კომერციული საიტის პროგრამული უზრუნველყოფის სრულყოფა და ფუნქციობის გამოკვლევა

გიორგი კუჭავა
საქართველოს ტექნიკური უნივერსიტეტი

რეზიუმე

შემოთავაზებულია კომერციული საიტის კონტენტის მართვის სისტემის პროგრამული უზრუნველყოფის სრულყოფის მეთოდიკა, რომელიც დაფუძნებულია დაპროგრამების Python-ენისა და Framework Django ვებ-პლატფორმის გამოყენებაზე. პრაქტიკულმა აპრობაციამ მის ბაზაზე დაპროექტებული ინტერნეტ-მაღაზიისათვის აჩვენა, რომ შემუშავებული პროგრამული უზრუნველყოფა არსებულთაგან განსხვავებული ლოგიკით ამუშავებს სერვერს და მაღალი სამომხმარებლო ტრაფიკის პირობებშიც კი მისი აპარატურული რესურსების საგრძნობ ეკონომიას იძლევა.

საკვანძო სიტყვები: ელექტრონული კომერცია. ვებ-საიტი. საიტის ტრაფიკი. პროგრამული უზრუნველყოფა. ინტერნეტ-მაღაზია. სერვერი. აპარატურული რესურსები. Python, Django.

1. შესავალი

თანამედროვე ინფორმაციული ტექნოლოგიები და მათი შესაბამისი პროგრამული უზრუნველყოფა სულ უფრო მეტ კომპიუტერულ რესურსს მოიხმარს, რაც ახალ მოთხოვნებს უყენებს მათ ბაზაზე აგებული სისტემების როგორც პროგრამულ, ასევე აპარატურულ უზრუნველყოფას. ეს გარემოება თავს იჩენს ინტერნეტ სივრცეშიც, კერძოდ, ელექტრონული კომერციის სფეროში, სადაც მწვავედ დგას ტრაფიკის პრობლემა.

საკითხს ორი ასპექტი აქვს: ერთი მხრივ, თუ კომერციულ საიტზე არ გვექნება ახალი პროგრამული აპლიკაციები, რომლებიც პასუხობს მომხამარებელთა მზარდ მოთხოვნებს, საიტს არ ეყოლება მუდმივი სტუმრები და რეგისტრირებული მომხმარებლები; მეორე მხრივ, ეფექტიანი პროგრამული და ინფორმაციული უზრუნველყოფა გაზრდილ მოთხოვნებს უყენებს კომპიუტერის აპარატურულ რესურსებს. მაგალითად, არც თუ იშვიათია შემთხვევა, როცა კომპიუტერები და სერვერები, რომლებზეც განთავსებულია ვებ-საიტები, ვერ უძლებს დატვირთვას და საიტი ავტომატურად ითიშება. ამგვარი მტყუნებები განსაკუთრებით მიუღებელია კომერციაზე პირდაპირ ორიენტირებული საიტებისათვის, მაგალითად, ინტერნეტ-მაღაზიებისთვის, სადაც განუწყვეტლივ ხდება ფულადი გადარიცხვები. მართალია, ასეთი პრობლემების დროს, ტრაფიკის შესანარჩუნებლად და გასაზრდელად, პროვაიდერები გვთავაზობენ საიტის უფრო მძლავრ სერვერზე გადატანას, მაგრამ ხშირად, განსაკუთრებით დამწყები საიტებისათვის, ეს პრობლემის კომერციულად წამგებიანი გადაწყვეტაა.

დღესდღეობით ვებ–პროგრამისტები სულ უფრო ხშირად ცდილობენ MySQL ტექნოლოგიების ნაცვლად გამოიყენონ ASP.NET და ORACLE ტექნოლოგიები. თუმცა, ORACLE-ს მონაცემთა ბაზებთან სამუშაოდ უფრო მძლავრი აპარატურაა საჭირო და პრა–ქტიკულად, ყოველ საიტს, საკუთარი სერვერი და ORACLE სისტემა უნდა გააჩნდეს.

ამგვარი კონფიგურაციის სისტემის შექმნა და გამოყენება დიდ დანახარჯებს მოითხოვს როგორც პროვაიდერის, ასევე შემკვეთის მხრიდან [1].

ამგვარად, კომერციული საიტების პროგრამული უზრუნველყო-ფის სრულყოფისა და მისი ანალიზის ახალი მეთოდებისა და ინსტრუმენტების შემუ-შავება ელექტრონული კომერციის აქტუალური ამოცანაა, რაც ხელს შეუწყობს კომერ–ციული საიტის ტრაფიკის გაზრდას და მის ეფექტიან ფუნქციობას მზარდ კონკურენტულ გარემოში.

2. ძირითადი ნაწილი

შემუშავებულია კომერციული საიტის კონტენტის მართვის სისტემის პროგრამული უზრუნველვყოფის სრულყოფის მეთოდიკა, რომელიც დაფუძნებულია დაპროგრამების სისტემა Python-ის გამოყენებაზე. ანალიზი აჩვენებს, რომ Python სინტაქსურად და პროცედურულად მსუბუქი ენაა და ძალუძს აპარატურული რესურსებისადმი მცირე მოთხოვნებით ამუშაოს თანამედროვე დონის კომერციული საიტი შემავსებელი აპლიკაციებითურთ. ამასთანავე, ვინაიდან Python ინტერპრეტატორია, მისი კოდი და შესაბამისად საიტი უფრო სწრაფად მუშაობს, რადგან არ საჭიროებს დამატებით დროს კომპილაციისათვის [2].

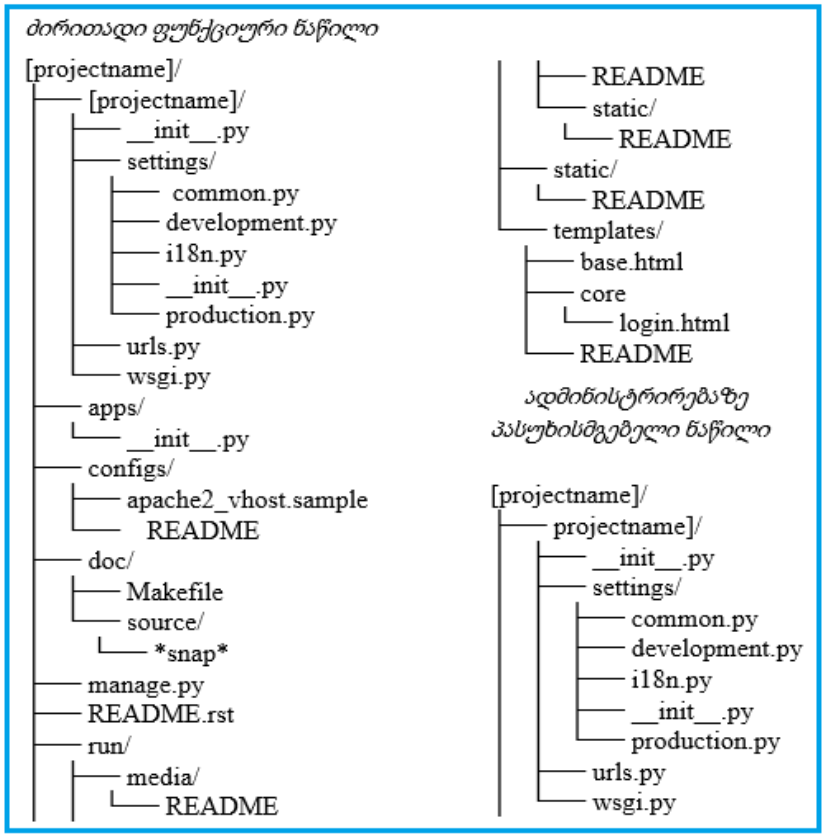
პროგრამული უზრუნველყოფის შემუშავებისას გათვალისწინებულ იქნა Python-ის საინტერესო შესაძლებლობა – გადაანაწილოს ოპერატიული მეხსიერების დატვირთვა მყარ დისკოზე შექმნილ ვირტუალურ ოპერატიულ მეხსიერებაში. ეს გარემოება, რომელიც Python სისტემაში ვირტუალიზაციის swapper ფუნქციითაა უზრუნველყოფილი, განსა-კუთრებულ შესაძლებლობას გვაძლევს ოპერატიული მეხსიერების ოპტიმალური დატვი-რთვის მხრივ. საიტზე ტრაფიკის დიდი რაოდენობისას, როდესაც ოპერატიული მეხსი-ერება სრულადაა დატვირთული და დამატებით რესურსებს საჭიროებს, მას დახმარებას უწევს მყარი დისკო. კერძოდ, swapper ფუნქცია მყარი დისკოს გარკვეულ ნაწილს გარდაქმნის ვირტუალურ ოპერატიულ მეხსისიერებად, რაც ზრდის სერვერის მუშაობის სისწრაფეს. გარდა ამისა, მყარ დისკოზე დატვირთვის გადატანით საგრძნობლად მცირდება ოპერატიული მეხსიერების დატვირთვა. პროცედურულად აღნიშნული გარემოება ასე მიიღწევა: Python ყოფს ოპერატიულ მეხსიერებას ცალკეულ გვერდებად (Pages). რის შემდეგაც ხდება გვერდის გადატანა მყარ დისკოზე წინასწარ გამოყოფილ არეში (Swap Space). მაგალითად, თუ ოპერატიული მეხსიერება არის 3 GB, მყარი დისკდან გამოიყოფა 5 GB ვირტუალური ოპერატიული მეხსიერების შესაქმნელად [2,3].

პროგრამული უზრუნველყოფის კარკასის შერჩევისას ყურადღება მიექცა იმ გარემოებას, რომ ის სამართავად მარტივი ყოფილიყო. ამ მოსაზრებით შერჩეულ იქნა Framework Django, რომელიც აგრეთვე სისტემაშია დაპროექტებული [4].

1-ელ ნახაზზე მოყვანილია PyCharm გარემოში შემუშავებული კომერციული საიტის პროგრამული უზრუნველყოფის ფაილური სტრუქტურა, რომლის ცალკეული კომპონენტების დანიშნულება შემდეგია:

- **init_.py** –კოდის გამშვები ანუ მოქმედებაში მომყვანი პროგრამული მოდულია;
- **common.py** – უზრუნველყოფს ფაილური სისტემის განთავსებას, აპლიკაციების კონფიგურირებას, დაცვის სისტემის გამართვას, ვებ-საიტის კარკასში მუშაობისას ცვლილებების შეტანას, შეცდომების გასწორებას, რეგიონული მომართვების მხარდაჭე-რასა და ადაპტირებას;

- **development.py** – ემსახურება მონაცემთა ბაზის კონფიგურირებას;



ნახ.1

- **i18n.py** – პროგრამული მოდულია, რომელიც ემსახურება საიტის ძირითადი ენის გადაყვანას ნებისმიერ სხვა ენაზე;
- **production.py** – საშუალებას გვაძლევს შევარჩიოთ ვებ-სერვერის სასურველი კონფიგურაცია. მოდული შეიცავს ქსელში სამუშაოდ საჭირო Host/Domain სინტაქსისებს;
- **urls.py** – ქმნის URL(Uniform Resource Locator) ლოგიკას, რომელიც განსაზღვრავს ვებ-გვერდის მდებარეობას ინტერნეტის ქსელში;
- **wsgi.py** (Web Server Gateway Interface) – წარმოადგენს ინტერფეისს ვებ-სერვერსა და ვებ-აპლიკაციებს შორის, დაპროექტებულს Python სისტემაში;
- **apps** – მოდული ამატებს სამომხმარებლო აპლიკაციებს ვებ-გვერდზე;
- **configs** – მოდული შეიცავს Apache2 სერვერზე ფაილების განლაგების პროტოკოლს;
- **doc** – დირექტორიის ეს ნაწილი მოიცავს ინფორმაციას სტრუქტურაში შემავალი ფაილების შესახებ. მისი ამოშლა სტრუქტურიდან არ აზიანებს საიტს;
- **manage.py** – შეიცავს საიტის ამძრავ ფაილებს, რომელთაც რეალურ მოქმედებაში მოჰყავს საიტის ადმინისტრირების სისტემა;
- **run/media/** – ინახავს მომხმარებლის მიერ ატვირთულ ფაილებს, ასეთი ნებართვის არსებობის შემთხვევაში;
- **run/Static** – უზრუნველყოფს CSS, Javascript, image ტიპის ფაილების მართვას;
- **Templates** – შეიცავს შაბლონებს, რომლებიც უზუნველყოფს საიტის ვიზულურ მხარეს.

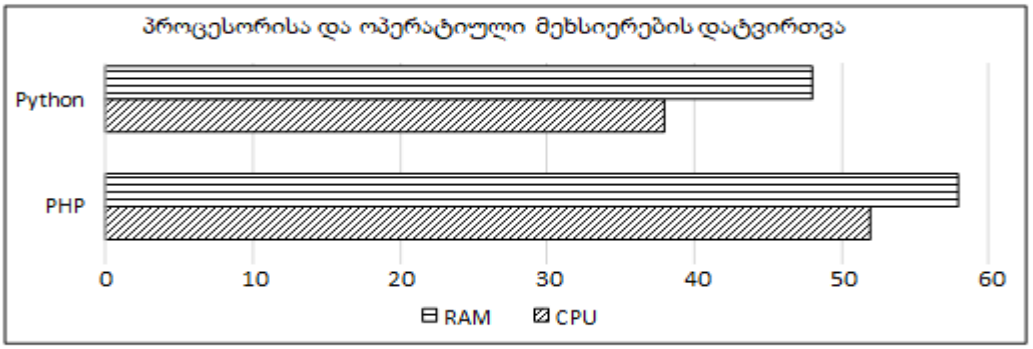
საზოგადოდ, Python სისტემა ორიენტირებულია Linux ოპერაციულ სისტემაზე. ამიტომ, თუ სერვერზე, ამ „ოჯახის“ რომელიმე სისტემას დავაყენებთ, მაგალითად CentOS–ს, შევ–ძლებთ მის გარსში, სპეციალური კონსოლის Terminal-ის საშუალებით, შევიყვანოთ შე–საბამისი Python-კოდი, რომელიც მყარი დისკოს გარკვეულ ნაწილს ვირტუალურ ოპერა–ტიულ მეხსიერებად გარდაქმნის. შედეგად, კომერციული საიტის სერვერი, რომელსაც შეზღუდული ოპერატიული მეხსიერება და მაღალი ტრაფიკი აქვს, ოპერატიულ მეხსი–ერებაზე დატვირთვას გადაუნაწილებს მყარ დისკოზე გამოყოფილ დანაყოფს. უნდა გავი–თვალისწინოთ, რომ ამ ოპერაციის შესრულება შეუძლია მხოლოდ ერთსა და იმავე პრო–გრამულ კოდზე მომუშავე სისტემას. ეს ნიშნავს, რომ თუ ვებ-საიტი არ არის აგებული Python-ის კოდზე, დატვირთვის გადანაწილება მყარ დისკოზე შეუძლებელი იქნება სხვა კოდზე მომუშავე საიტისათვის. ამ შემთხვევაში, საჭიროა Swapper მოდულის ჩაშენება Framework Django-ში, რათა მასზე დაფუძნებულმა საიტმა შეძლოს კავშირი დაამყაროს მყარი დისკოს swap space არესთან. ეს პროცედურა ხორციელდება შემდეგი პროგრამული კოდის მეშვეობით (ნახ. 2):

```
import swapper
Parent = swapper.load_model("reusableapp", "Parent")
Child = swapper.load_model("reusableapp", "Child")
def view(request, *args, **kwargs):
qs = Parent.objects.all( )
 # ...
```

ნახ. 2

შემუშავებული პროგრამული უზრუნველყოფა აპრობირებულ იქნა მის ბაზაზე დაპროექტებული ინტერნეტ-მაღაზიის საიტისათვის, რომელსაც დღის პიკურ პერიოდებში საკ–მაოდ მაღალი სამომხმარებლო ტრაფიკი აქვს (კვირის მანძილზე საშუალოდ 47 000 მნახველი). ანალიზმა აჩვენა, რომ სამომხმარებლო ტრაფიკის 60%-ით გაზრდისას, სერვე-რის პროცესორი, ჩვეულებრივზე მხოლოდ 6.54 %-ით მეტად დაიტვირთა.

მე-3 ნახაზზე ნაჩვენებია ინტერნეტ-მაღაზიის სერვერის მხარის აპარატურული რესურსების დატვირთვის შედარებითი დიაგრამა გაუმჯობესებული და სტანდარტული (PHP კოდის ბაზაზე აგებული) პროგრამული უზრუნველყოფის პირობებში. შედარებითი ანალიზი ცხადყოფს, რომ Python-ის ბაზაზე დაპროექტებული საიტის სერვერი, მუშაობს რა განსხვავებული ლოგიკით, საგრძნობლად ნაკლებ აპარატურულ რესურსებს მოიხმარს.



ნახ. 3

3. დასკვნა

შემუშავებულია კომერციული საიტის კონტენტის მართვის სისტემის პროგრამული უზრუნველყოფის სრულყოფის მეთოდიკა, რომელიც დაფუძნებულია დაპროგრამების სისტემა Python-ისა და Framework Django-ს გამოყენებაზე.

შემუშავებული პროგრამული უზრუნველყოფა აპრობირებულია მოქმედი ინტერნეტ-მაღაზიის საიტზე. ანალიზი ცხადყოფს, რომ Python-ის ბაზაზე დაპროექტებული საიტი, არსებულთაგან განსხვავებული ლოგიკით ამუშავებს სერვერს და მაღალი ტრაფიკის პი–რობებშიაც კი საგრძნობლად ნაკლებ აპარატურულ რესურსებს მოიხმარს, რაც ხელს უწ–ყობს სერვერისა და მთლიანად საიტის ეფექტიან ფუნქციობას.

ლიტერატურა-References-Литература:

1. Hemant J. (2017). Data Structures & Algorithms Using PHP. CreateSpace Independent Publishing Platform.

2. Cay S. Horstmann, Rance D. Necaise. (2016). Python for Everyone, 2-nd Edition., Wiley Publ.

3. Rance D. Necaise. (2016). Data Structures and Algorithms Using Python. Reprint edition. Wiley Publ.

4. Holovaty A., Kaplan-Moss J. (2017). The Definitive Guide to Django: Web Development Done Right. Second edition. Apress, Berkley.

IMPROVEMENT AND STUDY OF FUNCTIONING OF COMMERCIAL WEBSITE SOFTWARE

Kuchava Giorgi
Georgian Technical University

Summary

The article proposes a technique of improvement of the software of a commercial website based on the use of Python programming system and platform for web applications framework Django. The developed software allows for more efficient use of hardware resources of the server on which are hosted the site and to maintain a high level of user traffic. Practical testing of the software, for the online store built on its basis, showed, that in conditions of high user activity there is a significant saving of hardware resources of the server.

СОВЕРШЕНСТВОВАНИЕ И ИССЛЕДОВАНИЕ ФУНКЦИОНИРОВАНИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ КОММЕРЧЕСКОГО САЙТА

Кучава Г.Т.
Грузинский Технический Университет

Резюме

Предложена методика совершенствования программного обеспечения коммерческого веб-сайта основанная на использовании системы программирования Python и платформы для веб-приложений Framework Django. Разработанное программное обеспечение позволяет более эффективно использовать аппаратные ресурсы сервера на котором размешен саит и поддерживать высокий уровень пользовательского траффика. Практическая апробация программного обеспечения для построенного на его основе интернет-магазина, показала что и в условиях высокой пользователеской активности имеет место существенная экономия аппаратных ресурсов сервера.