ივანე ჯავახიშვილის სახელობის თბილისის სახელმწიფო უნივერსიტეტი

ზუსტ და საბუნებისმეტყველო მეცნიერებათა ფაკულტეტი

სტუდენტის ინდივიდუალური ცხრილის ფორმირების სისტემა

პროექტის შემსრულებელი სტუდენტები

- მეგრელი გიორგი
- მინდორაშვილი ირაკლი
- მჭედლიშვილი სანდრო
- პაპავა ირაკლი

ხელმძღვანელი: მანანა ხაჩიძე ტექნიკის მეცნიერებათა კანდიდატი

თბილისი

2014 წ.

შინაარსი

ანოტაცია

პროექტის მიზანია შეიქმნას პროგრამული სისტემა ინტეგრირებადი აპრიკაციის სახით, რომელიც სტუდენტს მაქსიმალურად მისაღები სასწავლო ცხრილის შერჩევაში დაეხმარება. მომხმარებლეს (სტუდენტს) საშუალებას აძლევს მისთვის პრიორიტეტული საგნების, დღეების და საათების შერჩევის შემდეგ შესთავაზოს ცხრილის სხვადასხვა ვარიანტი მეცადინეობის ფორმატის მითითებით (საერთო ლექცია, პრაქტიკული მუშაობა, ლაბორატორიული მუშაობა და ა.შ.).

აპლიკაციის ფუნქციონირების მიზანია მომხმარებელმა მიიღოს ოპტიმალურთან მაქსიმალურად მიახლოვებული შედეგი, თუმცა ეს უმეტეს შემთხვვაში ვერ მოხერხდება, რადგან ალგორითმი დამოკიდებულია რამდენიმე შეზღუდვაზე.

ეს შეზღუდვებია:

- თითოეულ საგანზე დაშვებული სტუდანტთა რაოდენობა.
- მაღალ პრიორიტეტულ დროის მონაკვეთებზე საგნების არარსებობა.
- სავალდებულო საგნებზე დამოკიდებულება.
- საგნების ასაღებად არსებული წინაპირობები.

ამ ყველაფრის გათვალისწინებით, იმ საგნების რაოდენობა, რომლებიც მომხმარებელს შეუძლია რომ აირჩიოს საკმაოდ მცირდება, რაც თავისთავად ამცირებს ოპტიმალური შედეგის მიღების შანსს.

აპლიკაცია წარმოადგენს პროექტის ავტორების მიერ მოდიფიცირებულ ევრისტიკულ ალგორითმს, რომელიც რეალიზებილია PHP ენაზე, ორიგინალურია და იყენებს მხოლოდ PHP - ის სტანდარტული ფუნქციებს.

Annotation

The goal of this project is to let students generate near optimal university schedule for them. User should be able to specify high priority days, hours and subjects and application with generate schedule for him/her.

Our goals was to offer user near optimal, while we would like to be able to offer optimal results, it is impossible in most cases, because application has to work with number of constraints.

These constraints include:

- Maximum number of students on each subject.
- Absence of subjects during high priority hours.
- Dependence on mandatory subjects.
- Prerequisites of individual subjects.

Because of all these, number of subjects that are available to individual student, decreases drastically, which in turn decreases chances of getting optimal results.

Our biggest priority was creating system that would be comfortable and easy to interact with for users.

Application is heuristic algorithm, modified by project authors, written in PHP.

შესავალი

საქმიანობის დაგეგმვა ნებისმიერი ბიზნესის წარმატების საწინდარია. თუ ჩავთვლით, რომ სტუდენტისათვის სწავლა ერთგვარი საქმიანობაა, მისი რაციონალური დაგეგმვა მეტად მნიშვნელოვანია. მიუხედავად იმისა, რომ მრავლად არსებობენ სახვადასხვა ინფორმაციული სისტემები, რომლების ბიზნესის სხვადასხვა სფეროს მენეჯმენტში წარმატებით გამოიყენება (მათ შორის სასწავლო პროცესში), ყოველი უმაღლესი სასწავლებელი განსხვავდება სასწავლო პროცესის რეგულირების წესით და ამის გამო არის რიგი საკითხები, რომლებიც ცალკეულ მიდგომას მოითხოვს. ერთერთ ასეთ პროცესად შეიძლება ჩაითვალოს ივ.ჯავახიშვილის თბილისის სახელმწიფო უნივერსიტეტში სტუდენტისათვის ინდივიდუალური ცხრილის ფორმირება.

ყოველი სტუდენტისთვის, სემესტრის დასაწყისში, ძალიან მნიშვნელოვანია ისეთი საგნების შერჩევა რომლებიც მისთვის აუცილებელია, საინტერესოა და ამასთან ერთად საშუალებას აძლევს მას რაციონალურად გაანაწილოს დრო მისთვის მორგებული განრიგით. თუმცა ამ ამოცანის გადაჭრა ხშირად დიდ პრობლემას წარმოადგენს. ჩვენი მიზანია სტუდენტს მივცეთ საშუალება მაუსის ღილაკზე რემდენიმე დაჭერით გადალახოს ეს წინაღობა.

ამ პრობლემის გადასაჭრელად დაიგეგმა პროგრამული აპლიკაციის შექმნა ვებ გვერდის სახით, რომელზეც რეგისტრაცია შეუძლია გაიაროს ნებისმიერმა თსუ–ს სტუდენტმა და მინიმალურ დროში მიიღოს მისთვის მისაღები სასწავლო განრიგი. სისტემის ეს ვერსია ჯერჯერობით მუშაობს მხოლოდ თსუ–ს ზუსტ და საბუნებისმეტყველო მეცნიერებათა ფაკულტეტის სტუდენტებისთვის.

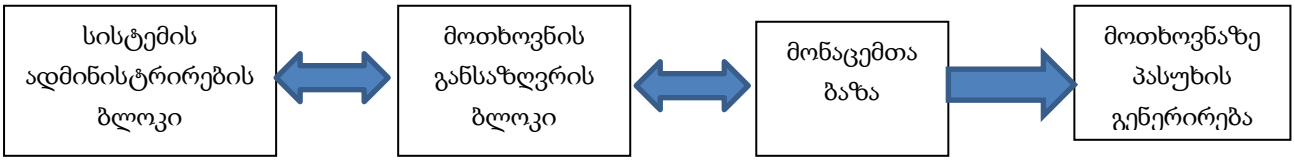
მნიშვნელოვანია ვებ-გვერდი იყოს მაქსიმალურად კომფორტული და ადვილად გამოსაყენებელი. მომხმარებელმა საიტზე შემოსვლის მომენტიდან უნდა იცოდეს თუ რა გააკეთოს მისთვის საჭირო შედეგის მისაღებად, იყოს მაქსიმალურად ინტერაქტიული და ადვილად აღქმადი. საიტი ადეკვატურად უნდა უმკლავდებოდეს მომხმარებლის ქმედებებს და აძლევდეს მას ინფორმაციას, თუ რისი გაკეთება შეუძლია ამ სისტემის გამოყენებით. ამისთვის საჭიროა, ერთად მომუშავე რამდენიმე კომპონენტი, რომლებიც მთლიანობაში გვაძლევს მოქნილ სისტემას, რომელიც გაუმკლავდება ნებისმიერ შემთხვევას.

ეს კომპონენტებია:

- მონაცემთა ბაზა.
- სისტემის სერვერის მხარე, რომელიც პასუხისმგებელია სტუდენტების სასწავლო განრიგების შედგენაზე.
- სისტემის კლიენტის მხარე, რომელიც პასუხისმგებელია, ვებგვერდის შეხედულობაზე,
- ის უნდა იყოს მაქსიმალურად კომფორტული და მიმზიდველი.

ინდივიდუალური ცხრილის ფორმირების სისტემის ლოგიკური სქემა

ნებისმიერი პროგრამული სისტემის რეალიზაციისათვის უპირველეს ყოვლისა უნდა განისაზღვროს ის ძირიტადი ლოგიკური ბლოკები რომლებიც უზრუნველყოფენ სისტემისადმი წაყენებული მოთხოვნების რეალიზებას. გამომდინარე დასმული ამოცანიდან პირობითად სისტემის ლოგიკური სქემა შეიძლება წარმოვადგინოთ შემდეგი სახით:



სერვერის მხარე

მონაცემთა ბაზა

ავღწეროთ მონაცემთა ბაზა რომელსაც იყენებს საიტი.

პირველი ცხრილი 'users' (სურათი 1), შეიცავს ინფორმაციას დარეგისტრირებულ მომხმარებლემზე.

სურათი 1

P_ID	username	password	salt	email	name	joined	groups
20001067222	user1	5215a02e7b8784c208c473d1(	22b0c77727f06a7d5fc23b11874c51e6b6fd3b3c6	san6090@gmail.com	sandro	2014-06-23 01:36:39	1
20004567123	user2	9435458c0fbe82dc30006f2db2	1ee20db2457e187686e2c19558e796745b3adf97	bbbbbb@gmail.com	GIORGI	2014-06-23 21:05:33	1
20001067223	user3	cd05913bbb5a41c8fbed573c4!	e908aea4e9b8bc5d502167f9d986a1e27029a0a5	san60290@gmail.com	Sandro	2014-06-23 12:41:57	1

- P_ID წარმოადგენს მომხმარებლის პირად ნომერს და სისტემა მას იყენებს როგორც მომხმარებლის მტავარი იდენტიფიკატორი.
- username მომხმარებლის სახელი საიტზე.
- password მომხმარებლის ჰეშირებული პაროლი.
- salt გამოიყენება პაროლის ჰეშირებისთვის, დამატებითი უსაფრთხოებისთვის.
- email მომხმარებლის ელექტრონული მისამართი.
- joined დარეგისტრირების თარიღი.
- groups მომხმარებლის ჯგუფის ნომერი (ამ დროისთვის არსებობს ორი ჯგუფი, 'ჩვეულებრივი მომხმარებელი' და 'ადმინისტრატორი')

შემდეგი ცხრილი 'subjects', შეიცავს ინფორმაციას საგნების შესახებ:

სურათი 2

subject_ID	name	krediti	cinapiroba	semestri	rec_semestri	savaldebulo
10	დაპროგრამების საფუძვლები	5	NULL	-1	1	1
11	წრფივი ალგებრა და ანალიზური გეომეტრია	5	NULL	-1	1	1
12	ალგორითმები და მონაცემთა სტრუქტურები	10	10	1	2	1
13	ობიექტზე ორიენტირებული დაპროგრამება 1 (C++)	10	10	1	2	1
14	კალკულუსი კომპიუტერული მეცნიერებისათვის	5	3,11	1	2	1
15	დისკრეტული სტრუქტურები	5	11	-1	3	1
16	რიცხვითი მეთოდები: ალგორითმების შექმნა, ანალიზი და...	5	3,10,11	-1	3	1
17	ალგორითმების აგება	5	12	-1	3	1
18	ობიექტზე ორიენტირებული დაპროგრამება 2 (Java)	5	1,13	0	3,4,5,6	1

- subject_ID საგნის ID, სისტემა იყენებს მას როგორც საგნის მთავარ იდენტიფიკატორს.
- name საგნის სახელი.
- krediti საგნის კრედიტი
- cinapiroba საგნის ცინაპირობების ID–ები
- semestri ინფორმაცია თუ რომელ სემესტრში ისწავლება საგანი.(–1 შემოდგომის სემესტრი, 1 გაზაფხულის სემესტრი, ორივე = 0)
- rec_semestri რეკომენდირებული სემესტრ(ებ)ი
- savaldebulo არის თუ არა საგანი სავალდებულო.

შემდეგი მნიშვნელოვანი ცხრილია timeslot: (სურათი 3), ინახავს ინფორმაციას საგნების ინდივიდუალურ ჯგუფებზე.

სურათი 3

ID	subject_ID	type	groups	time	room	day
1	15	1	1	13:00:00	107	1
2	15	3	1	11:00:00	219	1
3	15	3	2	12:00:00	219	1
4	15	3	4	14:00:00	427	1
5	15	3	4	14:00:00	203	1
6	15	3	5	10:00:00	204	1
7	15	2	1	09:00:00	208	1
8	15	2	2	14:00:00	210	1
9	15	2	3	11:00:00	319	1
10	15	2	4	11:00:00	209	1
11	15	2	5	09:00:00	209	2
12	15	2	6	11:00:00	209	2

- ID timeslot–ის ID
- subject_ID საგნის ID, რომლის timeslot–საც წარმოადგენს
- type კლასის ტიპი (1 –საერთო ლექცია, 2 – პრაქტიკული, 3 ჯგუფური მუშაობა, 4 ლაბორატორიული)
- groups ჯგუფის ნომერი
- time timeslot–ის ჩატარების ტრო
- room ოთახის ნომერი
- day ჩატარების დღე (1–6 შეესაბამება ორშაბათი–შაბათი)

შემდეგი ცხრილია ‘passed_subjects’, (სურათი 4) რომელიც ინახავს ინფორმაციას დარეგისტრირებული მომხამებლის მიერ გავლილი საგნების შესახებ.

სურათი 4

user_ID	passed_subject_id
20001067222	1,2,3,4,5,10,11,12,13,14,15,16,17,18,19,20,22,23,2...
20001067223	14,10,17,20
20004567123	1,2,3,4,5,10,11,12,13,14,15,16,17,18,19,20,22,23,2...

- user_ID მომხმარებლის ID (მისი პირადი ნომერი)
- passed_subject_id მომხმარებლის მიერ გავლილი საგნების ID–ები.

შემდეგი ცხრილი – ‘taken_subjects’ (სურათი 5), ინახავს ინფორმაცის ამჟამად აღებულ საგნებზე.

სურათი 5

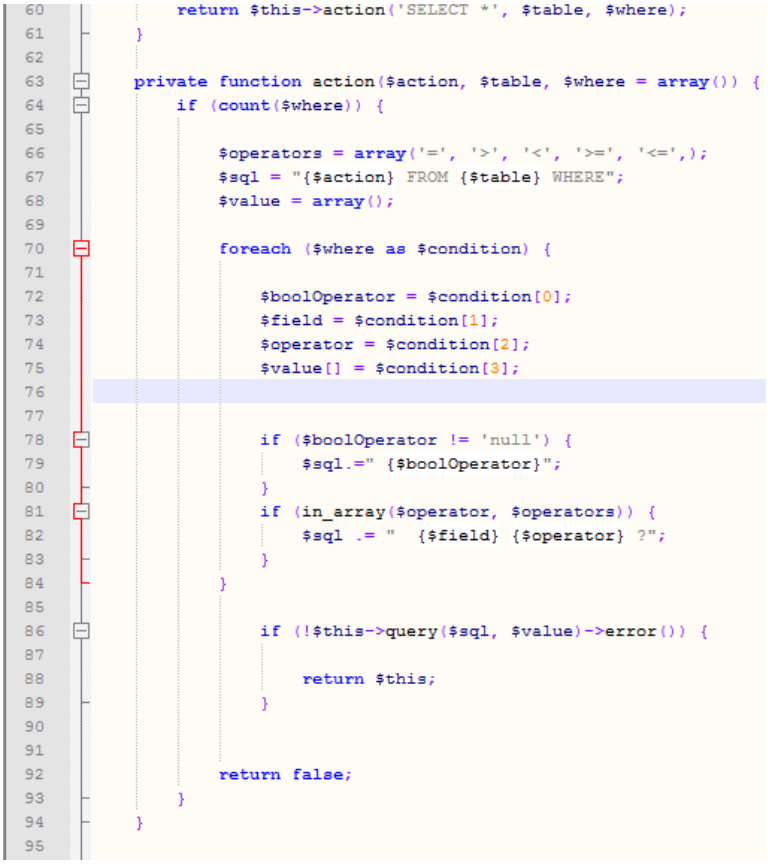
ID	P_ID	subject_ID	groups
193	20001067222	53	1,,,2
194	20001067222	69	1,,,2
252	20001067222	51	1,,,1
253	20001067222	44	1,,1,
254	20001067222	64	1,,1,2
255	20001067222	49	1,,1,

- ID ჩანაწერის ID
- P_ID დარეგისტრირებული მომხმარებლის ID
- subject_ID მომხმარებლის მიერ არჩეული საგნის ID
- groups საგნის არჩეული ჯგუფები (პირველი პოზიცია– საერთო ლექციის ჯგუფი, მეორე – პრაქტიკული მუშაობის ჯგუფი, მესამე – ჯგუფური მუშაობის ჯგუფი, მეოთხე ლაბორატორიული მუშაობის ჯგუფი, ცარიელი ადგილი ნიშნავს რომ ამ საგანს არ აქვს ამ პოზიციის შესაბამისი ჯგუფი, მაგალითად 49 – რომელიც ჩვენს ბაზაში წარმოადგენს კრიპტოგრაფიულ ალგორითმებს, არ აქვს პრაქტიკული და ლაბორატორიული მეცადინეობები)

კოდში მონაცემთა ბაზასთან მუშაობა ევალება DB კლასს. ბაზასთან მუშაობისთვის ვიყენებთ PDO (PHP Data Object)-ს. DB კლასი პასუხს აგებს, ბაზიდან მონაცემების ამოღებაზე, ასევე ბაზაში მონაცემების ჩაწერა, განახლება და წაშლაზე.

ბაზიდან მონაცემების ამოღება ხდება get() მეთოდით (სურათი 6)

სურათი 6



```
60          return $this->action('SELECT *', $table, $where);
61      }
62
63      private function action($action, $table, $where = array()) {
64          if (count($where)) {
65
66              $operators = array('=', '>', '<', '>=', '<=',);
67              $sql = "{$action} FROM {$table} WHERE";
68              $value = array();
69
70              foreach ($where as $condition) {
71
72                  $boolOperator = $condition[0];
73                  $field = $condition[1];
74                  $operator = $condition[2];
75                  $value[] = $condition[3];
76
77
78                  if ($boolOperator != 'null') {
79                      $sql.=" {$boolOperator}";
80                  }
81                  if (in_array($operator, $operators)) {
82                      $sql .= "  {$field} {$operator} ?";
83                  }
84              }
85
86              if (!$this->query($sql, $value)->error()) {
87
88                  return $this;
89              }
90
91
92              return false;
93          }
94      }
95
```

როგორც ვხედავთ get მეთოდი ძალიან მარტივი ფუნქციაა, მისი მთავარი დანიშნულებაა მონაცემები გადასცეს DB::action (სურათი 6) მეთოდს შესაბამისი სტრინგით. action მეთოდის დანიშნულებაა გაამზადოს $sql სტრინგი და გადასცეს ის DB::query მეთოდს.

გამოძახების მაგალითი:

DB::getInstance()->get('subjects', array(array('null', 'semestri' , '<=', 0), array('AND','savaldebulo', '=', 1)));

get მეთოდი action მეთოდს გადასცემს მონაცემებს, ანუ $action = 'SELECT *', $table = 'subjects' და $where =array(array('null', 'semestri' , '<=', 0), array('AND','savaldebulo', '=', 1)). რის შედეგადაც action გაამზადებს სტრინგს

$sql = 'SELECT * FROM subjects WHERE semestri = ? AND savaldebulo = ?

და გადასცემს მას მეთოდ query–ს(სურათი 7), და მასთან ერთად $value მასივს, სადაც ამ შემთხვევაში ჩაწერილია მნიშვნელობები 0 და 1.

სურათი 7

```
27
28      public function query($sql, $params = array()) {
29          $this->error = false;
30
31          if ($this->_query = $this->_pdo->prepare($sql)) {
32              $x = 1;
33
34              if (count($params)) {
35
36                  foreach ($params as $param) {
37
38                      $this->_query->bindValue($x, $param);
39
40                      $x++;
41                  }
42              }
43
44              if ($this->_query->execute()) {
45
46                  $this->_results = $this->_query->fetchAll(PDO::FETCH_OBJ);
47                  $this->_count = $this->_query->rowCount();
48
49              } else {
50                  $this->_error = true;
51              }
52          }
53
54          return $this;
55      }
56
```

მეთოდი query შესრულების შემდეგ დააბრუნებს DB კლასის ობიექტს, რომლის $_results პარამეტრი იქნება მასივი რომელიც შეიცავს stdClass ტიპის ობიექტებს, რომელთაგან თითოეული ფაქტრობრივად შეესაბამება ბაზაში რომელიმე კონკრეტულ ჩანაწერს. ამ შემთხვევაში $_results მასივში, ჩაიწერება ის სავალდებულო საგნები რომლებიც ისწავლება შემოდგომის სემესტრში. ხოლო $_count ცვლადში ჩაიწერება ბაზიდან ამოღებული საგნების რაოდენობა. უნდა აღინიშნოს, რომ ამ მეთოდით მუშაობა საკმაოდ აადვილებს საქმეს რადგან თითოეულ საგანზე რაიმე ინფორმაციის მისაღებად საკმარისია მხოლოდ ერთი ხაზი კოდის დაწერა, მაგალითად თუ გვინდა საგნის ID–ით 35 კრედიტის გაგება საკმარისია შემდეგი კოდი.

DB::getInstance()->get('subjects',array(array('null','subject_ID','=',35)))->results()[0]->krediti;

ეს დაგვიბრუნებს საგნის კრედიტის რიცხვით მნიშვნელობას.

```
29      private $_weekSchedule = array(
30          'monday'=>array(
31              '09:00:00'=>'',
32              '10:00:00'=>'',
33              '11:00:00'=>'',
34              '12:00:00'=>'',
35              '13:00:00'=>'',
36              '14:00:00'=>'',
37              '15:00:00'=>'',
38              '16:00:00'=>'',
39              '17:00:00'=>'',
40              '18:00:00'=>''
41          ),
42          'tuesday'=>array(
43              '09:00:00'=>'',
44              '10:00:00'=>'',
45              '11:00:00'=>'',
46              '12:00:00'=>'',
47              '13:00:00'=>'',
48              '14:00:00'=>'',
49              '15:00:00'=>'',
50              '16:00:00'=>'',
51              '17:00:00'=>'',
52              '18:00:00'=>''
53          ),
```

კოდში, ცხრილთან დაკავშირებული ყველა მოქმედება ევალება კლას Scedule–ს. მისი ერთერთი ყველაზე მნიშვნელოვანი ცვლადია მასივი $_weekSchedule: სურათი 8. მომხმარებლის საიტზე ყოველი შემოსვილისას, ას მასივი შესაბამისი სიდიდეები ივსება მის მიერ ამ სემესტრში არჩეული საგნებით.

ალგორითმი შედგება ორი ნაწილისგან, პირველი ნაწილია დღეების და დროების პრიორიტეტების ეგრედწოდებული რუკის შედგენა, ხოლო მეორე ნაწილი წარმოდაგენს ამ რუკით სასწავლო ცხრილის შედგენას. ეს მეთოდი ალგორითმს აძლევს მოქნილობას გაუმკლავდეს ნებისმიერ მოთხოვნას, თუ მას გადავაწვდით შესაბამის რუკას. მაგალითად, თუ მომხმარებელს სურს მიიღოს ისეთი ცხრილი, რომ დაკავებული ჰქონდეს რაც შეიძლება ნაკლები რაოდენობის დღეები, მაშინ რუკა მიიღებს შემდეგ სახეს.

დღე/დრო	9:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00
ორშაბათი	2	2	2	2	2	-1	-1	2	2
სამშაბათი	1	1	1	1	1	1	1	1	1
ოთხშაბათი	3	3	3	3	3	-1	-1	-1	3
ხუთშაბათი	1	1	1	1	1	1	1	1	1
პარასკევი	1	1	1	1	1	1	1	1	1
შაბათი	2	2	2	-1	-1	2	2	2	2

სადაც -1 ნიშნავს რომ ის უკვე დაკავებულია სავალდებულო საგნის მიერ რომელებსაც ვერსად გავექცევით, ამიტომ იძულებულნი ვართ ცხრილი შევქმნათ მათ გარშემო. ხოლო დანარჩენი მნიშვნელობები გამოითვლება შემდეგი პრინციპით: თავისუფალ უჯრაში ჩაიწერება ამ დღეს

დაკავებული უჯრების რაოდენობა, ანუ ამ შემთხვევაში, (გვინდა რომ დავტვირთოთ მინიმალური დღეების რაოდენობა), მაღალი პრიორიტეტები ენიჭება უკვე დატვირთული დღეების უჯრებს. წინააღმდეგ შემთხვევაში ენიჭება 1. ალგორითმი შეეცდება დატვირთოს რუკის მაღალ პრიორიტეტიანი უჯრები,

მაგალითი 2. მაგალითად სტუდენტს უნდა აირიდოს საგნები რომელთა ლექციები ტარდება 12–დან 14 საათებზე.

დღე/დრო	9:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00
ორშაბათი	1	1	1	-1	-1	-1	-1-	1	1
სამშაბათი	1	1	1	-1	-1	-1	1	1	1
ოთხშაბათი	1	1	1	-1	-1	-1	-1	-1	-1
ხუთშაბათი	1	1	1	-1	-1	-1	1	1	1
პარასკევი	1	1	1	-1	-1	-1	1	1	1
შაბათი	1	1	1	-1	-1	1	1	1	1

მიღებული რუკით ალგორითმი მაქსიმალურად აარიდებს თავს ისეთი საგნების არჩევას რომლის ლექციები ტარდება 12–დან 14–მდე.
ასევე აუცილებელი არაა, მხოლოდ ერთი მოთხოვნის მიცემა ალგორითმისთვის, შესაძლებელი იქნება სხვადასხვა მოთხოვნების კომბინირება, მაგალითად მოყვანილი ორი მაგალითის კომბინირებული რუკა იქნება:

დღე/დრო	9:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00
ორშაბათი	2	2	2	-1	-1	-1	-1-	2	2
სამშაბათი	1	1	1	-1	-1	-1	1	1	1
ოთხშაბათი	3	3	3	-1	-1	-1	-1	-1	3
ხუთშაბათი	1	1	1	-1	-1	-1	1	1	1
პარასკევი	1	1	1	-1	-1	-1	1	1	1
შაბათი	2	2	2	-1	-1	-1	2	2	2

შენიშვნა. ამ დოკუმენტის წერის მომენტში რუკის გენერირების ალგორითმი არ არის დასრულებული და საბოლოო ჩაბარამდე დიდი ალბათობით შეიცვლება.

ალგორითმის მეორე ნაწილი წარმოადგეს რუკის დამუშავებას, ის შედგება ორი ძირითადი ციკლისგან, გარე ციკლი პასუხს აგებს ინდივიდუალური დღეების დამუშავებაზე, ხოლო მასში ჩადგმული შიდა ციკლი პასუხისმგებელია კონკრეტული საათების დამუშავებაზე. ალგორითმი შეამოწმებს ყოველ საგანს რომელიც ამ დღეს ერთი ლექცია მაინც აქვს, და რუკის მიხედვით ადგენს ამ საგნის ლექციების თითოეული ტიპის (საერთო ლექცია, პრაქტიკული ლექცია, ჯგუფური პროექტი, ლაბორატორიული მუშაობა) ლექციებში საუკეთესო ვარიანტს, ხოლო იმ შემთხვევაში თუ შეუძლებელია რომელიმე ტიპის მეცადინეობის არჩევა, ალგორითმი გადადის შემდეგ საგანზე. მას შემდეგ რაც აარჩევს ყველა ტიპის საუკეთესო ვარიანტს, ალგორითმი დაითვლის საგნის შეფასებას და თუ შეფასება უკეთესია ამჟამინდელ ლიდერზე, განაახლებს ლიდერზე ინფორმაციას. მას შემდეგ რაც ალგორითმი შეამოწმებს ამ დღეს არჩევად ყველა საგანს ის აირჩევს ლიდერ საგანს და გაუშვებს ალგორითმს თავიდან, მას შემდეგ რაც თავიდან დააგენერირებს რუკას.

ქვემოთ მოყვანილია ალგორითმის მოქმედებების მიმდევრობა:

1. მიღებული რუკის მიხედვით დაითვალე ინდივიდუალური დღეების პრიორიტეტები.
2. დაალაგე დღეები პრიორიტეტების კლებადობის მიხედვით.
3. თითოეული დღისთვის იპოვე საგნები რომლებიც ტარდება ამ დღეს. (გარე ციკლი)
    I. თითოეული საგნისთვის იპოვე ამ საგნის ყველა მეცადინეობის timeslot.
        i. თითოეული საგნის მეცადინეობების ყოველი ტიპისთვის(საერთო ლექცია, პრაქტიკული ა.შ) იპოვე ყველა ტიპისთვის საუკეთესო ვარიანტი. (თითოეულისთვის ქულის გამოთვლით და მათგან საუკეთესოს ამორჩევით)
        ii. საგნის ყველაზე მისაღები ჯგუფების არჩევის შემდეგ, გამოთვალე საგნის საერთო ქულა და შეადარე ამ დროისთვის არსებულ ლიდერის ქულას.
    II. თუ ამ დღის ყველა საგანს შორის ლიდერის ქულა აკმაყოფილებს მოთხოვნას და ამ საგნის არჩევის შემდეგ სტუდენტის კრედიტების ჯამი არ გადააჭარბებს მის ლიმიტს, აირჩიე ეს საგანი, დააგენერირე რუკა თავიდან ახლად არჩეული საგნის გათვალისწინებით და გაუშვი ალგორითმი თავიდან, ახალი რუკით.
    III. თუ ლიდერი საგნის ქულა არ აკმაყოფილებს მოთხოვნას გადადი პრიორიტეტების მიხედვით შემდეგ დღეზე.
4. თუ ყველა დღის შემოწმების შემდეგ, სტუდენტს კვლავ შეუძლია საგნების არჩევა, გაანახევრე მოთხოვნები და გაუშვი ალგორითმი თავიდან ახალი მოთხოვნებით.
5. თუ სტუდენტს აღარ აქვს საგნების არჩევის უფლება დააბრუნე არჩეული საგნები.

ზემოდ მოყვანილი ალგორითმის მიხედვით, თუ სრულდება პუნქტი 4, ჩვენ ძალიან ვშორდებით ოპტიმალურ შედეგს, ამიტომ შეიძლება ჩაითვალოს რომ 4 პუნქტის შესრულება წარმოადგენს ყველა შესაძლო შემთხვევებიდან ყველაზე ცუდს, (რომლის თავიდან არიდებაც ჩვენზე არ არის დამოკიდებული, რადგან ჩვენ დამოკიდებულები ვართ თსუ–ს სასწავლო განრიგზე) ამიტომ შეიძლება პუნქტი 4–ს შესრულება გავხადოთ ალგორითმის გაჩერების პირობაც და მივცეთ მომხმარებელს საშუალება ხელით შეავსოს ცხრილი.

დასკვნა

პროექტის მიზანი იყო შექმნილიყოს პროგრამული სისტემა - აპლიკაცია - რომელიც საშუალებას მისცემს თსუ-ს სტუდენტს მოახდინოს ინდივიდუალური სასწავლო ცხრილის ფორმირება, რომელიც სინქრონირებული იქნება სასწავლო პროცესის მართვის სისტემასთან sms.tsu.ge.

მიზნის მისაღწევად შემუშავდა მოქნილი ევრისტიული ალგორითმი რომელიც დააგენერირებდა ცხრილს მითითებული საგნების მიხედვით, საგნების სავალდებულო/არჩევითობის პირობის გათვალისწინებით.

ალგორითმის რეალიზაცია მოხდა PHP დაპროგრამების ენაზე. რეალიზებული ალგორითმი არის ვებ-ზე დაფუძნებული პროგრამული სისტემის ერთერთი ძირითადი კომპონენტი. შეიქმნა ნაკლებად დატვირთული მარტივი და სადა დიზაინის მქონე ვებ აპლიკაცია, რომელიშიც მომხმარებელი თავს იგრძნობდა კომფორტულად და რაც მთავარია ინფორმაცია თავისი სპეციფიკიდან გამომდინარე არის მაქსიმალურად დაცული.

პროექტის შესრულების შედეგად შემუშავებული აპლიკაცია მცირევ დამუშავების შედეგად შეიძლება ინტეგრირდეს თსუ სასწავლო პროცესის მართვის სისტემაში.

ლიტერატურა

1. Tegegn Nuresu Wako. "Educational Management Information Systems An Overview", 2003
2. L. Carrizo, C. Sauvageot and N. Bella "Information tools for the preparation and monitoring of education plans", 2003
3. Criana Connal. "NFE-MIS Handbook Developing a Sub-National Non-Formal Education Management Information System", 2005
4. UNESCO. "Education Indicators Technical Guidelines", 2009.
5. www.php.net