    }
    return JsonResponse(data)

@login_required(login_url='/accounts/login')
def Reports(request):
    return render_to_response('reports.html',{'user':request.user})



def login(request):
    if request.user.is_authenticated():
        return HttpResponseRedirect('/')
    else:
        form=LoginForm()
        c={'form':form}
        c.update(csrf(request))
        return render(request,'registration/login.html',c)
@login_required(login_url='/accounts/login')
def logout(request):
    auth.logout(request)
    return render_to_response('registration/logged_out.html')

def auth_view(request):
    if request.method=='POST':
        response_data={}
        username = request.POST.get('username', '')
        password = request.POST.get('password', '')
        user = auth.authenticate(username=username, password=password)
        u=User.objects.filter(username=username)
        if not u:
            return HttpResponse('username')
        if user is not None:
            auth.login(request, user)
            return HttpResponseRedirect('/')
        elif request.user.is_authenticated():
            return HttpResponseRedirect('/')
```

```
        else:
            return HttpResponse('not')
def Isread(request,pk):
    e = Inbox.objects.get(id=pk)
    e.is_read = True
    e.save()
    return HttpResponse('OK')

@login_required(login_url='/accounts/login')
def ProjectTask(request,pk):

    p = Project.objects.all()
    pi = Project.objects.get(id=pk)
    t=Task.objects.filter(assign_id=request.user.id,project_id=pk)
    #t=Task.objects.filter(project_id=pk)
    return
render_to_response('home.html',{'project':p,'task':t,'pid':pi,'user':re
quest.user})


@login_required(login_url='/accounts/login')
def Email_rest(request):
    e = Inbox.objects.filter(receiver_id_id=request.user.id)
    cform = InboxForm()

    return
render(request,'email_rest.html',{'email':e,'inbox':True,'cform':cform}
)

def Email(request):
    e = Inbox.objects.filter(receiver_id_id=request.user.id)
    #e = Inbox.objects.all()
    cform = InboxForm()

    return
render(request,'email.html',{'email':e,'inbox':True,'cform':cform})



def EmailDetail(request,pk):
    e=Inbox.objects.get(id=pk)
    return render_to_response('email.html',{'mail':e,'inbox':True})
```

```
from .form import InboxForm

from django.template import RequestContext
def Compose(request):
    form = InboxForm()
    sform = SentForm()
    return
render(request,'email.html',{'compose':True,'form':form,'user':request.
user,'sform':sform})

from django.http.response import HttpResponse,HttpResponseRedirect

@login_required(login_url='/accounts/login')

def SentView(request):
    d=Sent_item.objects.all()
    #return HttpResponse(d)
    return
render_to_response('email.html',{'email':d,'sent':True,'user':request.u
ser})

def SentItem(request,pk):
    d=Sent_item.objects.get(id=pk)
    return
render_to_response('email.html',{'mail':d,'sent_item':True,'user':reque
st.user})


def sending(request):
    if request.method=='POST':
        form=InboxForm(request.POST)
        sent_form = SentForm(request.POST)
        if form.is_valid():
            f=form.save(commit=False)
            s=sent_form.save(commit=False)
            f.sender_id =
str(request.user.first_name)+'<'+str(request.user.email)+'>'
            s.sender_id =
str(request.user.first_name)+'<'+str(request.user.email)+'>'
            f.save()
            s.save()
```

```
            return HttpResponseRedirect('/email/')

@login_required(login_url='/accounts/login')

def CreateProject(request):
    if request.method == 'POST':
        form = ProjectForm(request.POST)
        if form.is_valid():
            form.save()
            return HttpResponseRedirect('/')
    else:
        form = ProjectForm()
        #project=True
        return
render(request,'home.html',{'form':form,'user':request.user,'project_c'
:True})


def Searching(request,srchkey):
    if request.method=='GET':
        search_q=request.GET['search']
        if(srchkey=='project'):
            query=Task.objects.filter(name__contains=search_q)
            project=Project.objects.filter(name__contains=search_q)

            context={
                'task':query,
                'keyword':search_q,
                'project':project,
                'user':request.user
            }
        else:
            temp=[]

t1=Inbox.objects.filter(content__contains=search_q,subject__contains=se
arch_q)
            t2=Inbox.objects.filter(subject__contains=search_q)
            t3=Inbox.objects.filter(content__contains=search_q)
            t4=Inbox.objects.filter(sender_id__contains=search_q)
            temp.extend(t1)
            temp.extend(t2)
            temp.extend(t3)
```

```
            temp.extend(t4)
            query=list(set(temp))
            temp=[]

t1=Sent_item.objects.filter(content__contains=search_q,subject__contain
s=search_q)
            t2=Sent_item.objects.filter(subject__contains=search_q)
            t3=Sent_item.objects.filter(content__contains=search_q)
            t4=Sent_item.objects.filter(sender_id__contains=search_q)
            temp.extend(t1)
            temp.extend(t2)
            temp.extend(t3)
            temp.extend(t4)
            sent_query=list(set(temp))


            temp=[]
            sent=Sent_item.objects.filter()
            context={
                'inbox':query,
                'sent':sent_query,
                'keyword':search_q,
                #'project':project,
                'user':request.user
            }

        return render_to_response('search_result.html',context)


@login_required(login_url='/accounts/login')
def Edit(request,pk):
    #form=TaskEditForm()
    if request.method =='POST':
        task=get_object_or_404(Task,id=pk)
        form = TaskEditForm(request.POST or None,instance=task)
        if form.is_valid():
            name=request.user.first_name+" "+request.user.last_name
            task.modify_by=name
            task.save()
            form.save()
            return HttpResponseRedirect('/')
        else:
```

```
            return
render(request,'home.html',{'form':form,'work':'taskedit','user':reques
t.user})
    else:
        task=get_object_or_404(Task,id=pk)
        form = TaskEditForm(instance=task)
        return
render(request,'home.html',{'form':form,'work':'taskedit','user':reques
t.user})


import csv
def export_csv(requests):
    from django.utils.encoding import smart_str
    response=HttpResponse(content='text/csv')
    row = csv.writer(response, csv.excel)
    response.write(u'\ufeff'.encode('utf8'))
    query=Task.objects.all()
    response['Content-Disposition'] = 'attachment;
filename="result.csv"'
    header=['ID','Name','Task Description','Comments','Created
By','Created Date','Modify By','Project
Name','Priority','TAT','Status']
    row.writerow(header)
    for f in query:
        row.writerow([
            smart_str(f.id),
            smart_str(f.name),
            smart_str(f.task_description),
            smart_str(f.comments),
            smart_str(f.created_by),
            smart_str(f.created_date),
            smart_str(f.modify_by),
            smart_str(f.project_id.name),
            smart_str(f.priority),
            smart_str(f.tat),
            smart_str(f.status)
        ])
    return response
```

Settings.py

```
import os
BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))
SECRET_KEY = 'a#ru=14f2nceeqp*&e-&#w)5hy53ew6-3bz_)0!-u8q3hvs0i0'
DEBUG = True
ALLOWED_HOSTS = []
CKEDITOR_UPLOAD_PATH = "upload/"
CKEDITOR_JQUERY_URL =
'https://ajax.googleapis.com/ajax/libs/jquery/2.2.4/jquery.min.js'
CKEDITOR_CONFIGS = {
    'default': {
        'toolbar': 'full',
        'height': 'auto',
        'toolbarCanCollapse': True,
        'filebrowserWindowHeight': 725,
        'filebrowserWindowWidth': 940,
        'maxwidth':600,
        'width':'700',
        'resize_maxWidth' :1000,
    },
}


INSTALLED_APPS = (
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'Workflow',
    'rest_framework.authtoken',
    'ckeditor_uploader',
    'ckeditor',
    'rest_framework',
)
```

```
MIDDLEWARE_CLASSES = (
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.auth.middleware.SessionAuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
    'django.middleware.security.SecurityMiddleware',
    'Workflow.middleware.ActiveUserMiddleware',
)

CACHES = {
    'default': {
        'BACKEND': 'django.core.cache.backends.locmem.LocMemCache',
        'LOCATION': 'default-cache'
    }
}

USER_ONLINE_TIMEOUT = 300
USER_LASTSEEN_TIMEOUT = 60 * 60 * 24 * 7

ROOT_URLCONF = 'TeamWorks.urls'

TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR, 'templates')]
        ,
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
                'django.core.context_processors.request',
            ],
        },
```

```
    },
]

WSGI_APPLICATION = 'TeamWorks.wsgi.application'

REST_FRAMEWORK = {

    'DEFAULT_AUTHENTICATION_CLASSES': (
        'rest_framework.authentication.TokenAuthentication',
        'rest_framework.authentication.SessionAuthentication',
    ),
    'DEFAULT_PERMISSION_CLASSES': (
        'rest_framework.permissions.IsAuthenticated',
    )
}

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': os.path.join(BASE_DIR, 'db.sqlite3'),
    }
}

LANGUAGE_CODE = 'en-us'
TIME_ZONE = 'UTC'
USE_I18N = True
USE_L10N = True
USE_TZ = True

STATIC_URL = '/static/'
```

Registration/l_base.html

```
{% load staticfiles %}
<!DOCTYPE html>
<html lang="en">

<meta name="viewport" content="width=device-width, initial-scale=1,
minimum-scale=1, maximum-scale=1" />
<meta name="theme-color" content="orange" />

<head>
    {% include 'css.html' %}
    {% include 'js.html' %}
    <meta charset="UTF-8">
    <title>{% block title %}{% endblock %}</title>
</head>
<style>
    body{
    background-image:url("{% static 'new.png' %}");
    font-size: 14px;
    width: 100%;
    position: fixed;
    background-repeat: no-repeat;
    -moz-background-size: cover;
    -o-background-size: cover;
    background-size: cover;
    }

</style>
<body>
        <div class="row">
            <div class='col-sm-12'>
                {% block content %}
                 {% endblock %}
            </div>
        </div>
</body>
</html>
```

Registration/login.html

```
{% extends 'registration/l_base.html' %}
{% block content %}
        <div class="col-sm-12">
            <h2 align=center>Welcome to TeamWorks</h4>
        </div>
        <div class="col-sm-5 col-sm-offset-3">
            <div class="auth-form-body mt-3" id='login'>
                <form action='/auth/' method="post">
                {% csrf_token %}
                    <div>
                        <label for="id_username" >Username</label>
                        {{ form.username }}
                    </div>
                    <div>
                        <label for="id_password" >Password</label>
                        {{ form.password }}
                    </div>
                    <div id='btn'>
                        <input type="submit" class="btn btn-block btn-
primary" name="submit" value="submit">
                    </div>

                </form>
            </div>
        </div>
        <style>
        #btn{
            margin-top: 10px;
        }
        #login{
            margin-left: 4cm;
        }
        .form-control{
         border-radius: 1px 1px 1px 1px;
         -moz-border-radius: 1px 1px 15px 15px;
         -webkit-border: 15px 15px 15px 15px;
    }
```

```
    .btn{
         border-radius: 1px 1px 1px 1px;
         -moz-border-radius: 1px 1px 15px 15px;
         -webkit-border: 15px 15px 15px 15px;
        </style>
{% endblock %}
```

Registration/logout.html

```
{% extends 'registration/l_base.html' %}

{% block content %}
        <div class="col-sm-12">
            <h2 align=center>Welcome to TeamWorks</h4>
        </div>

        <div class="col-sm-5 col-sm-offset-3">
            <div class="auth-form-body mt-3" id='login'>
                <form action='/auth/' method="post">
                {% csrf_token %}
                    <div>
                        <label for="id_username" >Username</label>
                        {{ form.username }}

                    </div>
                    <div>
                        <label for="id_password" >Password</label>
                        {{ form.password }}

                    </div>
                    <div id='btn'>
                        <input type="submit" class="btn btn-block btn-
primary" name="submit" value="submit">
                    </div>
```

```
                </form>
            </div>
        </div>
        <style>
        #btn{
            margin-top: 10px;
        }
        #login{
            margin-left: 4cm;
        }
        .form-control{
         border-radius: 1px 1px 1px 1px;
         -moz-border-radius: 1px 1px 15px 15px;
         -webkit-border: 15px 15px 15px 15px;
         /*-webkit-box-shadow: rgba(0, 0, 0, 0.5) 0px 0px 5px;
    -moz-box-shadow: rgba(0, 0, 0, 0.5) 0px 0px 5px;*/
    }
    .btn{
         border-radius: 1px 1px 1px 1px;
         -moz-border-radius: 1px 1px 15px 15px;
         -webkit-border: 15px 15px 15px 15px;

        </style>
{% endblock %}
```

Rest_framework/base.html

```
{% extends 'registration/l_base.html' %}
{% block content %}
        <div class="col-sm-12">
            <h2 align=center>Welcome to TeamWorks</h4>
        </div>
        <div class="col-sm-5 col-sm-offset-3">
            <div class="auth-form-body mt-3" id='login'>
                <form action='/auth/' method="post">
```

```
                {% csrf_token %}
                    <div>
                        <label for="id_username" >Username</label>
                        {{ form.username }}

                    </div>
                    <div>
                        <label for="id_password" >Password</label>
                        {{ form.password }}

                    </div>
                    <div id='btn'>
                        <input type="submit" class="btn btn-block btn-
primary" name="submit" value="submit">
                    </div>

                </form>
            </div>
        </div>
        <style>
        #btn{
            margin-top: 10px;
        }
        #login{
            margin-left: 4cm;
        }
        .form-control{
         border-radius: 1px 1px 1px 1px;
         -moz-border-radius: 1px 1px 15px 15px;
         -webkit-border: 15px 15px 15px 15px;
    }
    .btn{
         border-radius: 1px 1px 1px 1px;
         -moz-border-radius: 1px 1px 15px 15px;
         -webkit-border: 15px 15px 15px 15px;
        </style>
{% endblock %}
```

Base.html

```
{% load staticfiles %}
<!DOCTYPE html>
<html lang="en">
<meta name="viewport" content="width=device-width, initial-scale=1,
minimum-scale=1, maximum-scale=1" />
<meta name="theme-color" content="orange" />
<head>
    <meta charset="UTF-8">
    <title>{% block title %}{% endblock %}</title>
</head>
<body>
<div class="container">
    <div class="row">
      <div class="col-sm-12  col-md-12 col-lg-12">
        {% include 'nav.html' %}
      </div>
        <div class="col-sm-12 col-md-12 col-lg-12">
        </div>
          <div class="col-lg-3 col-md-3 col-sm-3">
              {% block sidebar %}
              {% endblock %}
          </div>
          <div class="col-lg-9 col-md-9 col-sm-9">
            {% block content %}
            {% endblock %}
        </div>
      </div>
    </div>
{% block script %}
{% endblock %}

{% block BackboneScript %}
{% endblock %}
</body>
</html>
```

Ck.html

```
{% load staticfiles %}
<script src="{% static 'ckeditor/ckeditor.js' %}" ></script>
<script src="{% static 'ckeditor/ckeditor-init.js' %}"></script>
```

Css.html

```
{% load staticfiles %}
<link rel="stylesheet" href="{% static
'bootstrap/css/bootstrap.min.css' %}">
<link rel="stylesheet" href="{% static 'style.css' %}">
```

Email.html

```
{% extends 'base.html' %}
{% load staticfiles %}

{% block title %}
{% if compose %} Compose | TeamWorks {% endif %} {% if inbox %} Inbox |
TeamWorks {% endif %} {% endblock %} {% block sidebar %}
<script src="{% static 'ckeditor/ckeditor.js' %}" ></script>
<script src="{% static 'ckeditor/ckeditor-init.js' %}"></script>
    <div class="affix-top hidden-print bs-docs-slidebar">
    <h3>Email</h3>
    <div class="list-group">
        <a class="list-group-item" href="#demo" data-
toggle="collapse">Collapse</a>
        <div id="demo" class="collapse in">
            {% if compose %}
            <a class="list-group-item active"
href="#compose">Compose</a> {% else %}
```

```
            <a class="list-group-item" href="#compose">Compose</a> {%
endif %} {% if inbox %}
            <a class="list-group-item active" href="#">Inbox</a> {%
else %}
            <a class="list-group-item" href="#">Inbox</a> {% endif %}
{% if sent %}
            <a class="list-group-item action" href="#sent">Sent
Items</a> {% else %}
            <a class="list-group-item" href="#sent">Sent Items</a> {%
endif %} {% if trans %}
            <a class="list-group-item active" href="#trans">Trans</a>
{% else %}
            <a class="list-group-item" href="#trans">Trans</a> {% endif
%}
        </div>
    </div>

</div>
{% endblock %}

{% block content %}

{% if mail %}
<div class="cont">
    <div style="margin-top: 15px;">
        <h3>{{ mail.subject }}</h3>
    </div>
    <hr>
    <div>
        <div style="float: left">
            <span style="font-weight: bold">From : </span> {{
mail.sender_id }}
        </div>
        <div style="float: right">
            <span><b>{{ mail.date }}</b></span>
        </div>
    </div>
    <br><br>
    <div>
        <p>{{ mail.content |safe }}</p>
    </div>
```

```
</div>
<hr>

<style>
    .marker {
        background: yellow
    }
</style>
{% elif compose %}
{% include 'ck.html' %}

<form action="/sending/" method="POST">
    {% csrf_token %}
    <div class="row messages">
        <div class="col-sm-1">
            <b>To</b>
        </div>
        <div class="col-sm-7">
            {{form.receiver_id}}
        </div>
    </div>
    <div class="row messages">
        <div class="col-sm-1">
            <b>Subject</b>
        </div>
        <div class="col-sm-10">
            {{form.subject}}
        </div>
    </div>
    <div class="row messages">
        <div class="col-sm-1">
            <b>Message</b>
        </div>
        <div class="col-sm-10">
            <div class="editor" id="editor">
                {{form.content}}
            </div>
            <div class="submit-button">
                <input type="submit" class="btn btn-primary btn-block"
value="Send" id="submit">
            </div>
```

```
        </div>
    </div>
</form>
<style>
    .messages {
        margin-top: 25px;
    }

    .texarea {
        min-height: 50%;
        width: 100%;
        max-width: 100%;
        min-width: 100%;
    }

    .subject {
        width: 100%;
    }

    .submit-button {
        margin-top: 5px;
    }
</style>


{% else %}
<div class="cont-here">
    <table class="table" id="table">
        <thead>
            <th><input type="checkbox" id="all"></th>
            <th> Name </th>
            <th> Subject </th>
            <th> Time </th>
            <th> Receiver </th>
        </thead>
        {% for i in email %} {% if i.is_read %}
            <a href='#email/{{i.id}}'>
        <tr href='#email/{{i.id}}'>
            <td><input onclick="not()" class="check" value="5"
type="checkbox"></td>
```

```
            <td href="#email/{{ i.id }}" style="cursor:pointer"><a
href="#email/{{i.id}}">{{ i.sender_id }}</a></td>
            <td href="#email/{{ i.id }}" style="cursor:pointer">{{
i.subject }}</td>
            <td href="#email/{{ i.id }}" style="cursor:pointer">{{
i.date.date }}</td>
            <td href="#email/{{ i.id }}" style="cursor:pointer">{{
i.receiver_id.first_name }}</td>
        </tr>
            </a>
        {% else %}
        <a href='#email/{{i.id}}'>
        <tr id="#email/{{i.id}}" class='row_data' style="background:
rgba(243,243,243,.85);font-weight: bold;draggable:true">

            <td><input onclick="not()" class="check" value="5"
type="checkbox"></td>

            <td href="#email/{{ i.id }}" style="cursor:pointer"><a
href="#email/{{i.id}}">{{i.sender_id }}</a></td>
            <td href="#email/{{ i.id }}" style="cursor:pointer">{{
i.subject }}</td>
            <td href="#email/{{ i.id }}" style="cursor:pointer">{{
i.date.date }}</td>
            <td href="#email/{{ i.id }}" style="cursor:pointer">{{
i.receiver_id.first_name }}</td>

        </tr>
        </a>
        {% endif %} {% endfor %}

    </table>
</div>
{% endif %}

<style>
    th {
        font-size: 1.2em;
    }

    table {
        margin-top: 5px;
```

```
    }

    .cont {
        margin-left: 15px;
        text-align: justify;
    }

    .affix-top {
        position: fixed;
        width: 19.45%;
    }

    @media (max-width: 767px) {
        .affix-top {
            width: auto;
        }
    }

    @media (max-width: 770px) {
        .affix-top {
            position: static;
            width: auto;
        }
    }

    .form-control {
        border-radius: 1px 1px 1px 1px;
        -moz-border-radius: 1px 1px 15px 15px;
        -webkit-border: 15px 15px 15px 15px;
    }

    form-control:focus {
        border-color: snow;
        outline: 0;
        -webkit-box-shadow: inset 0 1px 1px rgba(0, 0, 0, .075), 0 0
8px rgba(0, 0, 0, .6);
        box-shadow: inset 0 1px 1px rgba(0, 0, 0, .075), 0 0 8px
rgba(0, 0, 0, .6);
    }
</style>

{% endblock %} {% block script %}
```

```
{% include 'ck.html' %}
<script type="text/javascript">
    $(window).on('resize', function() {
        var win = $(this);
        if (win.width() < 770) {
            $('#demo').removeClass('collapse in')
            $('#demo').addClass('collapse')
        } else if (win.width() > 770) {
            $('#demo').removeClass('collapse')
            $('#demo').addClass('collapse in')
        }
    });



    $(document).ready(function() {
        //$('#cke_id_content').addClass('cke_focus')
        if ($(window).width() < 770) {
            $('#demo').addClass('collapse')
        }

    });


    $(document).ready(function() {
        console.log("sdlfkjs")
    });




    $("#all").click(function() {
        if ($("#all").is(':checked')) {
            $(".check").prop("checked", true);
        } else {
            $(".check").prop("checked", false);
        }
    });

    $(document).on('click', '.rowlink', function(e) {
        console.log($('.rowlink').attr('id'))
    });
```

```
    function func(e) {
        console.log(e);
    }
</script>
{% endblock %}

{% block BackboneScript %}

<script type="text/template" id='mail-here'>

<table class="table">
    <thead>
        <th><input type="checkbox" id="all"></th>
        <th> Name </th>
        <th> Subject </th>
        <th> Time </th>
         <% _.each(mail,function(i) { %>
         </thead>
         <% if (!i.is_read){ %>
         <tr style="background: rgba(243,243,243,.85);font-weight:
bold;draggable:true">
             <td>
             <input onclick="not()" class="check" value="5"
type="checkbox"></td>
             <td><a href='#email/<%=i.id%>'><%=i.sender_id%></a></td>
             <td><%=i.subject%></td>
             <td><%=i.date%></td>
        </tr>
             <% }else{%>
        <tr style="draggable:true">
                <td>
                <input onclick="not()" class="check" value="5"
type="checkbox"></td>
                <td><a
href='#email/<%=i.id%>'><%=i.sender_id%></a></td>
                <td><%=i.subject%></td>
                <td><%=i.date%></td>
            </tr>
          </td>
        <% }}); %>
</table>
</script>
```

```
<script type='text/template' id='msg-show'>
    <div style="margin-top: 15px;">
        <h3><%=mail.subject%></h3>
    </div>
    <hr>
    <div>
        <div style="float: left">
            <span style="font-weight: bold">From : </span>
<%=mail.sender_id%>
        </div>
        <div style="float: right">
            <span><b><%=mail.date%></b></span>
        </div>
    </div>
    <br><br>
    <div>
        <p><%=mail.content%></p>
    </div>
<hr>
<style>
    .marker {
        background: yellow
    }
</style>
</script>
{% load staticfiles %}
<script type='text/template' id='compose-msg'>
<style>
    .messages {
        margin-top: 25px;
    }
    .texarea {
        min-height: 50%;
        width: 100%;
        max-width: 100%;
        min-width: 100%;
    }
    .subject {
        width: 100%;
    }
```

```
    .submit-button {
        margin-top: 5px;
    }
</style>

    <form action="/sending/" method="POST">
    {% csrf_token %}
    <div class="row messages">
        <div class="col-sm-1">
            <b>To</b>
        </div>
        <div class="col-sm-7">
            {{cform.receiver_id}}
        </div>
    </div>
    <div class="row messages">
        <div class="col-sm-1">
            <b>Subject</b>
        </div>
        <div class="col-sm-10">
            {{cform.subject}}
        </div>
    </div>

    <div class="row messages">
        <div class="col-sm-1">
            <b>Message</b>
        </div>
        <div class="col-sm-10">
            <div class="editor" id="editor">
                {{cform.content}}
            </div>
            <div class="submit-button">
                <input type="submit" class="btn btn-primary btn-block"
value="Send" id="submit">
            </div>

        </div>
    </div>
</form>
```

```
</script>


<script>

    $.ajaxPrefilter( function( options, originalOptions, jqXHR ) {
        options.url = 'http://localhost:8000/api' + options.url;
    });
    var Email = Backbone.Model.extend({
        urlRoot:'/email/'
    });

    var Emails = Backbone.Model.extend({
        url:'/email/'
    });

    var Sent = Backbone.Model.extend({
        urlRoot:'/sent/'
    });

    var Sents = Backbone.Model.extend({
        url:'/sent/'
    })


    var EmailView = Backbone.View.extend({

        render:function(){
            console.log('sdf')
            var that = this;
            var email = new Emails()
            email.fetch({
                success:function(data){
                    var templates = _.template($('#mail-here').html())
                    $('.cont-
here').html(templates({'mail':data.attributes}))
                    //console.log(d)
                }
            })
        },
single:function(option){
            var that=this
            if(option.id){
```

```
                that.email = new Email({id:option.id})
                that.email.fetch({
                    success:function(data){
                        console.log(data.attributes)
                        var template = _.template($('#msg-
show').html())
                        $('.cont-
here').html(template({'mail':data.attributes}))
                    }
                })
            }
        },
        compose:function(){
            console.log('hsdjflkdjfsd')
                var template = _.template($('#compose-msg').html())
                $('.cont-here').html(template())
        },
        sent:function(){
            var sent = new Sents();
            sent.fetch({
                success:function(data){
                        console.log(data)
                var template = _.template($('#mail-here').html())
                $('.cont-
here').html(template({'mail':data.attributes}))
                }
            })
        },
    });
    var eview = new EmailView()
    var Router = Backbone.Router.extend({
        routes:{
            '':'home',
            'project/:id':'project',
            'email/:id':'email',
            'compose':'compose',
            'sent':'sent',
            'task/edit/:id':'taskedit',
            'task/update/:id':'taskupdate',
            'createproject':'createproject',
        }
    });
```

```
    var router = new Router()
    router.on('route:createproject',function(){
        console.log('here')
        //pview.CreateProject()

    });
    router.on('route:home',function(){
        eview.render()
    });
    router.on('route:project',function(id){
    });
    router.on('route:compose',function(){
        eview.compose()
    });
    router.on('route:sent',function(){
        eview.sent()
    })
    router.on('route:email',function(id){
        eview.single({id:id})
    });
    Backbone.history.start();
</script>
{% endblock %}
```

Home.html

```
{% extends 'base.html' %} {% block title %} {{ pid.name }} Project |
WorkFlow {% endblock %} {% block sidebar %}
<!--<div class="affix-top hidden-print bs-docs-slidebar hidden-xs">-->
<div class="affix-top hidden-print bs-docs-slidebar">
    <h3>Projects</h3>
    <div class="list-group">
        <a class="list-group-item" href="#demo" data-toggle="collapse">
Collapse</a>
        <div id="demo" style="cursor:pointer">
            {% if pid == 'home' %}
```

```
                <a class="list-group-item active" href="/">All</a>
             {% else %}
                <a class="list-group-item" href="/">All</a>
            {% endif %}
            {% for i in project %}
            {% if i.id == pid.id or i.id == task.project_id_id %}
                <a class="list-group-item active" href="#project/{{
i.id }}">{{ i.name }}</a>
             {% else %}
                <a class="list-group-item" href="#project/{{ i.id
}}">{{ i.name }}</a>
                {% endif %}
            {% endfor %}
            {% if 'Workflow.add_project' in user.get_all_permissions %}
                <a class="list-group-item" href="#createproject">Create
New</a>
            {% endif %}

        </div>
    </div>

</div>
{% endblock %}

{% block content %}

<div>
    <h5><a href='/'> Workflow</a>>><a
href="/project/{{pid.id}}">{{pid.name}}</a></h5>
</div>
<div>
    <div class='cont'></div>
</div>
<style>

    .borderless tr td{
        border:none ;
        padding: 3px;
    }

    th {
        font-size: 1.2em;
```

```
    }

    table {
        margin-top: 5px;
    }

    .affix-top {
        position: fixed;
        width: 19.45%;
    }

    @media (max-width: 767px) {
        .affix-top {
            width: auto;
        }
    }

    @media (max-width: 770px) {
        .affix-top {
            position: static;
            width: auto;
        }
    }

    .form-control {
        border-radius: 1px 1px 1px 1px;
        -moz-border-radius: 1px 1px 15px 15px;
        -webkit-border: 15px 15px 15px 15px;
    }

    form-control:focus {
        border-color: snow;
        outline: 0;
        -webkit-box-shadow: inset 0 1px 1px rgba(0, 0, 0, .075), 0 0
8px rgba(0, 0, 0, .6);
        box-shadow: inset 0 1px 1px rgba(0, 0, 0, .075), 0 0 8px
rgba(0, 0, 0, .6);
    }
</style>

{% endblock %} {% block script %}
<script type="text/javascript">
```

```
    $(window).on('resize', function() {
        var win = $(this);
        if (win.width() < 770) {
            $('#demo').removeClass('collapse in')
            $('#demo').addClass('collapse')
        } else if (win.width() > 770) {
            $('#demo').removeClass('collapse')
            $('#demo').addClass('collapse in')
        }
    });
    $(document).ready(function() {
        //$('#cke_id_content').addClass('cke_focus')
        if ($(window).width() < 770) {
            $('#demo').addClass('collapse')
        }
    });
</script>
{% endblock %}

{% block BackboneScript %}
<script type="text/template" id="task-list-template">
    {% if 'Workflow.add_task' in user.get_all_permissions %}
    <h4 align='center'><a href='#createtask/'>Create a new
Task</a></h4>
    {% endif %}
    <table class='table'>
        <thead>
                <th> Name </th>
                <th> Created Date </th>
                <th> Created By </th>
                <th> Priority </th>
                <th> Status </th>
        </thead>
        <% _.each(project,function(p) { %>
            <tr>
                <td><a href="#task/<%=p.id %>"><%= p.name %></a></td>
                <td><%= p.created_date %></td>
                <td><%= p.created_by %></td>
                <td><%= p.priority %></td>
                <td><%= p.status %></td>
            </tr>
        <% }); %>
```

```
    </table>
</script>
<script type="text/template" id="task-all-list-template">
       <table class='table'>
        <thead>
                <th> Name </th>
                <th> Created Date </th>
                <th> Created By </th>
                <th> Priority </th>
                <th> Status </th>
        </thead>
        <% _.each(task,function(p) { %>
            <tr>
                <td><a href="#task/<%=p.attributes.id %>"><%=
p.attributes.name %></a></td>
                <td><%= p.attributes.created_date %></td>
                <td><%= p.attributes.created_by %></td>
                <td><%= p.attributes.priority %></td>
                <td><%= p.attributes.status %></td>
            </tr>
        <% }); %>
    </table>
</script>

<script type="text/template" id='project-form'>
    <form id='prj-form'>
        {% csrf_token %}
        <h3 align=center>Create New Project</h3>
        <table class='table'>
            <tr>
                <td>Name : </td>
                <td><input type='text' name='name' class='form-
control'></td>
            </tr>
            <tr>
                <td>Description : </td>
                <td><textarea class='form-control' name='description'
cols="30" rows="10"></textarea></td>
            </tr>
            <tr>
                <td>Head : </td>
```

```
                <td><!--<input type='text' head='head' class='form-
control'>-->{{pform.head}}</td>
            </tr>
            <tr>
                <td></td>
                <td><input type='submit' class='btn btn-block btn-
primary' value='Create'/>
            </tr>
        </table>
    </form>
</script>

<script type='text/template' id='task-show'>
    <table class="table">
            <thead>
                <th>Attribute</th>
                <th>Value</th>
            </thead>
            <tr>
                <td>Name</td>
                <td><%=task.name%></td>
            </tr>
            <tr>
                <td>Created Date</td>
                <td><%=task.created_date%></td>
            </tr>
            <tr>
                <td>Created By</td>
                <td><%=task.created_by%></td>
            </tr>
            <tr>
                <td>Status</td>
                <td><%=task.status%></td>
            </tr>
            <tr>
                <td>Description</td>
                <td><%=task.task_description%></td>
            </tr>

            <tr>
                <td>Comment</td>
                <td><%=task.comments%></td>
```

```
            </tr>
            <tr>
                <td>Modify By</td>
                <td><%=task.modify_by%></td>
            </tr>
            <tr>
                <td>Priority</td>
                <td><%=task.priority%></td>
            </tr>
            <tr>
                <td>TAT</td>
                <td><%=task.tat%></td>
            </tr>
            <tr>
                <td>Assign</td>
                <td><%=task.assign_name%></td>
            </tr>
            <tr>
                    {% if 'Workflow.delete_task' in
user.get_all_permissions  %}
                    <td><input type=submit class='btn btn-danger btn-
block' value='Delete'/></td>
                {% endif %}
                <td colspan=2><a href='#task/edit/<%=task.id%>'><input
type=button class='btn btn-primary btn-block' value='Edit'/></a></td>
        </table>
</script>

<script type='text/template' id='creation-task'>
<form method='POST' class='task-creation'>
    {% csrf_token %}
    <table class="table">
        <thead>
            <th>Attribute</th>
            <th>Value</th>
        </thead>
        <tr>
            <td>Name</td>
            <td>{{taskform.name}}</td>
        </tr>
        <tr>
            <td>Created Date</td>
```

```
            <td>{{taskform.created_date}}</td>
        </tr>
        <tr>
            <td>Created By</td>
            <td><input type=text id={{user.id}} name='created_by'
value="{{user.get_full_name}}" class='form-control' disabled=true></td>
        </tr>
        <tr>
            <td>Status</td>
            <td>{{taskform.status}}</td>
        </tr>
        <tr>
            <td>Description</td>
            <td>{{taskform.task_description}}</td>
        </tr>

        <tr>
            <td>Comment</td>
            <td>{{taskform.comments}}</td>
        </tr>
       <tr>
            <td>Priority</td>
            <td>{{taskform.priority}}</td>
        </tr>
        <tr>
            <td>TAT</td>
            <td>{{taskform.tat}}</td>
        </tr>
        <tr>
            <td>Assign</td>
            <td>{{taskform.assign}}</td>
        </tr>
        <tr>
            <td colspan=2><input type=submit class='btn btn-primary
btn-block' value='Create'/></td>

    </table>
    </form>
</script>

<script type='text/template' id='task-edit'>
    <form class='task-edit-form' id='<%=task.id%>'>
```

```
            {% csrf_token %}
        <table class='table borderless'>
            <tr>
                <td>Name</td>
                <td><input type='text'  class='form-control'
name='name' value='<%=task.name%>'/></td>
            </tr>
            <tr>
                <td>Status</td>
                <td><input type='text'  class='form-control'
name='status'  value='<%=task.status%>'/></td>
            </tr>
            <tr>
                <td>Description</td>
                <td><input type='text'  class='form-control'
name='task_description'  value='<%=task.task_description%>'/></td>
            </tr>
           <tr>
                <td>Comment</td>
                <td><input type='text'  class='form-control'
name='comments'  value='<%=task.comments%>'/></td>
            </tr>
            <tr>
                <td>Priority</td>
                <td><input type='text'  class='form-control'
name='priority'  value='<%=task.priority%>'/></td>
            </tr>
            <tr>
                <td>TAT</td>
                <td><input type='text'  class='form-control' name='tat'
value='<%=task.tat%>'/></td>
            </tr>
            <tr>
                <td>Assign</td>

                <td>{{taskform.assign}}</td>
            </tr>
            <tr>
                <td colspan=2><input type=submit class='btn btn-primary
btn-block' value='Update'/></a></td>
        </table>
    </form>
```

```
    <script type='text/javascript'>
        var $inputProduct = $("select[id='id_assign']")
    </script>
</script>
<script>
        $.ajaxPrefilter( function( options, originalOptions, jqXHR ) {
          options.url = 'http://localhost:8000/api' + options.url;
        });
        var Projects=Backbone.Collection.extend({
            url:'/project/'
        })
        var Tasks=Backbone.Collection.extend({
            url:'/task/'
        })
        var Task = Backbone.Model.extend({
            urlRoot:'/task/'
        })
        var Project = Backbone.Model.extend({
            urlRoot:'/project'
        })
        var ProjectView = Backbone.View.extend({
            el:'.cont',
            events:{
                'submit .task-edit-form':'updateTask',
                'submit #prj-form':'newproject',
                'submit .task-creation':'createtask',
            },
            updateTask:function(ev){
                ev.preventDefault()
                id = $('.task-edit-form').attr('id')
                var api = '/task/'+$('.task-edit-form').attr('id')+'/'
                var task = new Task({id:id})
                this.form = $(ev.currentTarget);
                dt={}
                csrftoken =  $('input[name=csrfmiddlewaretoken]').val()
                $('input',this.form).each(function(){
                    var val=$(this).val()
                    var key=$(this).attr('name')
                    dt[key]=val
                    })
```

```
dt['assign']='http://localhost:8000/api/user/'+($('#id_assign
:selected').attr('value')).toString()+'/'
                    $.ajax({
                     type:'PUT',
                     url:api,
                     data : JSON.stringify(dt),
                     beforeSend:function(xhr,settings){
                         xhr.setRequestHeader("X-CSRFToken", csrftoken)
                     },
                     crossDomain:false,
                     contentType: "application/json; charset=utf-8",
                     success: function(d)
                      {
                         var t='task/'+id
                         router.navigate(t, {trigger:true});

                     },

                 })
             },
             createtask:function(ev)
             {
                 ev.preventDefault()
                 console.log(project_id)
                 var api = '/task/'
                 var task = new Task()
                 this.form = $(ev.currentTarget)
                 csrftoken =  $('input[name=csrfmiddlewaretoken]').val()
                 dt={}
                 dt['name']=this.form.context.elements.name.value
                 dt['status']=this.form.context.elements.status.value

dt['task_description']=this.form.context.elements.task_description.valu
e
dt['comments']=this.form.context.elements.comments.value
                dt['priority']=this.form.context.elements.priority.value
                 dt['tat']=this.form.context.elements.tat.value
dt['assign']='http://localhost:8000/api/user/'+this.form.context.elemen
ts.assign.value+'/'
dt['id_created_date']=this.form.context.elements.created_date.value
```

```
dt['created_by']=this.form.context.elements.created_by.value
                dt['project_id']=project_id
                console.log(dt)

                $.ajax({
                    type:'POST',
                    url:api,
                    data:JSON.stringify(dt),
                    beforeSend:function(xhr,settings){
                        xhr.setRequestHeader("X-CSRFToken", csrftoken)
                    },
                    crossDomain:false,
                    contentType: "application/json; charset=utf-8",
                    success: function(d)
                     {
                        console.log('succ')
                        router.navigate('', {trigger:true});
                     },
                })
            },
            newproject:function(e){
                console.log('click')
                var api = '/project/'
                var project = new Project()
                var form = $(e.currentTarget)
                console.log(form)
                e.preventDefault()
                csrftoken =  $('input[name=csrfmiddlewaretoken]').val()
                console.log($('input[name=name]').val())
                var dt={
                        'name':$('input[name=name]').val(),

'description':$('textarea[name=description]').val(),

'head':'http://localhost:8000/api/department/'+form.context.elements.he
ad.value+'/'
                    }
                $.ajax({
                    type:'POST',
                    url:api,
                    data:JSON.stringify(dt),
```

```
                beforeSend:function(xhr,settings){
                    xhr.setRequestHeader("X-CSRFToken", csrftoken)
                },
                crossDomain:false,
                contentType: "application/json; charset=utf-8",
                success: function(d)
                 {
                    router.navigate('', {trigger:true});
                 },
            })
        },
        render:function(option){
            var that = this;
            if(option.id){
                project_id=option.id;
                that.project = new Project({id:option.id})
                that.project.fetch({
                    success : function(e) {
                        //console.log(e)
                    var data = e.attributes.task_set
                    var template = _.template($('#task-list-
template').html())
                    console.log(e)
                    $('.cont').html(template({'project':data}))

                    }
                })
            }
        },
        CreateProject:function(e){
            var template = _.template($('#project-form').html())
            $('.cont').html(template())
        },

        CreateTask:function(e){
            var template = _.template($('#creation-task').html())
            $('.cont').html(template())
        },

        home:function(){
            var task = new Tasks();
            task.fetch({
```

```
                    success : function (e) {
                        var data = e.models
                        var template = _.template($('#task-all-list-
template').html())
                        $('.cont').html(template({'task':data}))
                    }
                })
            },
            task:function(option){
                var that = this;
                if(option.id){
                    var task = new Task({id:option.id})
                    task.fetch({
                        success : function(e){
                            var data = e.attributes
                            //console.log(data)
                            var template = _.template($('#task-
show').html())
                            $('.cont').html(template({'task':data}))
                        }
                    })
                }
            },
            taskedit:function(option){
                var that = this;
                if(option.id){
                    var task = new Task({id:option.id})
                    task.fetch({
                        success : function(e){
                            var data = e.attributes
                            //console.log(data)
                            var template = _.template($('#task-
edit').html())
                            $('.cont').html(template({'task':data}))
                        }
                    })
                }
            }
        })
        var pview = new ProjectView()
        var Router = Backbone.Router.extend({
            routes:{
```

```
                '':'home',
                'project/:id':'project',
                'task/:id':'task',
                'task/edit/:id':'taskedit',
                'task/update/:id':'taskupdate',
                'createproject':'createproject',
                'createtask/':'createtask'
            }
        });
        var router = new Router()
        router.on('route:createtask',function(){
            console.log('create')
            pview.CreateTask()
        });
        router.on('route:createproject',function(){
            console.log('here')
            pview.CreateProject()
        });
        router.on('route:home',function(){
            pview.home()
        });
        router.on('route:project',function(id){
            pview.render({id:id});
        });

        router.on('route:taskedit',function(id){
            pview.taskedit({id:id})
        })
        router.on('route:task',function(id){
            pview.task({id:id})
        });
        Backbone.history.start();
    </script>

{% endblock %}
```

Js.html

```
{% load staticfiles %}
<script src="{% static 'jquery.js' %}"></script>
<script src="{% static 'bootstrap/js/bootstrap.min.js' %}"></script>
<script src="{% static 'backbone/underscore.js' %}"></script>
<script src="{% static 'backbone/backbone.js' %}"></script>
<script src="{% static 'Chart.min.js' %}"></script>
```

nav.html

```
{% load staticfiles %}
{% include 'css.html' %}
{% include 'js.html' %}
    <nav class="nav navbar-inverse navbar-fixed-top navbar-light"
style="background-color: orange">
      <div class="container-fluid">
        <div class="navbar-header" >
          <button type="button" class="navbar-toggle collapsed" data-
toggle="collapse" data-target="#navbar" aria-expanded="false" aria-
controls="navbar">
            <span class="sr-only">Toggle navigation</span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
          </button>
          <a class="navbar-brand" style="color: white"
href="/">TeamWorks</a>
        </div>
        <div id="navbar" class="navbar-collapse collapse">
          <ul class="nav navbar-nav navbar-right">
            <li><a href="/" >Dashboard</a></li>
            <li><a href="/email">Email</a></li>
            <li><a href="/reports">Report</a></li>
            {% if user.is_authenticated %}
                <li><a href="/accounts/profile">{{ user.first_name
}}</a></li>
            {% else %}
                <li><a href="/accounts/login">Login</a></li>
```

```
            {% endif %}
          </ul>
          <form class="navbar-form navbar-left" action='/search/{% if
project %}project/{%else%}email/{%endif%}'>
            <input class="form-ctr" size=50 id=all_search name="search"
placeholder="Search..." style="background-color:#FEAB40;" type="text">
          </form>
        </div>
      </div>
    </nav>
    <nav class="nav navbar-inverse  ">
      <div class="container-fluid">
        <div class="navbar-header">
          <button type="button" class="navbar-toggle collapsed" data-
toggle="collapse" data-target="#navbar" aria-expanded="false" aria-
controls="navbar">
            <span class="sr-only">Toggle navigation</span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
          </button>
          <a class="navbar-brand" style="color: white"
href="#">TeamWorks</a>
        </div>
        <div id="navbar" class="navbar-collapse collapse">
          <ul class="nav navbar-nav navbar-right">
            <li><a href="#">Dashboard</a></li>
            <li><a href="#">Email</a></li>
            <li><a href="#">Report</a></li>
            <li><a href="#">Profile</a></li>
          </ul>
 <form class="navbar-search col-sm-5 docs-search" ng-submit="submit()">
<span class="glyphicon glyphicon-search search-icon "></span>
                        <input type="text"
                               name="q"
                               class="search-query"
                               placeholder="search"
                               autocomplete="off">
                    </form>
        </div>
      </div>
    </nav>
```

```
<style>
    .navbar-inverse .navbar-nav > li > a{
    color: white;
    }
    .form-ctr{

display: block;
    width: 100%;
    height: 34px;
    padding: 6px 12px;
    font-size: 14px;
    line-height: 1.42857143;
    color: #555;
    background-color: #fff;
    background-image: none;
    border: 1px solid gray;
    border-radius: 4px;
    -webkit-box-shadow: inset 0 1px 1px rgba(0,0,0,.075);
    outline: none;
    -webkit-transition: border-color ease-in-out .15s,-webkit-box-
shadow ease-in-out .15s;
    -o-transition: border-color ease-in-out .15s,box-shadow ease-in-out
.15s;
    transition: border-color ease-in-out .15s,box-shadow ease-in-out
.15s;}
    .sdfsdf{

    -webkit-transition: border-color ease-in-out .15s,-webkit-box-
shadow ease-in-out .15s;
    -o-transition: border-color ease-in-out .15s,box-shadow ease-in-out
.15s;
    transition: border-color ease-in-out .15s,box-shadow ease-in-out
.15s;

        box-shadow: inset 0 1px 1px rgba(0,0,0,.075);
    color: #555;
    background-color: #fff;
    background-image: none;
    -webkit-box-shadow: inset 0 1px 1px rgba(0,0,0,.075);
    }
</style>
```

Reports.html

```
{% extends 'base.html' %} {% block title %} Reports {% endblock %} {%
block sidebar %}
<div class="affix-top hidden-print bs-docs-slidebar">
    <h3>Reports</h3><div class="list-group">
<a class="list-group-item" href="#demo" data-
toggle="collapse">Collapse</a>
        <div id="demo" style="cursor:pointer">{% if project %}
            <a class="list-group-item active tree-toggler nav-
header">Projects</a>
            <ul style="display: none;" class="nav nav-list tree">
                <li><a href="#" style="cursor:pointer">Status</a></li>
                <li><a href="#"
style="cursor:pointer">Priority</a></li></ul>{% else %}
            <a class="list-group-item"
onclick="ajaxcl('Project')">Projects</a> {% endif %} {% if emp %}
            <a class="list-group-item active"
href="/employee">Employee</a> {% else %}
            <a class="list-group-item"
onclick="ajaxcl('Employees')">Employee</a> {% endif %} {% if department
%}
            <a class="list-group-item action"
href="/department">Department</a> {% else %}
            <a class="list-group-item"
onclick="ajaxcl('Department')">Department</a> {% endif %}
            <a class="list-group-item" href='/export/'>Download</a>
        </div>
    </div>
</div>
{% endblock %} {% block content %} {{project}}
<div>
    <div class="col-sm-7">
        <span class="coldata"></span>
        <div class="ch1">
            <canvas id="myChart"></canvas>
        </div>
    </div>
    <div class="col-sm-5">
        <span class="coldata1"></span>
        <div class="ch3">
```

```
            <canvas id="myChart3"></canvas>
        </div>
    </div>
</div>
<div>
    <br>
    <div class='cont' style="margin-top:15px;"></div>
</div>
{% endblock %}
{% block BackboneScript %}
<script type="text/template" id="task-list-template">
<div id='t_record'>
   <table class='table'>
        <thead>
                <th> Name </th>
                <th> Created Date </th>
                <th> Created By </th>
                <th> Priority </th>
                <th> Status </th>
        </thead>
        <% _.each(project,function(p) { %>
            <tr>
                <td><a href="/rest/#task/<%=p.pk %>"
target='_blank'><%= p.fields.name %></a></td>
                <td><%= p.fields.created_date %></td>
                <td><%= p.fields.created_by %></td>
                <td><%= p.fields.priority %></td>
                <td><%= p.fields.status %></td>
            </tr>
        <% }); %>
    </table>
</div>
</script>
{% endblock %}
{% block script %}
<script type="text/javascript">
    function ajaxcl(par) {
        if (par == 'Employees') {
            point = '/api/employee/',
                $.ajax({
                    method: 'GET',
                    url: point,
```

```
                    dataType: 'text',
                    success: function(data) {
                        sch(data, par)
                    }
                })
        } else if (par == 'Project') {
            point = '/api/project_data/' + par + '/'
            $.ajax({
                method: 'GET',
                url: point,
                dataType: 'text',
                success: function(data) {
                    project = []
                    count = []
                    stat = []
                    option = []
                    var json = $.parseJSON(data)
                    for (var i = 0; i < json.label.length; i++) {
                        project.push(json.label[i].fields.name)
                        count.push(json.count[i])
                        stat.push(json.status[i])
                        option.push(json.option[i])
                    }
                    $('#sub-con').remove()
                    $('#t_record').remove()
                    var hh="<h3 id=rtype
title="+par+">"+par.toUpperCase()+"<h3>"
      $('.coldata').html(hh)
                    $('canvas').remove()
                    html = '<canvas id="myChart"></canvas>'
                    html2 = '<canvas id="myChart2"></canvas>'
                    $('.ch1').html(html)
                    $('.ch2').html(html2)
                    showchart(par,'Task');
                }
            })

        } else {
            point = '/api/project_data/' + par + '/',
                $.ajax({
                    method: 'GET',
                    url: point,
```

```
                    dataType: 'text',
                    success: function(data) {
                        project = []
                        count = []
                        var json = $.parseJSON(data)
                        for (i = 0; i < json.label.length; i++) {
                            project.push(json.label[i].fields.name)
                            count.push(json.count[i])
                        }
                        console.log(project)
                        console.log(count)
                        var hh="<h3 id=rtype
title="+par+">"+par.toUpperCase()+"<h3>"
                        $('.coldata').html(hh)
                        $('canvas').remove()
                        $('#sub-con').remove()
                        $('#t_record').remove()
                        html = '<canvas id="myChart"></canvas>'
                        html2 = '<canvas id="myChart2"></canvas>'
                        $('.ch1').html(html)
                        $('.ch2').html(html2)
                        showchart(par,'Project')
                    }
                })
        }
    }
    pont = '/api/project_data/Project'
    $.ajax({
        method: 'GET',
        url: pont,
        success: function(data) {
            project = []
            count = []
            stat = []
            option = []
            var json = data

           for (var i = 0; i < json.label.length; i++) {
                project.push(json.label[i].fields.name)
                count.push(json.count[i])
                stat.push(json.status[i])
                option.push(json.option[i])
```

```
            }
            var par='Project'
            var hh="<h3 id=rtype
title="+par+">"+par.toUpperCase()+"<h3>"
           $('.coldata').html(hh)
            $('canvas').remove()
            $('#sub-con').remove()
            $('#t_record').remove()
            html = '<canvas id="myChart"></canvas>'
            html2 = '<canvas id="myChart2"></canvas>'
            $('.ch1').html(html)
            $('.ch2').html(html2)
            showchart('Project','Task');
        },
        error: function(error_data) {
            console.log('errro')
            console.log(error_data)
        }
    });
    function sch(data, par) {
        project = []
        count = []
        var json = $.parseJSON(data)
        for (var i = 0; i < json.label.length; i++) {
            nm = json.label[i].fields
            project.push(nm.first_name)
            count.push(json.count[i])
        }
        var hh="<h3 id=rtype title="+par+">"+par.toUpperCase()+"<h3>"
        $('.coldata').html(hh)
        $('canvas').remove()
        $('#sub-con').remove()
        $('#t_record').remove()
        html = '<canvas id="myChart"></canvas>'
        html2 = '<canvas id="myChart2"></canvas>'
        $('.ch1').html(html)
        $('.ch2').html(html2)
        showchart(par,'Task');
    }
    function dpt(department) {
        point = '/api/project_data/' + department + '/'
        $.ajax({
```

```
            method: 'GET',
            url: point,
            success: function(data) {
                console.log(data)
                console.log(data.count)
                console.log(data.label)
            }
        })
    }
    function graphClickEvent(evt, array) {
        var activePoints = myChart.getElementAtEvent(evt);
        var theElement =
myChart.config.data.datasets[activePoints[0]._datasetIndex].data[active
Points[0]._index]
        console.log(activePoints)
        var name = activePoints[0]._model.label
        console.log(array)
        GetJsonByName(name)
    }

    function GetJsonByName(par) {
        //console.log(par)
        //console.log($('#rtype'))
        if ($('#rtype').attr('title')=='Employees'){
            var api='/api/task_by_emp/' + par
        }
        else if($('#rtype').attr('title')=='Project'){
            var api='/api/single_project/' + par
        }
       else{
            var api='/api/project_by_department/'+par
            }
        $.ajax({
            method: 'GET',
            url:api,
            success: function(data) {
                sproject=data
                var project = data.option
                var count = []
                for (var i = 0; i < project.length; i++) {
                    count.push(data.label[project[i]].length)
                    $('#myChart3').remove()
```

```
                    html = '<canvas id="myChart3"></canvas>'
                    $('.ch3').html(html)
                }
                    singlechart(par, project, count)
            }
        })
    }
    function AreaChartClick(e)
    {
        var evn=myChartss.getElementAtEvent(e)
        var op=evn[0]._model.label
        console.log(evn[0]._model.label);
        var data = sproject.label[op]
        var template = _.template($('#task-list-template').html())
        $('.cont').html(template({'project':data}))
    }
    var myChartss
    function singlechart(par, project, count) {
        $('#t_record').remove()
        $('.coldata1').html('<h3 id="sub-con"
align=center>'+par.toUpperCase()+'<h3>')
        console.log(project)
        console.log(count)
        var ctx2 =
document.getElementById("myChart3").getContext('2d');
        myChartss = new Chart(ctx2, {
            type: 'polarArea',
            data: {
                labels: project,
                datasets: [
                    {
                    label: project,
                    data: count,
                    backgroundColor: [
                        'rgba(255, 99, 132, 0.2)',
                        'rgba(54, 162, 235, 0.2)',
                        'rgba(255, 206, 86, 0.2)',
                        'rgba(120, 190, 80, 0.2)',
                        'rgba(54, 216, 186, 0.2)',
                        'rgba(100, 129, 100, 0.2)',
                        'rgba(200, 109, 180, 0.2)',
                    ],
```

```
                borderColor: [
                    'rgba(255,99,132,1)',
                    'rgba(54, 162, 235, 1)',
                    'rgba(255, 206, 86, 1)',
                    'rgba(120, 190, 80, 1)',
                    'rgba(200, 216, 186, 1)',
                    'rgba(100, 129, 100, 1)',
                    'rgba(200, 109, 180, 1)',
                ],
                borderWidth: 1
            }]
        },
        options: {
            onClick: AreaChartClick,
            responsive : true,
var myChart
function showchart(par,vname) {
    var ctx = document.getElementById("myChart").getContext('2d');
    myChart = new Chart(ctx, {
        type: 'bar',
        data: {
            labels: project,
            datasets: [{
                label: vname,
                data: count,
                backgroundColor: [
                    'rgba(255, 99, 132, 0.2)',
                    'rgba(54, 162, 235, 0.2)',
                    'rgba(255, 206, 86, 0.2)',
                    'rgba(120, 190, 80, 0.2)',
                    'rgba(54, 216, 186, 0.2)',
                    'rgba(100, 129, 100, 0.2)',
                    'rgba(200, 109, 180, 0.2)',
                ],
                borderColor: [
                    'rgba(255,99,132,1)',
                    'rgba(54, 162, 235, 1)',
                    'rgba(255, 206, 86, 1)',
                    'rgba(120, 190, 80, 1)',
                    'rgba(200, 216, 186, 1)',
                    'rgba(100, 129, 100, 1)',
                    'rgba(200, 109, 180, 1)',
```

```
                    ],
                    borderWidth: 1
                }]
            },
            options: {
                onClick: graphClickEvent,
                scales: {
                    yAxes: [{
                        scaleLabel: {
                            display: true,
                            labelString: vname
                        },
                        ticks: {
                            beginAtZero: true
                        }
                    }],
                    xAxes: [{
                        scaleLabel: {
                            display: true,
                            labelString: par
                        },
                        ticks: {
                            beginAtZero: true
                        }
                    }]
                }
            }
        });

    }
    $(document).ready(function() {
        $('label.tree-toggler').click(function() {
            $(this).parent().children('ul.tree').toggle(300);
        });
    });
</script>
{% endblock %}
```

Search_result.html

```
{% extends 'base.html' %}
{% block title %} Result for {{ keyword }} {% endblock %}
{% block content %}{% if task %}
        <div style="background-color: transparent">
        <div style="font-size: 2em;height: 120px;padding-top: 10px;">
        <table width="80%">
        <tr><td>
            <h3>There are <b>{{task.count}}</b> task matching keyword
of <b>{{ keyword }}</b></h3>
        </td></tr></table>
    </div>
    <div >
    <div>
        <table class="table" id="table">
            <thead>
                <th> Name </th>
                <th> Created Date </th>
                <th> Created By </th>
                <th> Priority </th>
                <th> Status </th>
            </thead>
            {% for i in task %}
            <tr>
                <td><a href="/task/{{ i.id }}">{{ i.name }}</a></td>
                <td>{{ i.created_date }}</td>
                <td>{{ i.created_by }}</td>
                <td>{{ i.priority }}</td>
                <td>{{ i.status }}</td>
            </tr>
            {% endfor %}
        </table>
    </div>
    </div>
    </div>
    {% elif inbox%}
    <div style="font-size: 2em;height: 120px;padding-top: 10px;">
            <table width="80%">
            <tr><td>
                <h3>There are below Inbox matching keyword of <b>{{
keyword }}</b></h3></td>
```

```
            </tr>
            </table>
        </div>
        <table class="table" id="table">
                <thead>
                    <th><input type="checkbox" id="all"></th>
                    <th> Name </th>
                    <th> Subject </th>
                    <th> Time </th>
                    <th> Receiver </th>
                </thead>
                {% for i in inbox %}
                {% if i.is_read %}
                <tr draggable="true">
                    <td><input onclick="not()" class="check" value="5"
type="checkbox"></td>
<td onclick="window.document.location='/email/{{ i.id }}'"
style="cursor:pointer">{{ i.sender_id }}</td>
                    <td onclick="window.document.location='/email/{{
i.id }}'" style="cursor:pointer">{{ i.subject }}</td>
                    <td onclick="window.document.location='/email/{{
i.id }}'" style="cursor:pointer">{{ i.date.date }}</td>
                    <td onclick="window.document.location='/email/{{
i.id }}'" style="cursor:pointer">{{ i.receiver_id.first_name }}</td>
                </tr>
                {% else %}
                <tr id="{{ i.id }}" class='row_data' style="background:
rgba(243,243,243,.85);font-weight: bold;draggable:true">
                    <td><input onclick="not()" class="check" value="5"
type="checkbox"></td>
<td onclick="myfunction({{ i.id }})" style="cursor:pointer">{{
i.sender_id }}</td>
                    <td onclick="myfunction({{ i.id }})"
style="cursor:pointer">{{ i.subject }}</td>
                    <td onclick="myfunction({{ i.id }})"
style="cursor:pointer">{{ i.date.date }}</td>
                    <td onclick="myfunction({{ i.id }})"
style="cursor:pointer">{{ i.receiver_id.first_name }}</td>
                </tr>
                {% endif %}{% endfor %}
            </table>
        <br>
```

```
        <br>
            <h3>There are below sent item matching keyword of <b>{{
keyword }}</b></h3>
            <table class="table">
                    <thead>
                        <th><input type="checkbox" id="all"></th>
                        <th> Name </th>
                        <th> Subject </th>
                        <th> Time </th>
                        <th> Receiver </th>
                    </thead>
                    {% for i in sent %}
                    <tr>
                        <td><input class='check' type="checkbox"></td>
                        <td onclick="window.document.location='/sent/{{
i.id }}'" style="cursor:pointer">{{i.sender_id}}</td>
                        <td onclick="window.document.location='/sent/{{
i.id }}'" style="cursor:pointer">{{i.subject}}</td>
                        <td onclick="window.document.location='/sent/{{
i.id }}'" style="cursor:pointer">{{i.date.date}}</td>
                        <td onclick="window.document.location='/sent/{{
i.id }}'" style="cursor:pointer">{{i.receiver_id.first_name}}</td>
                    </tr>
                    {% endfor %}
                </table>
            {% else %}
    <h3>Sorry there is no keyword found
<strong>{{keyword}}</strong></h1>
    {% endif %}
{% endblock %}
{% block script%}
<script type="text/javascript">
$("#all").click(function() {
    if ($("#all").is(':checked')) {
        $(".check").prop("checked", true);
    } else {
        $(".check").prop("checked", false);
    }
});
</script>
{%endblock%}
```

Style.css

```
     @media (max-width: 767px) {
          .affix-top {
              width:auto;
          }
     }
     @media (max-width: 770px) {
     .affix-top {
          position:static;
         width: auto;
    }
          body{
              font-size: 12px;
          }
 }
.marker{
    background-color: yellow;
    font-size: 2em;
}
 .navbar-inverse .navbar-collapse, .navbar-inverse .navbar-form {
     border-color: transparent;
}
.navbar-inverse .navbar-toggle {
    border-color: transparent;
}
blockquote
{
    font-style: italic;
    font-family: Georgia, Times, "Times New Roman", serif;
    padding: 2px 0;
    border-left: solid;
    font-size: 15px;
    border-left-width: 10px;
    border-color: #ccc;
    border-width: 3px;
    padding-left: 8px;
    padding-right: 20px;
    border-right-width: 5px;
}
```

Urls.py

```
from django.conf.urls import include, url
from django.contrib.auth.views import logout,login
from django.views.generic import TemplateView
from django.contrib import admin
from rest_framework.authtoken import views as authtoken_views
from Workflow.views import *
from rest_framework import routers

router=routers.DefaultRouter()
router.register('task',TaskSet)
router.register('project',ProjectSet)
router.register('user',UserSet)
router.register('department',DepartmentSet)
router.register('inbox,InboxSet)
router.register('sent',SentItemset)
router.register('group',GroupSet)
router.register('contenttype',ContentTypeSet)
router.register('permission',PermissionSet)

urlpatterns = [
    url(r'^api/',include(router.urls)),
    url(r'^api-auth/', include('rest_framework.urls',
namespace='rest_framework')),
    url(r'^accounts/logout/$',Logout,name='logout'),
    url(r'^admin/', include(admin.site.urls)),
    url(r'^$',Home_rest,name='home'),
    url(r'^rest/',include('Workflow.url',namespace='rest-workflow')),
    url(r'^project/(?P<pk>\d+)/$',ProjectTask,name='ProjectTask'),
    url(r'^create/project$',CreateProject,name='crateproject'),
    url(r'^task/(?P<pk>\d+)$',TaskDetail,name='TaskDetail'),
    url(r'^task/(?P<pk>.*)/edit/$',Edit,name='Edit'),
    url(r'^email/$',Email,name='Email'),
    url(r'^email/(?P<pk>\d+)$', EmailDetail, name='ProjectTask'),
    url(r'^compose/',Compose,name='compose'),
    url(r'^reports/',Reports,name='reports'),
    url(r'^api/project_data$',project_data,name='project-data'),
    url(r'^api/project_data/(?P<what>.*)/$',prj_data,name='prj-data'),
    url(r'^api/single_project/(?P<project>.*)/$',Single,name='single-
project'),
```

```
    url(r'^api/task_by_emp/(?P<name>.*)/$',task_by_emp,name='emp-
task'),
url(r'^api/project_by_department/(?P<name>.*)/$',prj_by_dpt,name='prj-
by-dpt'),
    url(r'^api/employee/$',emp_data,name='emp-data'),
    url(r'^sending/',sending,name='sending'),
    url(r'^sent/$',SentView,name='sent-view'),
    url(r'^sent/(?P<pk>\d+)/$',SentItem,name='sent-item'),
    url(r'^project/create/$',CreateProject,name='create-project'),
    url(r'^isread/(?P<pk>\d+)$',Isread,name='isread'),
    url(r'^ckeditor/', include('ckeditor_uploader.urls')),
    url(r'^accounts/login/$',login,name='login'),
    url(r'^accounts/logout/$',logout,name='logout'),
    url(r'^auth/$',auth_view,name='auth'),
    url(r'^accounts/profile/',profile,name='profile'),
    url(r'^get_auth_token/$', authtoken_views.obtain_auth_token,
name='get_auth_token'),
    url(r'^search/(?P<srchkey>.*)/$',Searching,name='search'),
    url(r'^export/$',export_csv,name='csv'),

]
```

Wsgi.html

```
import os

from django.core.wsgi import get_wsgi_application

os.environ.setdefault("DJANGO_SETTINGS_MODULE", "TeamWorks.settings")

application = get_wsgi_application()
```

Manage.py

```
#!/usr/bin/env python
import os
import sys

if __name__ == "__main__":
    os.environ.setdefault("DJANGO_SETTINGS_MODULE",
"TeamWorks.settings")

    from django.core.management import execute_from_command_line

    execute_from_command_line(sys.argv)
```

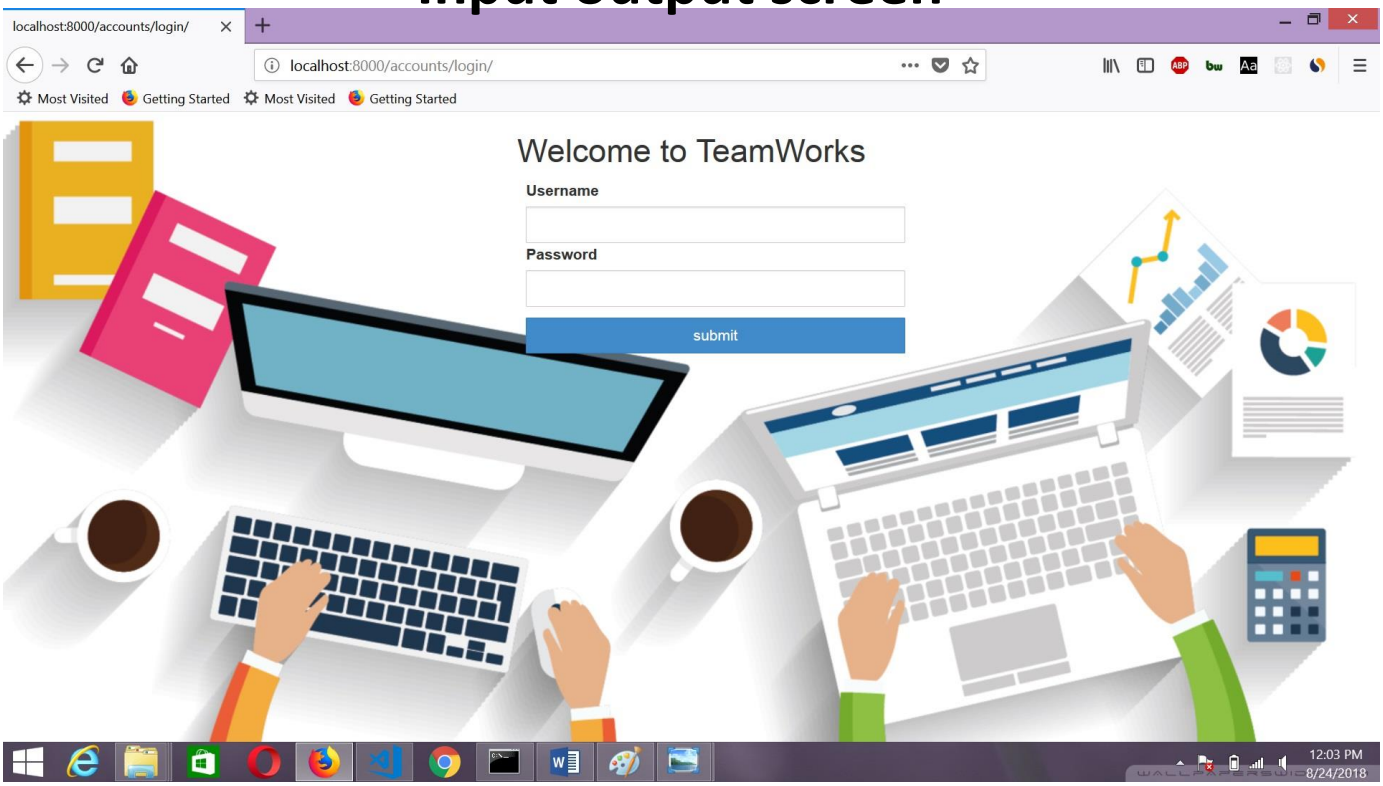
Code Efficiency

In an effort to maintain computational efficiency and to allow the eventual adaptation of the algorithm to face tracking applications, intense optimization of the code has been performed. Although further development is in progress, the algorithm is currently fast and compact enough to run interactively on most generic platforms.

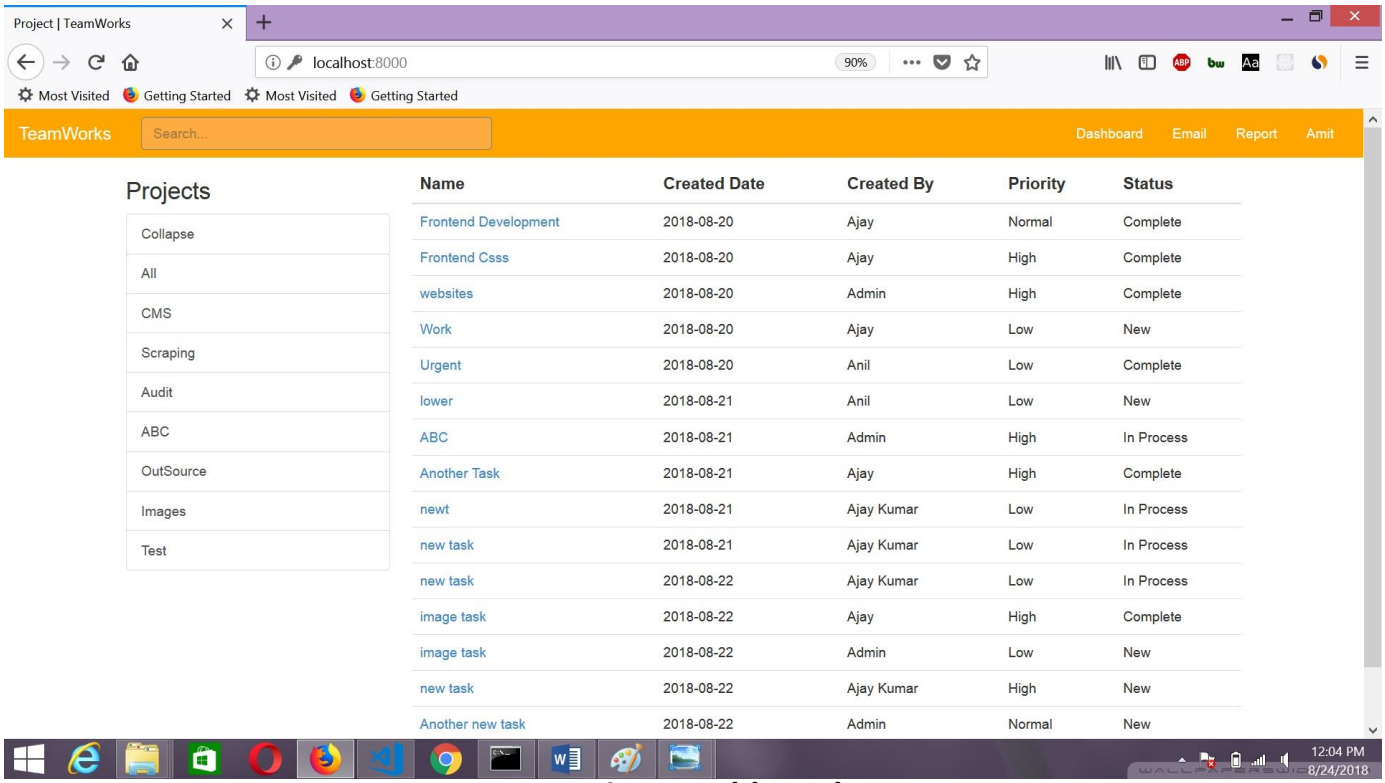
Note: - First the sequential hierarchical search which proceeds from large scales to small scales. This allows a rapid convergence if the face is dominant in the image. Furthermore, the algorithm does not always flow through the complete loop. It stops as soon as one of the modules reports a failure and loops back to an earlier stage. For example, we do not search for a mouth if no eyes are found. In this case, no time is wasted in the mouth module.

Additionally, in this Project I have utilized special programming techniques to reduce the run-time. The 3D-normalization algorithm is also extremely efficient and uses look-up tables and minimal calculations for increased speed. The 10 normalization's and DFFS calculations required for nose-localization also utilize small mug-shot images and increase efficiency.

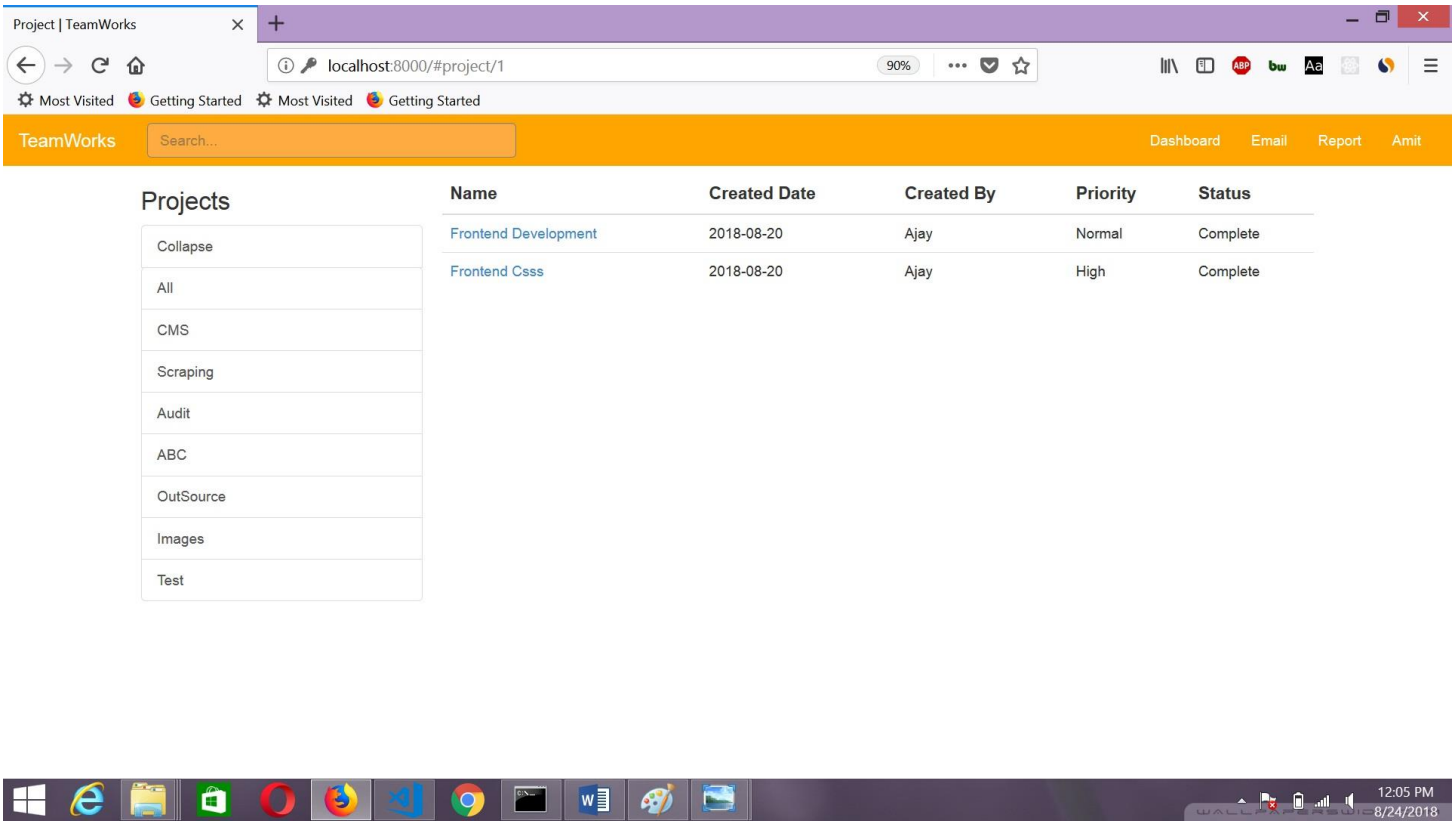
Input output screen



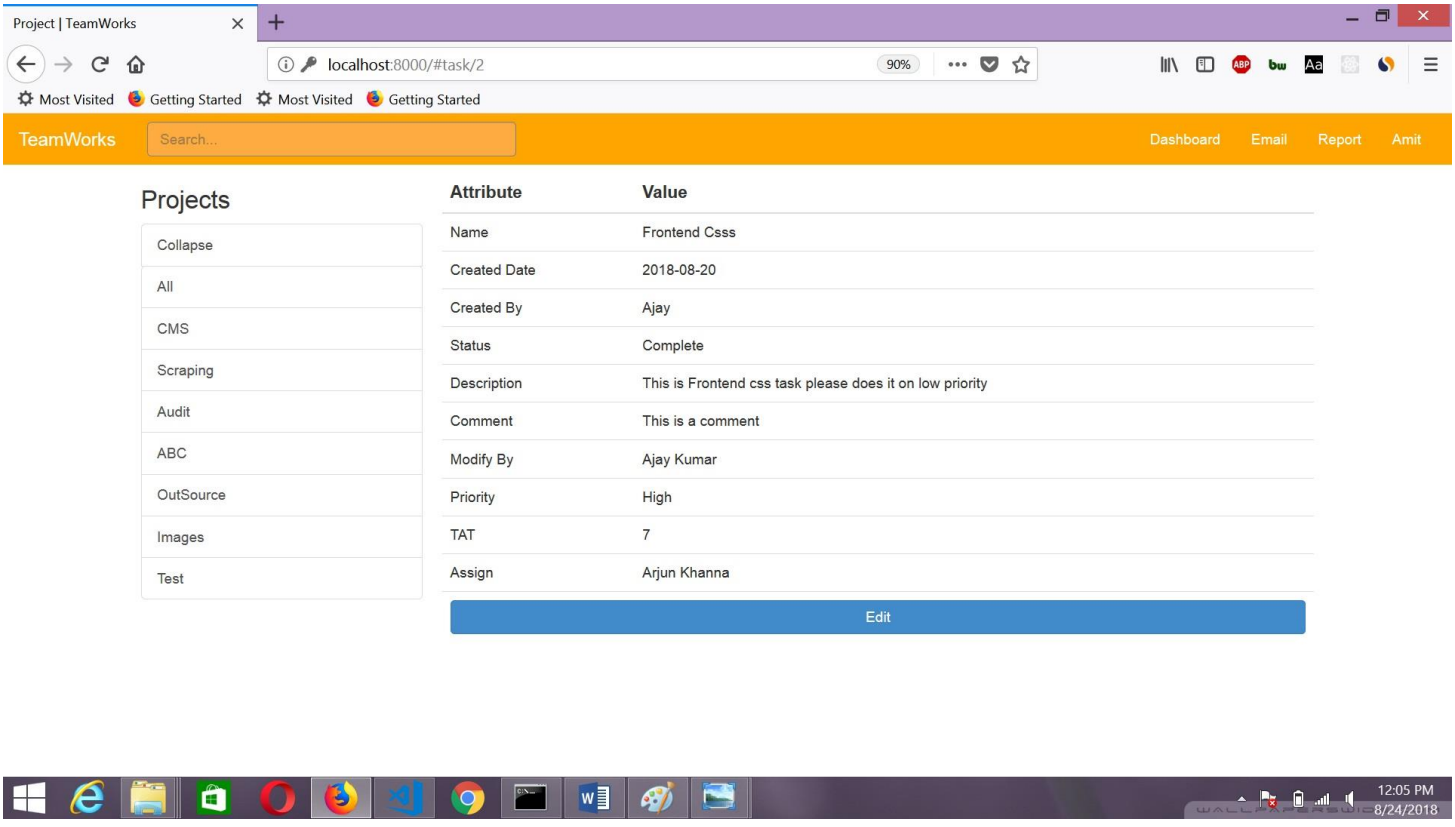
User Login



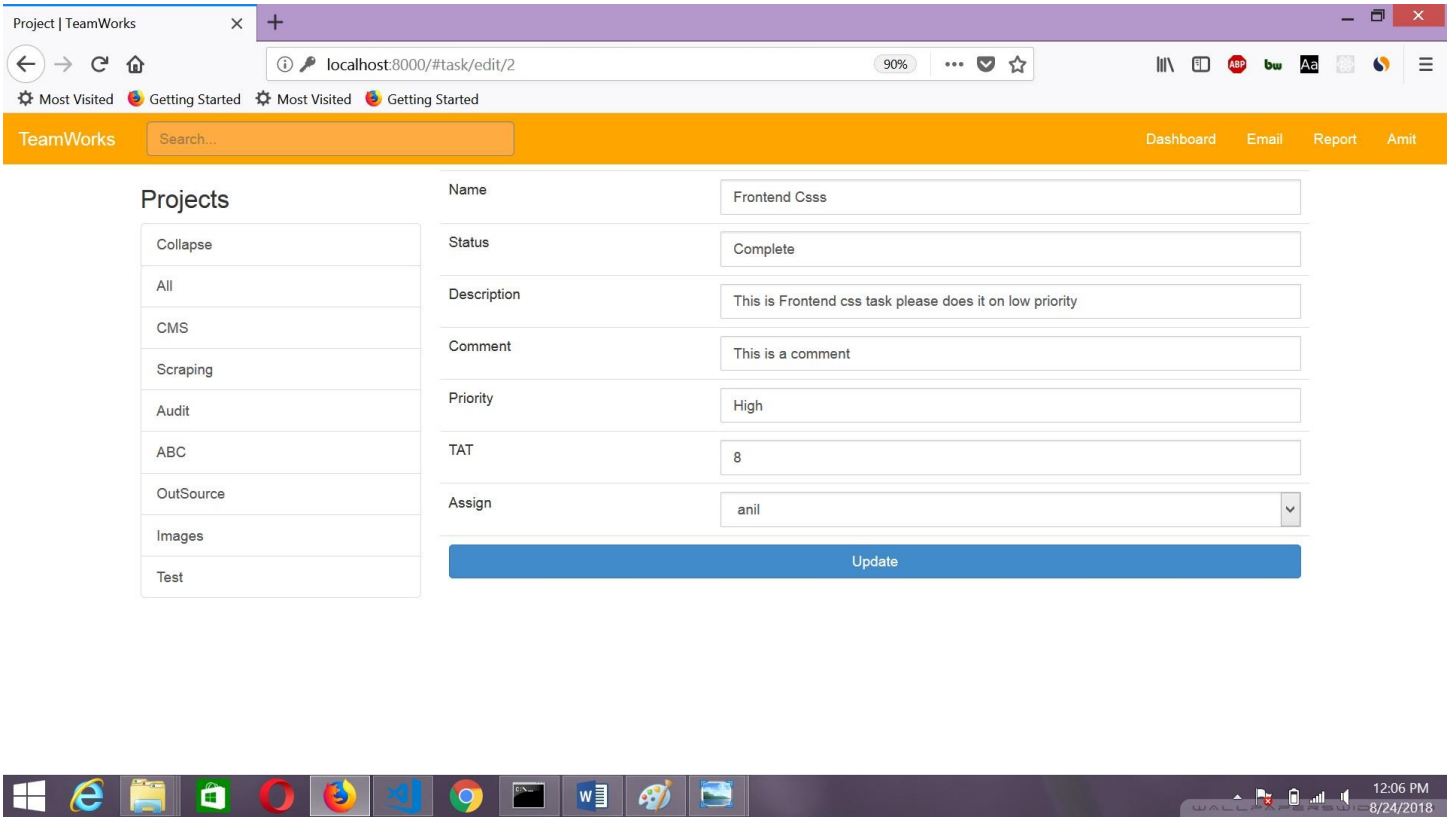
Associate Dashboard



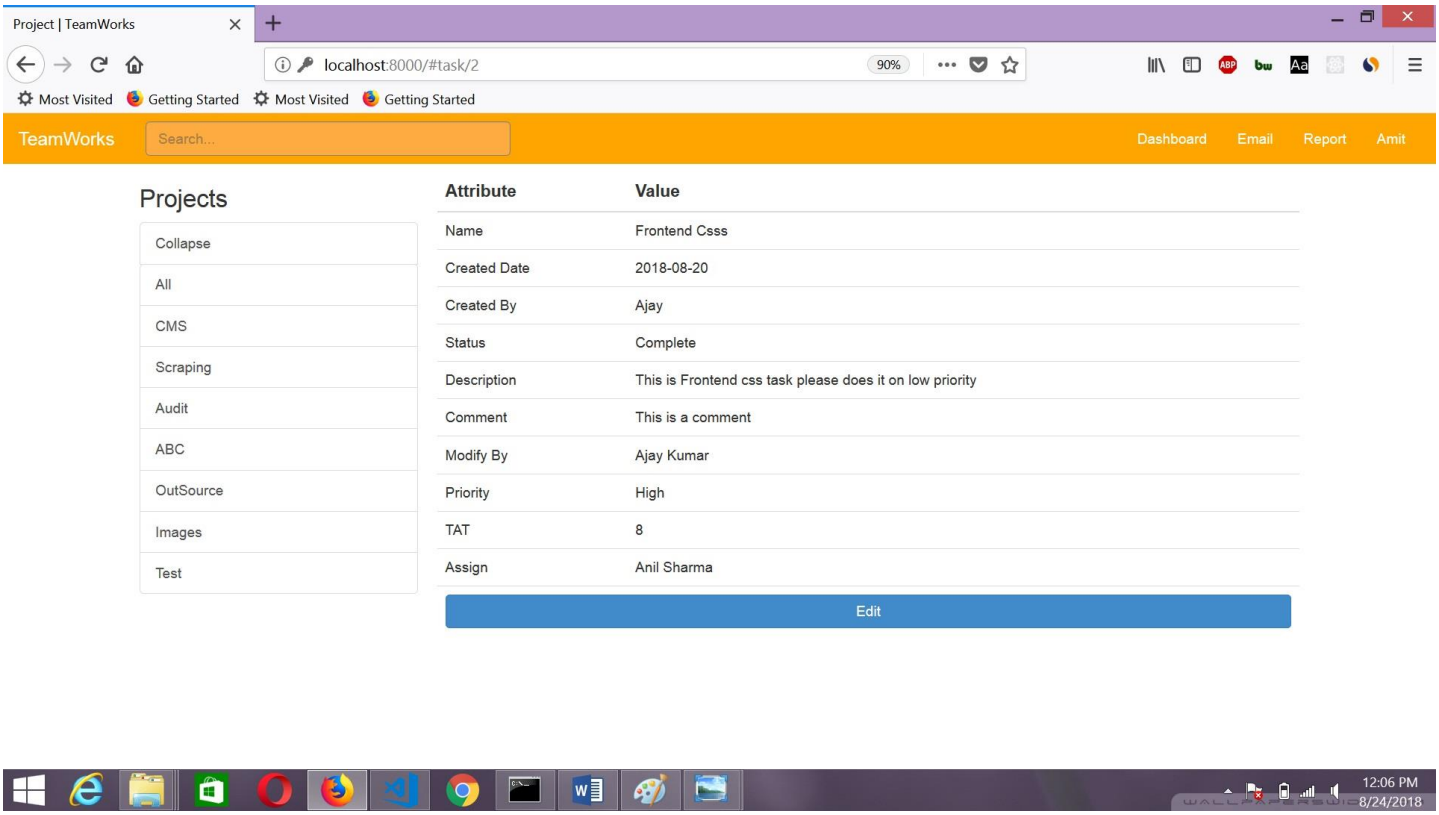
Single Page Application view(url changing after "#" on same page)



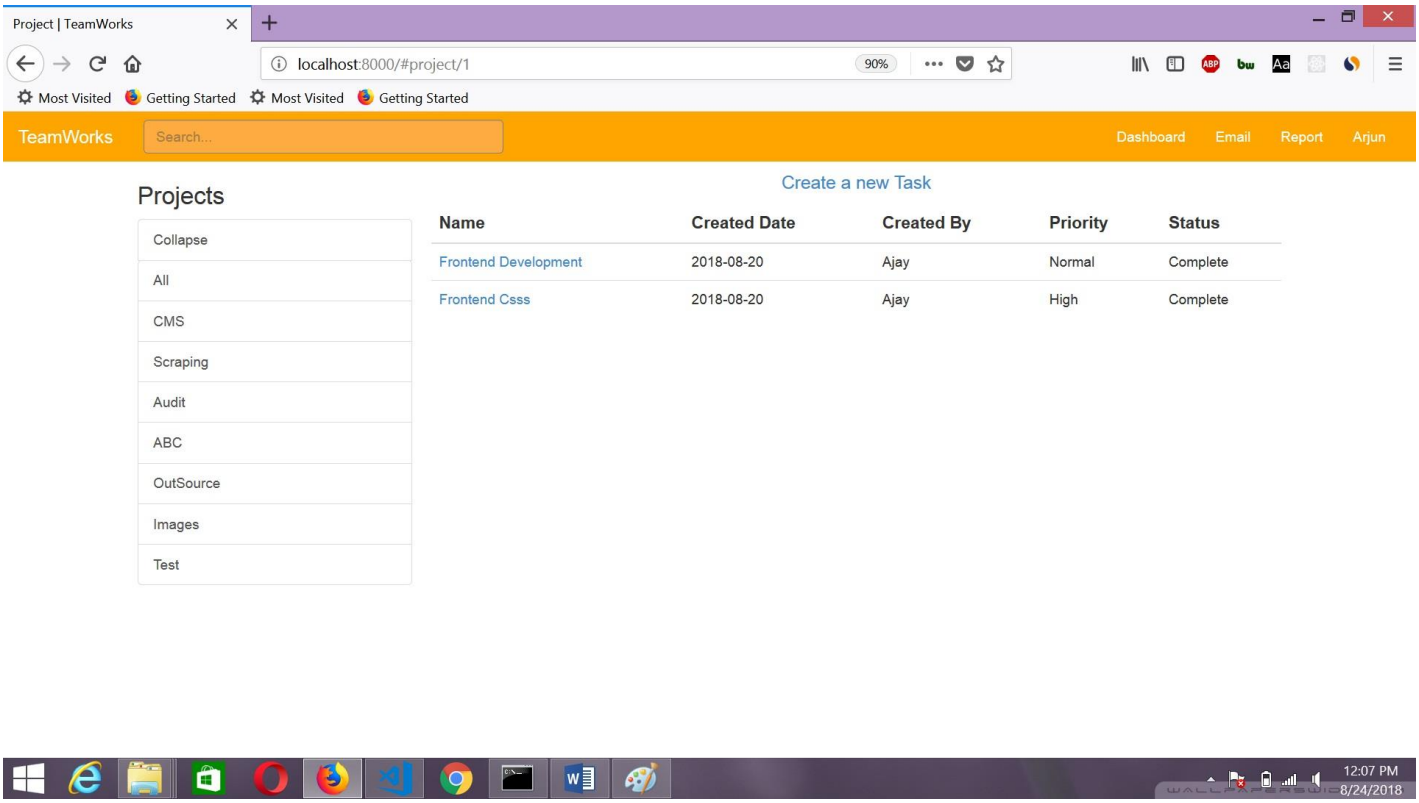
Edit Task



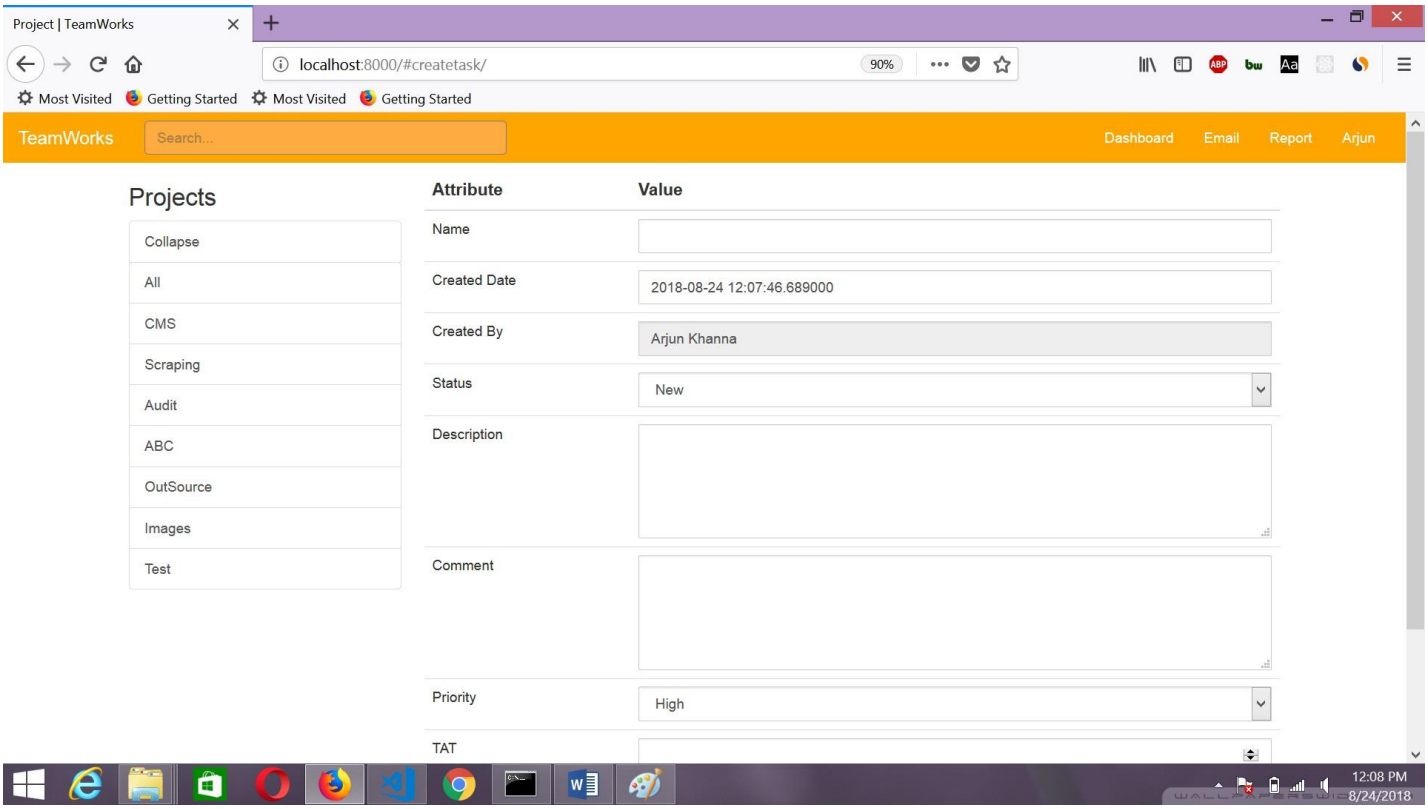
Update Task form



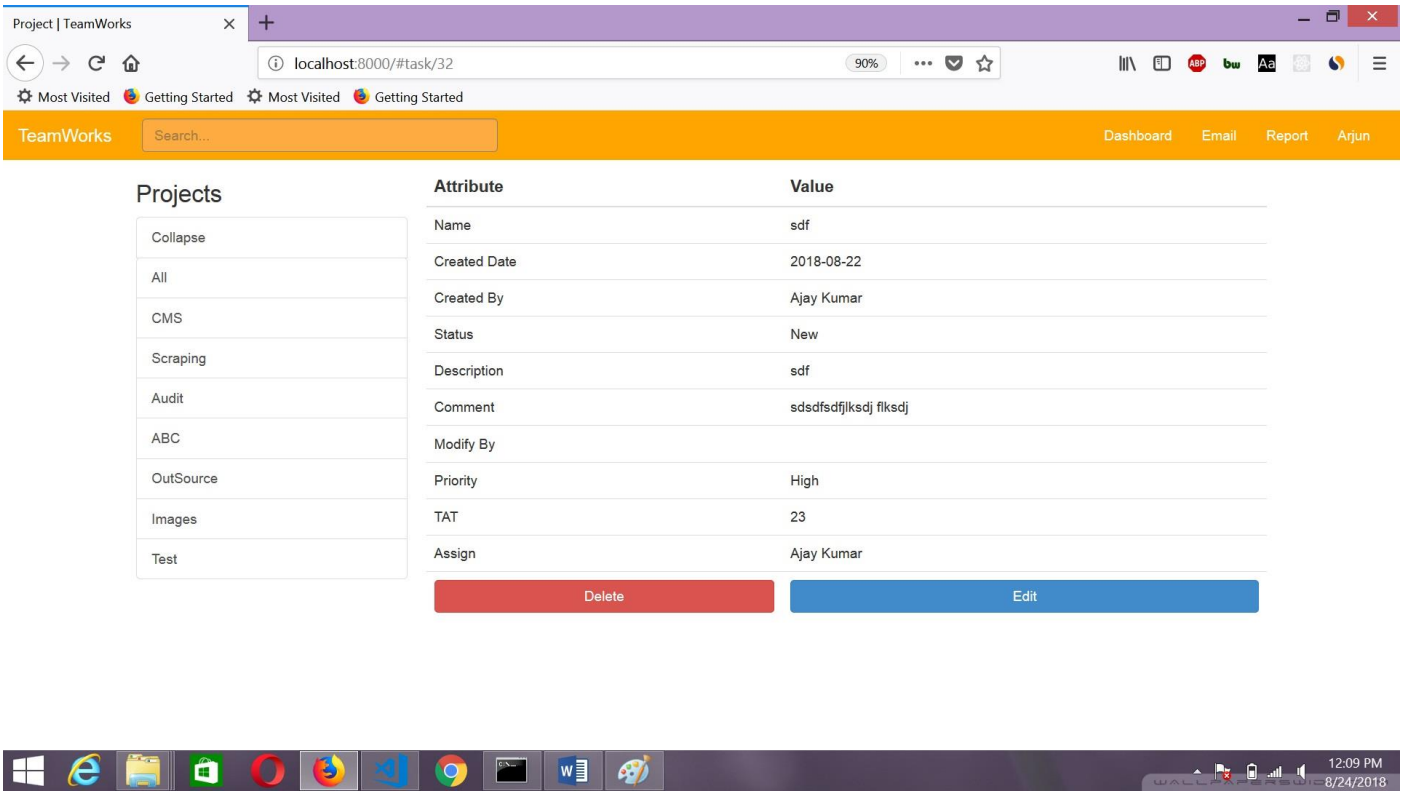
Update Task



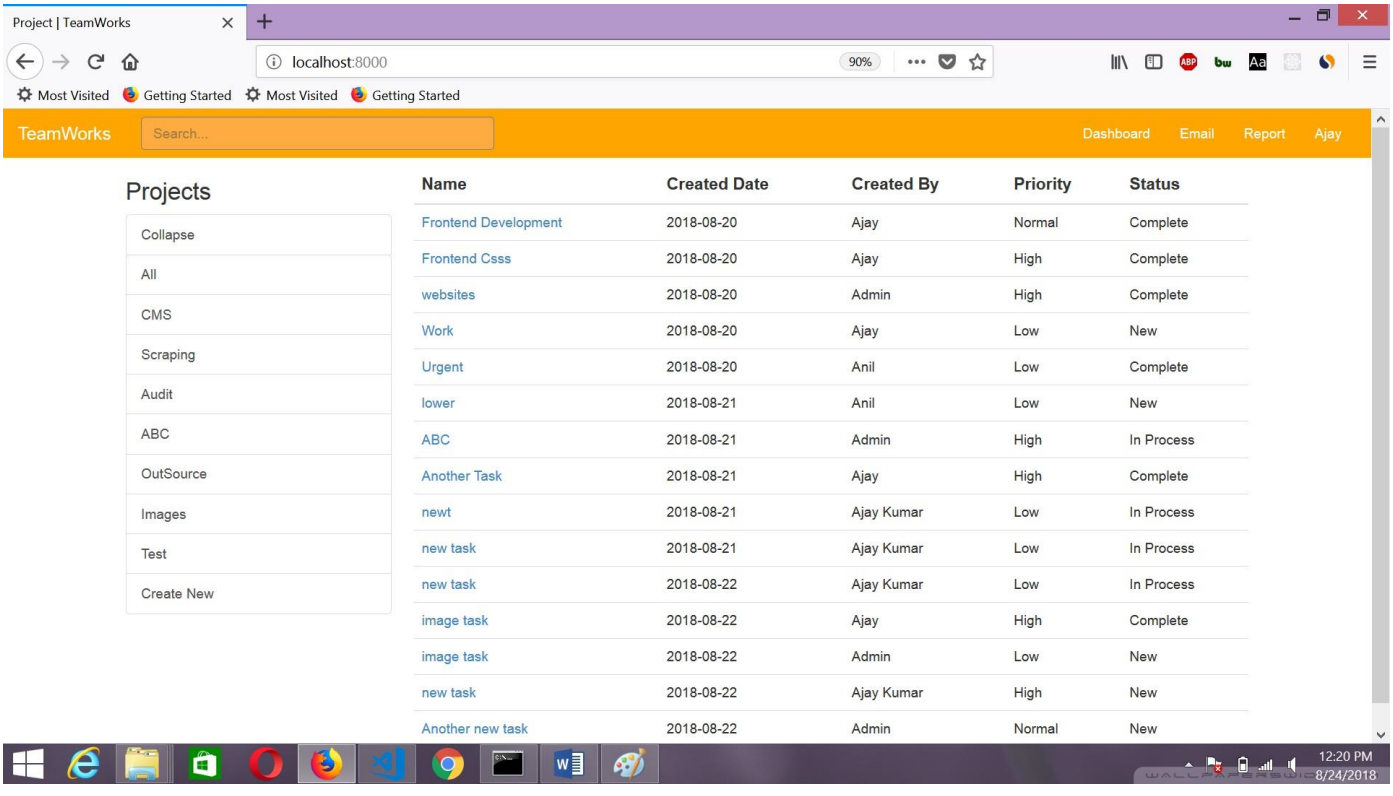
Manager dashboard view(with create task link)



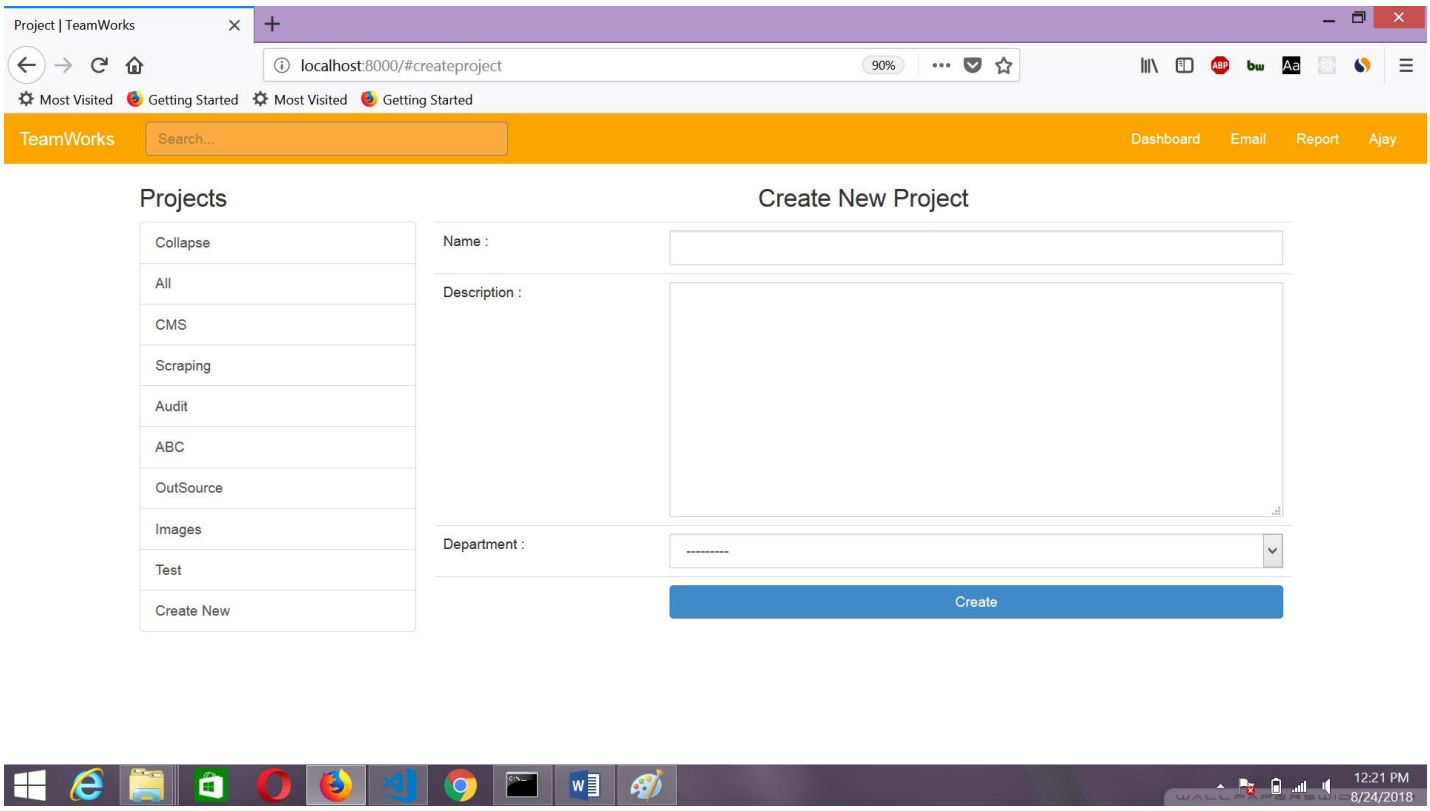
Create new task form (by manager)



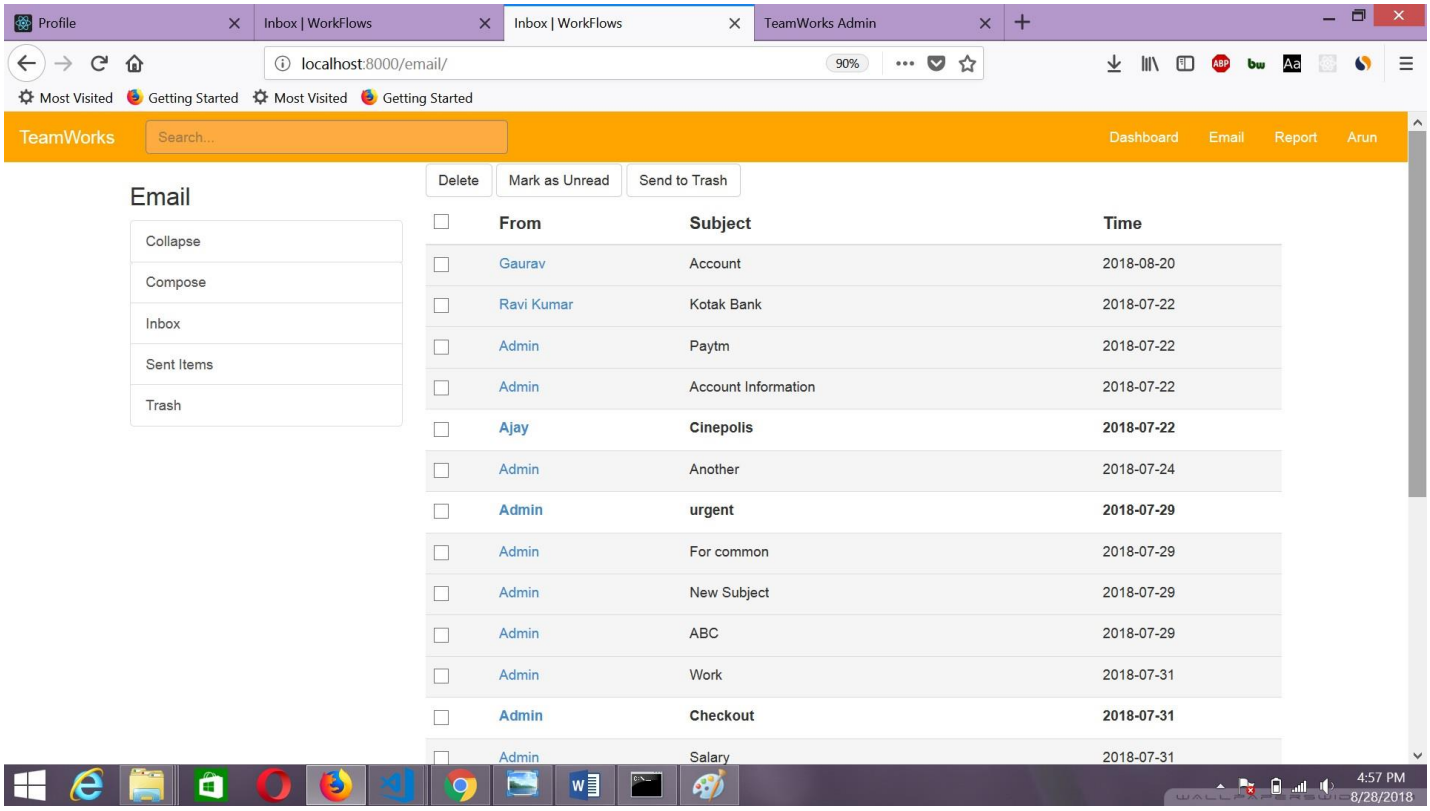
Manager view of Task(with delete button which is not in associate view)



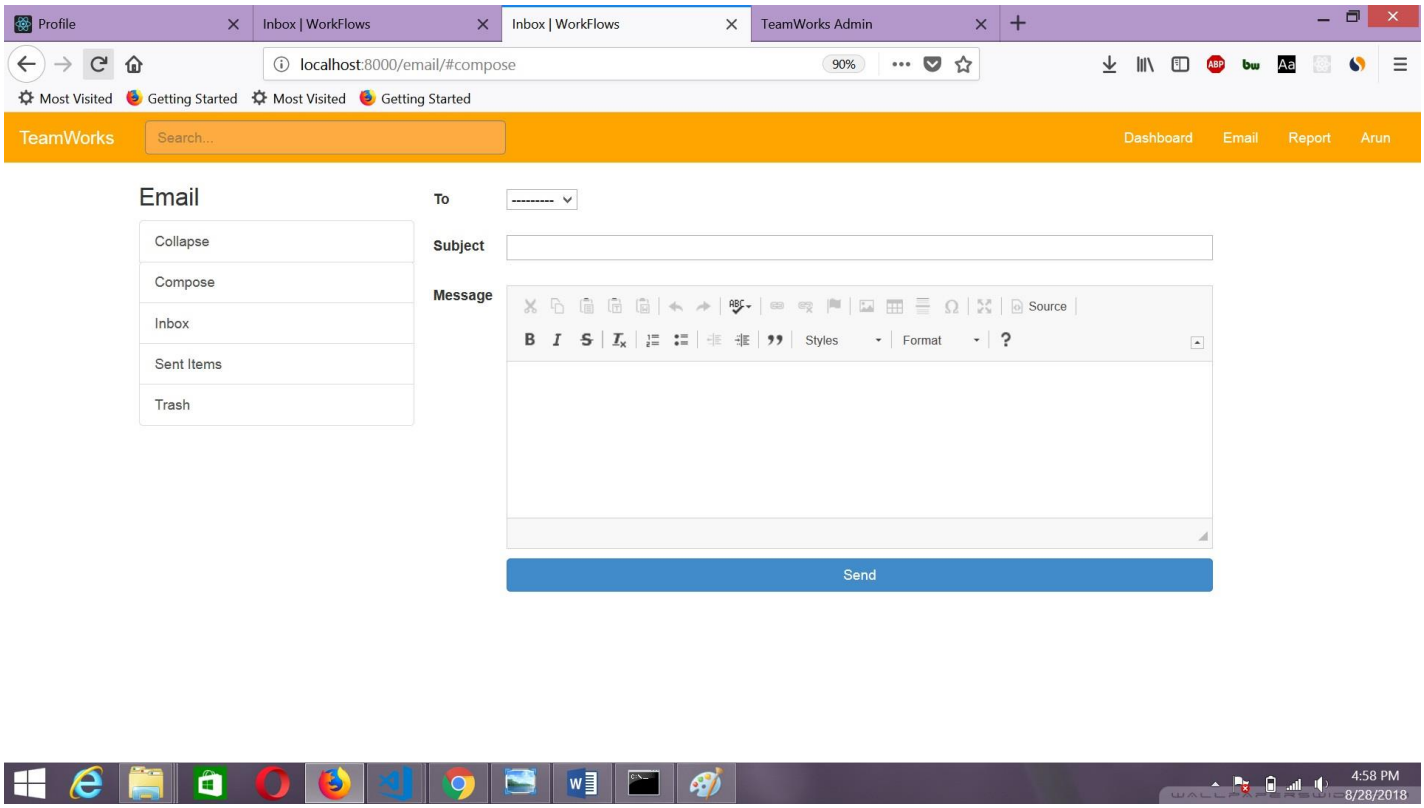
Department Head Dashboard(with “Create New” option below projects)



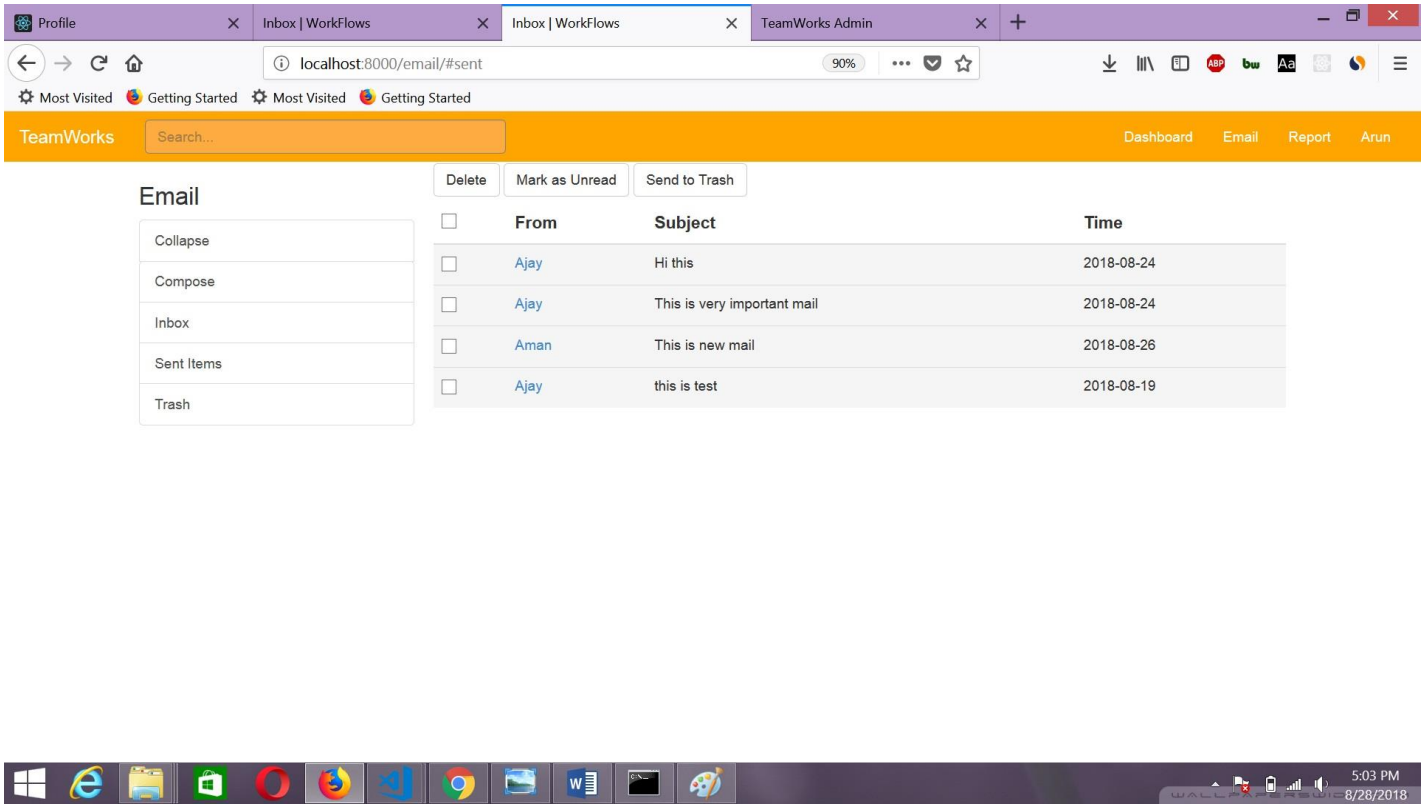
Create New Project(by Department Head)



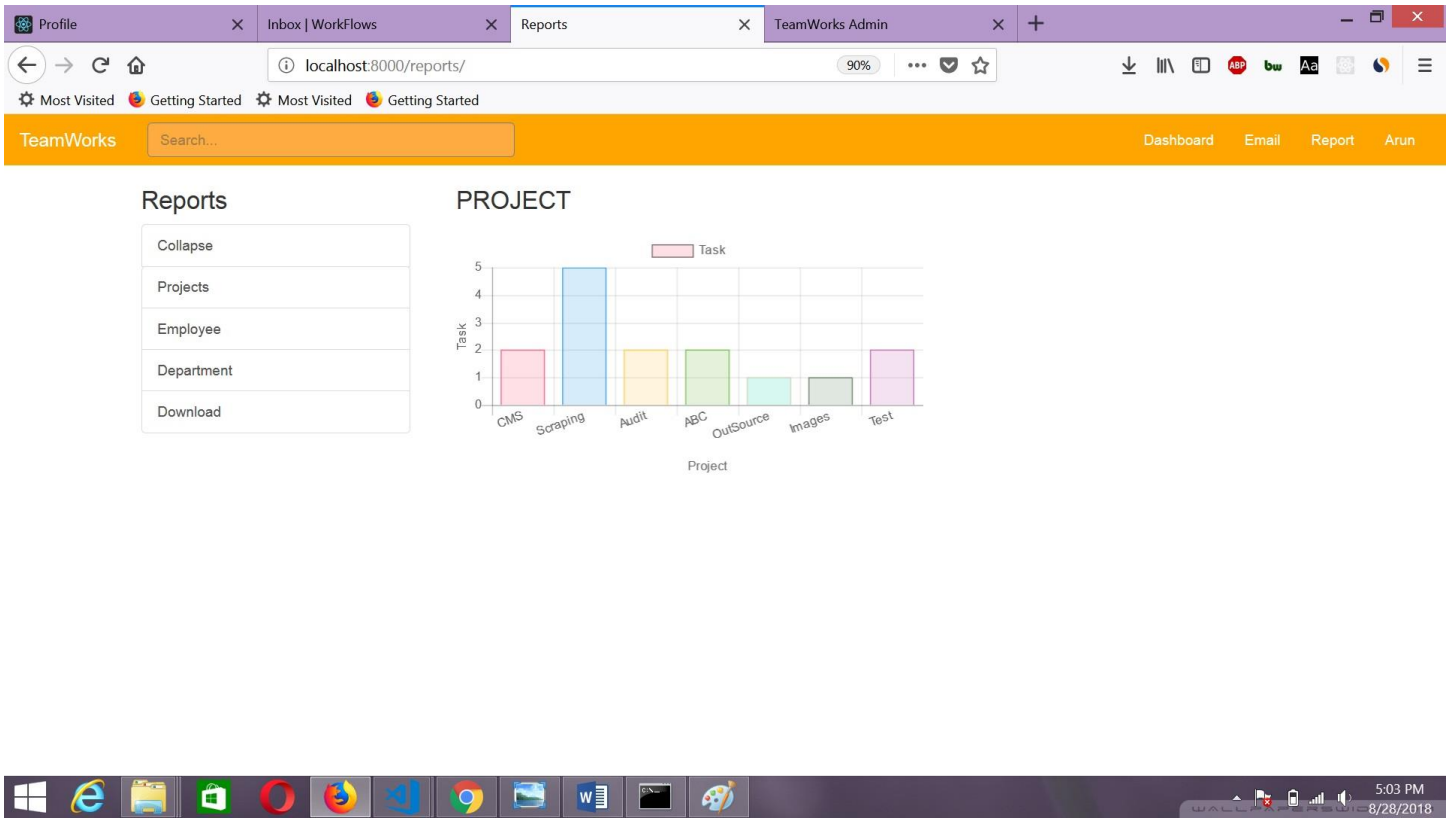
Email Dashboard(inbox default)



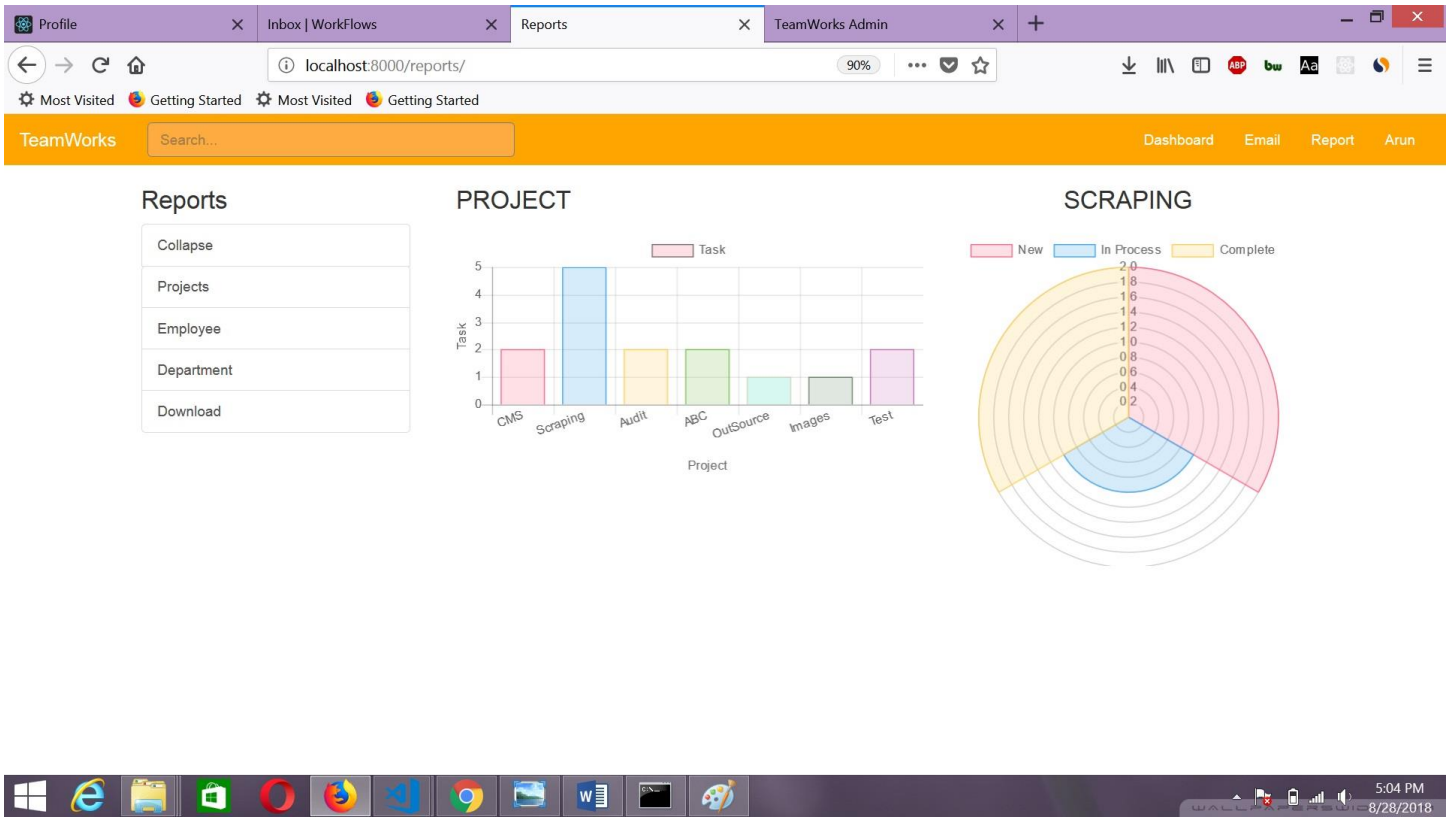
Compose mail



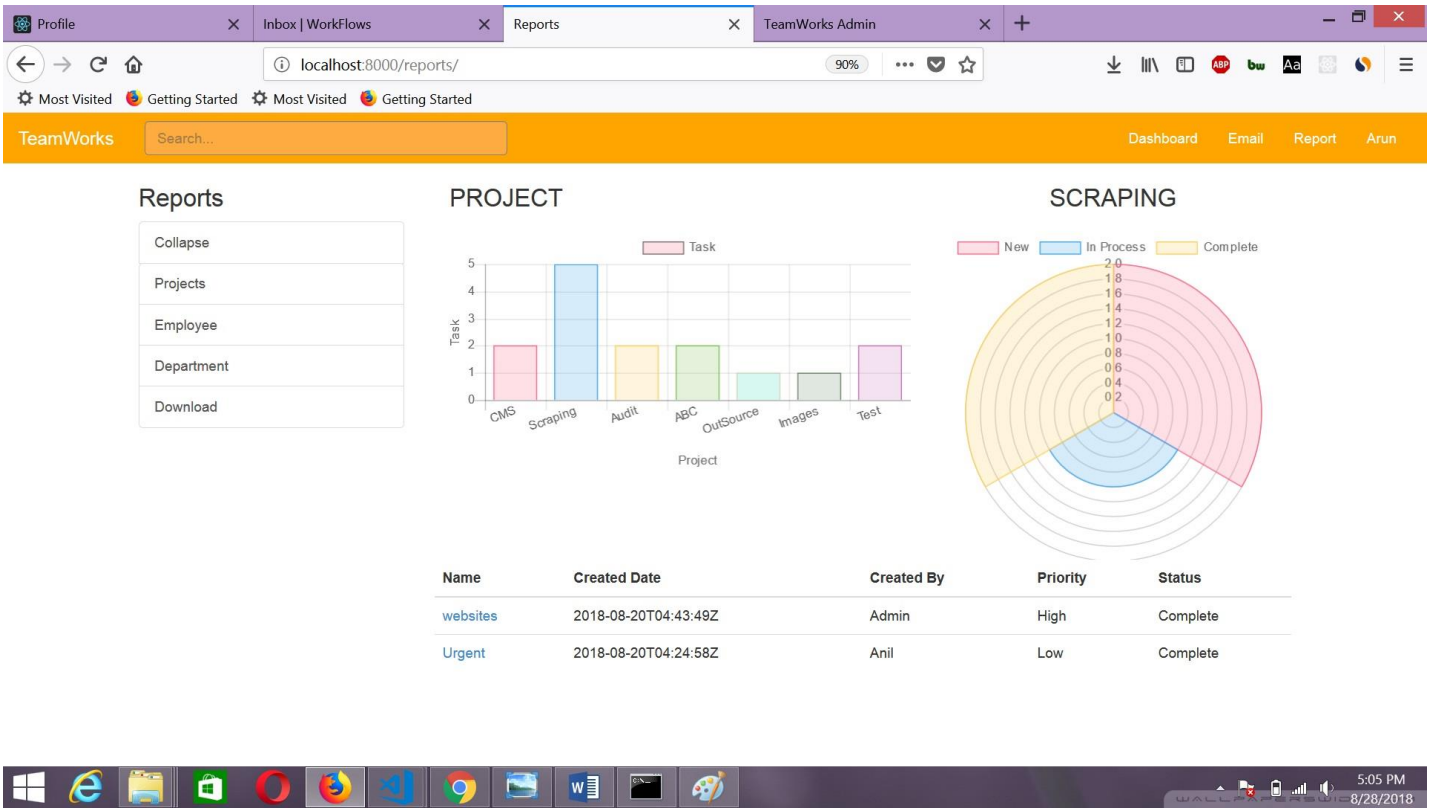
Sent mail



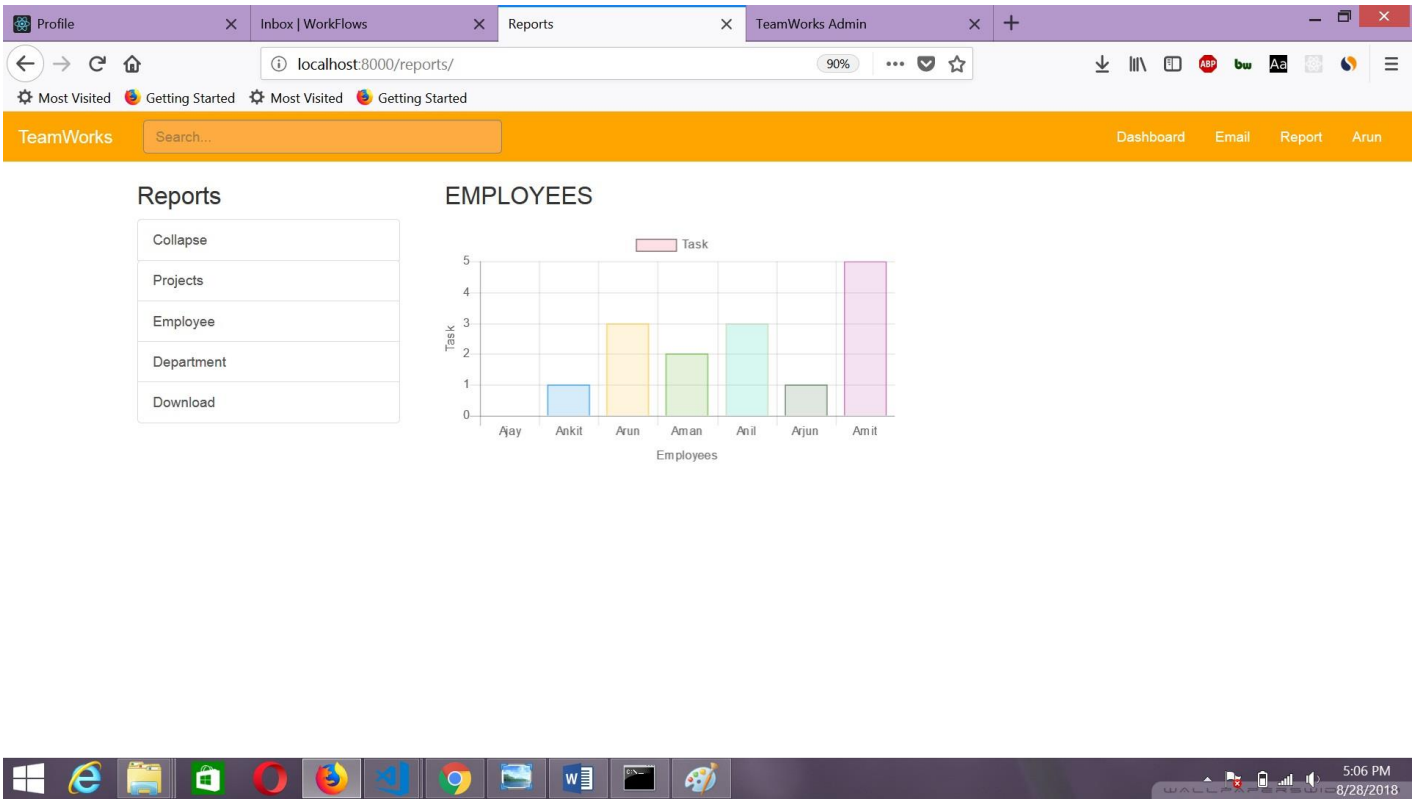
Report Dashboard(Project report default)



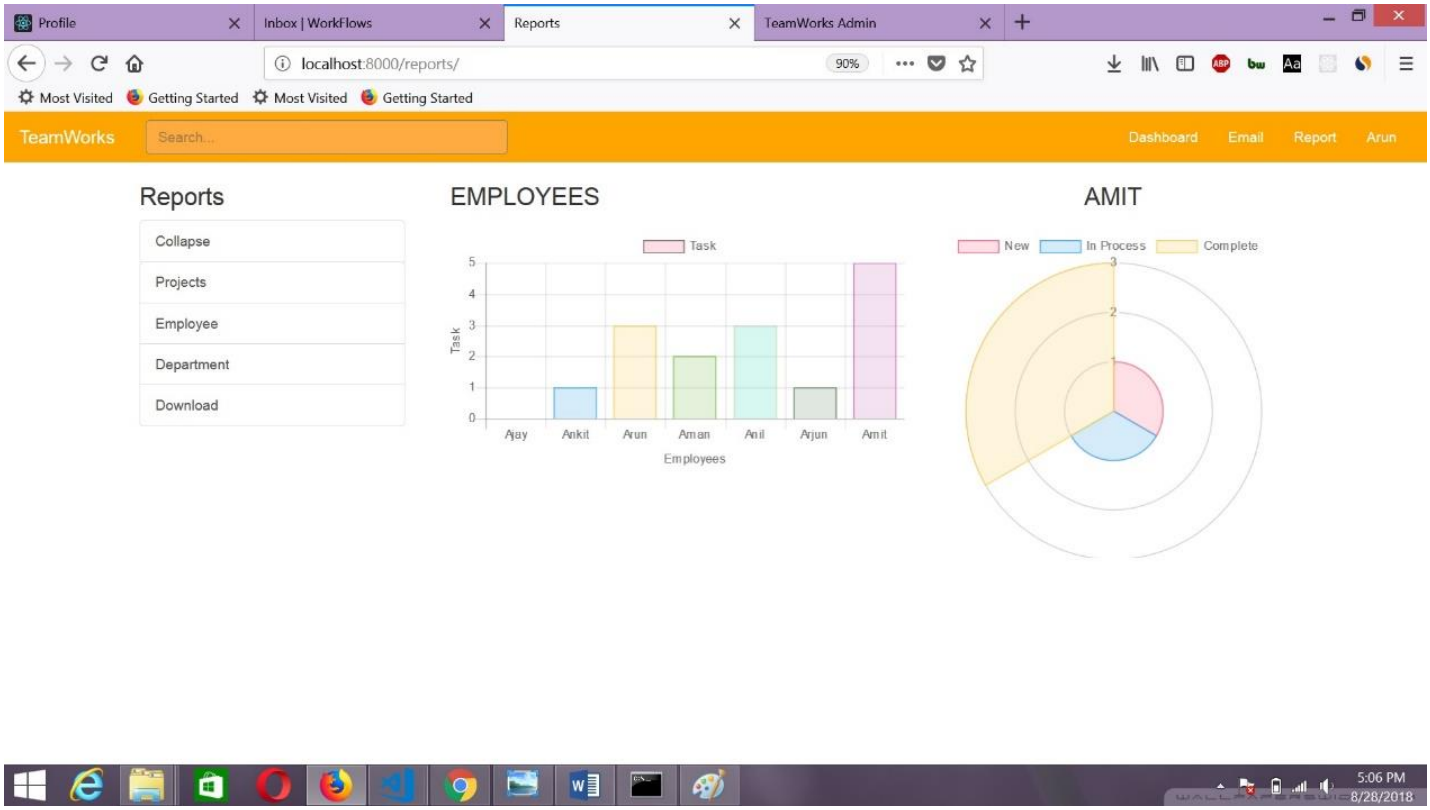
Status of particular project



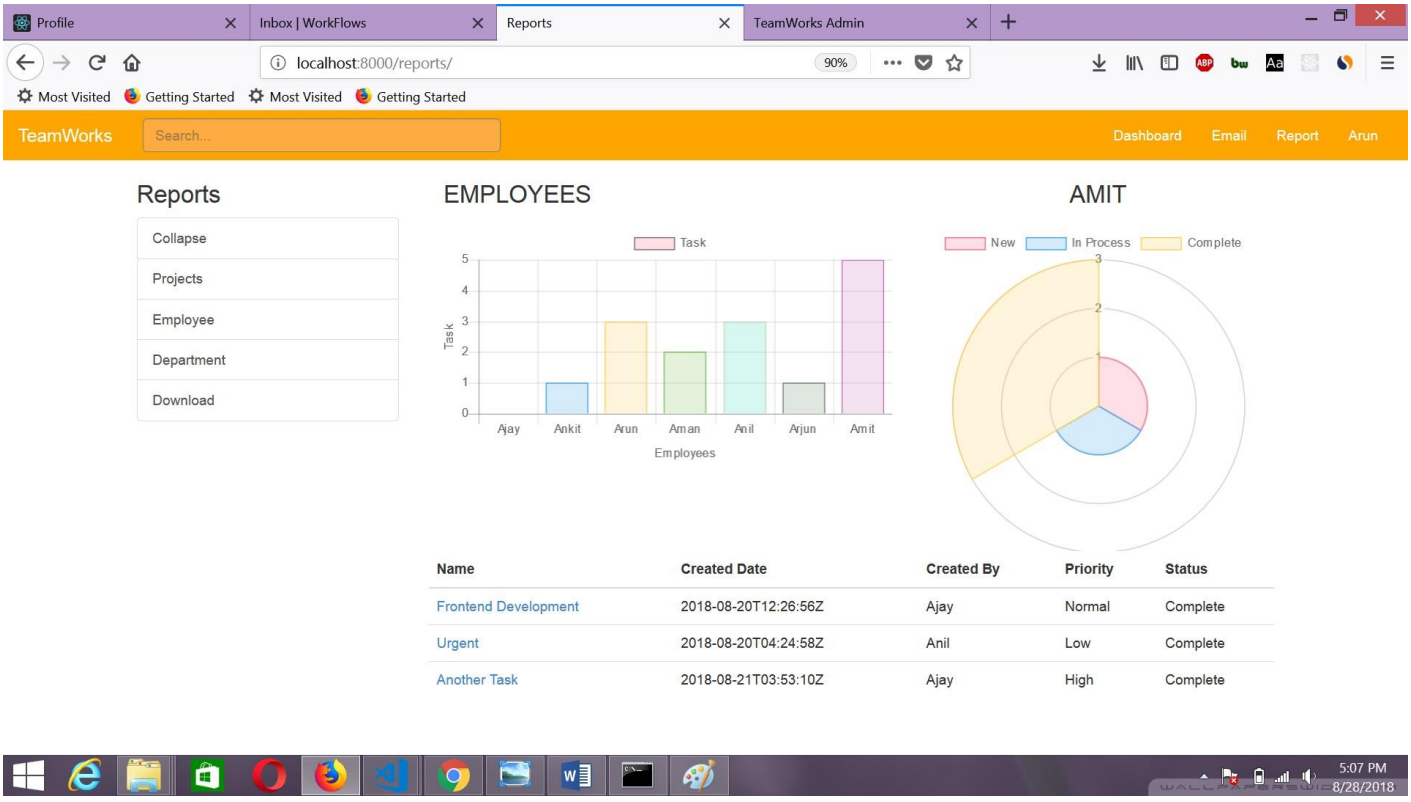
Task by their status(by click on graph area)



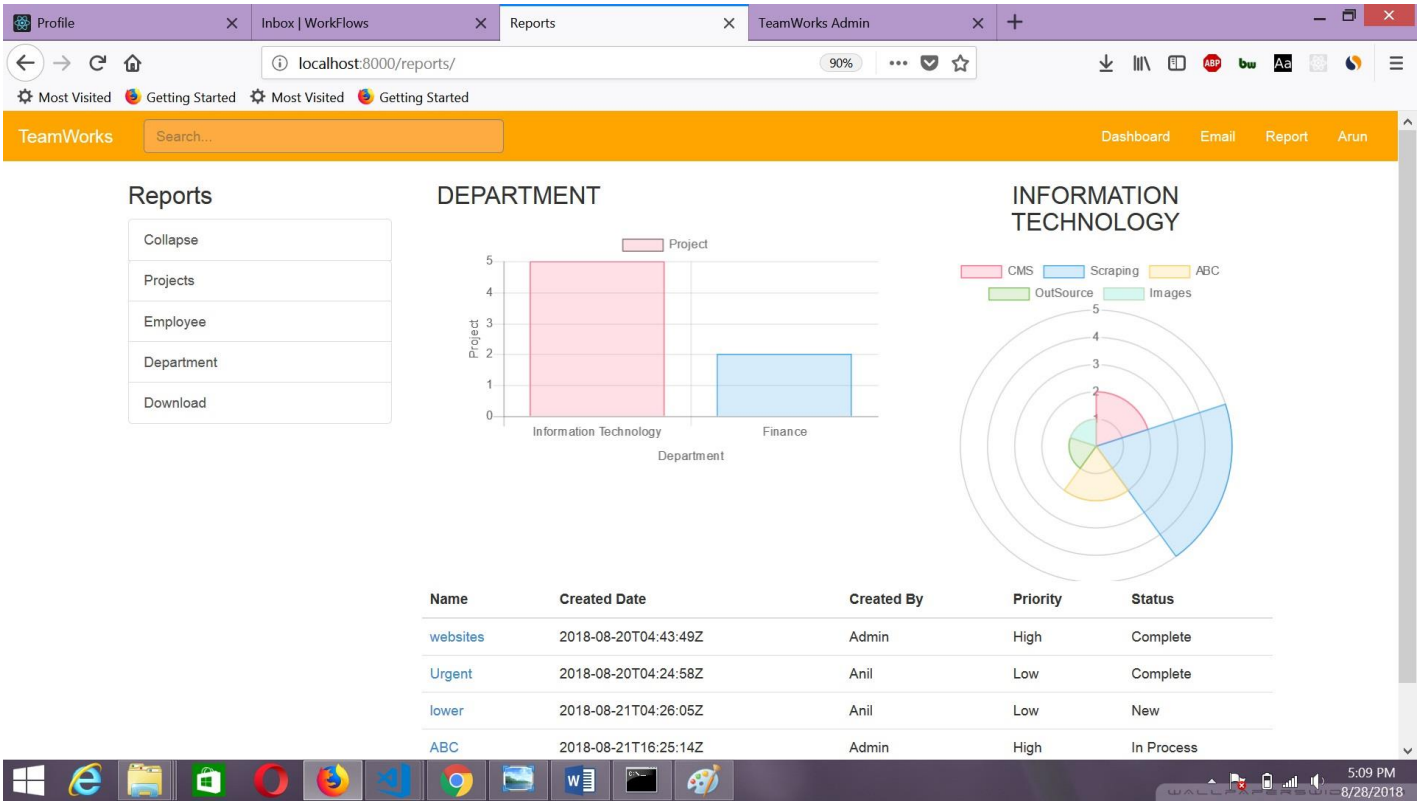
Employees report



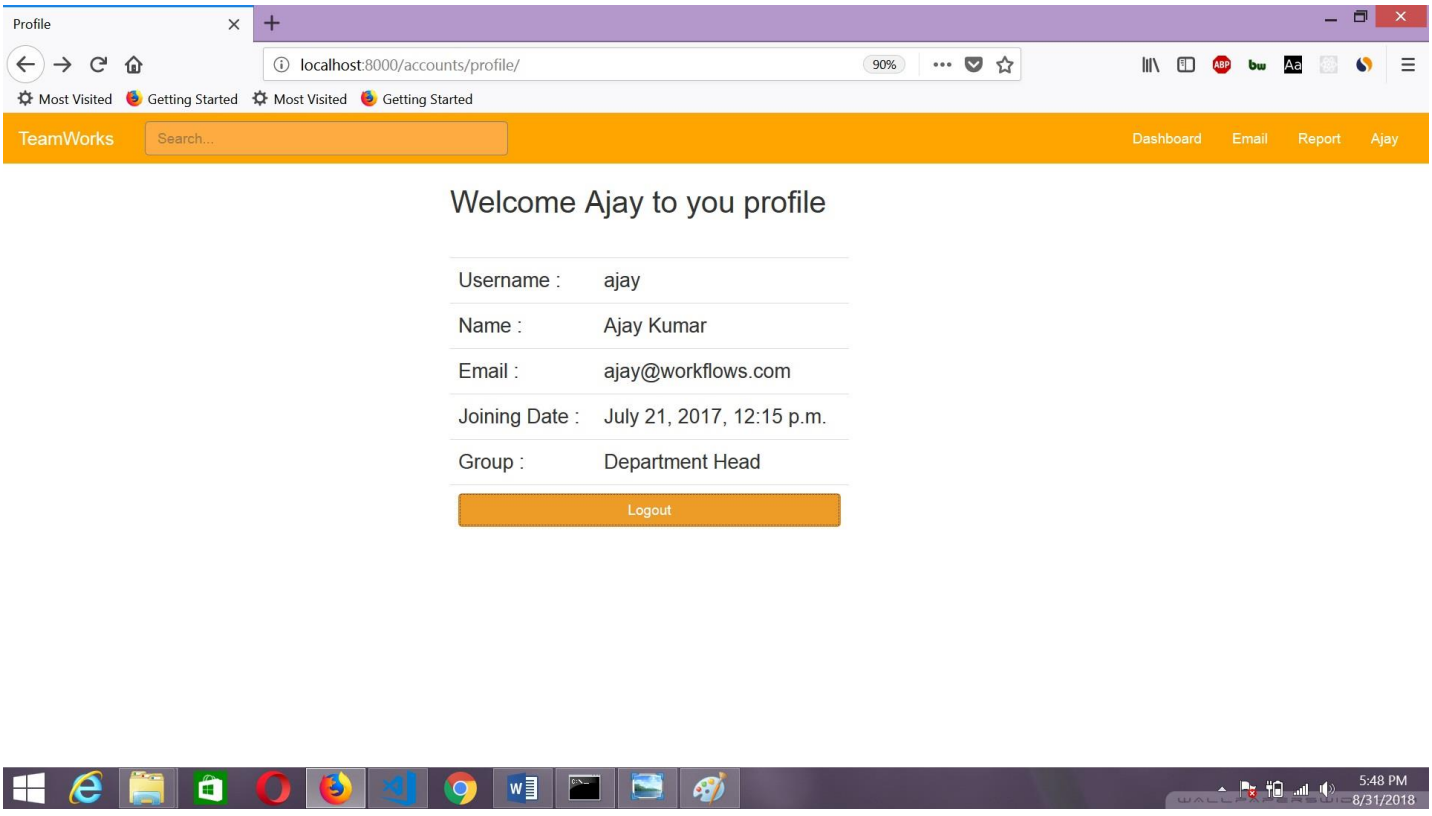
Check task of partiuclar employee(click on bar chart)



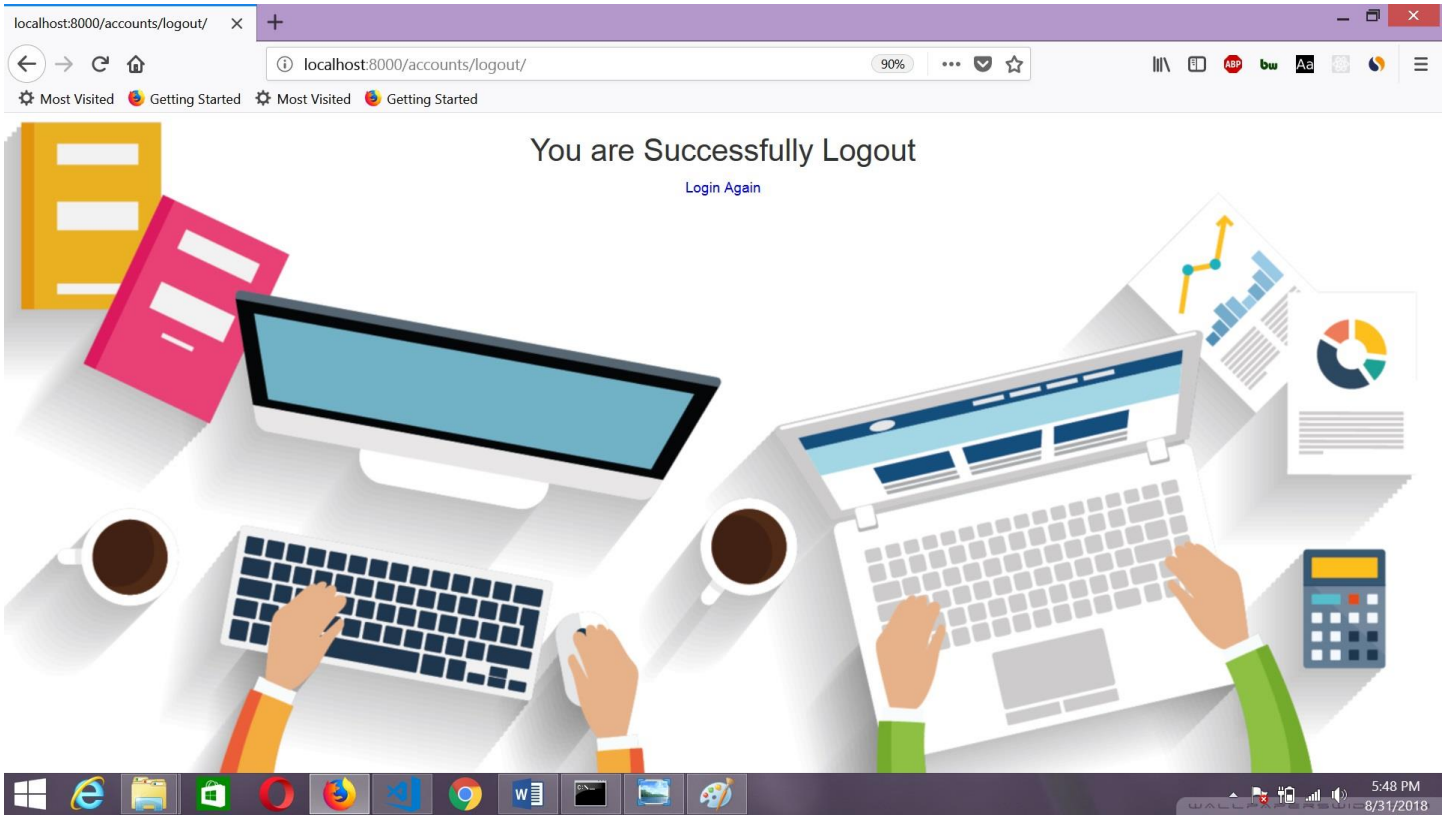
Task list by their status(by clicking on areabar chart)



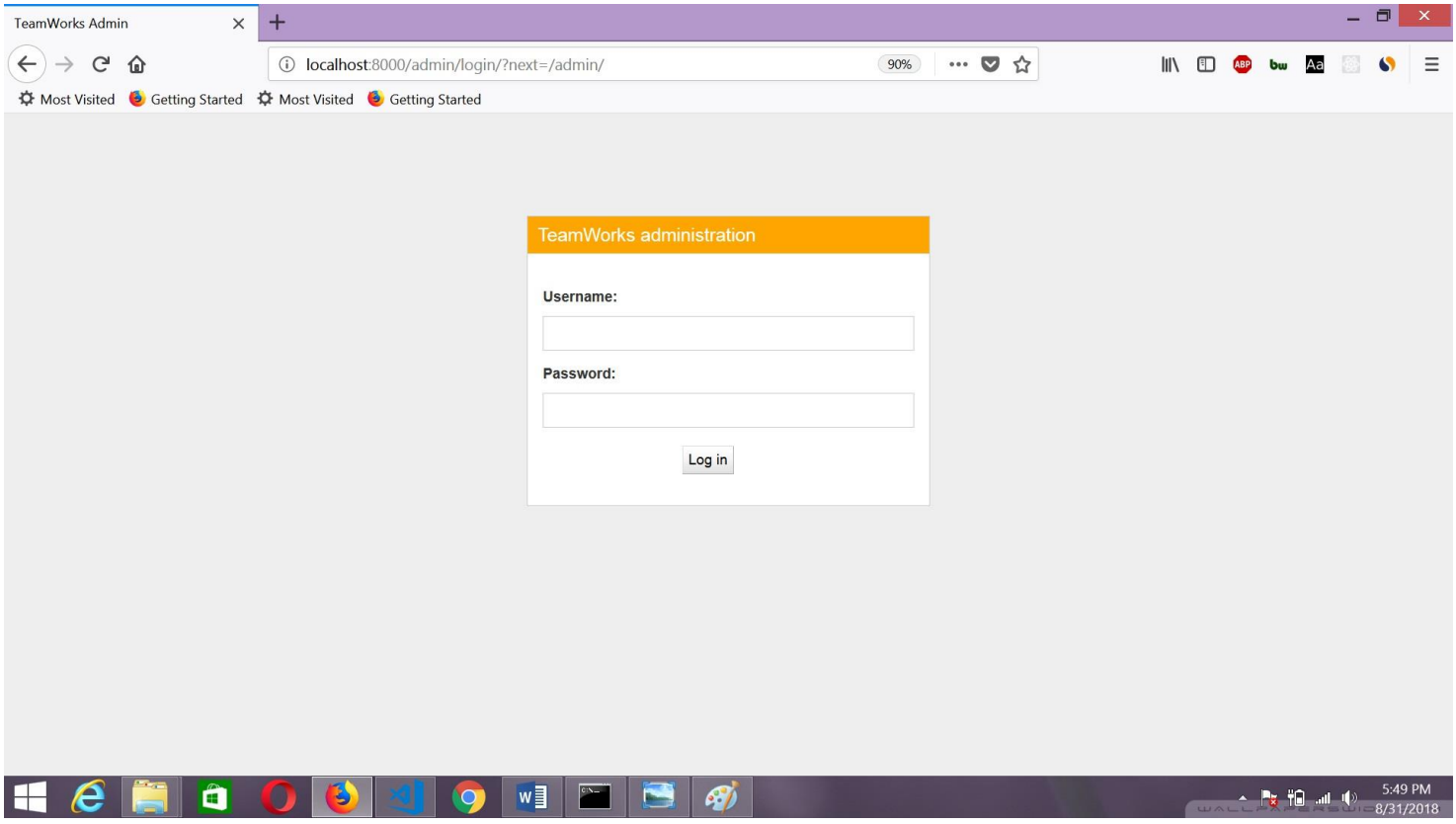
Department report as above steps



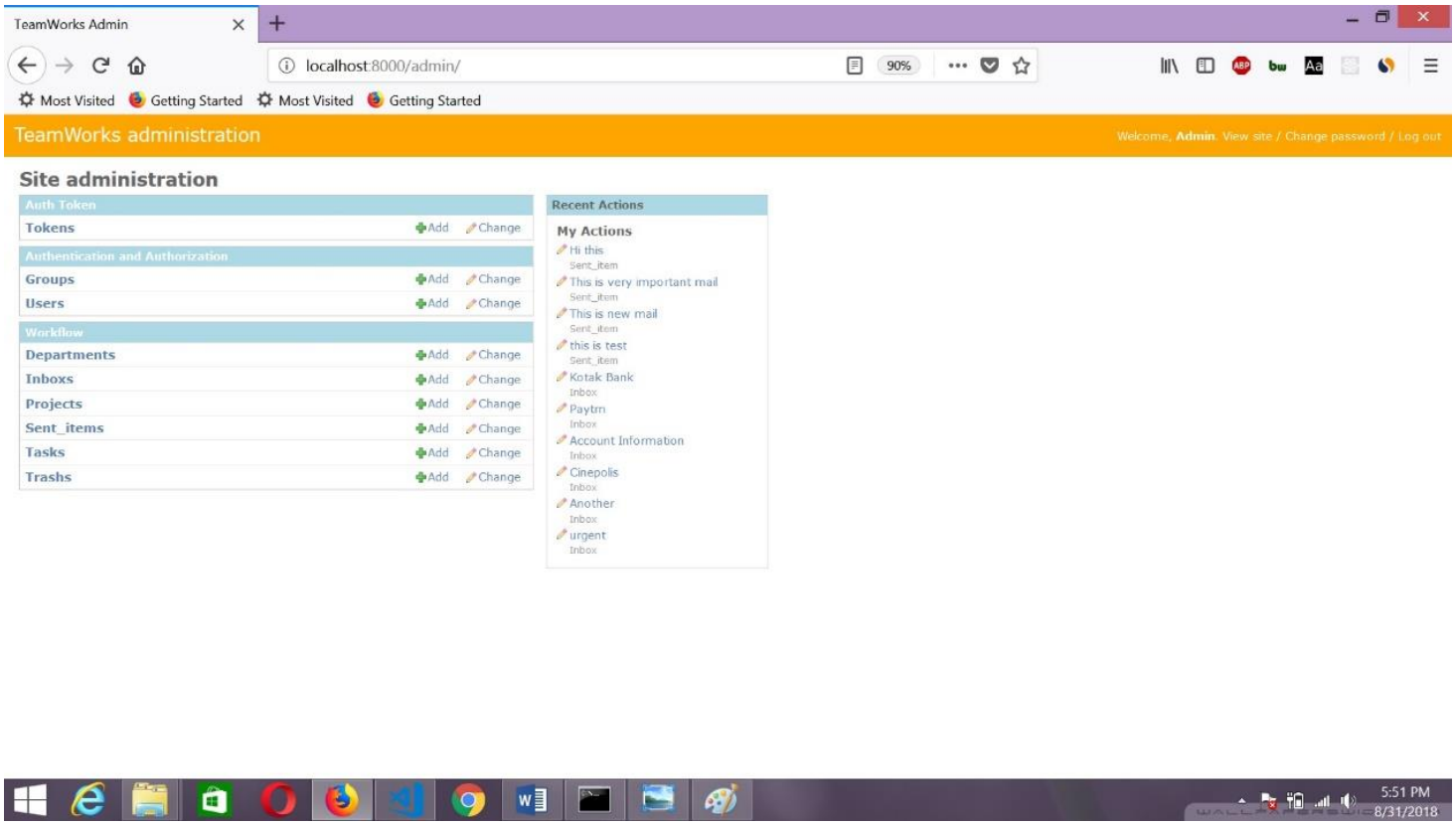
Head Profile view(Group Department Head)



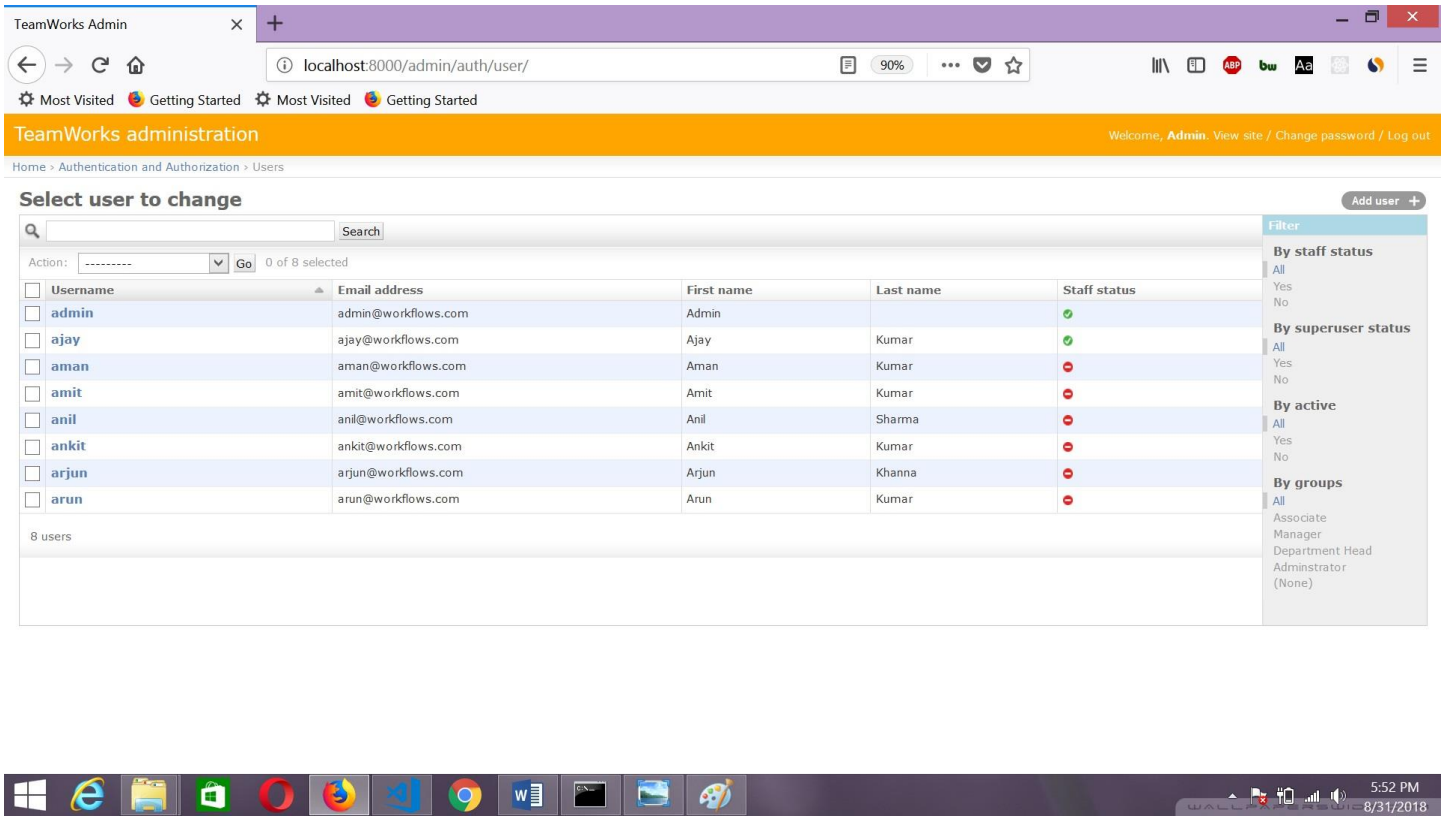
Logout



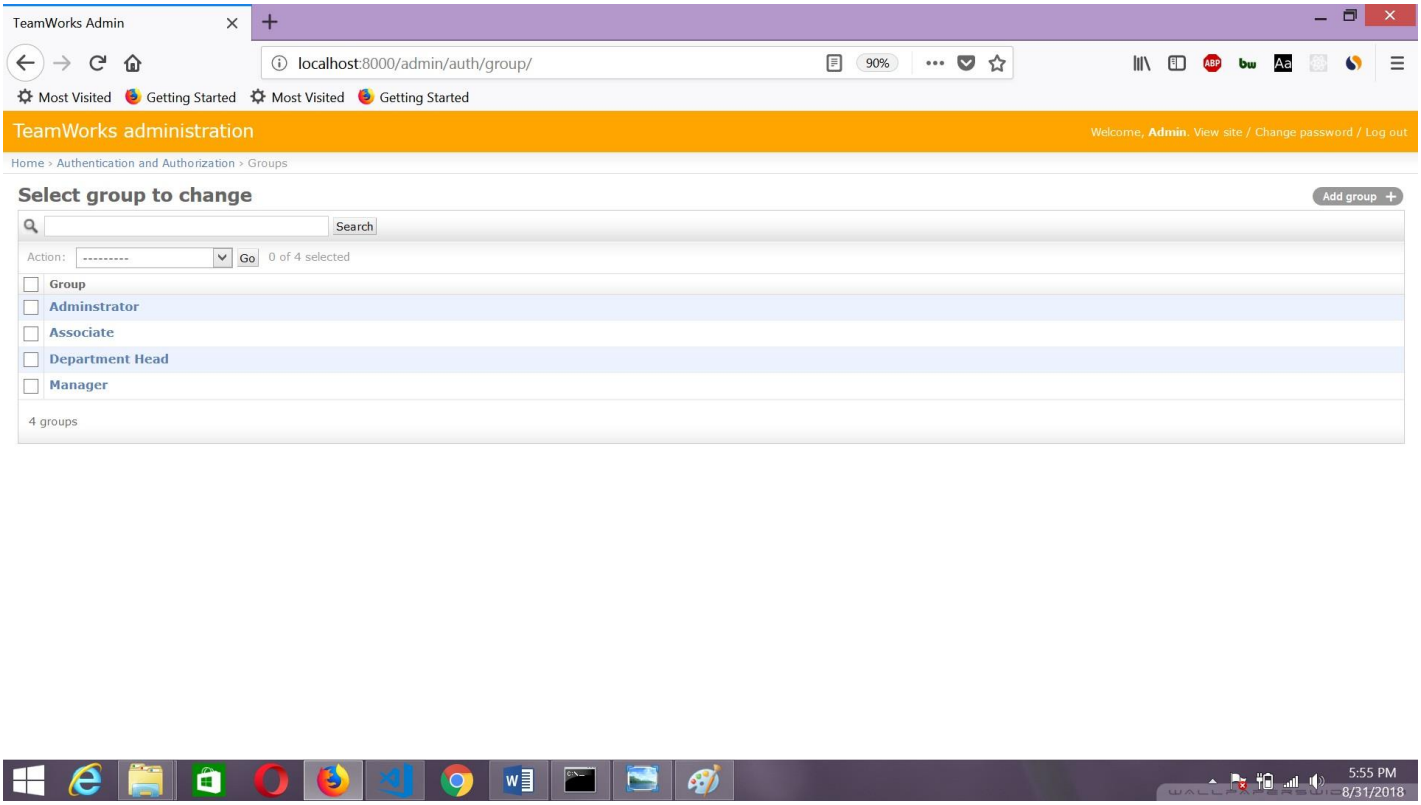
Site Adminstration login(only for admin)



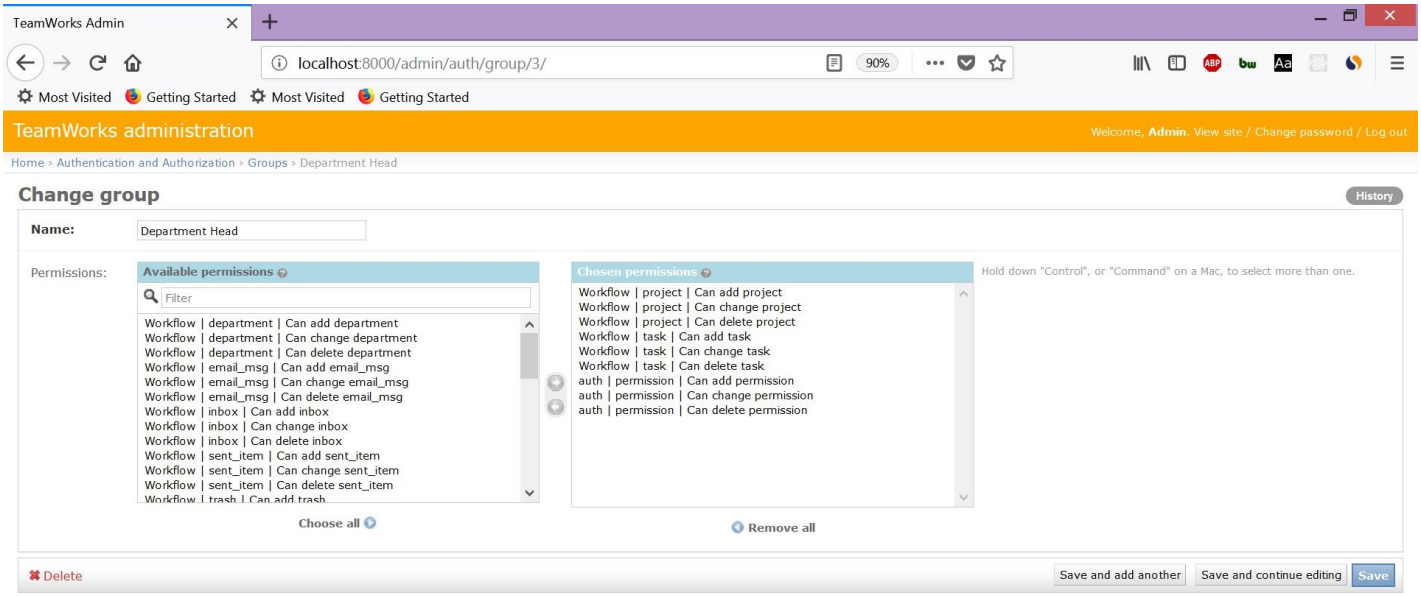
Admin Dashboard

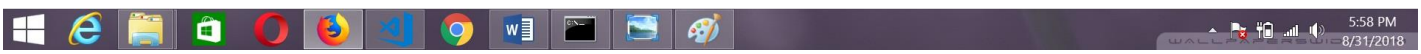


Admin users views

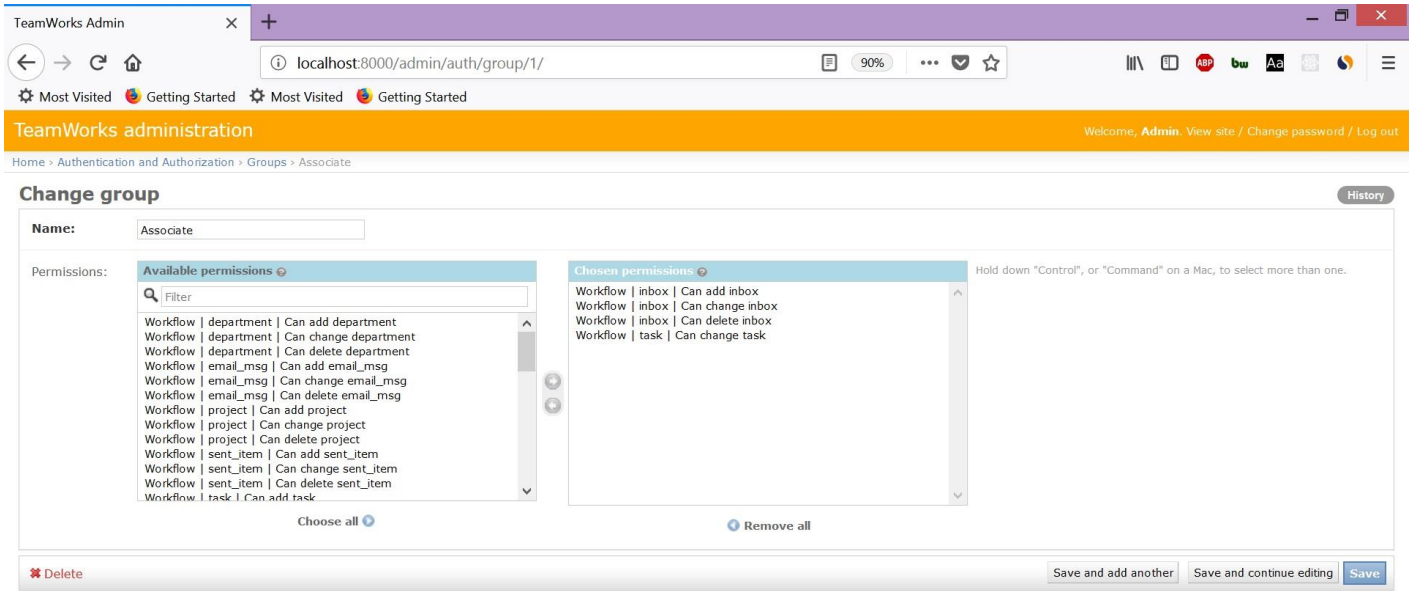


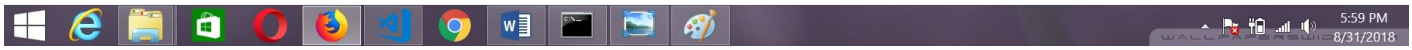
Admin Groups



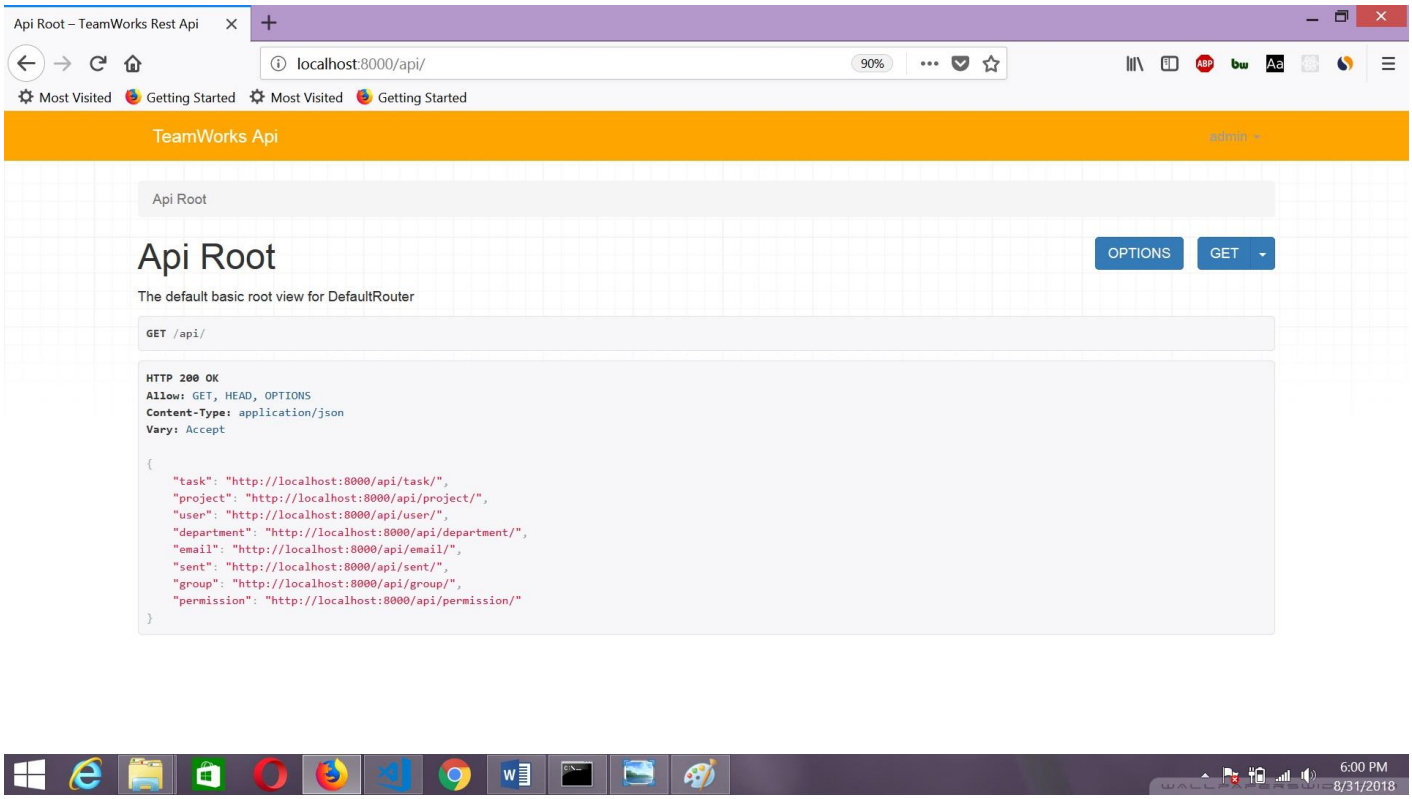


Edit/Create Department head permission view

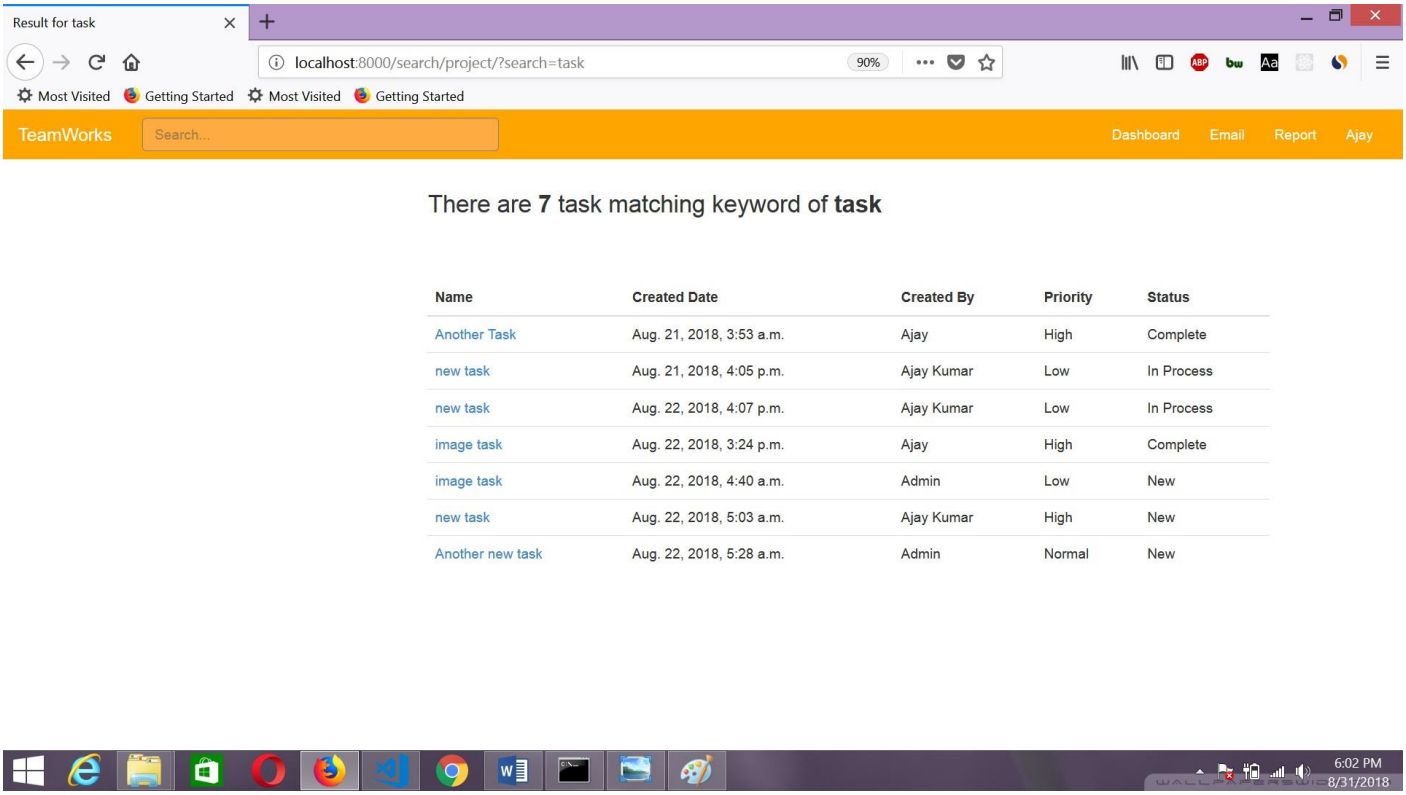




Associate permissions



Site Api Dashboard



Searching Task

Testing Techniques and Strategies

The philosophy behind the testing is to find errors. Test cases are devised with this purpose in mind. A test case is a set of data that the system will process as normal input. There are two general strategies for testing Software: Code Testing and Specification Testing. In code Testing, The analyst develops the cases to execute every instructions and path in the program. Under specification testing, the analyst examines the program specification and then writes test data to determine how the program operates under specific conditions.

Levels of Tests

Unit test

In unit testing I have tested the programs making up a system. For this reason, unit testing is sometimes called program testing. Unit Testing gives stress on the modules independently of one another, to find errors. TeamWorks consists of modules to handle login, modify or retrieve data and to respond to different types of export reports. The test cases needed for unit testing should exercise each condition and option.

Login	
Description	Login Page
Framework	Django 1.8
Operating System	Linux
Error Found	4
Status	None

Dashboard	
Description	Home Page
Framework	Django 1.8
Operating System	Linux
Error Found	8
Status	None

Create Task	
Description	Create task
Framework	Django 1.8
Operating System	Linux
Error Found	9
Status	None

Update task	
Description	Edit/Update Task
Framework	Django 1.8
Operating System	Linux
Error Found	4
Status	None

Delete Task	
Description	Deletion of Task
Framework	Django 1.8
Operating System	Linux
Error Found	3
Status	None

Add user	
Description	Create new user
Framework	Django 1.8
Operating System	Linux
Error Found	6
Status	None

Add group	
Description	Create new group
Framework	Django 1.8
Operating System	Linux
Error Found	8
Status	None

Permissions	
Description	Assign/create permission
Framework	Django 1.8
Operating System	Linux
Error Found	12
Status	None

Project	
Description	Create/Delete
Framework	Django 1.8
Operating System	Linux
Error Found	10
Status	None

composing mail	
Description	Send mail to user
Framework	Django 1.8
Operating System	Linux
Error Found	5
Status	None

Inbox	
Description	Mailbox
Framework	Django 1.8
Operating System	Linux
Error Found	9
Status	None

Sent	
Description	Sent item
Framework	Django 1.8
Operating System	Linux
Error Found	2
Status	None

Reports	
Description	Download/check reports
Framework	Django 1.8
Operating System	Linux
Error Found	15
Status	None

Integration testing

Integration name	Integrated Modules
Integration details	Login → Dashboard Dashboard → Add task Dashboard → Email Dashboard → Add Project Dashboard → Projects Dashboard → Report Dashboard → Profile Dashboard → Logout task details → Edit/Update task task details → Delete task Admin dashboard → Users Admin dashboard → Groups
Found Errors	20
Status	None

Implementation

One of the crucial phases in the Software Development Life Cycle is the successful implementation of the software. Implementation includes all those activities that take place to convert from the old system to the new one.

The new system may be completely new, replacing an existing manual or automated system or it may be major modification to an existing system. In either case, proper implementation becomes necessary so that a reliable system based on the requirements of the organization can be provided.

Successful implementation may not guarantee improvement in the organization using the new system, but improper installation will prevent it. It has been observed that even the best system cannot show good result if the analysts managing the implementation do not attend to every important detail. This is an area where the systems analysts need to work with utmost Propertye.

Limitation of the projects

- Can’t cover the financial data of the company
- Can’t store full information of Employees/users
- Can’t show history of action of particular task or project
- Email can’t be sent to other mailing system such as gmail,yahoo.
- Can’t track personal activity of employee

Cost Estimation

Task Traking System is very useful for any organization to track and collaboration the team, and organization has dynamic number of employees so the cost estimation of this project for company based on their number of employee, company has to pay for particular login on TeamWorks per Year.

Number of Emplyees	Cost(per login)
1-10	1000 INR
10-100	900 INR
100-500	850 INR
500-2000	800 INR
2000-10000	750 INR
More than 10K	700 INR

Future Scope

The future scope of this project is as follows:-

- This project will cover only the project of organization problems/solution etc. prescribed by the Employee but it would be prepared in such a manner that all the modules related to a organization can be easily merged to this project.
- This project can be further enhanced to make the company completely automatic management.
- With the uses of more organization as many as possible we can make this project more helpful for the organization.
- This project can improve the productivity of work in organization.
- For big organization It will add some useful features such as full HR work, telecommunication etc.

Bibliography

- O'Reilly Python/Django
- Stackoverflow.com
- gitHub.com
- djangoproject.com
- Python.org
- Backbonejs.org
- Underscorejs.org