MARMARA UNIVERSITY

FACULTY OF ENGINEERING

COMPUTER ENGINEERING

CSE3063 Object Software Design Java Project

Course Registration and Tracking System

Iteration #2

Group 16

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to give a detailed description of our Course Registration Analysis System project. This document contains the purpose, requirements and functions of our project and is a kind of a guide to those who will use our software to analyse their course registration processes.

1.2 Intended Audience and Intended Use

Our project can be used by both Marmara University Computer Engineering students, and advisor teachers who check whether the students meet the compulsory requirements in order to take the courses they have chosen, in the course registration process.

1.3 Scope of Project

At the beginning of every academic year, we choose our courses and certain conditions must be met in order for us to be able to enroll in these courses. For this course registration process, we have created a software that simulates the course registrations of our department and observes the problems and bottlenecks.

Our project aims to check whether a student can enroll in the course they have chosen, based on the rules of our department, transcript information of the student and the prerequisites and quotas of the courses.

2. Overall Description

2.1 Product Functions

Our course registration and tracking system is based on the completion of the course registrations by processing the data taken from the JSON files. In this project we analyze, design and implement a registration system in an object oriented manner.

We created an object based design that includes reading and writing JSON files, taking teachers, students and courses and randomly generated registration process.

-TeacherJSONReader: Reads all the teachers and puts them on a list.

-CourseJSONReader: Reads the courses and determines the teachers of these courses. It also divides the courses into three categories as non-technical elective, technical elective and regular department courses.

-StudentJSONReader: Reads students name and ID information from student.json file. It also creates a student list by assigning the semester information etc.

After these processes, each student's transcript is arranged according to the random success rate. If there are courses that the student has failed before, they are asked to take them first. In addition, the courses that the student can take are determined according to the semester they are in. For example, in the fall period, s/he can take the courses opened in the fall semester.

A student can take a maximum of 10 courses in a semester, and in order to take the courses s/he has chosen, s/he must complete the prerequisite requirements of that course. In addition to these, there are some conditions and restrictions for each course category:

> -Techinal Elective: If there is enough quota and the student has completed at least 155 credits, they can take these courses, maximum two in the spring semester and maximum three in the fall semester.
>
> -Non-Technical Elective: There are certain quotas for these courses. If there is enough quota, the student can enroll in these courses.
>
> -Regular Department Courses: There is no quota for these courses. If the student can meet the prerequisites for these courses, they can enroll in these courses. However, for the graduation project, it is necessary to complete at least 165 credits.

3. Specific Requirements

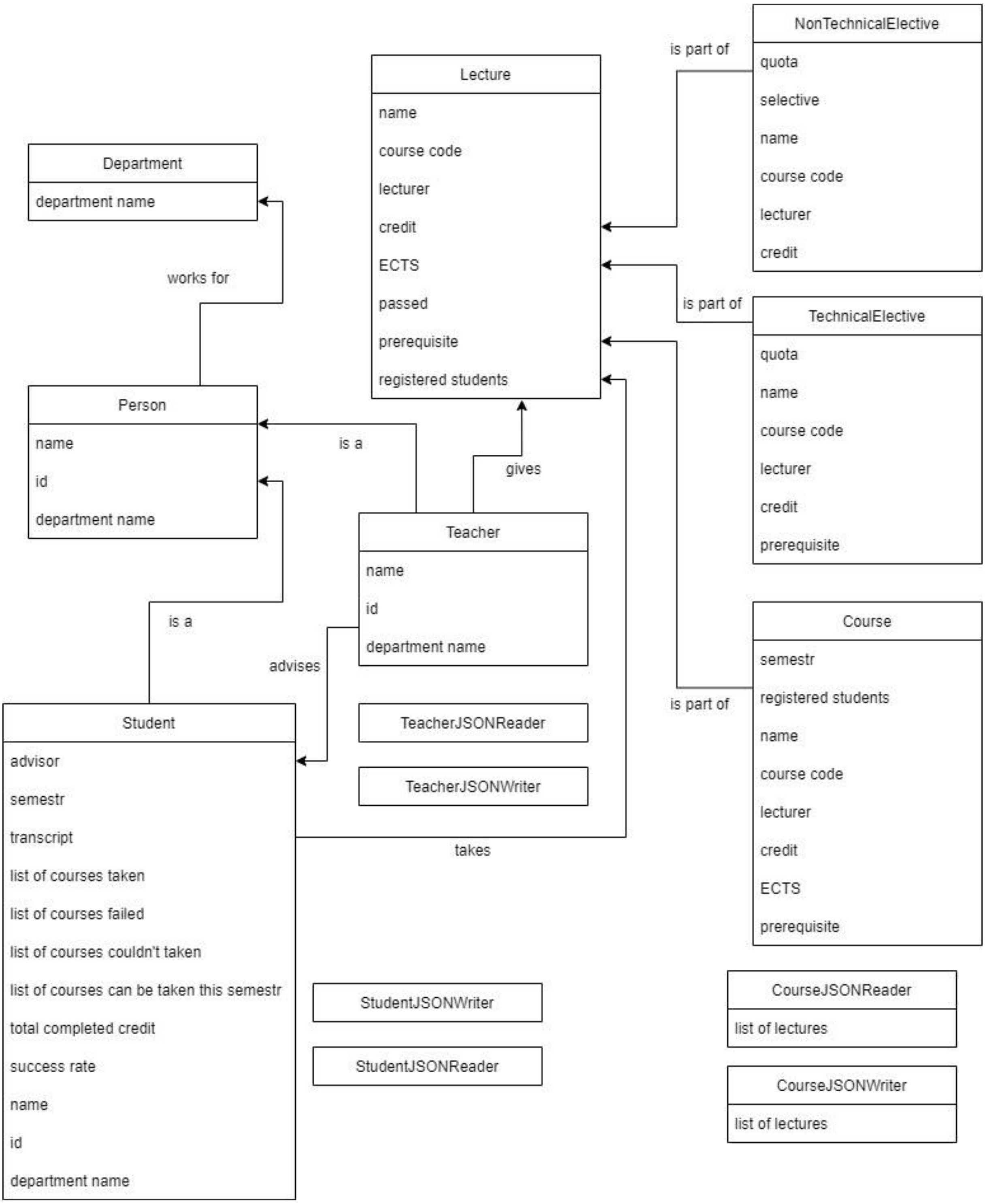
3.1 Functional Requirements

- There are taken input parameters from a JSON file.
- There are quotas for each course especially for elective courses in the JSON file.
- There are prerequisites of the courses.
- There are students that randomly generated that can be in each of the 8 semester with equal probability.
- There are students and their transcripts in JSON files.
- There is a JSON file for each student.
- In order to simulate course registration process, either a randomly selected student or a student's JSON filename will be chosen.

3.2 Non-Functional Requirements

- **Portability:** Our software does not cause any problems when running on different environments. It works on any platform that can run Java.
- **Usability:** Our software is simple and easy to use by users.
- **Performance:** In our project, there are no immediate delays and response time is as little as possible.

4. Domain Model


NonTechnicalElective
quota
selective
name
course code
lecturer
credit
is part of
Lecture
name
course code
lecturer
credit
ECTS
passed
prerequisite
registered students
Department
department name
works for
is part of
TechnicalElective
quota
name
course code
lecturer
credit
prerequisite
Person
name
id
department name
is a
gives
Teacher
name
id
department name
is a
advises
Course
semestr
registered students
name
course code
lecturer
credit
ECTS
prerequisite
is part of
Student
advisor
semestr
transcript
list of courses taken
list of courses failed
list of courses couldn't taken
list of courses can be taken this semestr
total completed credit
success rate
name
id
department name
TeacherJSONReader
TeacherJSONWriter
takes
StudentJSONWriter
StudentJSONReader
CourseJSONReader
list of lectures
CourseJSONWriter
list of lectures