Institute Automation System

Object Design

Version 1

4 May 2018

Oğuzhan Ulusoy

Prepared for
CSE490 Bc.Thesis Project

IŞIK UNIVERSITY
COMPUTER
SCIENCE AND
ENGINEERING

Table of Contents

OBJECT DESIGN DOCUMENT

1. Introduction

1.1. Object Design Trade-offs

- We implemented the “active course” attribute in Taken Course model with Course foreign key. We have changed it as Section foreign key, instead Course foreign key. Semantically, Taken Course represents students have registered courses currently.
- We have worked with Django User model. We used it as abstract user model. Phone number attribute was char field. We thought people might type invalid input for phone number. Therefore, we have evaluated by regular expression.
- We have added some extra attributes in to Student model. These are orderly “hold state”, “reg open statue” and “approval statue”. Without these, the controlling mechanism is weak.
- Instead of defining academic staff completely, we have satisfied to inherit from Staff model. So that, complexity decreased.
- Instead of removing an existing course, we have defined an attribute that is called “is deleted”. Thanks to this variable, we will not remove a course from the database. It is stored in database similar to archive.
- Instead of defining another model for available courses, we have defined an attribute that is called “is valid” in Course model.
- Instead of defining all courses in Curriculum model, we have followed a strategy to solve this. We implemented CCR Course model and Curriculum model. We are able to add a course in CCR Course for a selected curriculum.
- Instead of defining a role model, we used Django permissions.

1.2. Interface Documentation Guidelines

Front-end design:

- Static content, such as CSS and Bootstrap files, is made a folder that is called static, in main.
- Django Template Language (DTL), in other words jinga2, is used for public side of project.
- There are five bases. Child bases are inherited from parent base.
- Common areas where assets – footer, header, head, navigation - and body, in other say content area, are tagged with jinga2.
- All pages extend a base through the activity.
- Django forms are used for input operations. They are implemented in pages as classical.
- Django Group Permissions are implemented in pages for security.

1.3. Definitions, Acronyms, and Abbreviations

- DTL: Django Template Language
- IAS: Institute Automation System

Institute Automation System

- RAD: Requirement Analysis Document
- SDD: System Design Document

1.4. References

- RAD: Requirement Analysis Document
- SDD: System Design Document
- Model Implementation

2. Packages

IAS: Institute Automation System

- Media
- IAS
    - init.py
    - settings.py
    - urls.py
    - wsgi.py
- Institute
    - migrations
    - init.py
    - admin.py
    - apps.py
    - forms.py
    - lists.py
    - managers.py
    - models.py
        - user
        - visitor
        - personal information
        - student
        - staff
        - academic staff
        - institute
        - department
        - program
        - quota manager
        - curriculum
        - course
        - section
        - schedule

    - course type
    - ccr course
    - offered course
    - taken course
    - completed course
- tests.py
- views.py
    - home
    - academic staffs
    - institute staffs
    - grad students
    - institute
    - institutes
    - institute details
    - departments
    - department details
    - programs
    - program details
    - curriculums
    - curriculum details
    - courses
    - course details
    - available courses
    - remove course
    - open/close course
    - course types
    - sections
    - section details
    - academic staff registration
    - all academic staffs
    - academic staff details
    - institute heads
    - department heads
    - program heads
    - quota managers
    - all institute staffs
    - all grad students
    - applications for visitors

    - applications
    - application details
    - remove selected application
    - remove all applications
    - successfully
    - invalid
    - all completed courses
    - all completed course details
    - selected completed course details

- Pages
  - assets
    - contents
      - modal (html)
      - table (html)
    - menu
      - academic staffs (html)
      - grad students (html)
      - institute staffs (html)
      - institute (html)
    - footer (html)
    - head (html)
    - menu (html)
    - navigation (html)
  - registration
    - login (html)
  - academic staff details (html)
  - academic staff registration I (html)
  - academic staff registration II (html)
  - academic staffs (html)
  - all academic staff (html)
  - all completed courses details (html)
  - all grad students (html)
  - all institute staffs (html)
  - all completed courses (html)
  - application details (html)
  - applications (html)
  - apply (html)
  - available courses (html)

    - base (html)
    - curriculum details (html)
    - curriculums (html)
    - close course details (html)
    - close course (html)
    - course details (html)
    - course types (html)
    - courses (html)
    - department details (html)
    - department heads (html)
    - departments (html)
    - grad student details (html)
    - grad students (html)
    - institute details (html)
    - institute heads (html)
    - institute staffs (html)
    - institute (html)
    - institutes (html)
    - invalid (html)
    - program details (html)
    - program heads (html)
    - programs (html)
    - quota managers (html)
    - remove course details (html)
    - remove course (html)
    - section details (html)
    - sections (html)
    - selected completed course details (html)
    - student application I (html)
    - student application II (html)
    - successfully (html)
- Static
    - css
    - vendor
- Staticdev
    - admin
    - css
    - vendor

▪ DB.sqlite3
▪ Manage.py

3. Class Interfaces

Model name: User
Model attributes – explanation – dependencies:
Model attributes – explanation:

- Username – Unique username, it should be defined as university identification number.
- E-mail – E-mail address of the user.
- First name – First name of the user.
- Last name – Last name of the user.
- Phone regex – It is regular expression to get valid phone number.
- Phone number – Phone number of the user.
- Date joined – Date that user has been joined into the system.
- Is active – It is for ban.
- Is super user – It is for defining system administrator.
- Is staff – It is for defining staff.
- Avatar – Picture of the user
- Personal information – All personal information of the user, it comes from another model.

Dependencies:

- It is dependent with Personal information model.

Model operations:

- Generate unique username – It is for generating unique username.
- Full name – It is toString method.
- E-mail user – It is for sending an e-mail to user to give as password.
- Save – Saves the all data.

Model name: Visitor
Model attributes – explanation – dependencies:
Model attributes – explanation:

- User – All user information of visitor, it comes from User model.
- TC – Turkey Citizenship number of visitor.
- Birthday – Birthday of visitor.
- Gender – Gender of visitor.
- Application Date – Auto generated date for application.
- Address – Home address of visitor.
- City – City where visitor has been lived.
- Degree – Last education degree of visitor.

- University – University that visitor has been graduated.
- GPA – General point average of visitor that is belonging to university.
- ALES – ALES exam point of visitor.
- YDS – YDS exam point of visitor.
- Acceptance – It is statue for defining the application.
- Program – University program that visitor has been applied.

Dependencies:

- It is dependent with User, Personal Information and Program models.

Model operations:

- There is no model operation.

Model name: Student
Model attributes – explanation – dependencies:
Model attributes – explanation:

- User – All user information of student, it comes from User model.
- Student ID – Unique student identification number.
- Student E-mail – An e-mail address is defined by school for student.
- Curriculum – Curriculum that is actual for a program. It is automatically defined according to the last curriculum in a program.
- Program – Program that student has been applied to.
- Advisor – It is defined by school, in other words academic advisor of a student.
- Hold State – Boolean field to check financial statue of student.
- Reg Open Statue – Boolean field to check registration time of student.
- Approval Statue – Boolean field to check approval statue of student.

Dependencies:

- It is dependent with User, Curriculum, Program, Academic Staff models.

Model operations:

- There is no model operation.

Model name: Staff
Model attributes – explanation – dependencies:
Model attributes – explanation:

- User – All user information of staff, it comes from User model.
- TC – Turkey citizenship number of staff.
- Birthday – Birthday of staff.
- Gender – Gender of staff.

- Joined date – It is obtained automatically by system, the date that staff has been joined to system.
- Main E-mail - Personal e-mail address of staff.
- School E-mail – E-mail address of staff, it is defined by school.
- Address – Home address of staff.
- City – City that staff has been lived.

Dependencies:

- It is dependent with User model.

Model operations:

- There is no model operation.

Model name: Academic Staff
Model attributes – explanation – dependencies:
Model attributes – explanation:

- Staff – All academic staff information, it comes from Staff model.
- University – University that academic staff has been worked.
- Institute – Institute that academic staff has been belonged to.

Dependencies:

- It is dependent with Staff (User, Personal Information) and Institute models.

Model operations:

- There is no model operation.

Model name: Institute
Model attributes – explanation – dependencies:
Model attributes – explanation:

- Name – Name of institute
- Head – Head of institute, it comes from Academic Staff model.
- Established Date – Date that it is obtained automatically.

Dependencies:

- It is dependent with Academic Staff model.

Model operations:

- There is no model operation.

Model name: Department
Model attributes – explanation – dependencies:
Model attributes – explanation:

- Name – Name of department.
- Head – Head of department, it comes from Academic Staff model.
- Institute – Institute that department belongs to.

Dependencies:

- It is dependent with Academic Staff and Institute models.

Model operations:

- There is no model operation.

Model name: Quota Manager
Model attributes – explanation – dependencies:
Model attributes – explanation:

- Quota Manager – Name of quota manager.

Dependencies:

- It is dependent with Academic Staff model.

Model operations:

- There is no model operation.

Model name: Program
Model attributes – explanation – dependencies:
Model attributes – explanation:

- Name – Name of program.
- Code – Code of program.
- Type – Type of program.
- Thesis – Boolean field to show program details.
- Department – Department that program belongs to.
- Head – Head of program.
- Quota Manager – Quota Manager of program.

Dependencies:

- It is dependent with Academic Staff, Department, Quota Manager models.

Model operations:

- There is no model operation.

Model name: Curriculum
Model attributes – explanation – dependencies:
Model attributes – explanation:

- Program – Program that curriculum belongs to.
- Year – Time that curriculum has been defined.

Dependencies:

- It is dependent with Program model.

Model operations:

- There is no model operation.

Model name: Course
Model attributes – explanation – dependencies:
Model attributes – explanation:

- Code – Code of course.
- Title – Full title of course.
- Description – Full details, in other say description, of course.
- Credit – Credit of course.
- ECTS Credit – ECTS credit of course.
- Program – Program that course belongs to.
- University – University that course belongs to.
- Is Valid – Boolean field to check course is open.
- Is Deleted – Boolean field to show course has been removed.
- Created Date – Time that course has been created. It is automatically defined by system.

Dependencies:

- It is dependent with Program model.

Model operations:

- There is no model operation.

Model name: Section
Model attributes – explanation – dependencies:
Model attributes – explanation:

- Course – Course is selected to define a section.
- Number – Unique number of section.
- Quota – People amount of section.
- Instructor – Instructor of section.

- Year – Year that section has been defined.
- Semester – Semester that section has been defined.
- Special quota – It is for opening a special quota for a student.
- Students – Full list of students in this section.

Dependencies:

- It is dependent with Course, Academic Staff, Student model.

Model operations:

- There is no model operation.

Model name: Schedule
Model attributes – explanation – dependencies:
Model attributes – explanation:

- Section – Section of a certain schedule.
- Day – Day of section.
- Slot – Slot of section.
- Place – Place of section.

Dependencies:

- It is dependent with Section model.

Model operations:

- Write Roman – It is to write Roman number.

Model name: Course Type
Model attributes – explanation – dependencies:
Model attributes – explanation:

- Title – Full title of course type.
- Code – Code of course type.

Dependencies:

- There is no dependency.

Model operations:

- There is no model operation.

Model name: CCR Course
Model attributes – explanation – dependencies:
Model attributes – explanation:

- Curriculum – To define a certain curriculum.
- Type – To define course type.
- No – Course number.
- Semester – Semester of CCR course.

Dependencies:

- It is dependent with Curriculum and Course Type model.

Model operations:

- There is no model operation.

Model name: Offered Course
Model attributes – explanation – dependencies:
Model attributes – explanation:

- CCR Course – Course that must be taken.
- Active Course – Course that has been taken.

Dependencies:

- It is dependent with CCR Course and Course models.

Model operations:

- There is no model operation.

Model name: Taken Course
Model attributes – explanation – dependencies:
Model attributes – explanation:

- Student – Student that is taking course.
- CCR Course – Course in curriculum of student.
- Active Course – Section of student.
- Accepted – Boolean field to define draft schedule.

Dependencies:

- It is dependent with Student, CCR Course, Section models.

Model operations:

- There is no model operation.

Model name: Completed Course
Model attributes – explanation – dependencies:
Model attributes – explanation:

- Student – Student of completed course.
- CCR Course – Course in curriculum of student.
- Active Course – Course that is selected by student.
- Grade – Grade of student, in that course.

Dependencies:

- It is dependent with Student, CCR Course, Course models.

Model operations:

- There is no model operation.

Implementation Planning:
We have planned service-oriented architecture, not class based.
The web-services in project are similar to each other. We already developed a web-service for “Institutes” tab.

The web-service is called in a page, receives the incoming HTTP request.
def *institutes(request):*
Rendered URL is defined.
url = 'institutes.html'
The purpose of this page is to display all institutes. Therefore, a typeless variable is defined and all Institute objects are received. It can be received with “.objects.all()” function. We can order them by some feature. We receive Institute objects ordering by “established date” attribute.
institutes = Institute.objects.order_by('-establishedDate')
Normally, we can return “institutes” object, but we want to return a form to add new institute object. We took a request for web-service. If we create a form in secure, we should check it, if it is POST or GET.
if request.method == 'POST':
When request is POST, we define a “form” object. We call relevant form with request, match to this form object.
form = AddInstituteForm(request.POST)
We assume user has filled the form. If the form is valid, we open new indent.
if form.is_valid():
Form is saved.
form.save()
Received data is cleaned. This is required for invalid text.

name = form.cleaned_data['name']
head = form.cleaned_data['head']
else:
Otherwise (form is not valid), it is returned to page that is called invalid.
return redirect('/invalid')
else:
form = AddInstituteForm()
We return everything by rendering. Returning objects are request, url, a dictionary that is including “institutes” and “form” objects.
return render(request, url, {'institutes' : institutes, 'form' : form})

It might be desired to see details of any item. The web-service is called in “Institutes” page. When any item is clicked, new request goes to relevant method. The web-service receives request and identification number of selected item.
def instituteDetails(request, id=None):
Rendered url is defined.
url = 'institute-details.html'
All item has default unique primary key. All details of selected item is received with “get_object_or_404” function. It can be received with other functions, such “.objects.all() – objects.filter() - .objects.get() – and so on”. “get_object_or_404” needs two fields. These are model name and primary key. Model name is Institute, and pk is identification number of clicked item.
instituteDetails = get_object_or_404(Institute, pk=id)
It is returned by rendering request, url and all content.
return render(request, url, {'instituteDetails' : instituteDetails})