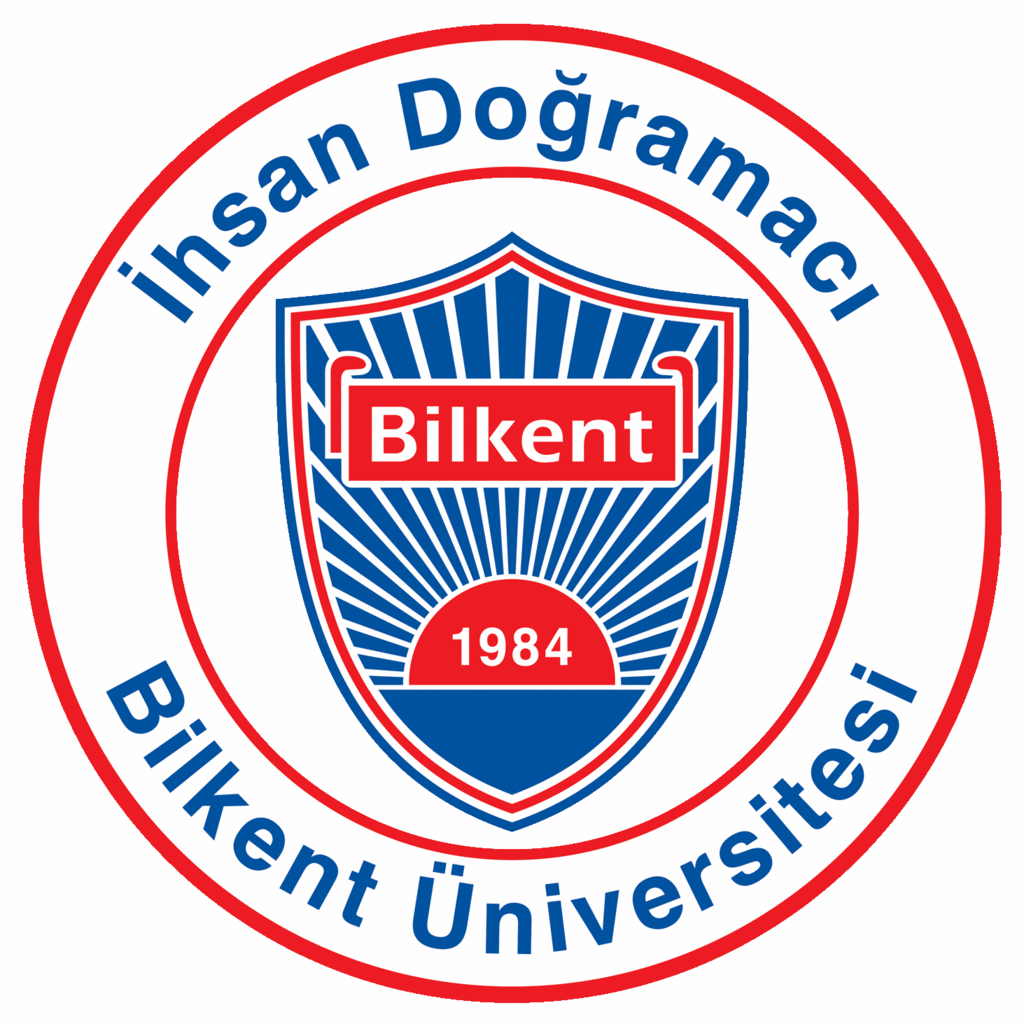
****

Bilkent University

Department of Computer Engineering

**Software Architecture Design**

**Useful ProDers App for Headmaster of Schools**

**Report**

**Group-16**

**Ahmet Ay**

**Anıl Göllü**

**Cüneyt Erem**

**Muratcan Katırcı**

TABLE OF CONTENTS

1. INTRODUCTION..............................................................................................................3
2. PROBLEM STATEMENT ..................................................................................................4
3. THE SOFTWARE ARCHITECTURE DESIGN PROCESS........................................................4
4. REQUIREMENTS ANALYSIS

4.1. STAKEHOLDERS IDENTIFICATION............................................................................5

4.2. FUNCTIONAL REQUIREMENTS................................................................................8

4.3. NON-FUNCTIONAL REQUIREMENTS .....................................................................10

4.4. USE CASE MODELS................................................................................................11

4.5. ARCHITECTURAL SCENARIOS.................................................................................12

4.6. USER INTERFACE PROTOTYPES.............................................................................15

5. TECHNICAL PROBLEM ANALYSIS

5.1. COMPUTER SCIENCE PROBLEMS ..........................................................................23

5.2. MATHEMATICS DOMAIN PROBLEMS....................................................................24

5.3. APPLICATION DOMAIN PROBLEMS ......................................................................24

5.4. QUALITY PROBLEMS .............................................................................................24

6. DOMAIN ANALYSIS

6.1. DATABASE MANAGEMENT SYSTEMS....................................................................25

6.2. DISTRIBUTED SYSTEMS .........................................................................................25

6.3. USER INTERFACE DOMAIN....................................................................................26

7. REFERENCE.....................................................................................................................27

**1. INTRODUCTION**

Education has very big effect on society in various ways from the ancient history because it directly related to almost all different types of parts of human lives such as science, technology, art and human attitudes. Because education is one of the most important issue for society, developing an application for educational advancements is also significant and may provide huge advantage for the society.

The aim of this project is to increase the quality of education by providing headmaster of school with rescheduling the whole curriculum of the entire school for each different school. Almost all schools try to enhance their curriculum by changing their schedule but it is difficult for the headmasters to reconsider all classes, teachers, branches etc. so that this app provides headmasters to reschedule their plans in a fast way. Headmaster can open/close specific hour. Each day has 18 hours and there are 7 days. User can also define ‘lunch’ hours and determine the starting/ending time of the classes. User can create class and its name, schedule etc. and charge administrator.  He can add classes to the schedule one by one or randomly from the current MEB system. He can create teachers and their names, branches etc. Each lesson can be assigned to the classes. The convenient part of the program is to define some borders and define priority of the all classes and teachers. For example, “Mustafa Bayar” does not have any class on Monday, and he will have max number of classes on Friday. By using high/low priority feature, he can be assigned to scheduled program which app uses lots of algorithm to find the best option.  So, this app helps headmasters to schedule the curriculum and assign teachers to the classes properly fast.

Because of good facilities, it is very useful for the headmasters. Besides the introduction of this app, Section 2 includes the problem statement that why we choose this topic. Section 3 includes the approach that explains the steps of the requirement analysis, technical problem analysis and domain analysis problem. Section 4 covers the requirement anaysis section that includes stack holder, functional/non-functional requirements, and technical problem analysis and domain problem analysis. Section 5 involves the technical problem analysis that computer science problems, mathematics domain problems, application domain problems, and quality problems.  Section 6 contains domain analysis which is related to the solution domains and knowledge sources are explained, also glossary is constructed for those identified domains.  Section 7 contains the software architecture design process in which shows the conceptual software architecture designed by UML notations. In Section 8 top level context diagrams which show the boundaries of this app is defined. Report continues with the modeling of software architecture, Views and beyond approach is chosen for description of software architecture model. Section 9, Module Views which contains uses, decomposition, generalization, layer, aspect and data flow views in order to documents the systems principal units of implementations. Section 10, Component and Connector Views which contains Client-Server, Publish-Subscribe and Repository views to show that system is structured as a set of elements that have run-time behavior.

**2. PROBLEM STATEMENT**

There are 61,200 high schools in Turkey which means there are around sixty thousand headmasters in our country. There are lots of different types of curriculums and some of them proved their usability. But almost every year, schedules have to change because education system changes almost biyearly. Therefore, headmasters need to change their plans but MEB does not have particular system to do it properly. Headmasters should decide teachers, classes, hours etc. Some mini types of scheduling programs exist but not proper for the MEB systems because each school contains hundreds of teachers, classes etc. that headmasters mostly give lots of time to do their schedule plan.

The aim of this project is to provide headmasters with providing a useful app which makes their work fast. Headmaster can see the schedule and apply his/her thoughts by using high quality features. User can determine the priority of the teachers and classes so that scheduling algorithm finds the best results to be applied on curriculum.

**3. APPROACH**

Requirements analysis, technical problem analysis, domain analysis and software architecture design are the steps to be followed in the design process of software architecture of Proders. These steps are essentially based on the synthesis-based software architecture design approach.

Requirements Analysis: This phase is defined to determine the needs for a new system that will be developed. It is crucial process for the success of the project. Requirements analysis phase starts with the identification of stakeholders. After that, the functional/non-functional requirements are specified for each stakeholder by conceiving their prior concerns and expectations. After the specification of stakeholders with their concerns, the requirements of the project are defined with use cases, textual requirements, architectural scenarios and UI prototypes to make the system more clear and understandable.

Technical Problem Analysis: Technical problem analysis defines the technical problems which considering the mathematics domain problems, application domain problems, computer science problems, quality problems and the related concerns which have to be solved properly are specified. The process starts by defining identification of the problems. Then the current systems are evaluated in keeping with these problems and states which are found. Then application is explained.

Domain Analysis: It is the process of analyzing related software systems in a domain to find their common and variable parts. Firstly solution of the domains are identified using the technical problems which is described in the “Technical Problem Analysis” section. Then the source f the knowledge are evaluated and described. Also the derived concepts and the structure and description of concepts are specified.

**4. REQUIREMENTS ANALYSIS**

Requirements Analysis process is a must for a software. The customers’ or users’ needs should be completely described in this phase. In order to provide a better Requirement Analysis, stakeholders, textual requirements, use cases, architectural scenarios and prototypes will be mentioned in detail.

**4.1 Stakeholders**

Stakeholder is anyone or group that play role in the system by either affecting it or being affected by it. Stakeholders of Proders can be divided into two groups:

End Users:

. School Administration

. Teacher

. System Administrator

. Student’s Parent

Technical Developers:

. Software Designer

. Implementer

. Project Manager

. Software Architect

. Tester

. System Maintainer

**4.1.1. End Users**

End users are actual users of the system that analyzed requirements derived from. The system should requirements their needs. Otherwise, the system cannot have any user.

School Administration: School Administration’s role in the system is the most important end user part. They create curriculum according to different classes, teachers, difficulty of the courses and time constraint. The curriculum created by system administrator should be approved by school administration.

Teacher: Teachers are able see their schedules that are sent by their School Administrator via system’s e-mail service.

System Administrator: System Administrator differs from school administration by not having approval permission.

Student’s Parent: This stakeholder has ability to see their children’s curriculum or course schedule with assigned teacher.

**4.1.1.1. Concerns of End Users**

Concerns of end users are mostly about usability of the system. The user should know every part of detail that what she/he can do with the system. It can be called also as ability knowledge.

Besides, the user also expects the system to be ease of use. Proders is a complex system that provides too much features and opportunities to users to handle arranging curriculum or course schedule.

Accessibility to system is another concern for the end users. They should be able to access the system whenever with the username and password they are given. Additionally their access should be secured.

Finally, the user expects sensible and feasible curriculums from the system regarding with teachers’ availability and for example not adding difficult courses on same day.

**4.1.2. Technical Developers**

Software Designer: Software designer designs the system as it compensate the requirements.

Implementer: An implementer provides the essential software according to software designer’s and architect’s work.

Software Architect: A software architect is obligated to provide a proper, implementable architect for the system.

Tester: This stakeholder tests the system whether it has any bugs or not. It has highly importance that the system should pass the tests in order to provide a good quality service to users.

Project Manager: From beginning to the end, a project manager is the worker that irresponsible with every part of the system. Also, each part of system and their process should be confirmed by project manager.

System Maintainer: The person or group who are responsible for fixing any errors after the system provided in the market. Any update or maintenance is done by System Maintainer.

**4.1.2.1 Concerns of Technical Developers**

No doubt that, a project manager (PM) would like to deliver or release a project to market in time. Staying behind of the schedule is an undesirable event for them. That's why timing is one of their most important concerns. A PM also have to manage the budget and work progress in the project. Exceeding budget or misdirected work would threatens the purpose of the project. It is also PM's important concern that the project should be completed with desired qualifications.

For a tester, the system should be testable. Implemented and designed works need to be fit with testing tools.

For a software designer, the design of the system must be identified clearly. Next designers, architects and implementers should be able to understand the design clearly. It must be done with universal design methods not personal discovered ones.

An implementer should pay attention to system's performance. It should not be affected by time-wasting codes such as including lots of iterative and huge loops (E.g. n = 100000). Also the system should be optimized with different kind of operating systems, browser et cetera...

Software architect should take flawless and proper design into consideration. The design should exactly detect the stakeholder's requirements.

System maintainer should consider doing updates according to stakeholder's refreshed needs. The errors or bugs in the system should be fixed immediately if it blocks user's acts in the system.

**4.2. FUNCTIONAL REQUIREMENTS**

**.** The system should allow the user to log in with their username and password.

**.** If the user forgets password the system should provide an alternative way user to gaining a new password such as security question or sending an e-mail to registered account.

**.** The system should allow the user to create a new course schedule.

**.** The system shall be able to let user creating different degreed classes.

**.** The system should provide user to option of identifying the lectures and its instructions.

**.** The system should allow user to edit any information user entered.

**.** The system shall allow user to assign lectures to the specific classes.

**.** The system shall allow user to set difficulty level of a lecture.

**.** The system shall allow user to enter dependence level of lectures in each other.

**.** The system shall allow user to add/delete new lectures to a class' curriculum.

**.** The system should provide user the ability of saving created curriculum.

**.** The system shall provide user the ability of loading a curriculum.

**.** The system shall allow user to choose prior factors to determine the schedule.

**.** The system shall allow user to set lecture and break times including afternoon break.

**.** The system shall allow user to send curriculums to teachers and parents of students.

**.** The system shall allow user to display curriculum of a specific class.

**.** The system shall allow user to arrange elective lectures.

**.** The system shall allow user to set previous prepared curriculums.

**.** The system shall allow user to arrange teachers' watching regulations.

**.** The system shall allow user to display schemas and tables about each classes.

**.** The system shall allow user to calculate extra-lecturing cost for teachers.

**.** The system shall allow user to contact with system's personal about system regarding issues.

**4.3. NON-FUNCTIONAL REQUIREMENTS**

**.** The user population is existed by mostly school administrator. Thus, the system should be designed as user-friendly that a person who does not interested in computer works can easily use.

**.** The user-interface should provide user to access any option easily and it must be easily understandable.

**.** The system should be implemented as it can be updatable later without taking too much time to use.

**.** The system should be accessed by different kind of web browsers such as Mozilla, Opera, Chrome, Internet Explorer, Safari et cetera...

**.** The system should be accessed by mobile phones, tablets and computers that any devices which have internet connection.

**.**  The internet connection has to be at least 1 Mbit to use the system properly.

**.** The response time while the user arranging something should be less than 1 second.

**.** The system should be reliable.

**.** The system safety should be paid attention that the information of any user cannot be seen by other users except system administrator.

**.** The system's self-regulation of curriculum service should provide sensible tables and schemas.

**.** Two users with same account cannot be able to access the system at the same time.

**.** The database that records are kept should be big enough to store more data for more users.

**4.4 USE CASE MODELS**

In this part, the use case diagrams of the system and related scenarios explaining the diagrams are explained

**4.4.1 USE CASE DIAGRAM**

There is one user in Proders. Figure 1 explains the interaction of instructors with the system. Instructor logs into the system. He/she can add/drop the course, instructor and classroom on the board. Instructor can assign the course to classroom and instructor. He/she set restriction for course and instructor and can create course schedule automatically. Additionally, instructor can create schedule for guard.

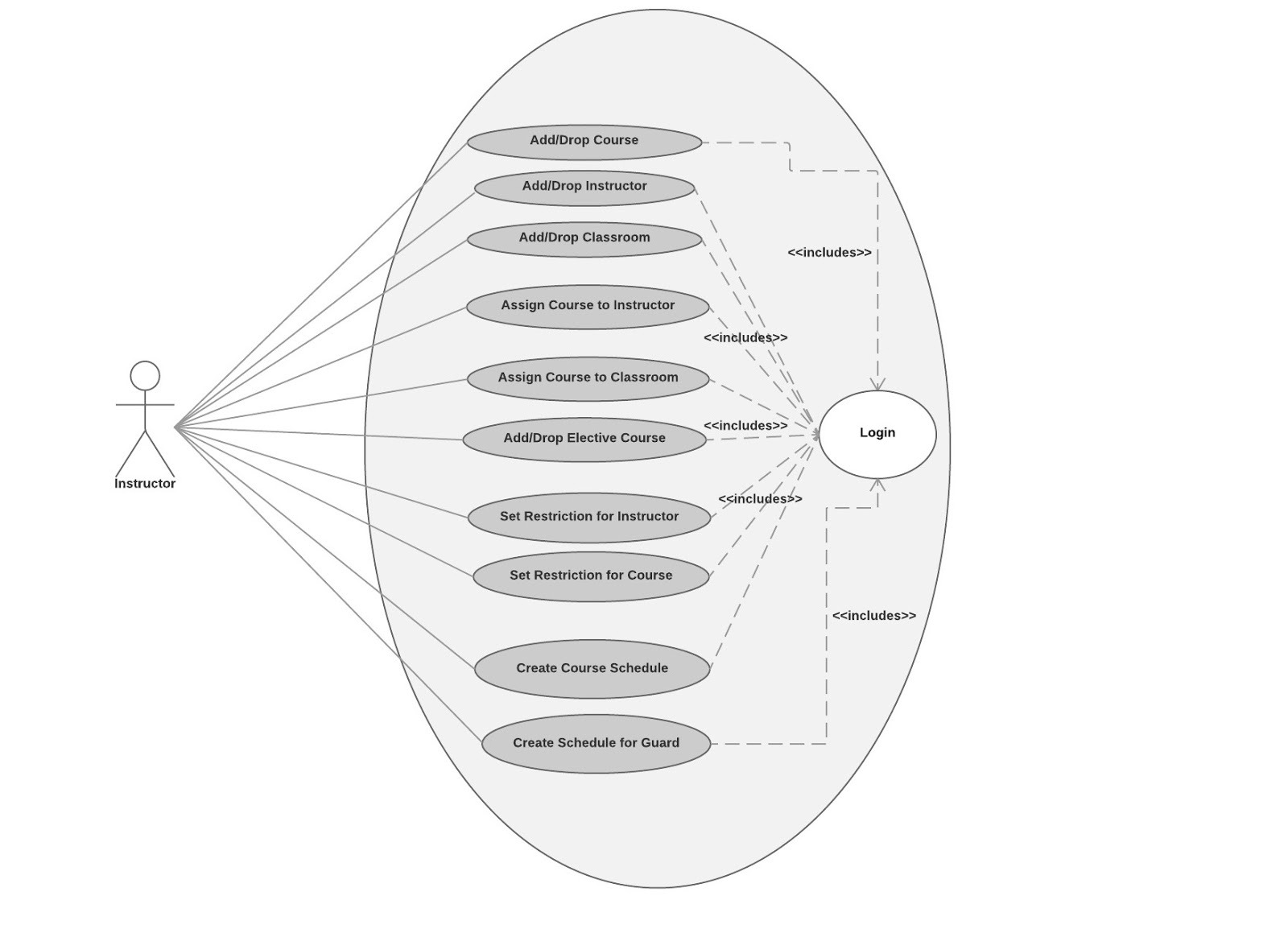


Figure 1

**4.5. ARCHITECTURAL SCENARIOS**

**Logging in and Registering Proders**

Mr. Namık Kemal is a school administrator. Proders is a web-based system. In order to access this system, Mr. Kemal should create an account by registering the web-site by providing necessary information such as user name, password, e-mail, job title, phone number, security question and answer. After these steps, Mr. Kemal should approve his membership by clicking on a link that has been sent to his e-mail account.

Then, Mr. Kemal becomes a member of Proders Freeware version. Mr. Kemal is now able to access his account in any device that has internet connection and a web-browser. Mr. Kemal uses IPhone 4S so he can use Proders in Safari browser. If Mr. Kemal forgets his password and becomes unable to register the system, he can ask for either security question or an e-mail that provides a link to create a new password. Then, he is again able to access his account on the system.

**Creating New Lectures, Classes and Assigning Teachers**

After he registered the system from his computer, Mr. Kemal can see a set of options in the left panel of the browser. By clicking ‘Program Verileri' and then 'Öğretmenler', he adds necessary information for each teachers in the school. These information include name, faculty, phone number, e-mail, social club. Mr. Kemal then adds lessons for each classes by indicating lesson name, difficulty level of the lesson, distribution of the lesson (i.e. two hours consecutively) and hours of lesson in a week. Such extra information will be used when the system generates a curriculum. After Mr. Kemal introduced lessons and teachers in school, the components of the system has been created. However, he still have some works to do. Assigning teachers to their branches (i.e. Physics, Chemistry, Biology, and History) is the next step. After Mr. Kemal assigned to teachers, finally he does not need paper work anymore. He can handle the rest from his computer.

**Arranging Breaks and Elective Lesson Hours in Class Schedules**

The next step what Mr. Kemal should do is arranging classes schedule' empty hours. By clicking 'Program Verileri' and then 'Sınıflar' Mr. Kemal can add each classes in the school (i.e. 9-A, 10-C, 12-F). When class addition process is completed, Mr. Kemal sees each classes’ button of the screen and after clicking one of these classes the system displays its schedule. In this schedule days and hours are displayed. If Mr. Kemal would like to add an afternoon break for a class he clicks the blank of corresponding day and hour in table. Table is painted red in order to mean that this hour is partitioned for the break.When he would like to choose elective lesson's hours, Mr. Kemal clicks the checkbox 'Sınıfa ait seçmeli ders saatlerini işaretle' and then clicks the hours as he does for break hours. This checkbox is placed under the Schedule.

**Adding Classrooms and Assigning Classes to Corresponding Classroom**

For some particular lessons such as Physics, Chemistry, Computer, the classrooms are different than a class' standard classroom because of using different equipment. By clicking 'Program Verileri' and then 'Derslikler' option, Mr. Kemal adds specific classrooms by indicating its student quota. After adding the classrooms, Mr. Kemal assigns the related lessons to these classrooms from 'Atama İşlemleri' options. The closed hours of these classrooms can also be specified by clicking the desired hour as Mr. Kemal does while rearranging the break hours.

**Constraints and Adjustments**

This is one of the most important part of the system. Mr. Kemal clicks 'Kısıtlamalar ve Ayarlamalar' option and the system displays all teachers, their working hours branches, minimum and maximum empty days in a week, their priority between each other’s and their individual class schedule. After meeting with teachers, Mr. Kemal adjusts their free-days and hours, weekly class hours, daily class hours. For weekly work hours, Mr. Kemal can make arrangement from the schedule table. Also Mr. Kemal is able to see any teacher's classes and classrooms in this menu.

**Creating the Schedule**

After Mr. Kemal provides all necessary information for the schedule, he clicks on 'Program Oluştur' option that placed on the left of the screen. Then the system asks Mr. Kemal to choose priority levels of parameters from 1 star to 5 stars (1= very low priority, 5 = very high priority).

These parameters are:

. Importance of teachers' weak constraints

. Importance of teacher's free day count

. Importance of opposite-timed appointment of the teachers

. Importance of teacher's daily maximum number of classes

. Importance of lessons' weak constraints

. Importance of selected lessons' appointment to early hours

. Importance of difficult lessons' scheduling consecutively

. Importance of length of the free hours of a teacher's classes in a day.

Mr. Kemal carefully evaluate each parameter and clicks on 'Program Oluştur' button. After waiting a little, the system comes up with a successful curriculum for each classes and teachers. If necessary Mr. Kemal makes some adjustment to created curriculum. Mr. Kemal also has opportunity to save, or delete the program.

**User settings, User Manual and Contact**

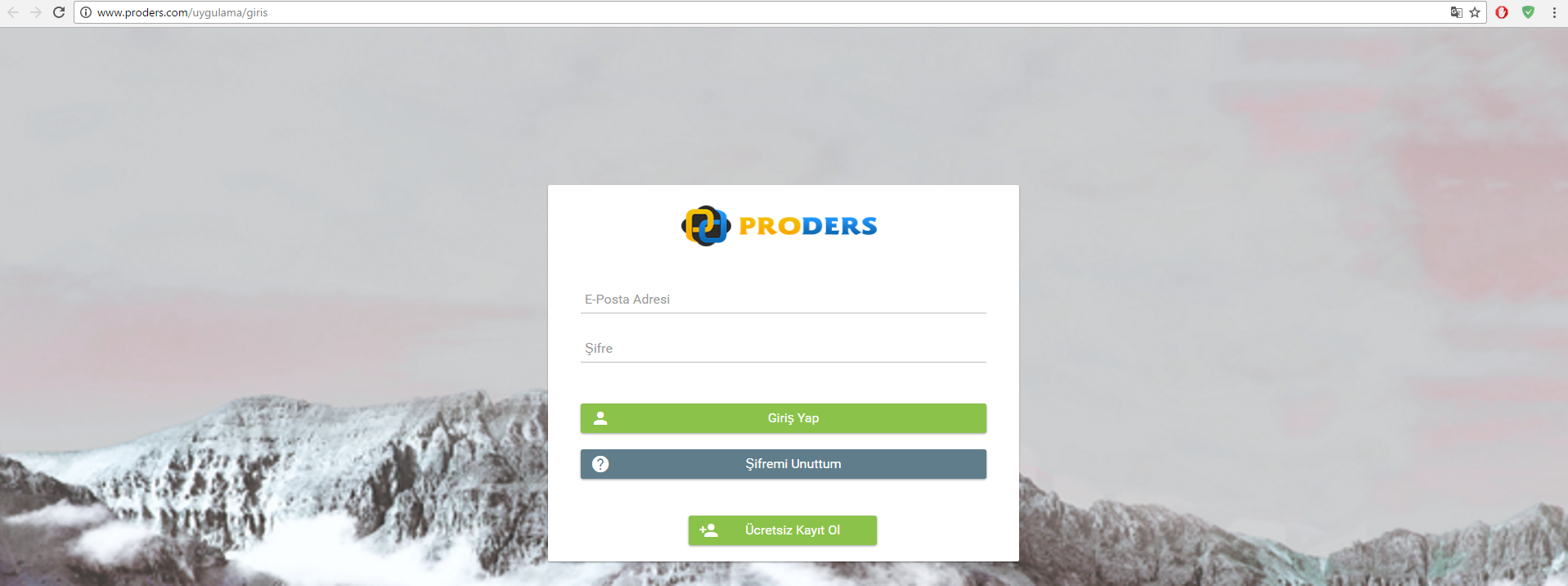
In order to use the system efficiently, Mr. Kemal can open User-Manual by clicking on the question mark shaped button which placed right top corner of the screen. Each steps are explained in detail there.

Should Mr. Kemal does not understand something or when an error happened, Mr. Kemal can make contact with the maintainer of the system by acquiring his phone number by clicking the telephone handset shaped button which placed right top corner of the screen. This phone number is partitioned for the users to make contact in office hours.

Again at right top corner of the screen, there is an owl image. By clicking on this image Mr. Kemal can change his personal information and password that recorded in the system. Also, Mr. Kemal logs out from the system by clicking 'Log out' at the end of the day.

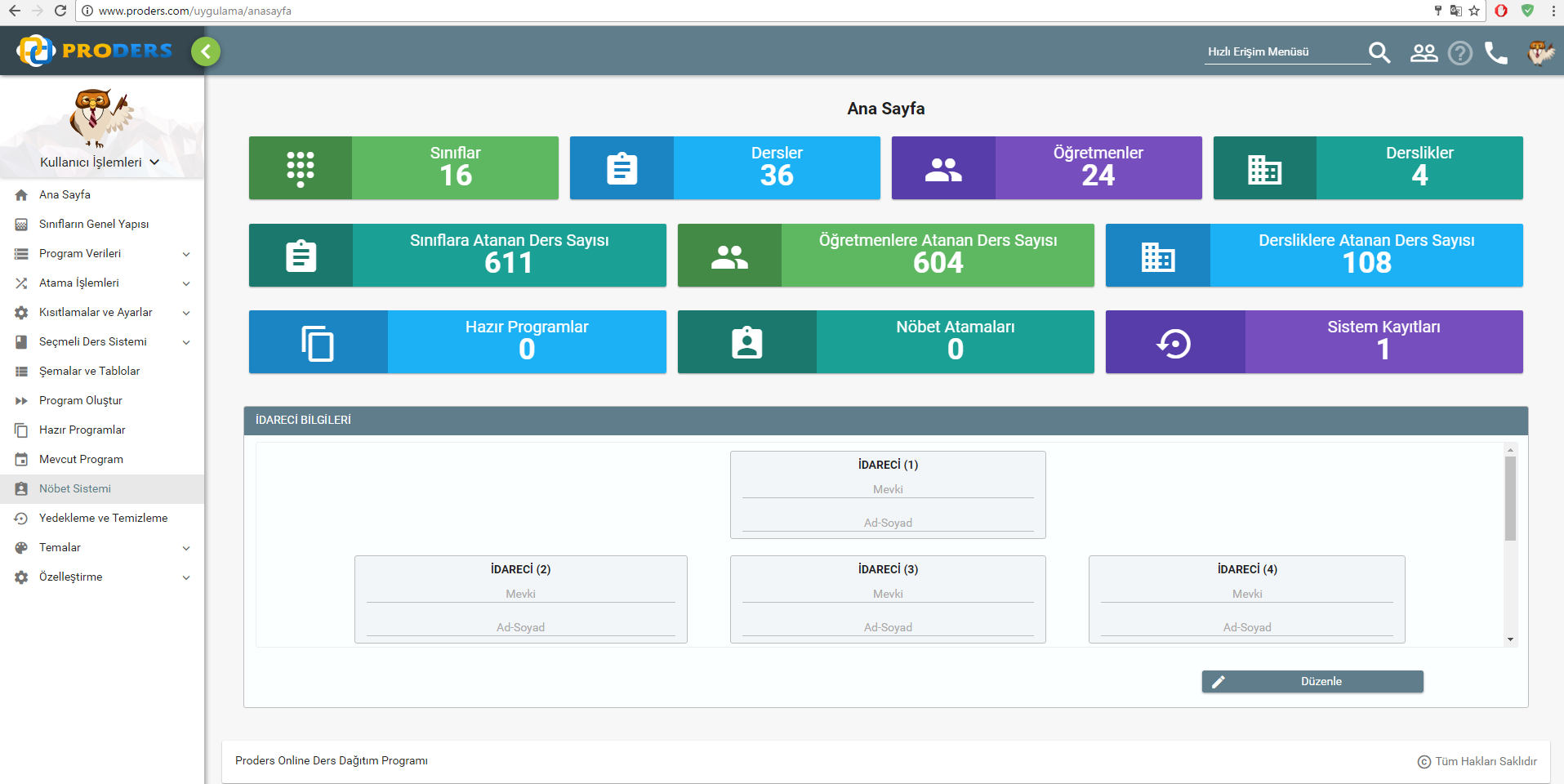
**4.6. USER INTERFACE PROTOTYPES**

Important parts of the graphical user interface of the system are shown below

****

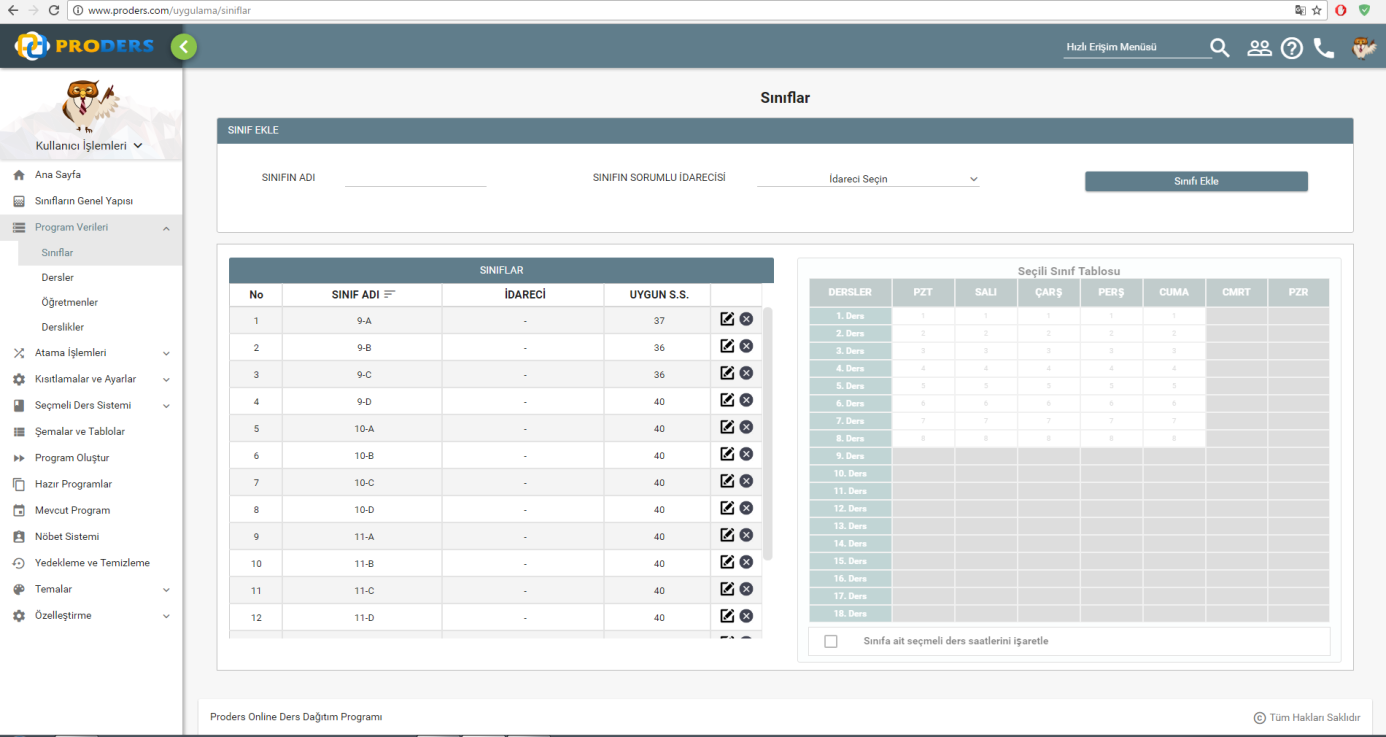
**Figure 1: Login Page**

The user gains access the system by entering e-mail address and password that he/she provided when registering the system. If the user does not have account, it is also possible to signing up an account in this panel.

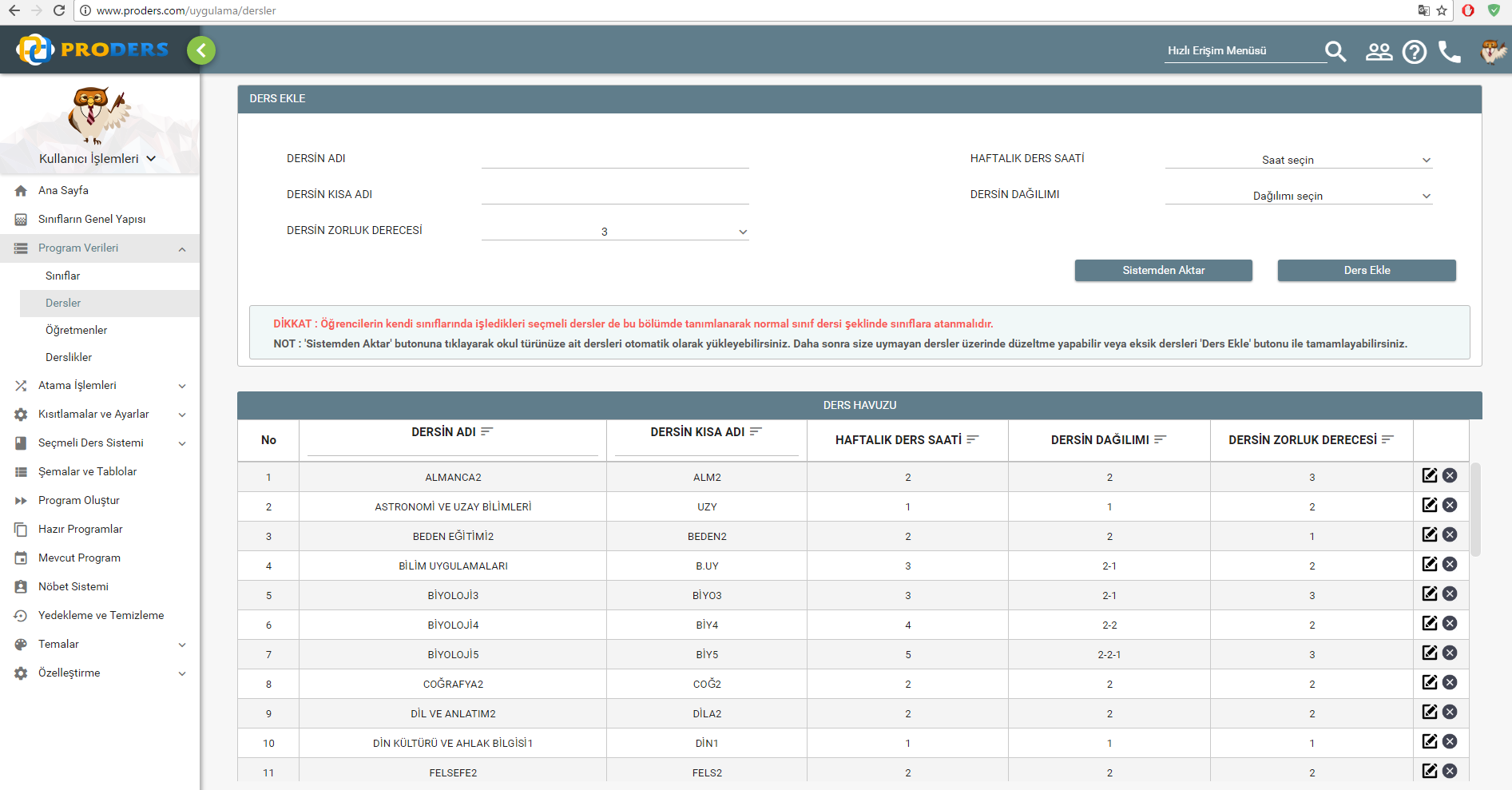


**Figure 2: Main Menu**

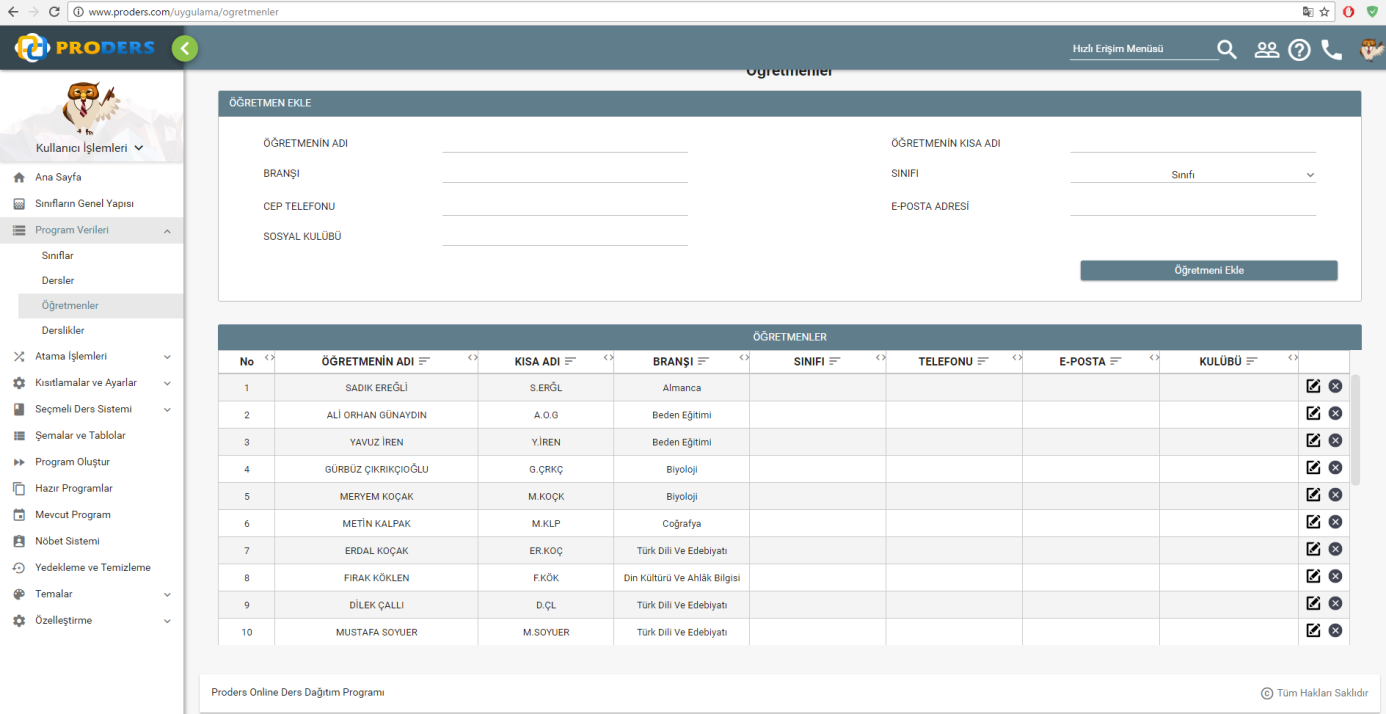
In main menu screen, the user can see his options to do, information about his works and some schemas related personals of his/her school. Also searching, user-manual, making contact, account adjustments are done in this screen.



**Figure 3: Adding Class Information**

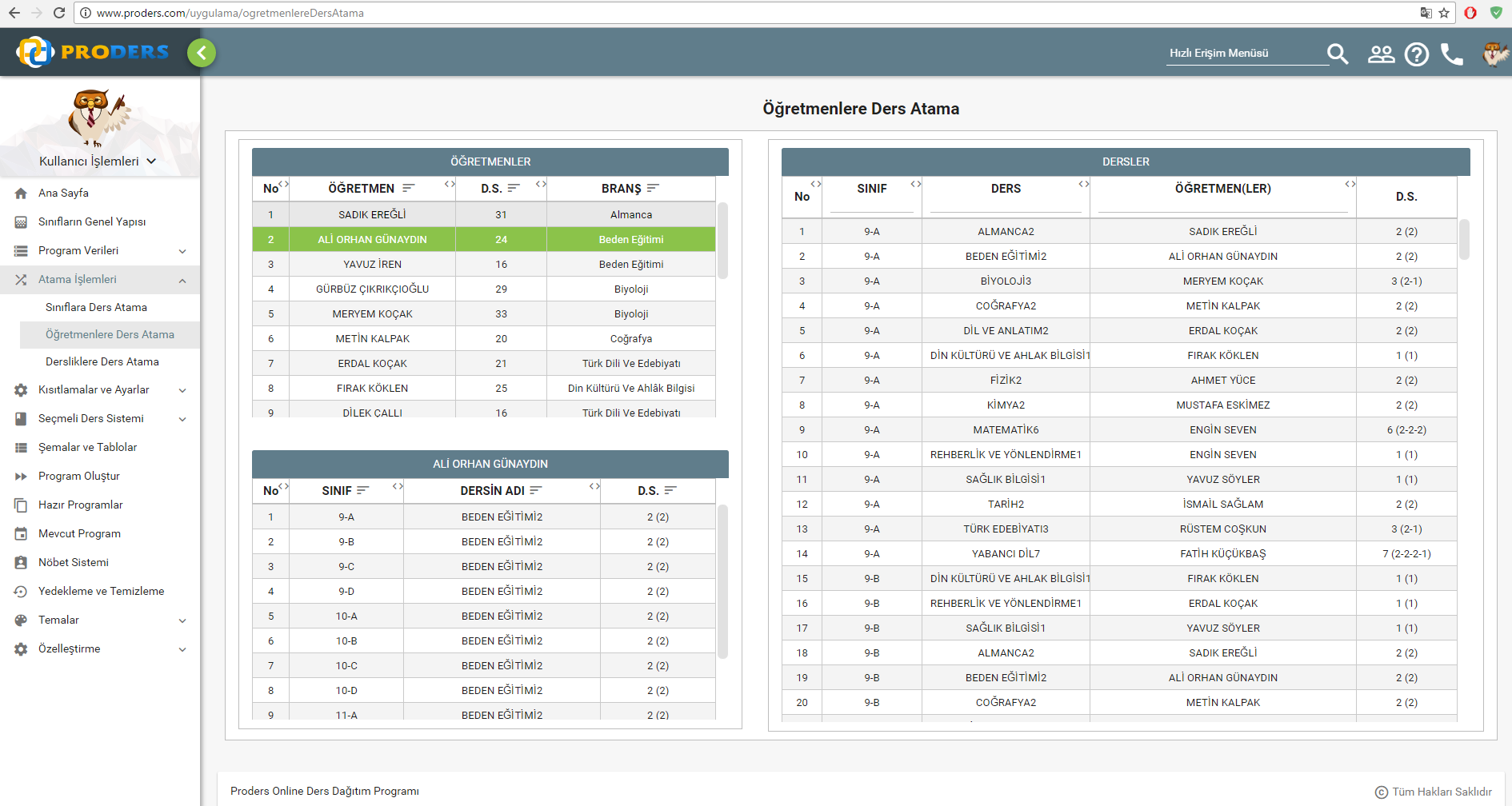
****

**Figure 4: Adding Lesson Information**

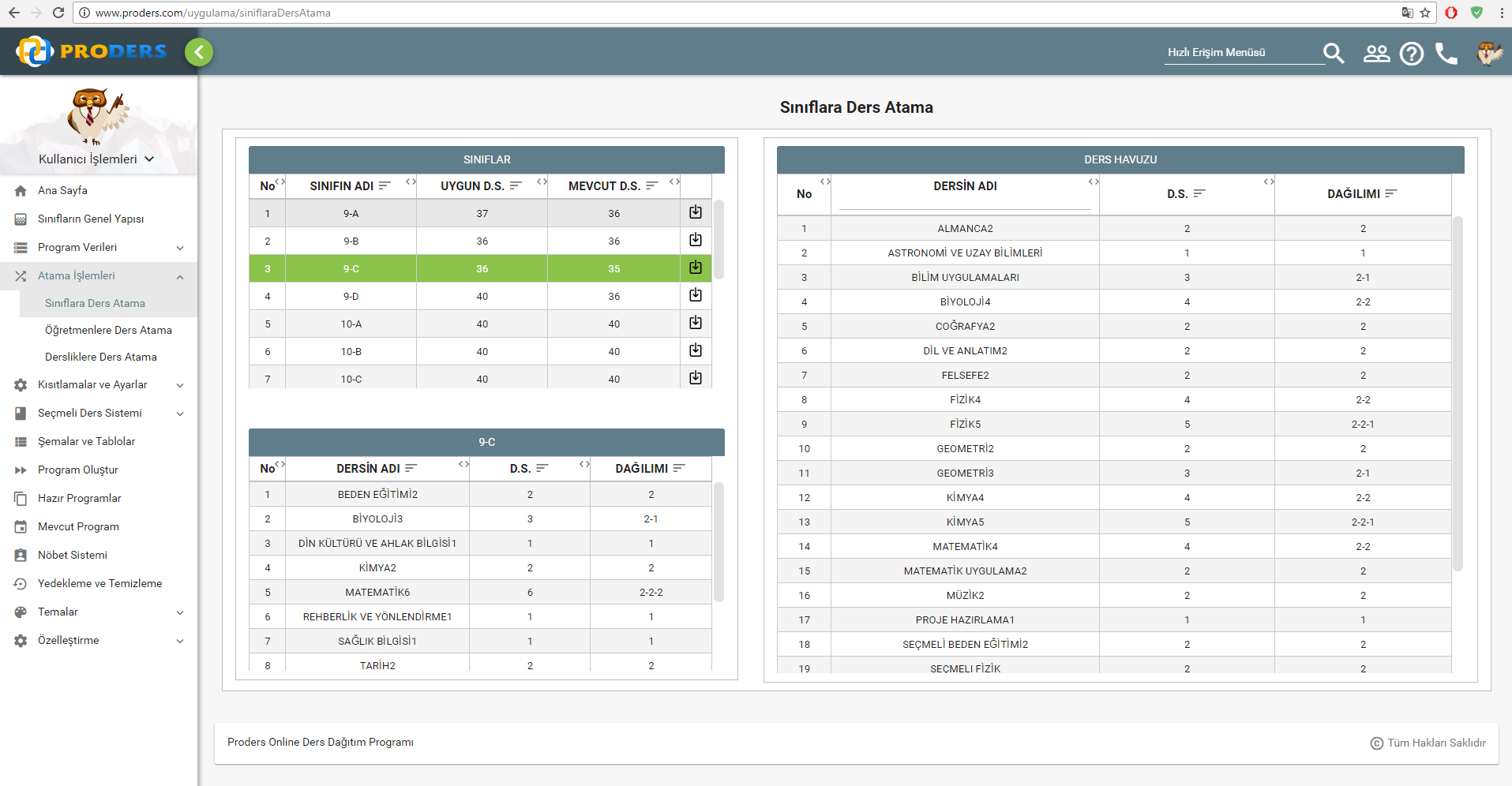
****

**Figure 5: Adding Teacher Information**

As shown in Figure 3, Figure 4, and Figure 5 the user provides necessary information for creating a curriculum. Also, the user can edit the any information from these panels about teachers, classes, and lessons.

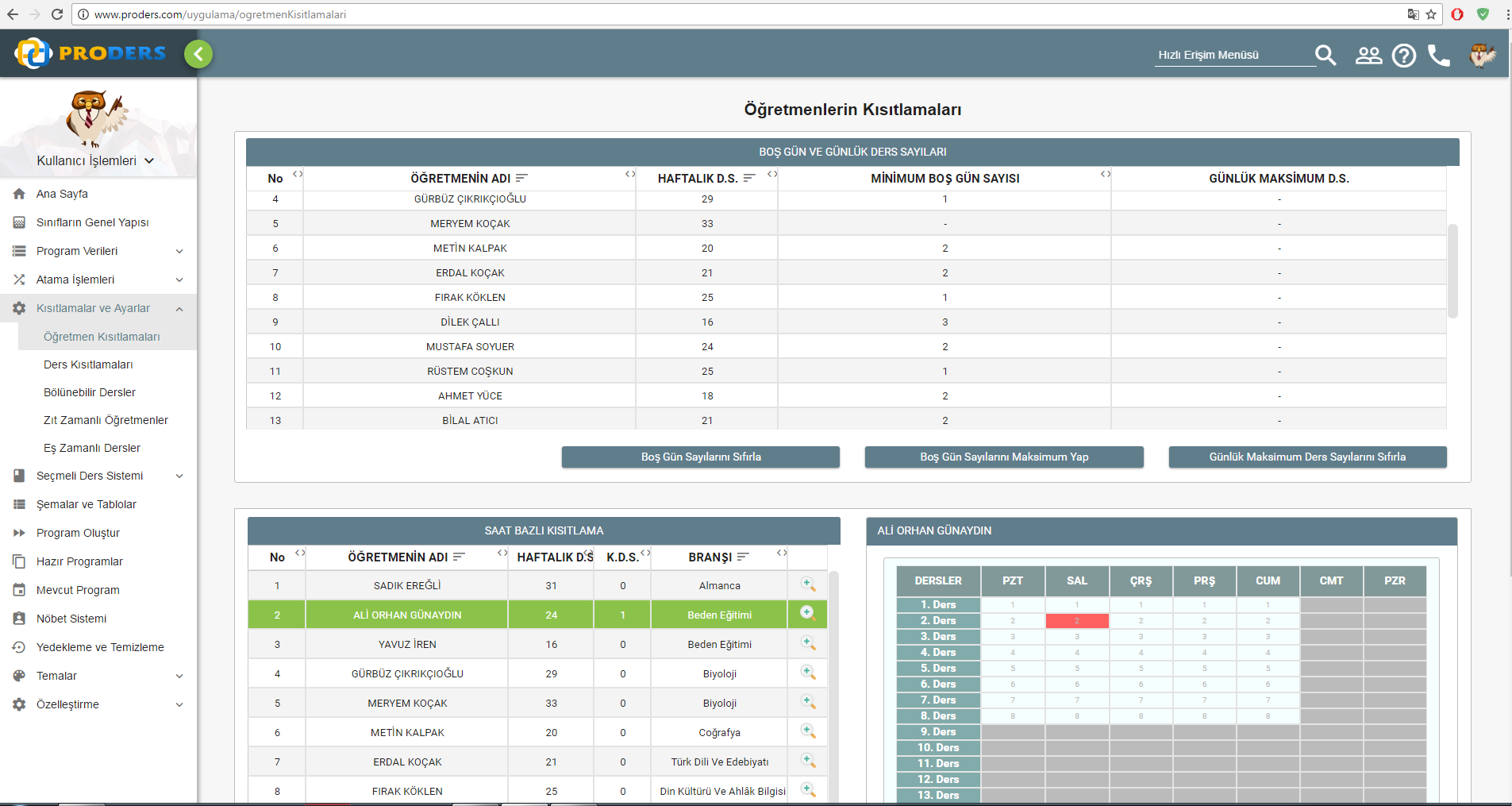


**Figure 6: Assigning teachers to their branches**

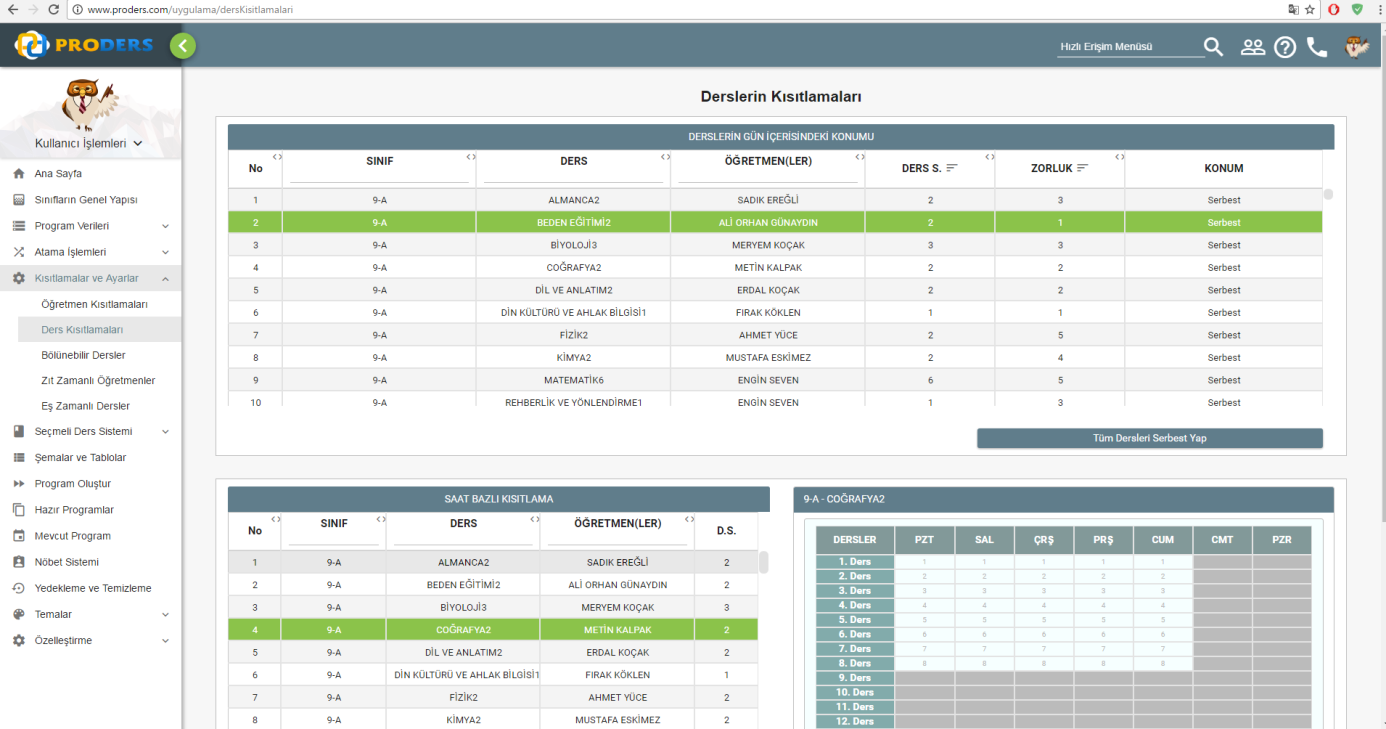
****

**Figure 7: Assigning lessons to classrooms**

As shown in Figure 6, and Figure 7, the user assigns each teacher to their branches and each lessons to classes.



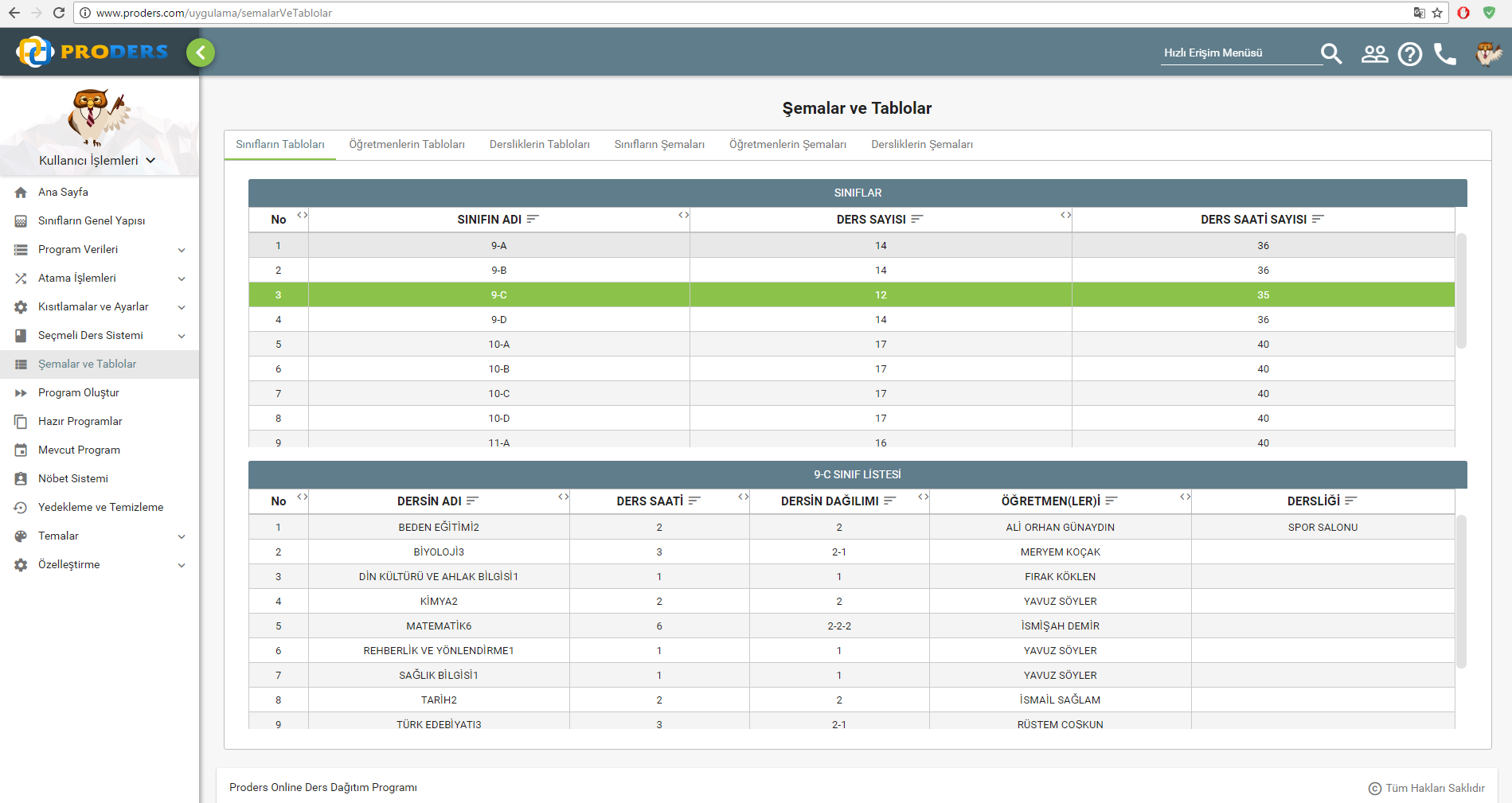
**Figure 8: Teacher Constraints**

****

**Figure 9: Lesson Constraints**

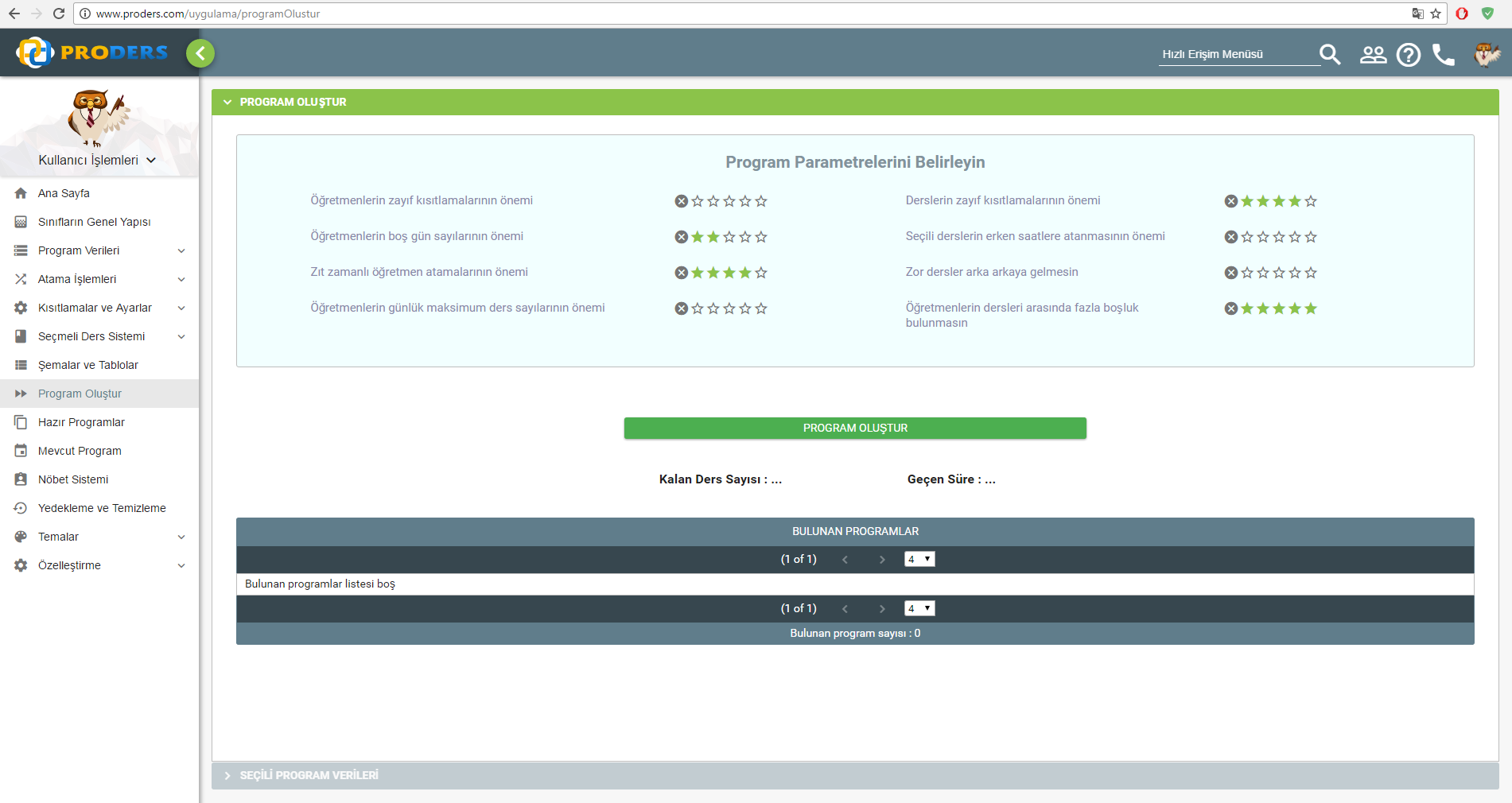
In Figure 8, the user decides any teacher's free days, maximum and minimum free days, weekly working hours and free hours from the table.

In Figure 9, the user adjusts lessons with respect to difficulty level, assigned teachers, and weekly hour of the lesson and its assigned time such as free, early, late.



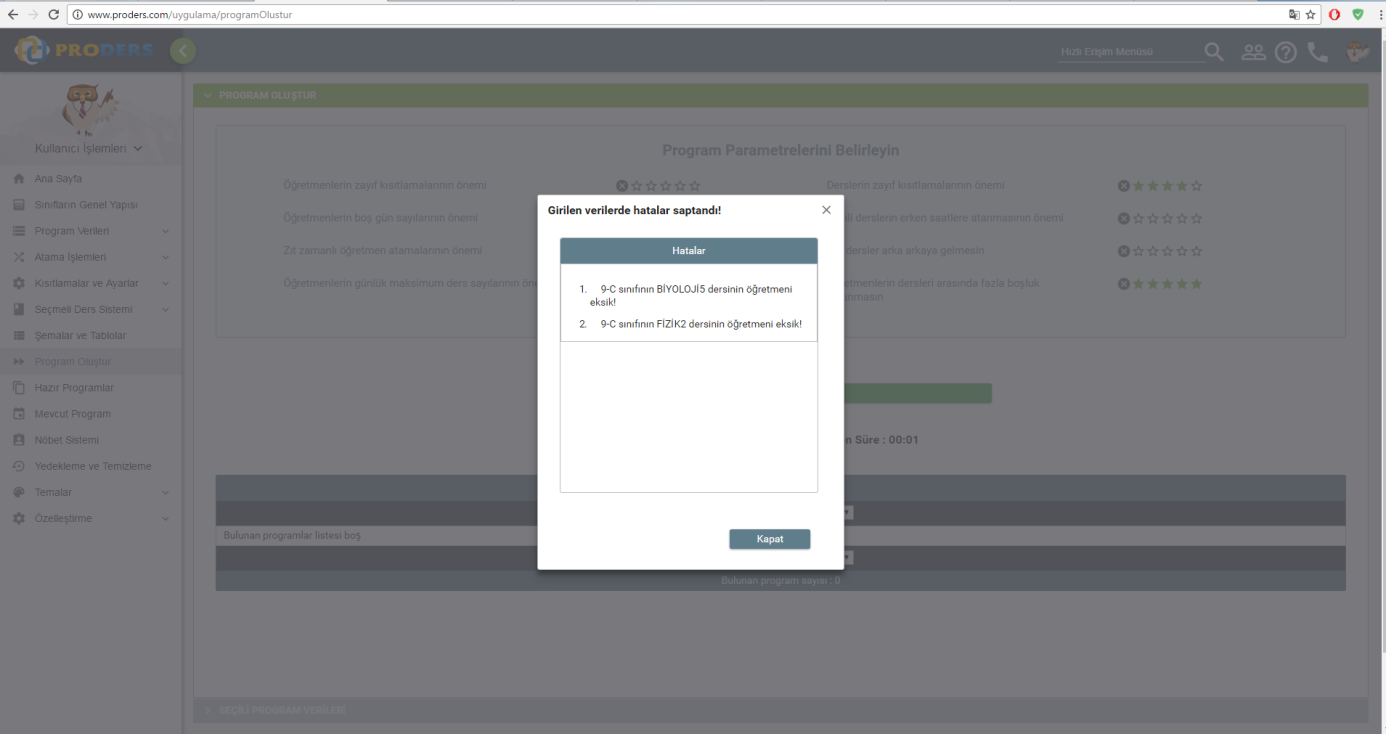
**Figure 10: Schemas and Tables of Classes**

In this panel, the user can display every information about classes, classrooms, and teachers. This part can be useful for printing stuffs and checking the information whether there is any error or not.



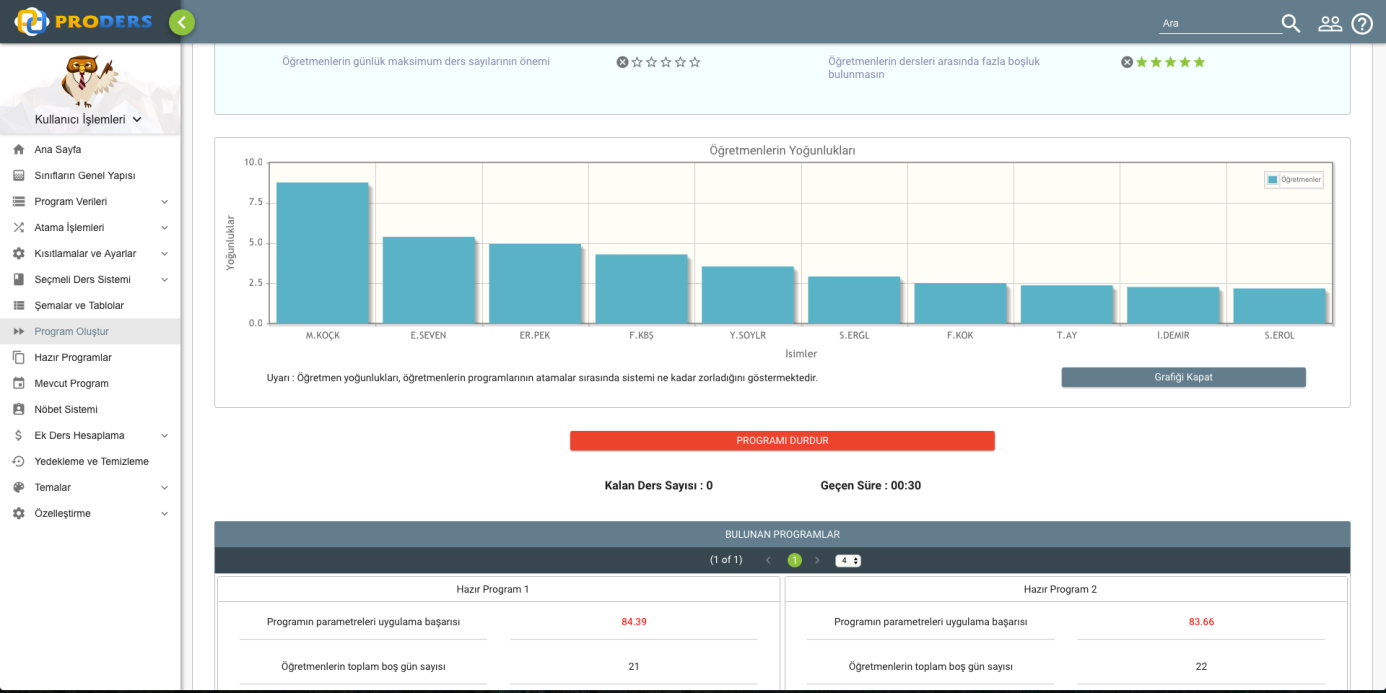
**Figure 11: Creating the program**

After providing all information about classes, classrooms, teachers, and lessons, the user chooses the parameter and their priority and clicks on 'Program Oluştur' button.

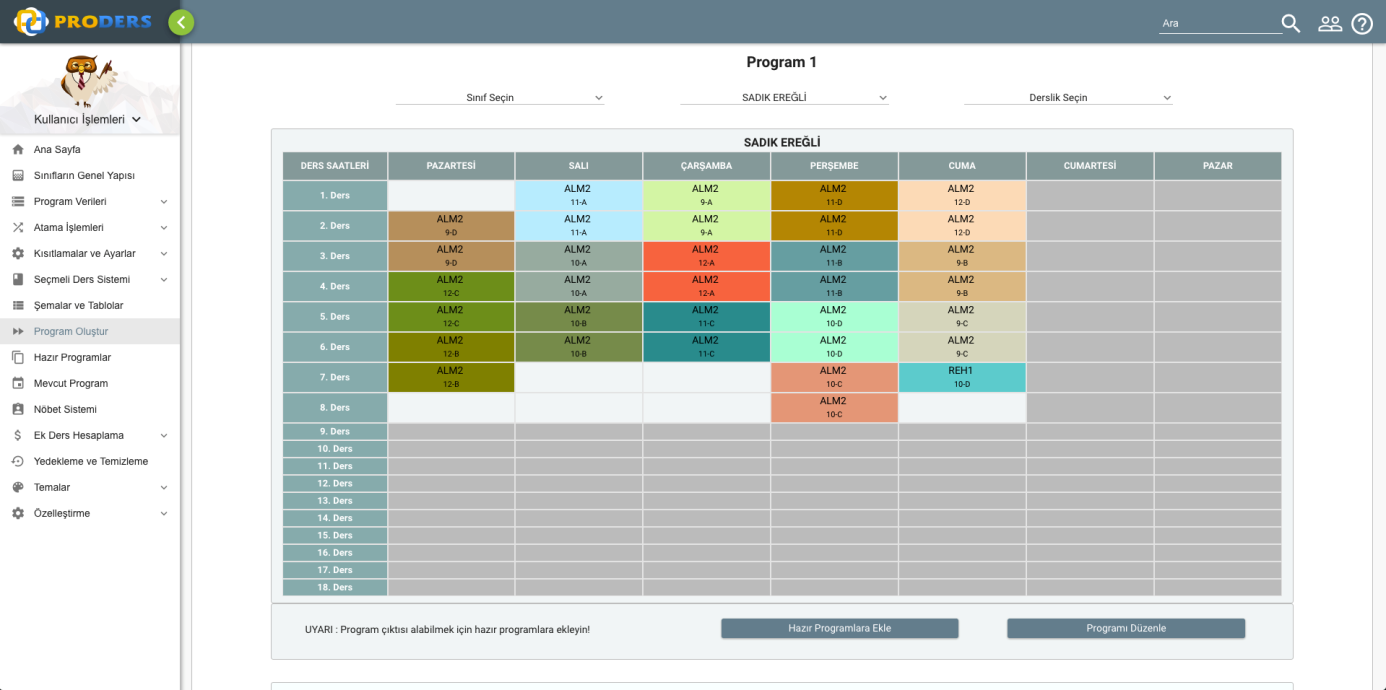


**Figure 12: Error while creating the Program**

If any inconvenient situation happens such as deficient information, the system does not create a program and displays an error message as seen in Figure 12.



**Figure 13: Program is being created**

****

**Figure 14: The Program**

In figure 14 program has been created successfully and shown in the table. If user does not satisfy he can edit the program.

1. **TECHNICAL PROBLEM ANALYSIS**

**5.1. COMPUTER SCIENCE PROBLEMS**

**Question**: How to design a distributed system?

**Current State**: Scheduling and optimization operations works in another application. By this means more than one CPU can be used for the core algorithm of the program. Many programs performing scheduling algorithm runs simultaneously and creates plenty of different course scheduling. However each algorithm program runs individualistically and holds just own data and analysis. Each algorithm program sends the best distributed course program that it found. Main application selects the best programs sent by those algorithm programs.

**Desired State**: The main system should be able to inform other algorithm programs when an algorithm program find a better solution according to others. This architecture decreases cost of the all system and more importantly time cost of main application.

**Question**: How to store the data?

**Current State**: Current system uses both file system and SQL to hold program data. SQL is used just for user list. File system is used to hold all information of user in his account. Application data of a user is hold as serialized objects in user database files. With this method database operations are not affected by database size.

**Desired State**: All database system must use SQL to hold all application data. This increase maintainability and scalability of the system. Time costs of database operations must be optimized with a layered architecture.

**Question**: How to recover damaged database?

**Current State**: Current system uses file backup periodically. When a user file is damaged, administrator of the system can change file with its backup manually.

**Desired State**: System must use SQL and backup all database on a regular basis. When some information damaged, system must be restore data from backup automatically.

**5.2. MATHEMATICS DOMAIN PROBLEMS**

**Question**: How to handle the optimization of cost of the project?

**Current State**: The cost of existing project is expensive because of insufficient input analysis system.

**Desired State**: The system will check user input comprehensively in order to prevent restriction overlaps. Some constraint can prevent finding complete and feasible solutions. This constraints or wrong inputs must be analyzed before running distribution algorithm and inform user in any favorableness.

**5.3. APPLICATION DOMAIN PROBLEMS**

**Question**: How to make modifications on an account by multiple users simultaneously?

**Current State**: Current system does not support multiple users simultaneously. Because of some architectural reasons single user principle is implemented in the system.

**Desired State**: The system must be redesigned for usage of multiple user at the same time.

**Question**: How to handle authorization/permission?

**Current State**: Only school administration can access the system at this moment.

**Desired State**: Instructors should be able to access their own course schedules and enter their own constraints to system.

**Question**: How to reach the systems which are backed up?

**Current State**: Current system allows user to reload any backed up system. However system does not allow partial reloading from backed up systems.

**Desired State**: System should be able to provide partial reloading from backup. For example user should be able to get only teacher list from backup.

**5.4. QUALITY PROBLEMS**

**Question**: How to create a high performance system?

**Current State**: The system creates same number of algorithm thread for every user at any time.

**Desired State**: The system must be able to analyze user system and load on the CPU and create algorithm threads according to current requirements. If there is no so much load on CPU, system must create more algorithm thread for course distribution of the user.

**Question**: How to create a user-friendly system?

**Current State**: Current system is a little bit difficult to understand due to its complex user interface design. They do not fully provide usefulness and easiness for its users.

**Desired State**: The system may be used by the people who do not know using technological devices very well. Therefore, the system should be easy to use. Its user interface gadgets should be clear, simple and understandable.

**Question**: How to provide robustness for system?

**Current State**: Current systems have some minor errors.

**Desired State**: The system should be error-free to provide robustness and it should be tested many times before being put on the market.

**6. DOMAIN ANALYSIS**

**Domain Identification**

The solution area has been determined to solve the problems for technical problems. In this step, each field element is derived from technical problems.

The project contains the following domains:

User Interface

Performance

Reliability

Concurrency

Availability

Maintainability

Data Storage

Authorization

Scheduling

Solution Improvement

Analysis of distribution

|  | | |
| --- | --- | --- |
| how to solve problem | solution domain | priority |
| providing user-friendly interface | User Interface | 2 |
| making system faster | Performance | 8 |
| making system more reliable | Reliability | 1 |
| providing many services for multiple users | Concurrency | 1 |
| it can be accessed easily | Availability | 1 |
| making system maintenance during updating | Maintainability | 1 |
| storing user info | Data Storage | 1 |
| giving access rights to the user and admin | Authorization | 1 |
| making scheduled time tables | Scheduling | 7 |
| finding more feasible solutions | Solution Improvement | 7 |
| helping user to improve current table | Analysis of distribution | 7 |

Knowledge Sources Identification

|  |  |  |
| --- | --- | --- |
| Overall Knowledge Source |  |  |
| **ID** | **Knowledge Source** | **Form** |
| A1 | Constraint-based  Timetabling (T. Müller, Ph.D. Thesis, KTIML MFF UK, Prague, 2005) | article |
| A2 | Interactive Timetabling  (T. Müller, Master Thesis, KTIML  MFF UK, Prague, September 2001) | arthical |
| A3 | Java Distributed Computing  by Jim Faber, 1998 | arthical |
| A4 | Thomas Müller | person |
| A5 | primefaces | E-book |
| A6 | Çağatay Çivici | person |
| B1 | Chapple, Mike (2005)  "SQL Fundamentals" | book |
| B2 | Richard A. Henle, Boris  W. Kuvshinoff, C. M. Kuvshinoff  (1992) | book |
| B3 | Ahmet Ay | person |
| C1 | John lewis and William  Loftus Software Solutionns,  Addison-Wesley, 2014 | book |
| C2 | Paul Deitel, Harvey Deitel,  Java, How to Program,  Addison-Welley, 2012 | book |
| C3 | Java Programming, Derek Banas | E-book |

|  |  |  |
| --- | --- | --- |
| Scheduling Algorithm  Knowledge Source |  |  |
| **ID** | **Knowledge Source** | **Form** |
| A1 | Constraint-based  Timetabling (T. Müller, Ph.D. Thesis, KTIML MFF UK, Prague, 2005) | article |
| A2 | Interactive Timetabling  (T. Müller, Master Thesis, KTIML  MFF UK, Prague, September 2001) | arthical |
| A3 | Java Distributed Computing  by Jim Faber, 1998 | arthical |
| A4 | Thomas Müller | person |

|  |  |  |
| --- | --- | --- |
| Database/Server Knowledge Source |  |  |
| **ID** | **Knowledge Source** | **Form** |
| B1 | Chapple, Mike (2005)  "SQL Fundamentals" | book |
| B2 | Richard A. Henle, Boris  W. Kuvshinoff, C. M. Kuvshinoff  (1992) | book |
| B3 | Ahmet Ay | person |

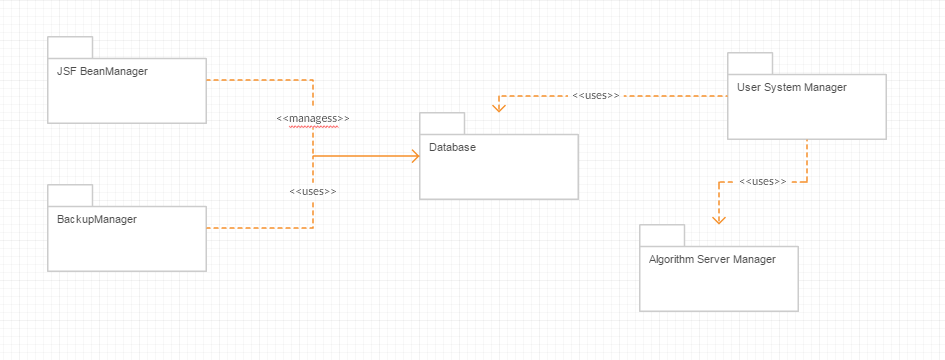
|  |  |  |
| --- | --- | --- |
| Java Knowledge Source |  |  |
| **ID** | **Knowledge Source** | **Form** |
| C1 | John lewis and William  Loftus Software Solutionns,  Addison-Wesley, 2014 | book |
| C2 | Paul Deitel, Harvey Deitel,  Java, How to Program,  Addison-Welley, 2012 | book |
| C3 | Java Programming, Derek Banas | E-book |

|  |  |  |
| --- | --- | --- |
| Presentation Layer Knowledge Source |  |  |
| **ID** | **Knowledge Source** | **Form** |
| A5 | primefaces | E-book |
| A6 | primeTek | company |

Concepts

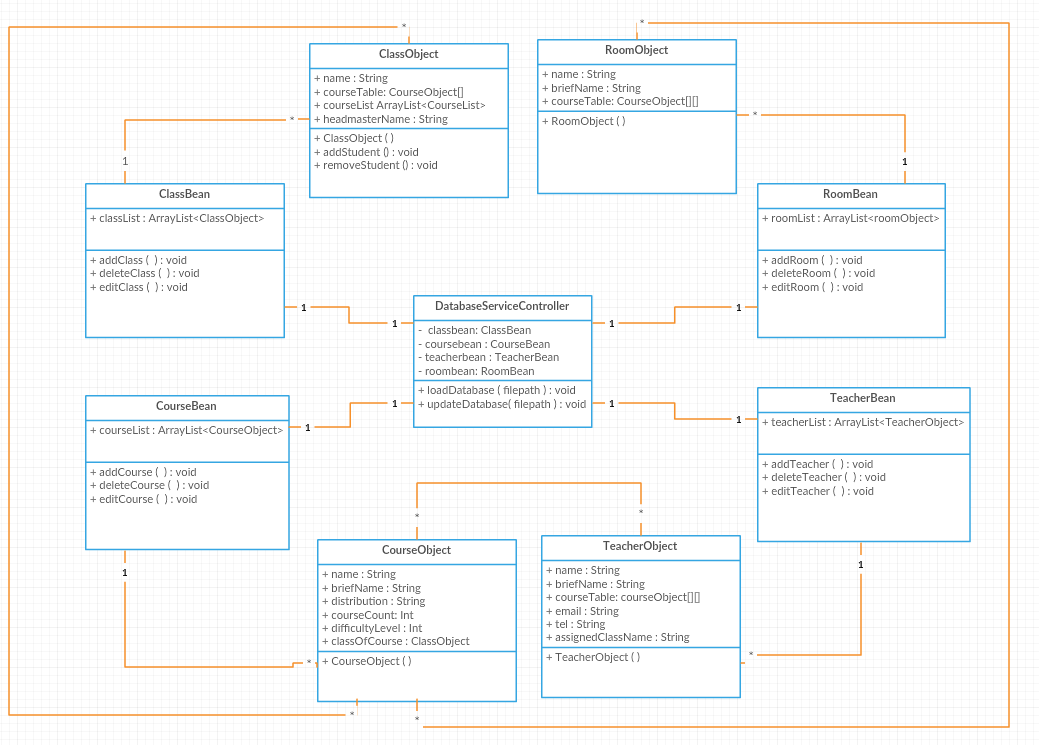
| Table 1 | |
| --- | --- |
| solution domain | solution domain concepts |
| User Interface | primefaces component |
| Performance | distributed systems |
| Reliability | backup systems |
| Concurrency | JSF Bean manager |
| Availability | glassfish server |
| Maintainability | glassfish server |
| Data Storage | data manager |
| Authorization | authorization manager |
| Scheduling | genetic algorithm |
| Solution Improvement | hill climbing algorithm |
| Analysis of distribution | constraint based statistics |

Structure of Concepts



|  |  |
| --- | --- |
| Concept | Description of the Solution Domain Concept |
| JSF BeanManager | responsible for creting and destroying ManagedBeans |
| BackupManager | responsible for backup of user data |
| Database | responsible for storing user data in file |
| User System Manager | responsible for creating user schedules |
| Algorithm Server Manager | Interact with distributed algorithm servers |

Domain Model



**REFERENCES**