

CANKAYA UNIVERSITY

FACULTY OF ENGINEERING

COMPUTER ENGINEERING DEPARTMENT

Project Report

Innovative System Design and Development

201901

PATENT COMPARISON SYSTEM

*Özlem Simay CELKAN*

*201511009*

*Neval Zeynep GÜNDEŞLİ*

*201511027*

*Öznur ÖZTÜRK*

*201511045*

*Mert KIZILDAĞ*

*201411032*

**Advisor:** Dr. Öğr. Üyesi Murat SARAN

[**Abstract**](#_b4wry150ggz8)5

[**Key words**](#_t9xgpf1nf9x)5

[**1. Introduction**](#_so8uf48ltdv1)5

[1.1 Company Background](#_wqetxjk3vno0) 6

[1.2 Problem Statement](#_idcx7cf7h9id) 6

[1.3 Related Work](#_75x8wtvu68d) 6

[1.4 Solution Statement](#_8y2x0lsstiui) 7

[1.5 Contribution](#_1m8743ey0rm6) 7

[**2. Literature Search**](#_ajbctosd7vke)8

[**2.1 Library Research**](#_ez6e47stg7tx)8

[2.2 Internet Research](#_7yqbs9wqvp43) 8

[**3. Summary**](#_yzcfxvnua1sv)8

[3.1 Summary of Conceptual Solution](#_72zf7l3703uf) 8

[3.2 Technology Used](#_f5xkfy7kpsib) 8

[**4. Software Requirements Specification**](#_1m0one2jmbf5)8

[4.1 Introduction](#_rtye6blo79qx) 8

[4.1.1 Purpose](#_eb5syi72k2ao) 8

[4.1.2 Scope of Project](#_96jspaiycgp) 9

[4.1.3 Glossary](#_l2ec34ddepb5) 9

[4.1.4 References](#_rthly15jyt5b) 10

[4.1.5 Overview of Document](#_tx2fn4lwz3mm) 10

[4.2 Overall Description](#_3r9l92kblgmh) 10

[4.2.1 Product Perspective](#_gnte8ccnyejj) 11

[4.2.2 Development Methodology](#_th72ik478727) 11

[4.2.3 Product Functions](#_k7mtjrlxh26m) 11

[4.2.3.1 Login Page](#_mpf8lopghd3q) 11

[4.2.3.3 Add Patent](#_g80yz19ia9o3) 12

[4.2.3.4 Description Page](#_yx64i4kihw2t) 12

[4.2.3.5 Contact Page](#_s4em6zdzyyez) 12

[4.2.3.6 Result Page](#_xpth9vg3opr4) 12

[4.2.4 User Characteristics](#_pcifdlbfjaws) 12

[4.2.4.1 Patent Owners](#_6xqlhtibw57s) 12

[4.2.4.2 People Who Want to Apply for a Patent](#_9gfp85ms104p) 12

[4.2.5 Constraints](#_h5bwfgpkrpwr) 13

[4.2.6 Assumptions and Dependencies](#_wer23nql25su) 13

[4.3 Requirements Specification](#_lhf5j4etfy3b) 13

[4.3.1 External Interface Requirements](#_58zqq7lifqlv) 13

[4.3.1.1 User interfaces](#_j74ip3xhjrz) 13

[4.3.1.2 Hardware interfaces](#_44l5hyhlecc8) 14

[4.3.1.2.1 Server Side](#_9md5hkwgn37d) 14

[4.3.1.2.2 Client Side](#_g7kd1biig5uk) 14

[4.3.1.3 Software interfaces](#_oa6xjfs0ua9y) 14

[4.3.1.4 Communications interfaces](#_7hox6452br6r) 14

[4.3.2 Functional Requirements](#_p0dn36394nc4) 14

[4.3.2.1 Add Patent](#_g334cj1b3ah9) 14

[4.3.2.2 Create User](#_rrc9it5xqxqb) 14

[4.3.2.3 Create Result](#_51l9vlsejsrv) 15

[4.3.3 Performance Requirements](#_8pwj0x5yisfm) 15

[4.3.4 Design constraints](#_bl3cemhamx4) 15

[4.3.4.1 Usability](#_auqfu7chx7xt) 15

[4.3.4.2 Standards compliance](#_iskw41mj1vm3) 15

[4.3.5 Software system attributes](#_egt6grn0ki17) 15

[4.3.5.1 Reliability](#_plixgym87dw3) 15

[4.3.5.2 Security](#_jfcdw7b3a7xd) 16

[4.3.5.3 Portability](#_qkw79r7vczh7) 16

[4.3.5.4 Performance](#_t89eih6ajaxa) 16

[4.3.5.5 Usability](#_gnq9u14dgs27) 16

[4.3.5.6 Adaptability](#_qqqiiztmdf43) 16

[4.3.5.7 Scalability](#_jcsnease5hsr) 16

[4.3.5.8 Performance](#_fywdh3p1jy2h) 16

[4.3.6 Other Requirements](#_fchh98y3t2q4) 16

[**5. Software Design Description(SDD)**](#_4e6dqbdlhbgc)16

[5.1 Introduction](#_ka067cl2haz6) 16

[5.1.1 Purpose](#_2kwia8v4y4lv) 17

[5.1.2 Scope](#_qf0tzfl03vy6) 17

[5.1.3 Glossary](#_d8abxtkhllnn) 17

[5.1.4 References](#_lpnvxj36i41p) 18

[5.1.5 Overview of Document](#_9ol60rv76xi7) 19

[5.2 Deployment Diagram](#_emistbdpzbd6) 20

[5.3 Architecture Design](#_g5ki9wzcflug) 20

[5.3.1 Options Menu](#_q8t4vy84c9pl) 21

[5.3.2 Class and Activity Diagram](#_2i1hxub909ff) 21

[5.4 Data Structure Design](#_rzov0te3tr9a) 22

[5.5 Use Case Realizations](#_5wr4phv6icao) 23

[5.6 Interface Design](#_rqjlc8a6t1yl) 23

[5.7 Help System Design](#_glgygk53h2io) 28

[**6. Test Plan**](#_ih4ycf6p6aj)28

[6.1 Introduction](#_thtla4919lp4) 28

[6.1.1 Version Control](#_uuqlsclogp1k) 28

[6.1.2 Overview](#_i0i68115etll) 28

[6.1.3 Scope](#_k6zmovfn6zhi) 29

[6.1.4 Terminology](#_3b865ey47ajp) 29

[6.2 Features to be tested](#_z1j92m7g32ta) 29

[6.2.1 Adding a Patent (AP)](#_lr295elocgdc) 29

[6.2.2 Patent Comparison (PC)](#_kly2abajs902) 29

[6.2.3 Patent Result (PR)](#_q79lgj5m7ehb) 29

[6.3 Features not to be testes](#_ovd9cwi8gfk0) 29

[6.4 Item pass / fail criteria](#_69ne6yp5lfb6) 30

[6.4.1 Exit Criteria](#_ledoi82uk27e) 30

[6.5 References](#_oy3xqdvf179b) 30

[6.6 Test design specifications](#_rtd0kw736tol) 30

[6.6.1 Useful\_Information Button(GUI\_Info)](#_9qkqpm8wg3qj) 30

[6.6.2 User Login Button (GUI\_UserLog)](#_4qryjnnf01lh) 30

[6.6.3 Guest Login Button(GUI\_GuesstLog)](#_wapvc5dkvx6p) 30

[6.6.4 See\_Old\_Reports Button(GUI\_SeeRep)](#_bmshulwz2wce) 30

[6.6.5 Make\_New\_Comparison Button (GUI\_NewComp)](#_8ch69x8cu233) 31

[6.6.6 Yes\_Patent Button(GUI\_Yespat)](#_r10nk5wfhhbp) 31

[6.6.7 No\_Patent Button(GUI\_Nopat)](#_92dlgtup16z5) 31

[6.6.8 Save\_Info Button(GUI\_Save)](#_fkt2lntvt7z3) 31

[6.6.9 Exit\_Button(GUI\_Ex)](#_3o4aii75nbq0) 31

[6.6.10 Admin Login (GUI.ADLG)](#_590dxne3947o) 31

[6.6.11 Admin Control (GUI.ADC\_ADDQ)](#_7gmpkiggylzi) 31

[6.6.12 Admin Control Delete Patent (GUI.ADC\_DELQ)](#_huy5vo5frrqz) 31

[6.6.13 Admin Control Update Patent(GUI.ADC\_UPDQ)](#_nb1cp3ek6z9e) 31

[6.6.14 Test Cases](#_1k8awz1w62kl) 32

[6.6.15 Adding a Patent (AP)](#_ggm7c69v85af) 32

[6.6.16 Patent Comparison (PC)](#_e3o0ybuf915j) 32

[6.6.17 Patent Result (PR)](#_ap5o502vzxwy) 32

[6.6.17.1 Test Cases](#_aw97n7z75q3l) 33

[6.7 Detailed Test Cases](#_x3z6qgsfovjz) 33

[6.7.1 AP.AD.01](#_oupmhixsk1a7) 33

[6.7.2 AP.GT.01](#_hseal9zgffsa) 34

[6.7.3 PC.AD.01](#_bxigioe01ciy) 34

[6.7.4 PC.GT.01](#_yh212p5ypzzw) 35

[6.7.5 PR.AD.01](#_lnpe9epzpxbz) 35

[6.7.6 PR.GT.01](#_6d3uai1r5hu7) 36

[**7. Test Results**](#_n5ip9m1w6cwe)36

[7.1 Individual Test Results](#_t4qku5p5ogwj) 36

[7.2 Summary of Test Result](#_1wo8ha5cnzad) 36

[7.3 Exit Criteria](#_2kx1oeud2v4l) 36

[7.4 Known Problems](#_nzef37p5qa0v) 36

[7.5 Conclusion](#_sw5fm6y0gga4) 37

[**8. Conclusions**](#_tyhm2x8linpx)37

# 

**Figure Table**

[**Figure 1-Deployment Diagram** 19](#_Toc41684420)

[**Figure 2-Home Page** 23](#_Toc41684421)

[**Figure 3-Help System Menu** 23](#_Toc41684422)

[**Figure 4-Patent Information Form** 24](#_Toc41684423)

[**Figure 5-General Information** 24](#_Toc41684424)

[**Figure 6-Frequently Asked Questions** 25](#_Toc41684425)

[**Figure 7-Our Mission and Vision** 25](#_Toc41684426)

[**Figure 8-Contact page** 26](#_Toc41684427)

[**Figure 9-Result Page** 26](#_Toc41684428)

[**Figure 10-Login Page** 27](#_Toc41684429)

[**Figure 11-Reset Password** 27](#_Toc41684430)

[**Figure 12-Submit Request** 37](#_Toc41684431)

[**Figure 13-Bert Serving Starting** 37](#_Toc41684432)

[**Figure 14-Mongodb Screen** 38](#_Toc41684433)

# Abstract

Currently, there is no current practice in which the patent content can be compared.This situation; it may adversely affect people who have patent or want to have patent. For example, someone who wants to apply for a patent; may want to know if his invention has already been patented by someone else. If the patent holder is; they may wish to protect their rights when they learn whether their patent content has been copied by someone else.In this project; It is aimed that individuals who have a patent and want to have a patent find a patent content similarity rate by using our system.Dozens of different patents from different groups are already registered in our database.The user can continue as a guest or as a member while using our application. The similarity comparison of the two user types will be done differently.

# Key words

Patent, Similarity, Similarity Rate, Patent Comparison

# 1. Introduction

In order to gain ownership of an invention, it is necessary to obtain a patent. As a person or organization, it will be your first choice to own and protect your invention. In such cases, the best way to protect your invention is to patent it. However, more than one person may consider finding the same find. We need benchmarking systems to go with it and prove that it has never been done before. The Patent Comparison System has been designed with features not currently available. The system will be able to compare titles and content. The system can find similarity rates with two different algorithms. While selecting these algorithms, we made comparisons according to the tests of accuracy rates. The system will send you the report by e-mail and will also make the rate reflection on the screen. The system was designed with django and it was designed by choosing innovative technologies. A simple and useful system has been developed.

## 1.1 Company Background

Onur Engineering was established in 1980. It is an institution serving in the defense industry. Company; It conducts projects in areas such as Voice Communication Systems, Air Traffic Control Systems, Network Supported Talent and Communication Systems, Cyber ​​Security Systems and Meteorological Systems.

Onur Engineering, which has been in the sector for many years, is the main contractor of many important companies.

## 1.2 Problem Statement

When people apply for patents, it is a long process to look for similar patents and have patents approved. Here we have developed a platform to shorten time and enable people to learn patent results more easily. This platform has increased the quality of the patent search process in terms of time and scope in the research conducted at the Turkish Patent Institute. This project also continues to demonstrate limited comparison features of existing patent comparison systems and to develop an alternative, more advanced patent comparison system.

Briefly, the invention is defined in the art as the solution of a technical problem. The most convenient way for an individual to benefit from the invention is to provide patent protection for the invention. The transformation of an invention into a patent (registration as a patent) is the result of the process that begins when the inventor applies to a patent office. The inventors have in Turkey, Turkish Patent Institute is doing patent application. TPE has 45 patent classes. The purpose of our project is to find out which patent belongs to which class and whether there is another patent with the same content in the same class.

## 1.3 Related Work

Our program, which works on patent search and similarity display, has variables that perform similar operations. There are many different sites for patent comparison. Similarity algorithms allow us to search for similarities between patents using keywords and a patent number.

For example, as a similar application; USPTO Patent Database USPTO has its own patent database. This resource is free to use. You can search the USPTO below:

* classification number
* patent number
* Date of issue
* technical features in the patent
* keyword
* name of inventors
* company name
* application date

Second; Official database of the Espacenet European Patent Office (EPO).

You can download the full text part of the patents.

CIPO Canada Intellectual Property Office (CIPO) checks the Canadian patent database. Here you can search for patents obtained in French or English since the 1870s.

You can search patents using:

keyword

patent document number

patent date

Google translates patents into English. Like CIPO, you can search by patent number or keyword.

**Free Patents Online**

You can open an account in Free Patents Online and access the following.

US patents and applications

European patents

WIPO documents

Speed ​​dial is for US patents only, but advanced search also includes foreign patents.

You can save and download the patents you find. These search algorithms allow you to search among patents using specific phrases.

## 1.5 Contribution

Our solution is designed to develop any existing system. Background: Our database has been selected for serial and effective results in the background of the project. We chose mongoDB here because of the quick search function in the articles.

# 2. Literature Search

## 2.1 Library Research

We took care to choose the versions of tools suitable for the libraries we selected. We used the nltk, stopwords and word\_tokenize libraries for data cleaning.

## 2.2 Internet Research

While doing internet research, we first focused on finding text comparison articles. We paid attention to the fact that these articles were written by people who are experienced in this field. Then, we tried to read articles about the algorithm logic of sites that use the textual similarity algorithm such as google, yandex, etc.We shared every website we found with each other and tried to decide whether that site was reliable or not.When we decided that the website is reliable, we decided on the requirements related to our project .

# 3. Summary

# 3.1 Summary of Conceptual Solution

To accurately compare Patent texts, we first needed an algorithm. After doing the necessary research, the best algorithm for our project was the Bert algorithm. We integrated this algorithm into our project to ensure that the system provides accurate and reliable results. 2 different options were offered for the guest user and the member user. If he is a guest user, he can compare patents but only see the similarity ratio as a percentage. If the user is a member, he can request a detailed report from us after making patent comparisons. In this detailed report, an annotated report is sent on which patents and what parts of the patents are similar. We receive email information from the user when signing up. If the user requests a detailed report, the report to be prepared is sent to his / her email. When the research was done to write this project, we decided that Python was the most appropriate language. So we used Django for the website. We used MongoDB, which we thought was the most efficient NOSQL, for our database.

## 3.2 Technology Used

In this project, we used Python, MongoDB and Visual Studio Code as software tools. We preferred a NOSQL - based database for efficient search on the document.

# 4. Software Requirements Specification

## 4.1 Introduction

### 4.1.1 Purpose

The purpose of this document is to provide information and details about the project to be conducted on patent comparison. Patent comparison will be developed in our project to find similarities of different products with multiple features and to prevent the repeated use of the same names.The most important goal here is that companies will be able to follow up on their own without having to pay extra fees for patent follow-up.

### 4.1.2 Scope of Project

**Subsection of project**

**1.**We used MongoDB as the database management system. We preferred Django for web programming. We performed patent search with our search algorithm.

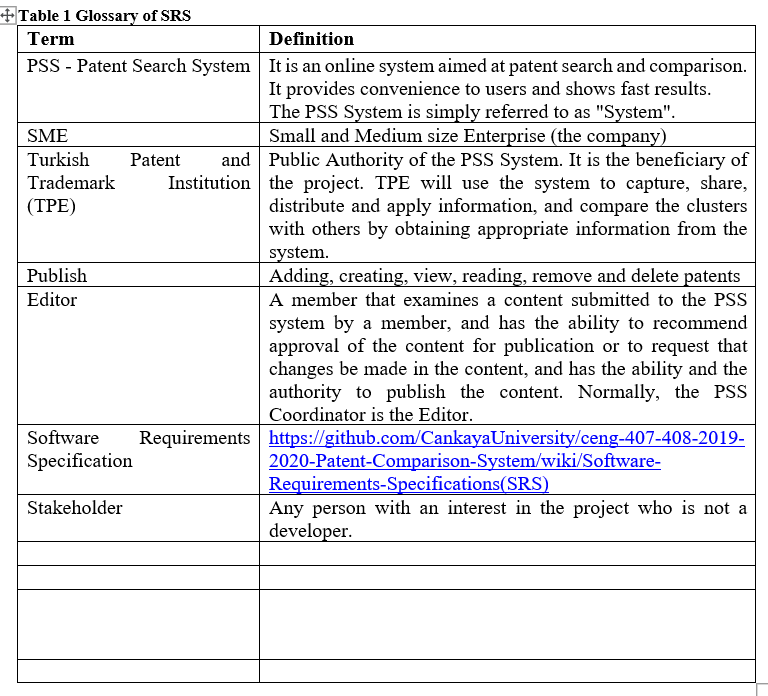
**2.**Our project makes patent classification according to title and content. The patent comparison relates to the comparison of a patented product with a product intended to be produced. Patent Analysis and Comparison is indispensable for preventing negative material and moral consequences that may be encountered later and for correct R & D configuration. We realized programming with Java or Python in our project. After our data set was provided by our company, our first job was to process this data in the database. In doing so, we used various classification and clustering algorithms. Thanks to the structure we created, we performed our transactions quickly and accurately.

**3.**Our goal is for users to quickly and actively compare patents and see similarity rates. The biggest goal of the system is to perform transactions by taking less time from the users

**4.**No specification is needed.

### 4.1.3 Glossary

A patent for an inventions is granted by government to the inventor, giving the inventor the right to stop others, for a Limited period, from making, using or selling the invention without their permission.



### 4.1.4 References

1. SCRUM NEDİR?

[https://medium.com/@secilcor/scrum-nedi%CC%87r-6a4326951dd8](https://medium.com/@secilcor/scrum-nedi%CC%87r-6a4326951dd8(Development)

1. Top 4 software development methodologies

<https://www.synopsys.com/blogs/software-security/top-4-software-development-methodologies>

### 4.1.5 Overview of Document

The remaining content of our SRS document includes product content and usage interfaces. Our SRS document is organized in two stages. The first is to determine the specifications at the beginning of the project and the second is during the project coding.

## 4.2 Overall Description

Nosql database was preferred because we have a large data set and a lot of data will be processed. We used visual studio code. Github was preferred for its support and interactivity. While doing this project, we wanted to use new and effective technologies. That's why we chose the Django framework.

### 4.2.1 Product Perspective

Patent Comparison System is a web project aiming to compare patents and give a certain ratio and reporting. The Patent Comparison System project will be implemented using technologies such as machine learning, language processing and search algorithms.

### 4.2.2 Development Methodology

Successful projects are managed well.Therefore, it is very important that we choose the methodology that will work best in our project.While developing the project, we have decided to use Scrum which is an agile software development methodology. Scrum; is one of the project management methodologies and it is used to manage complex software processes. In performing this management, it split the whole and follows a method based on repetition. It provides that the target is achieved through regular feedback and planning. It has a structure that is flexible for needs and open to innovations. Communication and teamwork are very important. The one most advantage of scrum is that reviewing each sprint before moving to another that testing is conducted throughout the process, so permits teams to change the scope of the project at whatever point.

### 4.2.3 Product Functions

Our website consists of multiple pages. Each page has its own function and image. We can list these pages as follows;

1. Login Page
2. Register Page
3. Add patent
4. Description Pages
5. Contact Page
6. Result Page

#### 4.2.3.1 Login Page

After the user becomes a member, they can login or continue as a guest.

The related sub-functions are the following:

User\_Login\_as\_Member;

4.2.3.2 Register Page

It is aimed to have 2 different user types in our application. These are Member user and guest user. User types are planned to have different rights

The related sub-functions are the following:

User\_Register\_as\_Guest;

User\_Register\_as\_Member;

#### 4.2.3.3 Add Patent

Member user and guest user can start the benchmarking process by entering the requested information on the patent comparison form

The related sub-functions are the following:

Member\_User\_Add\_Patent;

Guest\_User\_Add\_Patent;

#### 4.2.3.4 Description Page

There will be buttons in our application to inform the user. The user can go to these pages to get information about the patent.

#### 4.2.3.5 Contact Page

Users need to fill out a form to contact Site developers

The related sub-functions are the following:

User\_Contact\_Developers;

#### 4.2.3.6 Result Page

After the patent comparison is realized, the result page will be available for the user to see more clearly.

### 4.2.4 User Characteristics

The types of users that we aim to use this application are people who want to have a patent and people who already have a patent.

#### 4.2.4.1 Patent Owners

Patents owners using our web site can learn whether their patent content is copied or not using . In this way they can protect their own rights

#### 4.2.4.2 People Who Want to Apply for a Patent

People who are going to apply for a patent can find out whether their inventions have been made by someone else before using our application. In this way, he can change his invention before applying for a patent (if his invention is highly similar to other inventions).

### 4.2.5 Constraints

This subsection of the SRS should provide a general description of any other items that will limit the developer’s options. These include

1. Finding patent data;
2. Safety and security considerations
3. Higher-order language requirements;
4. Reliability requirements;
5. Time;

### 4.2.6 Assumptions and Dependencies

We accept the patent contents entered by the users and the patent contents entered before.

## 4.3 Requirements Specification

### 4.3.1 External Interface Requirements

1. Source of Input: Users
2. Timing: When the user encounters a malfunction
3. Accuracy of Knowledge: We accept the patent information entered by the user correctly.
4. Unit of Measure: Day.
5. Relationships to Other Inputs: None.
6. Screen Format/Organization Scroll Page.
7. Window Format/Organization: Multiple Page.
8. Command Formats: None.
9. End Messages: Result Page

#### 

#### 4.3.1.1 User interfaces

Our software will be able to work actively on all web-based platforms.

What you can do in the user interface:

* Register to the system,
* Login to the system,
* Search for patents,
* Update Patent database,
* Create Patent report,
* Create Simple Patent report (for Non-members)

Unlike the user, the administrator will be able to make the features listed below.

* Test and train the system,
* Can comment on files uploaded by the user,
* Add data to database,
* Will be able to access the information uploaded by the user,
* Can data statistics in uploaded files

#### 4.3.1.2 Hardware interfaces

##### 4.3.1.2.1 Server Side

Patent Comparion System , is a Digital UNIX (64bit), 128 RAM, 24 GB Hard Disk computer with a 100 Mbps access to the CU fiber-optic backbone.

##### 4.3.1.2.2 Client Side

Any Personal computer, which can support any X-window or Windows environment with a mouse support, is acceptable.

#### 4.3.1.3 Software interfaces

We developed a web-based software interface during the testing phase of this project. We used the following programs at the software stage.

* Visual Studio Code
* Django
* MongoDB

#### 4.3.1.4 Communications interfaces

There are no external communications interface requirements.

### 4.3.2 Functional Requirements

#### 4.3.2.1 Add Patent

In order to make a patent comparison, the user must enter the required information in the form that comes up.

#### 4.3.2.2 Create User

User must register by entering the requested information in order to use our application.

#### 4.3.2.3 Create Result

After the comparison is made, the user will be shown a result page with the similarity rate.

### 4.3.3 Performance Requirements

The large size of our data set will significantly delay the user finding patent similarity rate. So we're going to need a quick search algorithm in this project. In order to give the user similarity rate in a short time, we need a quick search algorithm in this Project. We also need to create word classes. According to these classes, it will be easier to find out which category the patent belongs to.

### 4.3.4 Design constraints

#### 4.3.4.1 Usability

This system should be designed as an easy-to-understand interface that people with little internet experience can use.

#### 4.3.4.2 Standards compliance

Our project reports are prepared in formats compliant with IEEE standards. By enumerating for each requirement, we saved time during the project construction phase. By creating tables and features that are easy to understand in the database, we have enabled the third parties to easily understand the project.

4.3.5 Software system attributes

#### 4.3.5.1 Reliability

* The personal information and patent information entered by the user will not be shared with third parties.

#### 4.3.5.2 Security

* The system has an authorization mechanism for users to enter their profiles. Therefore, different users will have different levels of authorization to access the data.
* Users' information will not be shared with third parties.

#### 4.3.5.3 Portability

* Patent Comparison System will be developed as a web application by using Java or Python programming language. So, our project will be available at all platforms like Mobile Phones, Tablets and PCs having an Internet browser application.
* The system is a web based project and it not require different hardware for users.
* All computers and phones will be available to use our system.

#### 4.3.5.4 Performance

* Patent name search should be fast.
* The user should not be able to obtain patent names previously obtained.
* The data should be classified well in the database.

#### 4.3.5.5 Usability

* If the participant does not fill the places that need to be filled, he/she will receive an error message.

#### 4.3.5.6 Adaptability

* Patents must be in English language.

#### 4.3.5.7 Scalability

* Many participant uses the system at a time, there is scalability requirement.
* The system may generate a similarity ratio in line with the criteria we set.

#### 4.3.5.8 Performance

* Patent name search should be fast.
* The user should not be able to obtain patent names previously obtained.
* The data should be classified well in the database.

### 4.3.6 Other Requirements

There is no other requirement in our project.

# **5. Software Design Description(SDD)**

## **5.1 Introduction**

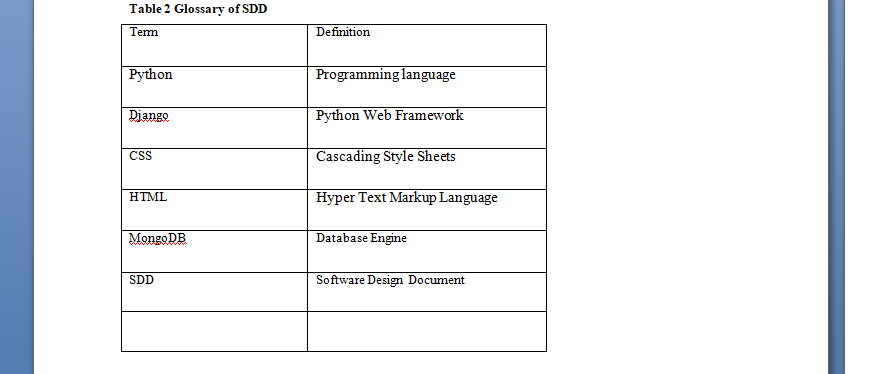
### **5.1.1 Purpose**

The purpose of this Software Design Description, "a program capable of Contents Benchmarking Benchmarking Patent" is entitled to provide the details of the project. Target audience, people who want to own and have all invention. The system will create opportunities to compare Patent contents. The program will serve to show us the similarity rate is another patent of the patent. The aim of the Patent Comparison System project is to design the “Content Comparative Patent System için to find the similarities of the contents, both innovative and different from other comparison programs, taking into account the currently used Patent comparison programs. The system will operate in two main modes. The first is “User” mode and the other is “Guest” mode. In User mode, the user will be able to compare the patent he / she wants to compare in detail and see the similarity ratio. It can also generate a patent report or search for patents. In Guest mode, the user will be able to perform limited operations on the system without being a member.

### **5.1.2 Scope**

This document provides a complete description of the design of the Patent Comparison System Project. Procedures for creating, programming, and designing the system environment through Python programming languages will be performed. We decided to use Clustering Algorithms for the operation of the system (K-Means, Neural Networks etc.) We will design our system in English. When the patents are published in Turkish patent institute, our system cannot read it because it is in pdf format. In finding similarity, we will set limits for ourselves and act within that framework. First of all, after the completion of these parts, we intend to do the patent valuation, which is the main purpose of our project.

### **5.1.3 Glossary**

****

Web Site A place on the world wide web

### 

### 5.1.4 References

**1)Onur Mühendislik, 2019.** [Online] **Available at:** <https://www.onur.net/tr/kurumsal/tarih%C3%A7e-1/>

**2) Wang Chong-feng "The quality difference analysis between environmental patents and non-environmental patents in China", 2011 International Conference on Management Science & Engineering 18th Annual Conference Proceedings.** [Online] **Available at:** <https://ieeexplore.ieee.org/document/6070135>

**3) Loet Leydesdorff; Dieter Franz Kogler Bowen Yan "Mapping patent classifications: portfolio and statistical analysis, and the comparison of strengths and weaknesses", September 2017.** [Online] **Available at:** <https://www.researchgate.net/publication/314080370_Mapping_Patent_Classifications_Portfolio_and_Statistical_Analysis_and_the_Comparison_of_Strengths_and_Weaknesses>

**4)**  **Patent Database Comparison: Everything You Need to Know**

<https://www.upcounsel.com/patent-database-comparison>

**5) Development Methodology** [Online] **Available at:**

<https://medium.com/@secilcor/scrum-nedi%CC%87r-6a4326951dd8>

**6) Development Methodology** [Online] **Available at:**

[**https://www.synopsys.com/blogs/software-security/top-4-software-development-methodologies/**](https://www.synopsys.com/blogs/software-security/top-4-software-development-methodologies/) **( Development Meth. )**

**7) D. Kukolj; Z. Tekic; Lj. Nikolic; Z. Panjkov; M. Pokric; M. Drazic; M. Vitas; D. Nemet "Comparison of Algorithms for Patent Documents Clusterization ", 2012 Proceedings of the 35th International Convention MIPRO,2012.** [Online] **Available at:** <https://www.sciencedirect.com/science/article/pii/S1877705814002379>

**8) Nadin Tamer ,**Natural Language: Swift ile Doğal Dil İşleme. [Online] **Available at:** <https://medium.com/turkishkit/natural-language-swift-ile-do%C4%9Fal-dil-i%CC%87%C5%9Fleme-ca316b8e218d>

**9) ]**Google Patents: The global patent search engine [Online ] **Available at:**

<https://www.researchgate.net/publication/280301154_Google_Patents_The_global_patent_search_engine>

### 

### 5.1.5 Overview of Document

The remaining chapters and their contents are listed below;

Section 5.2 contains Deployment Diagram;

* Section 5.3 is the Architectural Design which describes the project development phase. Also it contains class diagram of the system and architecture design of the simulation which describes actors, exceptions, basic sequences, priorities, preconditions and postconditions. Additionally, this section includes activity diagram of scenario generator.
* Section 5.4 contains Data Structure Diagram.
* Section 5.5 is Use Case Realization. In this section, a block diagram according to the SRS document is shown.
* Section 5.6 contains images of how our web interface is.
* In section 7, the Help System that our web interface will provide to the users is explained.

## **5.2 Deployment Diagram**

## 

**Figure 1-Deployment Diagram**

## 5.3 Architecture Design

**Summary:** This system is used by guest, user and administrator. The guest user does not need to log in. The guest can save personal information, exit system, compare patent. The user can enter, save and update personal information and exit system, compare patents, view reports and search for patents. In addition, an administrator can delete an account, approve participant accounts, and add a new administrator.

Basic arrays:

• If an account is not available, the user must register.

• The user must login to the system by entering the username and password.

• The user can update their personal information by selecting the update button from the user menu.

• User can search through patent search button.

• User can compare patent using patent comparison button.

• The user can view their reports.

• The user can browse the site without being a member.

• The administrator can delete a user account by selecting the delete button from the admin menu.

• The administrator can confirm a recently registered user account by selecting confirm from the admin menu.

• The administrator can add a new administrator by selecting the Add new administrator button from the admin menu.

• The user can exit the system by selecting the exit button.

Exception: Database connection may fail.

Message Conditions: None

### 5.3.1 Options Menu

**Summary:** The user can perform transactions on the system with or without a member. The user can see the similarity percentage of his searches with other patents.

**Actor :** User

**Prerequisite:** The user must be able to select the search from the options button.

**Basic Sequence:**

The user can search and become a member.

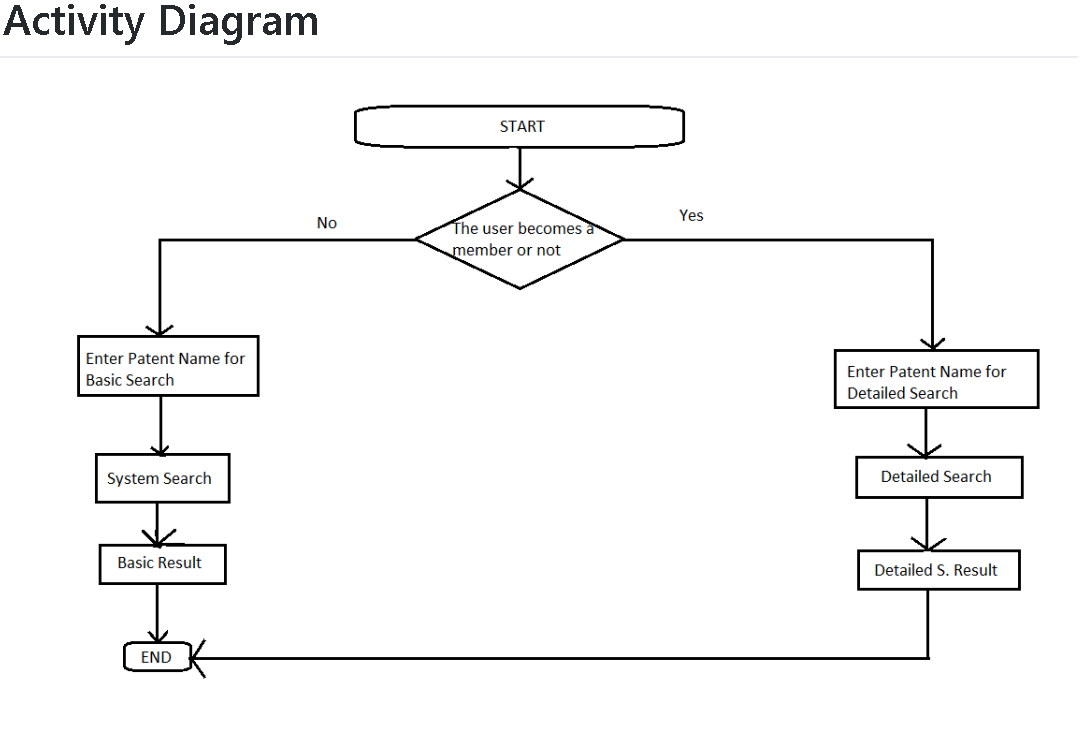
The user can select the continue button without being a member.

The user can view the search results.

**Exception:** None

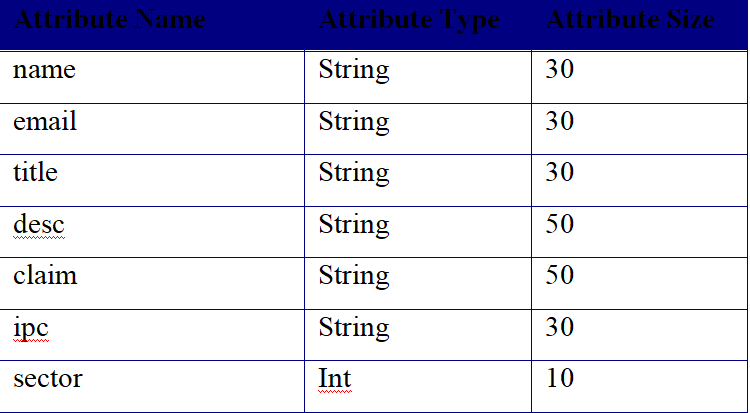
### 5.3.2 Class and Activity Diagram

### 

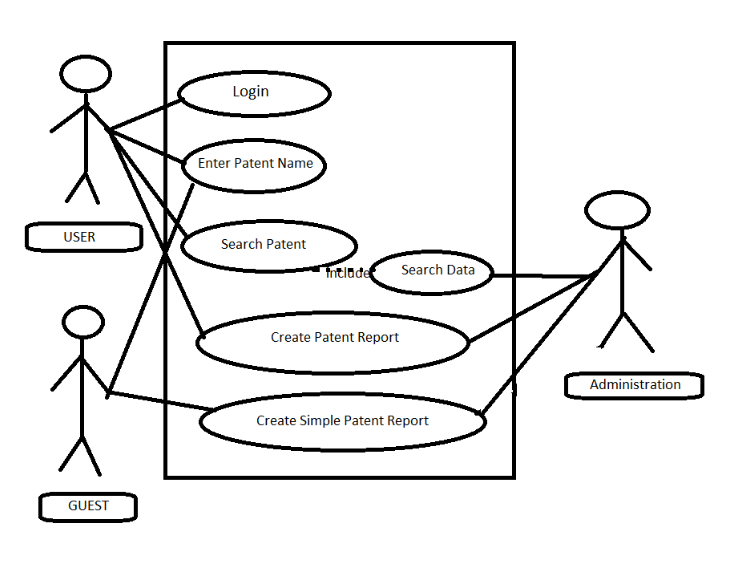


This diagram shows how our program can be used by users. The user can continue with or without a member. The user enters the search for the patent name he wishes to investigate. System searches according to similarity ratio. Then these rates are listed.

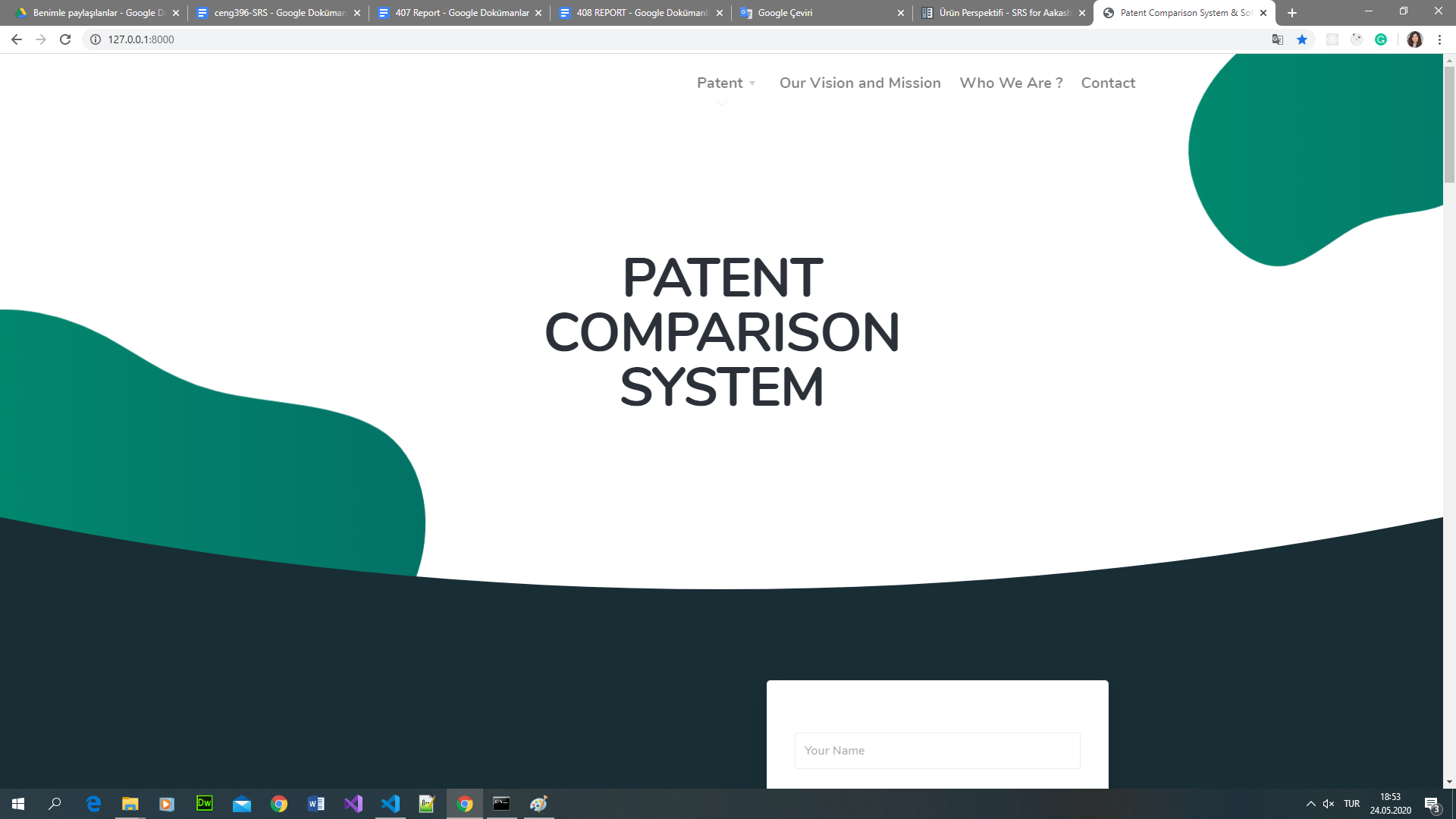
## 5.4 Data Structure Design



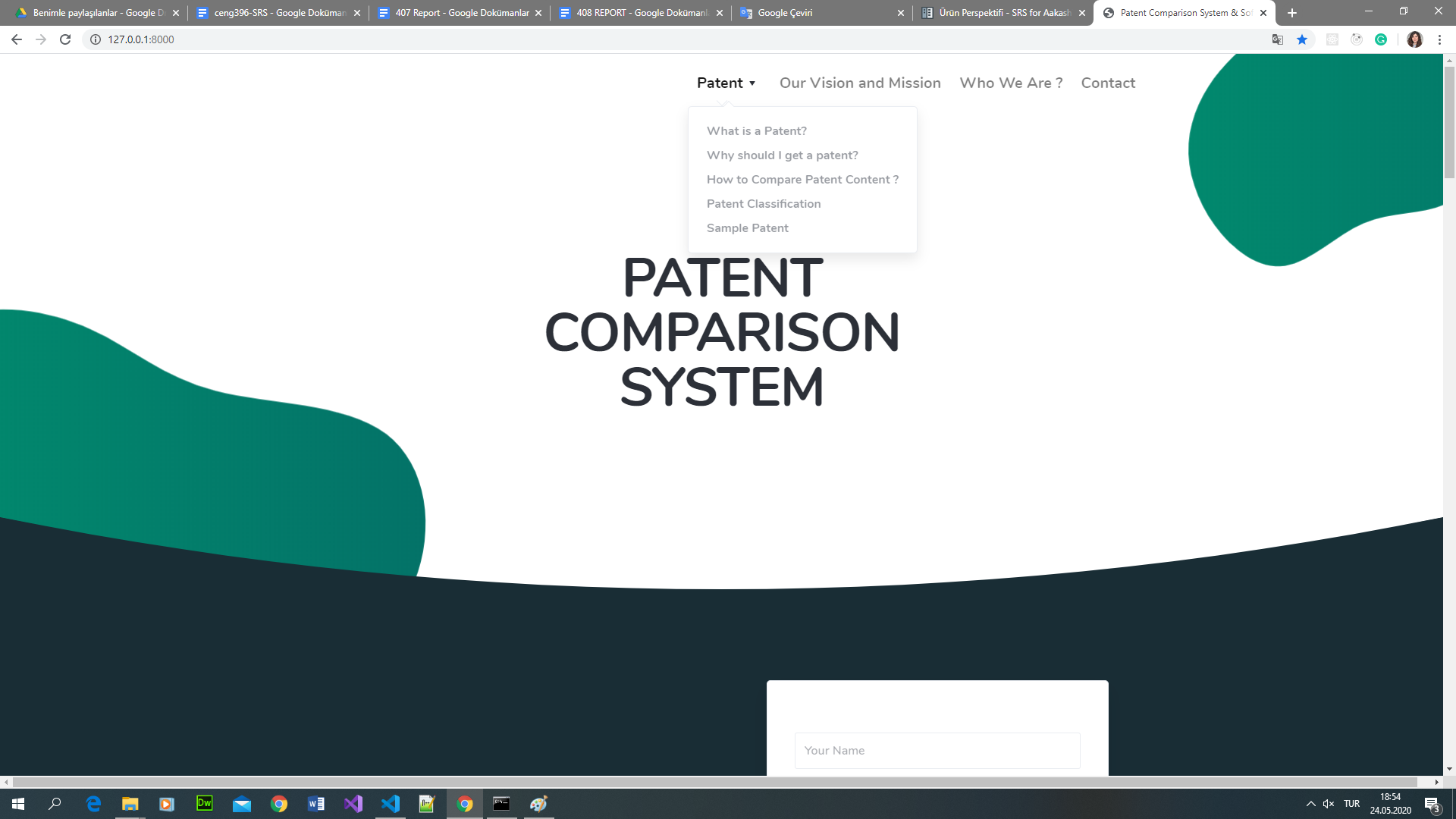
## 5.5 Use Case Realizations



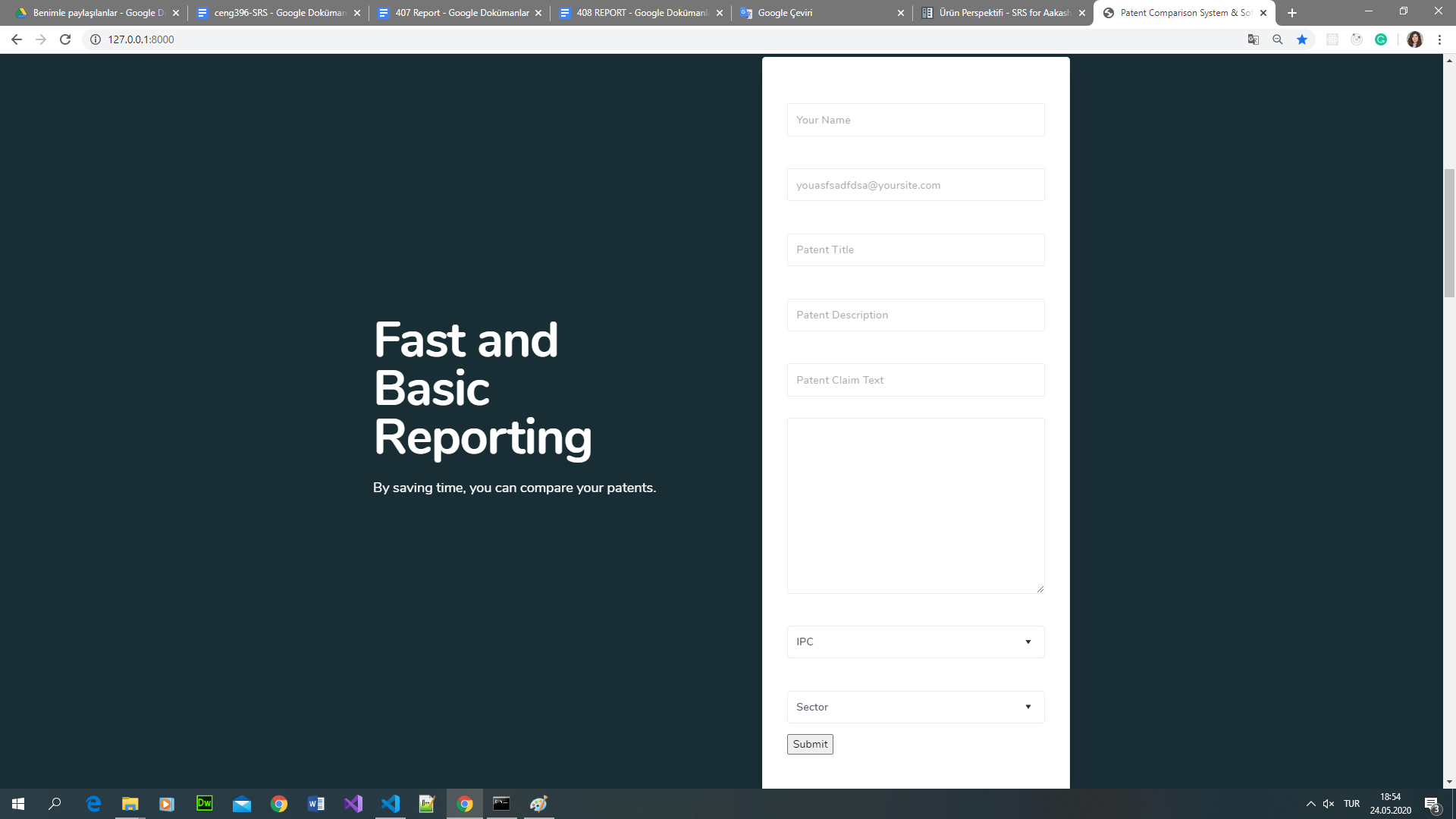
## 5.6 Interface Design



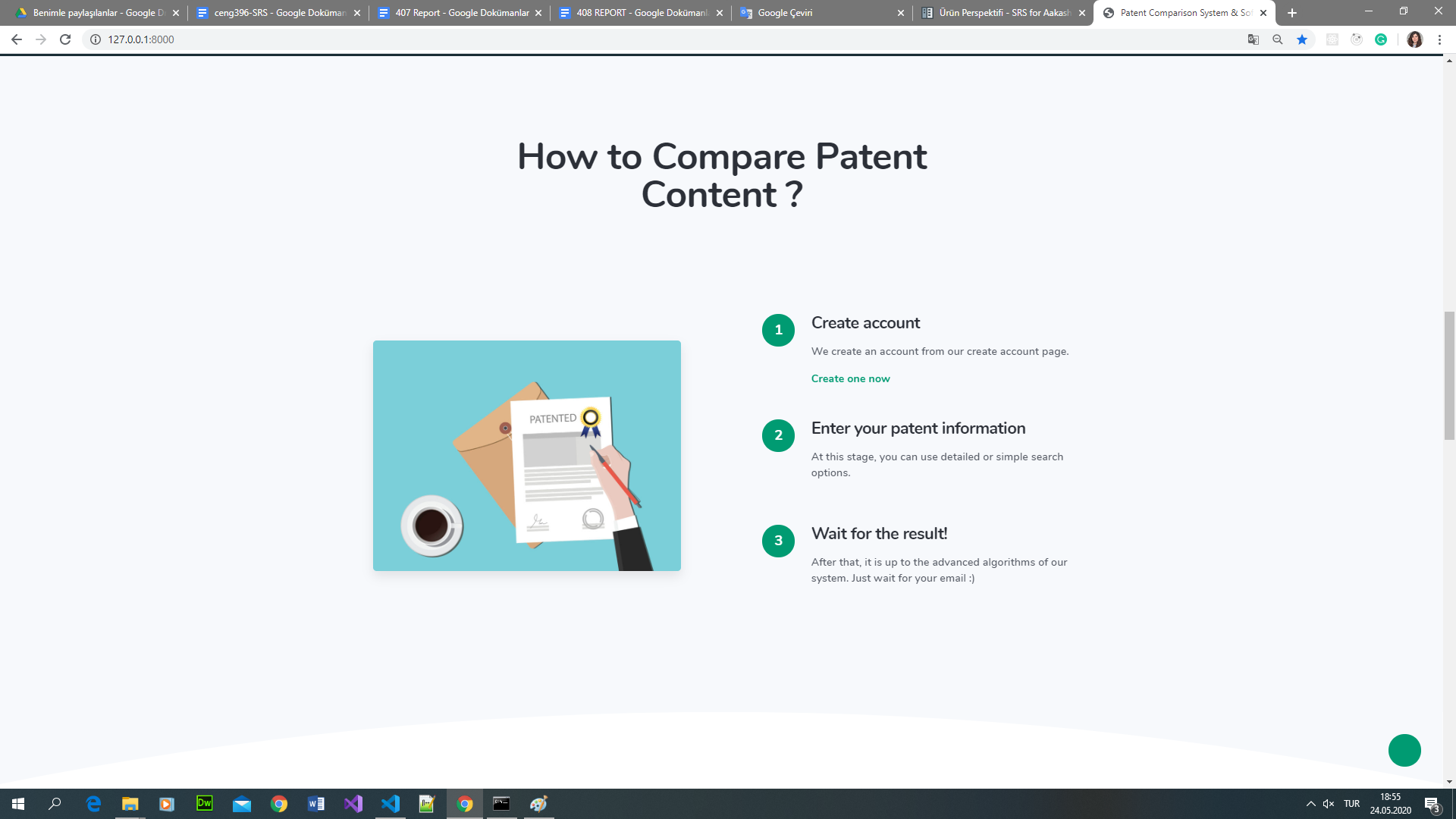
**Figure 2-Home Page**



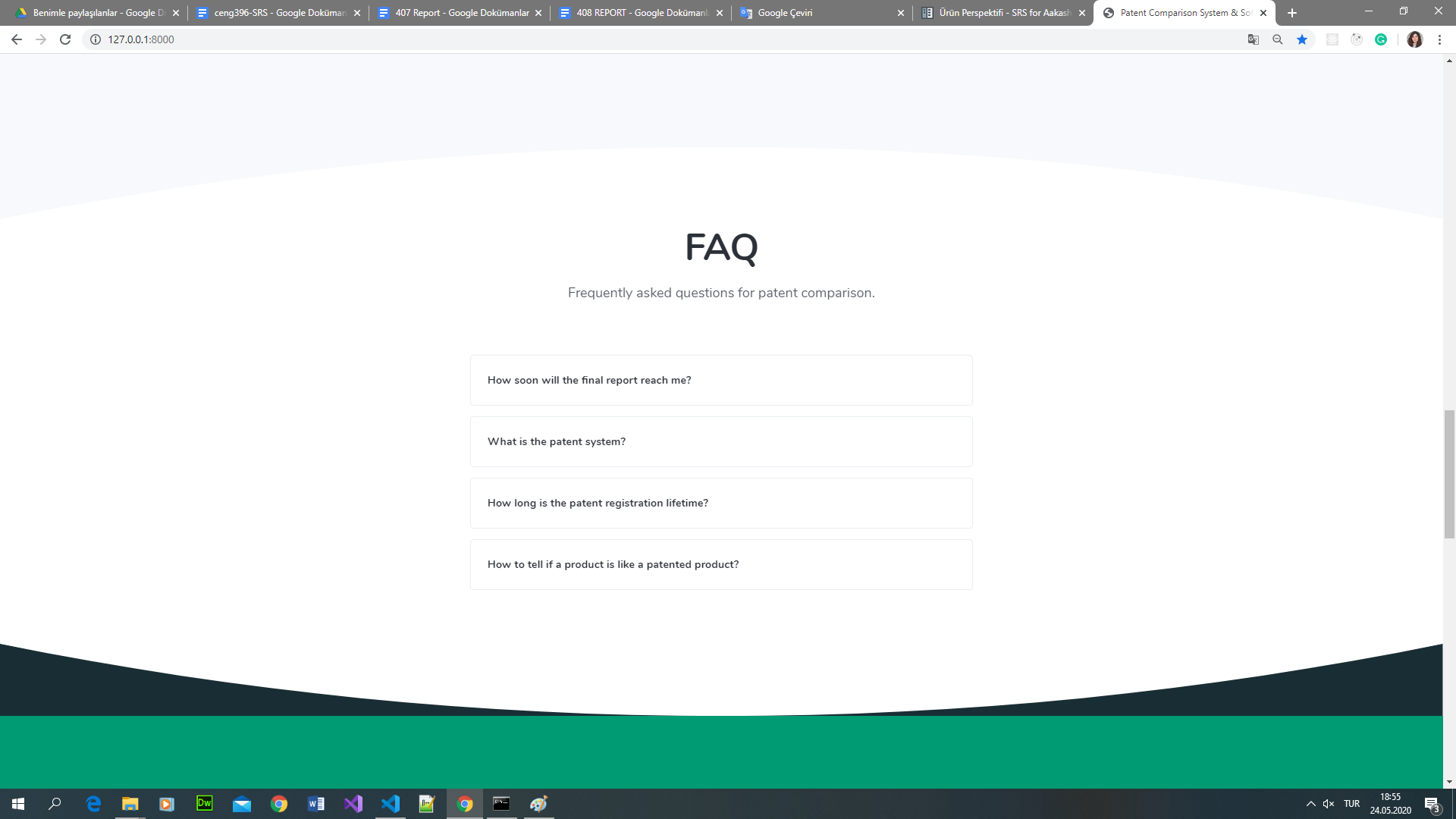
**Figure 3-Help System Menu**



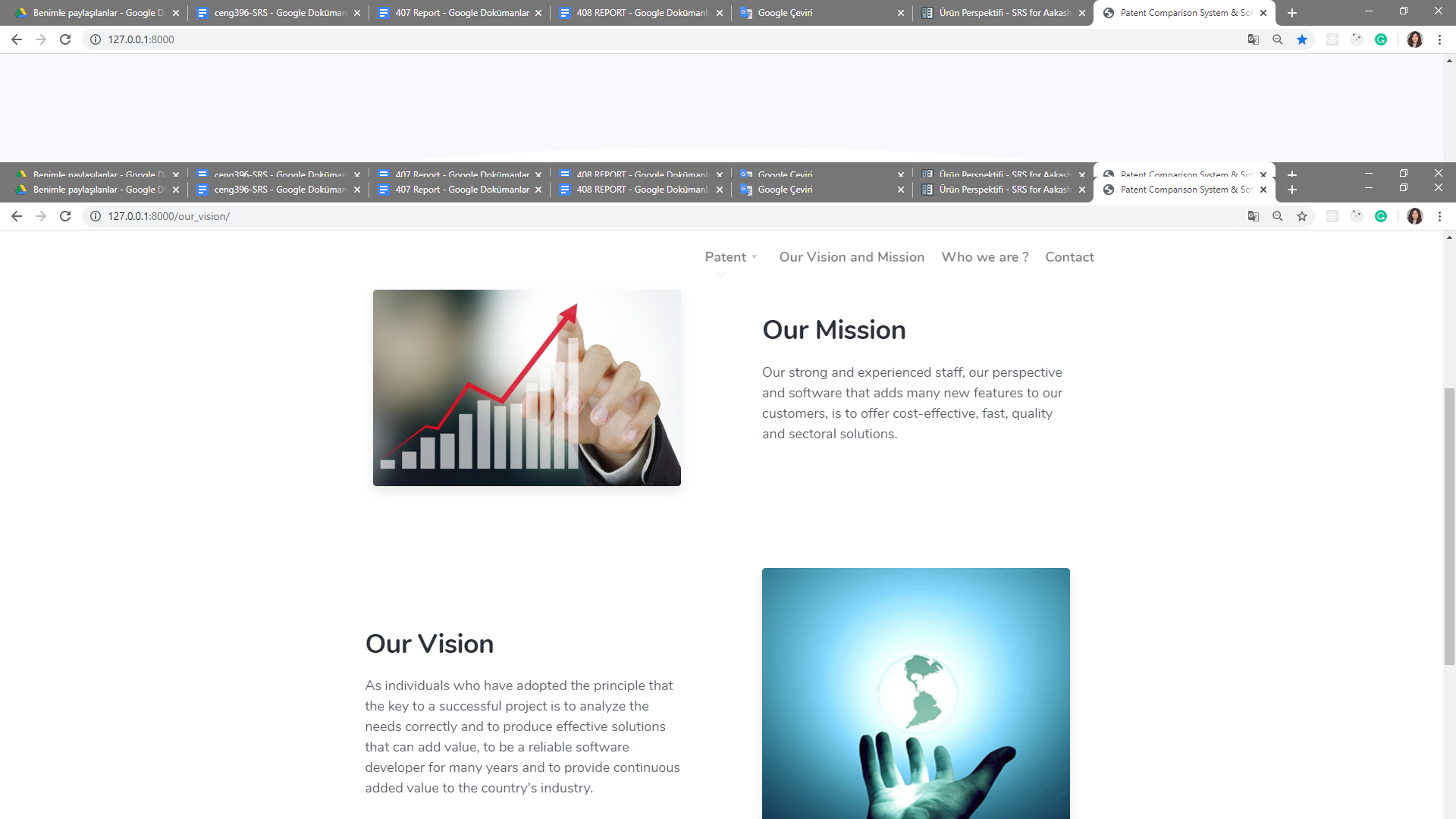
**Figure 4-Patent Information Form**



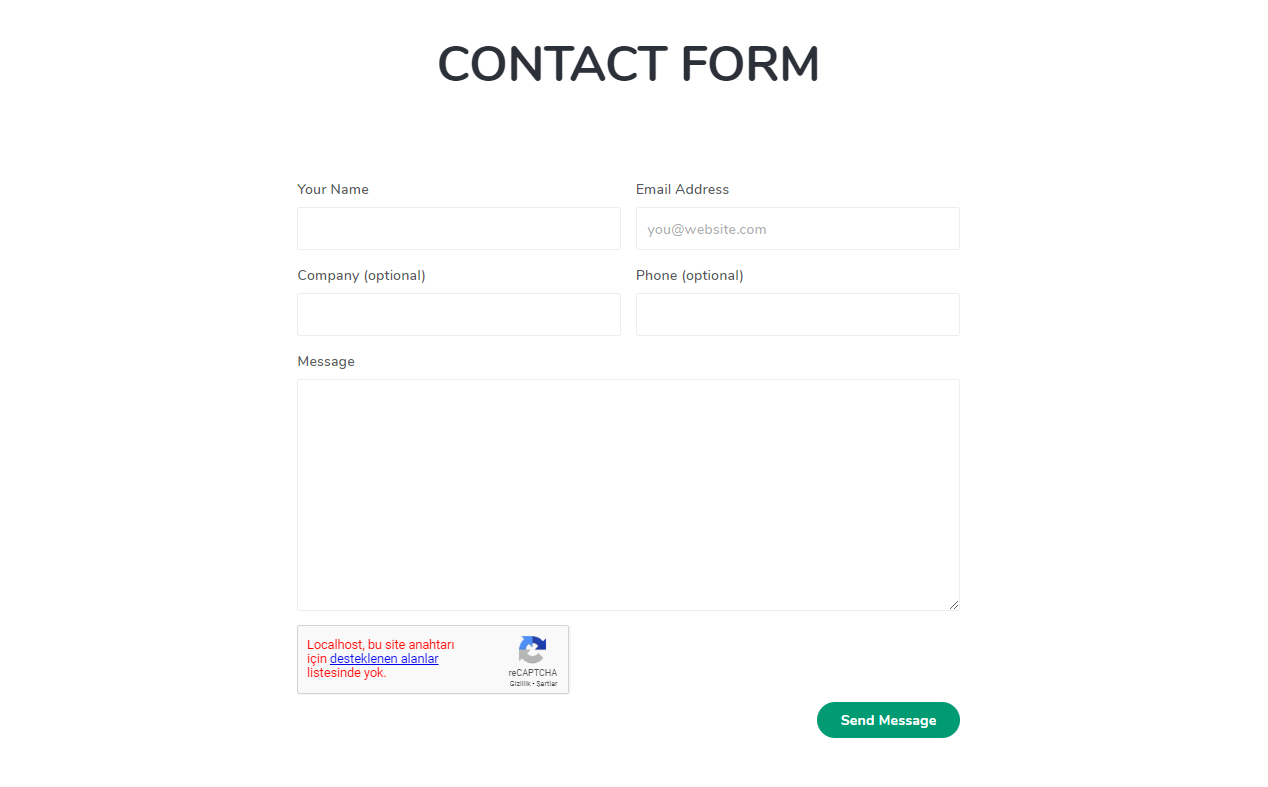
**Figure 5-General Information**



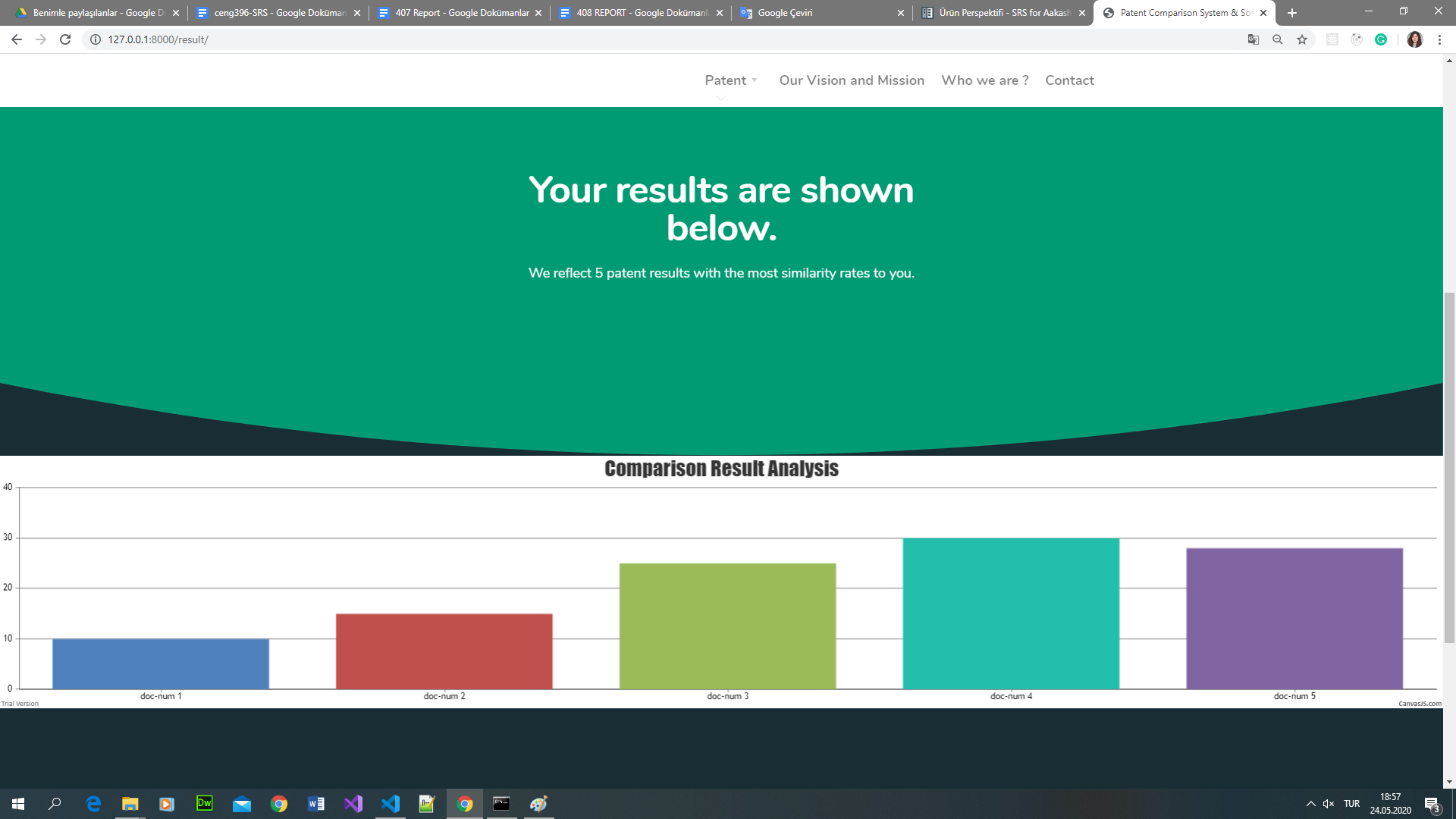
**Figure 6-Frequently Asked Questions**



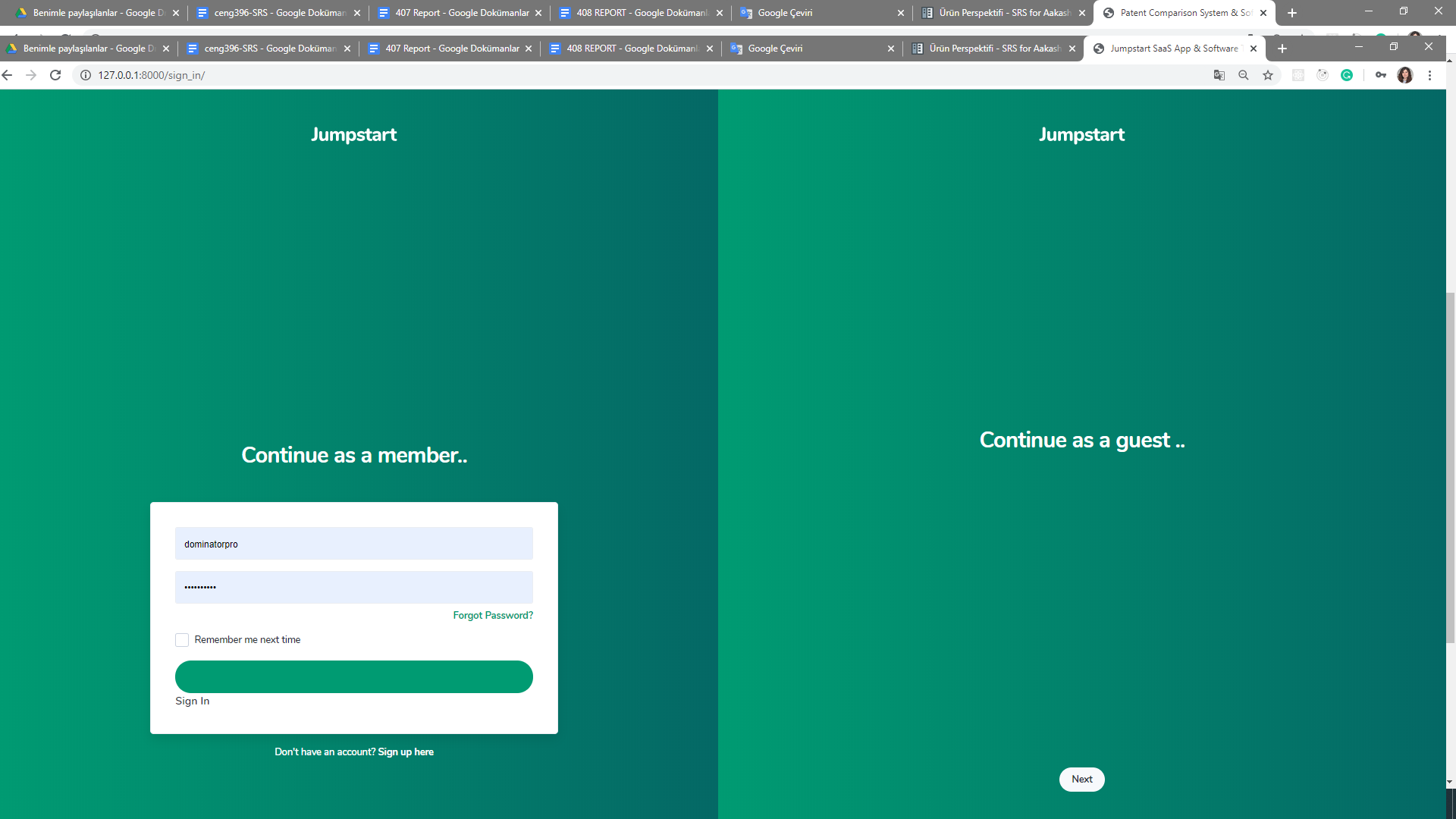
**Figure 7-Our Mission and Vision**



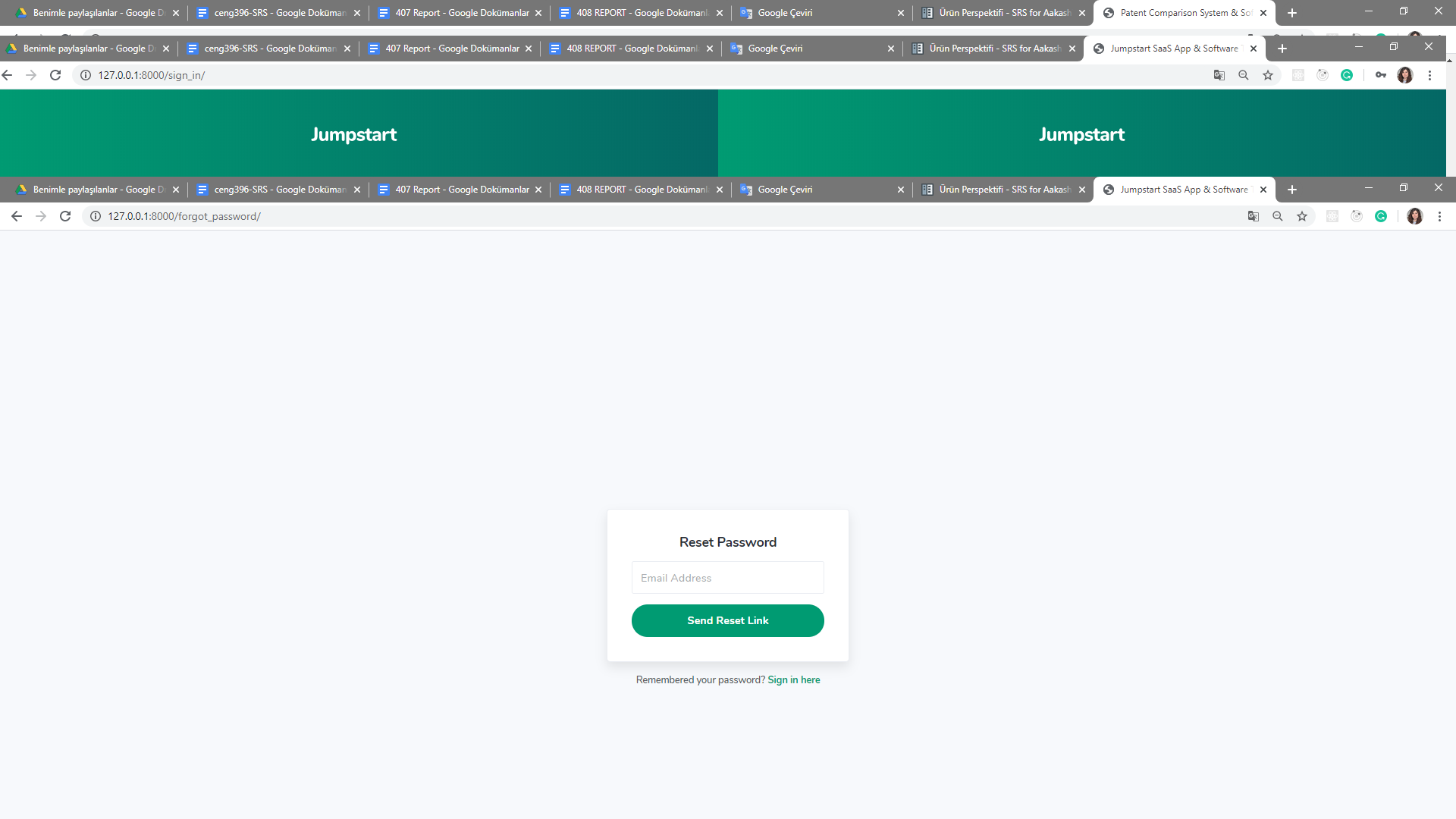
**Figure 8-Contact page**



**Figure 9-Result Page**



**Figure 10-Login Page**



**Figure 11-Reset Password**

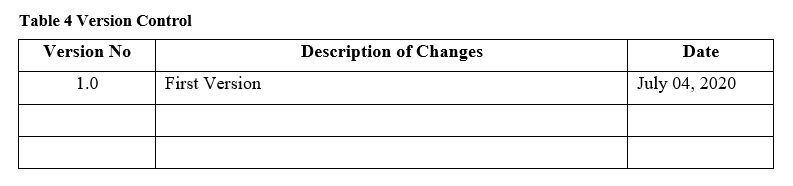
## 5.7 Help System Design

We have designed pages where the user can be informed about the patent. It is aimed to have enough information about our user application on our 'frequently asked questions' page. There is also a contact page where users can contact us

# **6. Test Plan**

## **6.1 Introduction**

### **6.1.1 Version Control**

****

### **6.1.2 Overview**

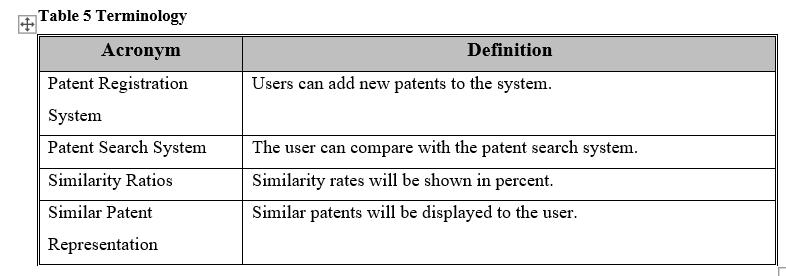
In the testing phase of our project, the effectiveness of our patent comparison system has been tested. At the end of this test, we aimed to show how reliable our project is and how useful it is.

### **6.1.3 Scope**

The efficiency of our project has been determined within the scope of this document. Briefly, it covers the topics on which our project has been tested. These topics

* General Analysis of Patents
* Similarity Ratios
* Ranking Speed of Similar Patents

### **6.1.4 Terminology**

****

## 6.2 Features to be tested

We tried to test every detail in our system. Our patent insertion system, patent similarity system, we finally tested our system of presenting similar patents.

### 6.2.1 Adding a Patent (AP)

In this section, the user enters the patent content. There are 2 types of users. Users will save the content of the invention to the system. This is the part where we check whether the patent data saved in the system by our users are recorded correctly or not.

### 6.2.2 Patent Comparison (PC)

In this section, we tested the system that allows users to compare patent files with other patent contents in the system and test the similarity given on the screen.

### 6.2.3 Patent Result (PR)

In this section, we tested the patents listed for our users.

## 6.3 Features not to be testes

We will determine the criteria for the patent comparison algorithm ourselves. These criteria may vary from person to person, so we cannot test the accuracy of the similarity rate.

## 6.4 Item pass / fail criteria

* Item Pass

In order to accept that our application is successful, the components in our system must work synchronously. All three components in our system (patent insertion, patent comparison, patent result) should work flawlessly.

* Fail Criteria

In order for our system to be considered unsuccessful, our patent addition, patent comparison and patent result display components should not work effectively.

### 6.4.1 Exit Criteria

* 100% Test scenarios all worked effectively.
* 60% Patent addition and patent comparison components worked effectively on the system.
* 0% No components we have prepared in our system have worked.

## 6.5 References

1. Software Design Document(SDD)

<https://github.com/CankayaUniversity/ceng-407-408-2019-2020-Patent-Comparison-System/wiki/Software-Design-Document(SDD)>

2) Software Requirements Specifications(SRS)

<https://github.com/CankayaUniversity/ceng-407-408-2019-2020-Patent-Comparison-System/wiki/Software-Requirements-Specifications(SRS)>

## 6.6 Test design specifications

### 6.6.1 Useful\_Information Button(GUI\_Info)

User can access detailed information about the patent by using this button.

### 6.6.2 User Login Button (GUI\_UserLog)

The user must login to the system by entering the password in order to use the system.

### 6.6.3 Guest Login Button(GUI\_GuesstLog)

Users who want to use the application as a guest can use this button

### 6.6.4 See\_Old\_Reports Button(GUI\_SeeRep)

Member users can use this button to see all the results of their comparison.

### 6.6.5 Make\_New\_Comparison Button (GUI\_NewComp)

All users can use this button to compare their patents.

### 6.6.6 Yes\_Patent Button(GUI\_Yespat)

User will use this button if he/she patent owner

### 6.6.7 No\_Patent Button(GUI\_Nopat)

If the user is not a patent holder, he/she will use this button

### 6.6.8 Save\_Info Button(GUI\_Save)

The user will use this button after entering the data to be entered. The data is saved to the database.

### 6.6.9 Exit\_Button(GUI\_Ex)

Participant can close the application with selecting “Exit” button.

### 6.6.10 Admin Login (GUI.ADLG)

Admin have to login to the system by entering password for being able to change patent.

### 6.6.11 Admin Control (GUI.ADC\_ADDQ)

Administrators can check the patent contents by entering the system.

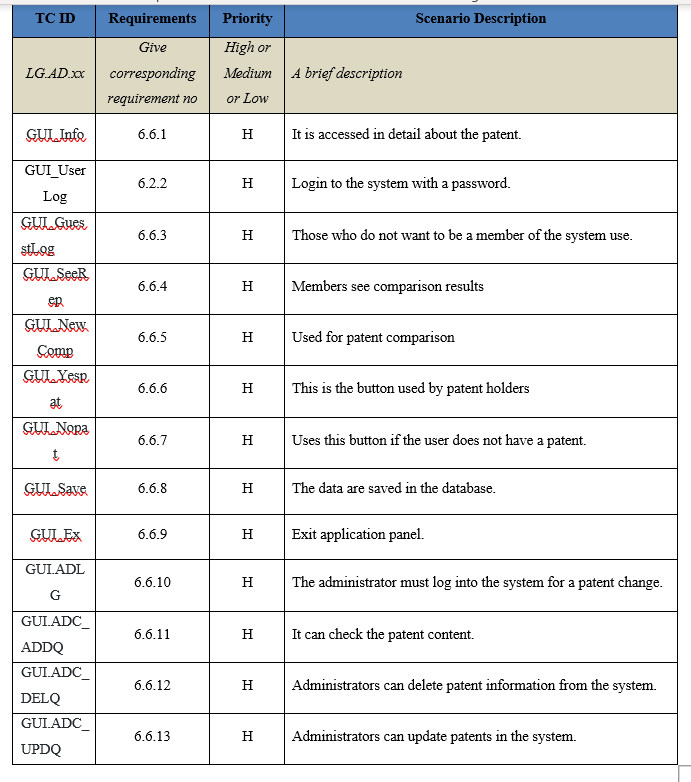
### 6.6.12 Admin Control Delete Patent (GUI.ADC\_DELQ)

Admins can delete patent from the system by entering the patent ID of the information from the information list.

### 6.6.13 Admin Control Update Patent(GUI.ADC\_UPDQ)

Admins can update patent from the system by entering the patent ID of the information from the information list.

### 6.6.14 Test Cases

****

### 6.6.15 Adding a Patent (AP)

Our members can add patents to our system.

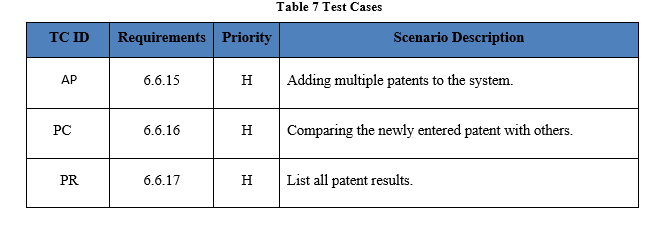
### 6.6.16 Patent Comparison (PC)

Those who use it with the patent comparison system can make a patent comparison.

### 6.6.17 Patent Result (PR)

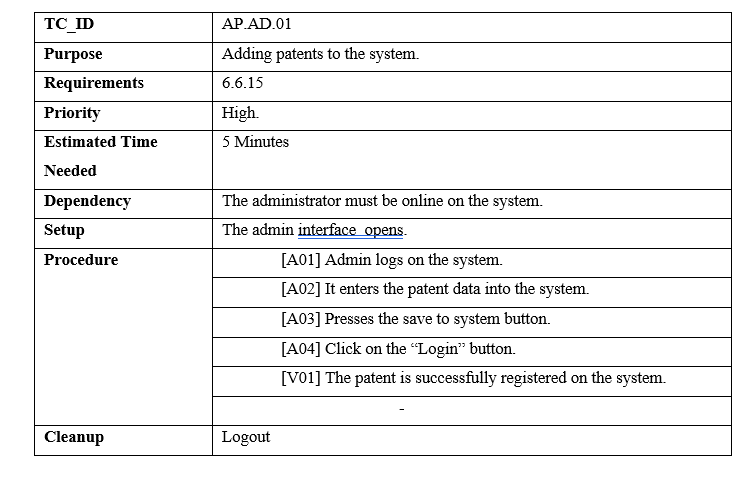
Users see the patent results they compare.

#### 6.6.17.1 Test Cases

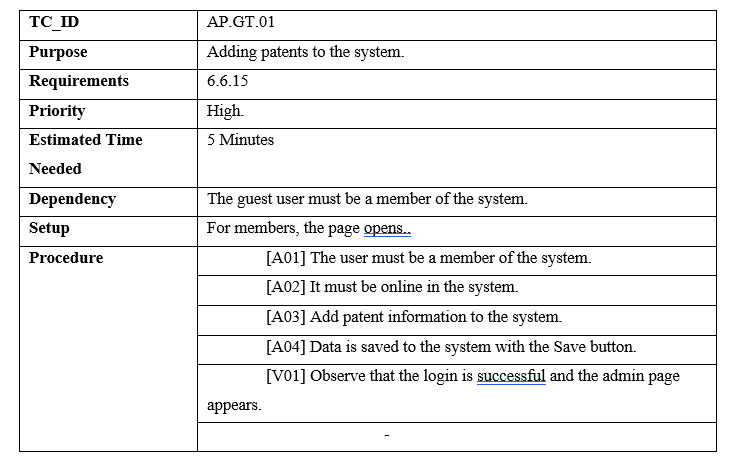
****

## 6.7 Detailed Test Cases

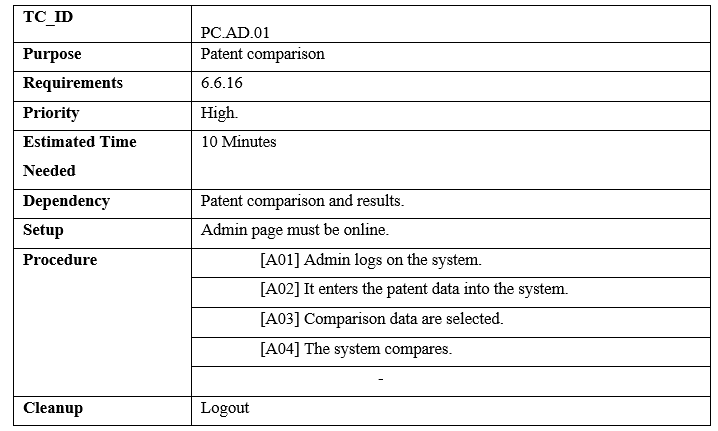
### 6.7.1 AP.AD.01

****

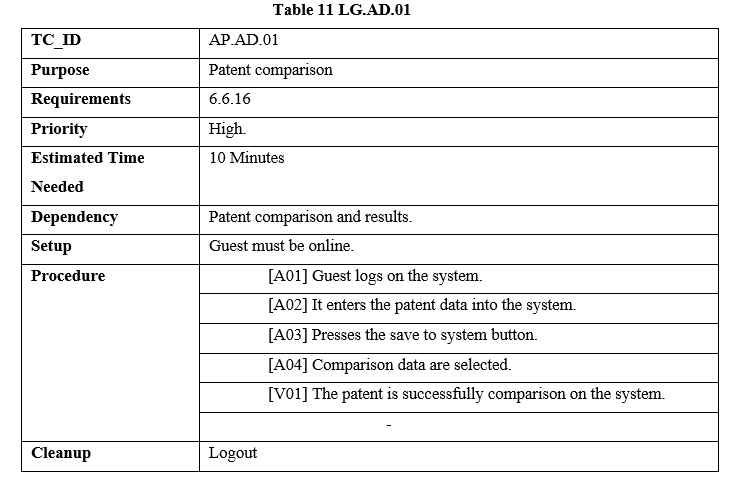
### 6.7.2 AP.GT.01

****

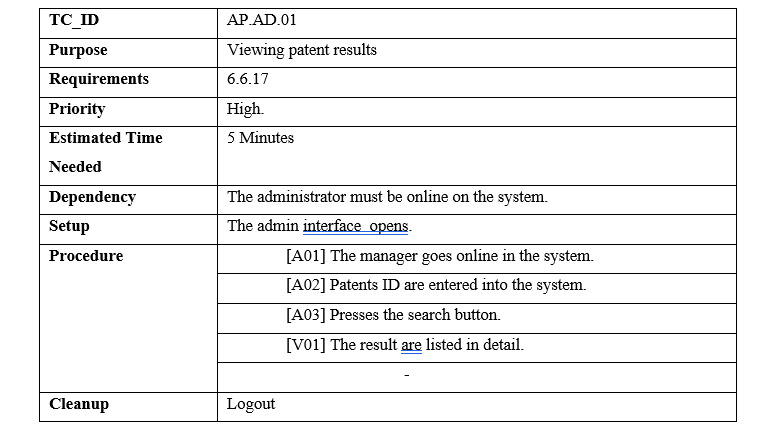
### 6.7.3 PC.AD.01

****

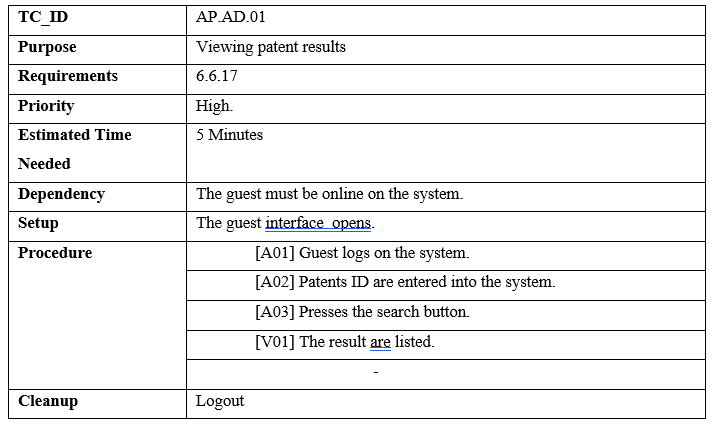
### 6.7.4 PC.GT.01

**** -

### 6.7.5 PR.AD.01

****

### 6.7.6 PR.GT.01

****

# 7. Test Results

## 7.1 Individual Test Results

All of the buttons that we will use in our web search have resulted successfully. Page redirects worked correctly. No problems were encountered during the transition between the pages. User registration was successful. User login was successful. The user's patent insertion was registered correctly in the database. The patent comparison of the user has been successful. The user exited from the system successfully.

## 7.2 Summary of Test Result

The features we have identified as the most important elements of the testing phase have been successfully realized. The extra features we want to add due to lack of time have not been tested since they are not added to the site. The patent addition and patent comparison, which are the center of our project, have been successful.

## 7.3 Known Problems

Since there is no distinction between guest and member users for the user type; a single patent comparison system was designed.”See Old Reports “ page not designed.

## 7.4 Conclusion

In this section describe the decision whether the product is ready for use or not.  

7.5 Conclusion

Our project successfully fulfills the critical features we have identified. These critical features are; is the person adding patents to the application, comparing patents and returning the similarity rate .In order to use the project more professionally, improvements should be made in the similarity algorithm.

# 8. Conclusions

We have designed an application in which patent contents can be compared. According to this application, the person will be able to find a similarity ratio  between the other patents registered in our database and his own patent. For this algorithm, we used the Bert algorithm, which is used by important search sites like google. We made text clearance to get more accurate results before the patent contents were compared. We made use of ready-made libraries using Python, the language with the program that contributed most to Natural Language Process.

We have learned that it may be much more efficient for the project to determine the requirements of the project as a result of more detailed research by someone who has done this work before. During the project construction phase, we learned that it is important to carefully examine all the tasks and distribute them according to the competencies of the group members. When the project started, we learned that it is important to save time for all its parts to proceed in parallel. In the software part, we learned how to use Django effectively. We gained experience in using Nosql database using MongoDB. We had an idea of ​​what to do in text analysis using natural language processing algorithms. We learned easy integration methods using Visual Studio Code for the software part.

One of the biggest advantages of our project; is to learn whether one's invention is similar to other inventions. If it is, is it at critical level, how much will it be able to find the answer.

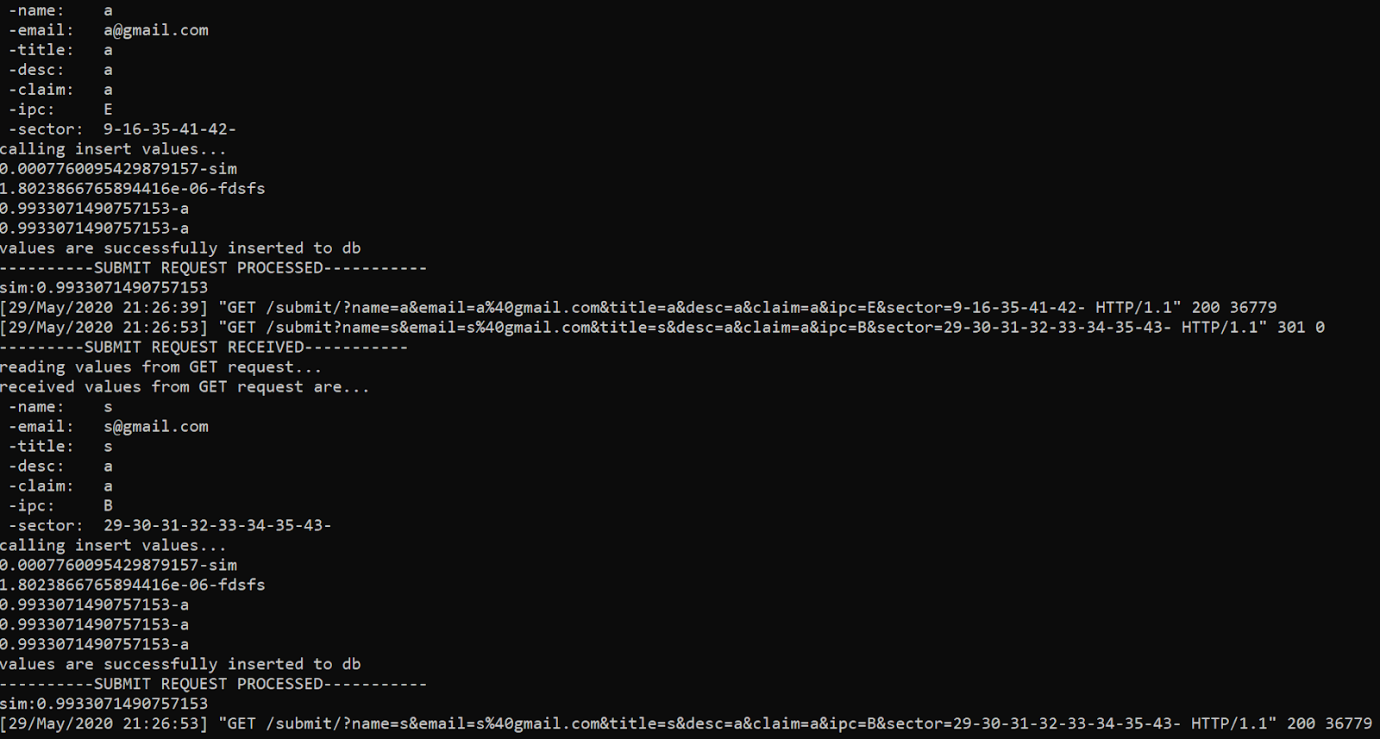
The disadvantage is that the accuracy of Natural Language Process algorithms can vary from person to person. In another algorithm that works with the same logic, this may result in different similarity results between the same patents. However, this does not mean that one of the algorithms is wrong.

**In what ways is your solution missing or incomplete?**

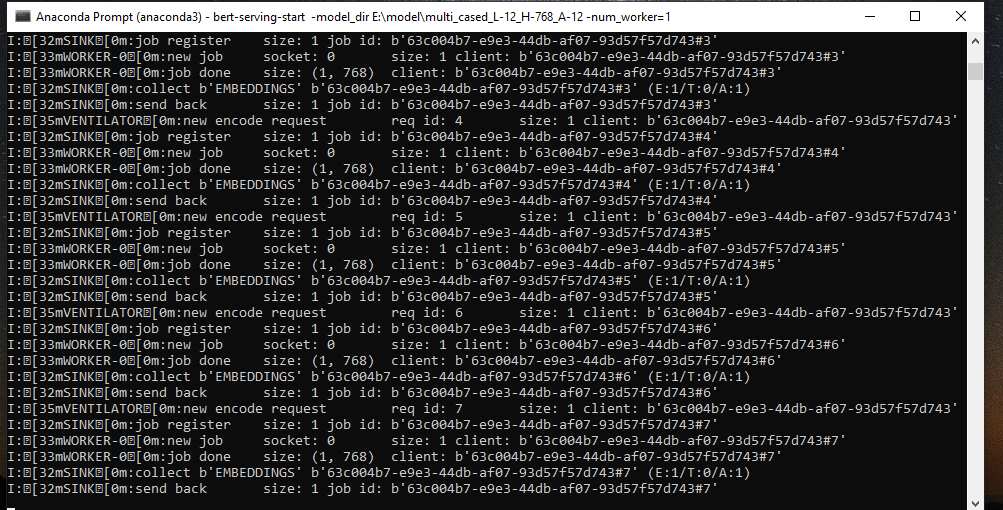
* The accuracy rates of the system's comparison algorithms vary according to the determined criteria. That is why our lack and improvement should be the criteria we make this similarity. In addition, the result screen can be further detailed and the user can be shown more results.

**Acknowledgement Acknowledge any individuals who have helped you during the course of the project, including your project advisor. If you have been supported by a company or a scholarship, then this should also be gratefully acknowledged.**

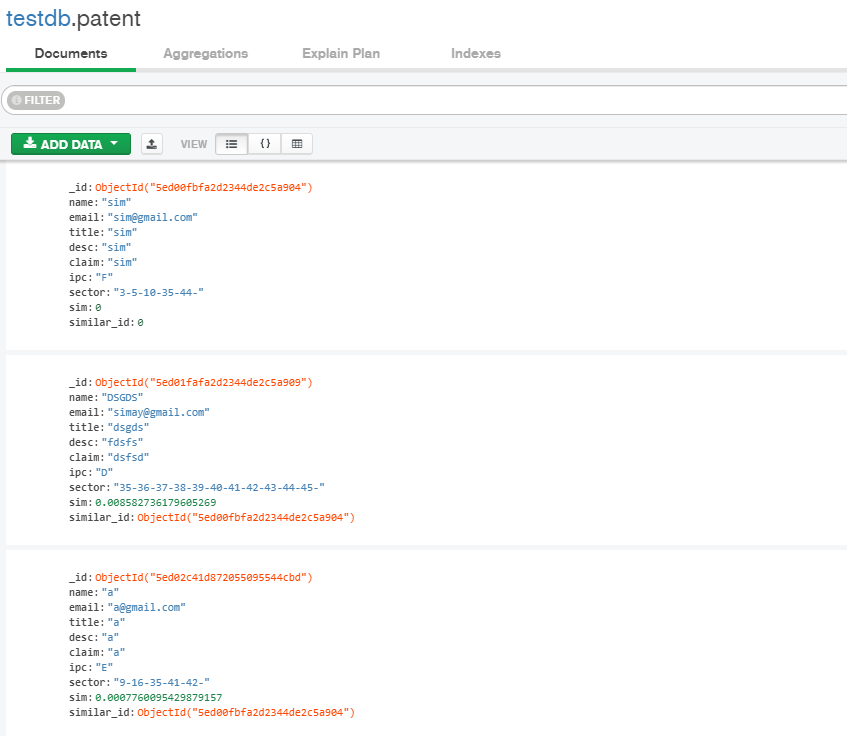
* As a support, we started the project by taking the project idea from Onur Muhendislik. We received the help of our valuable teachers from the system. Thanks to the company, we have provided access to our data.



**Figure 12-Submit Request**



**Figure 13-Bert Serving Starting**



**Figure 14-Mongodb Screen**

**8.1 Code**

import numpy as np

from bert\_serving.client import BertClient

from django.shortcuts import render

import pymongo

from django.http import JsonResponse

# Create your views here.

def home\_view(request):

    return render(request,'index.html',{})

def sign\_in\_view(request):

    return render(request,'account-sign-in-cover.html',{})

def sign\_up\_view(request):

    if 'email' in request.GET:

        print ("---------SIGN UP REQUEST RECEIVED-----------");

        print ("reading values from GET request...")

        email     = request.GET['email']

        password  = request.GET['pass']

        print("received values from GET request are...");

        print(" -email :   ", email)

        print(" -pass  :   ", password)

        print("calling insert values...")

        insertAccount(email, password)

        print("values are successfully inserted to db")

        print ("----------SIGNUP REQUEST PROCESSED-----------");

    return render(request,'account-sign-up-cover.html',{})

def insertAccount(email, password):

    client = pymongo.MongoClient("mongodb://localhost:27017/")

    #Database name: testdb

    db = client["testdb"]

    #collection name : accounts

    collection = db["accounts"]

    #query

    query = {"email": email, "pass": password}

    #execute query

    collection.insert\_one(query)

def forgot\_password\_view(request):

    return render(request,'account-forgot-password.html',{})

def patent\_one\_view(request):

    return render(request,'patent-1.html',{})

def patent\_second\_view(request):

    return render(request,'patent-2.html',{})

def patent\_third\_view(request):

    return render(request,'patent-3.html',{})

def patent\_fourth\_view(request):

    return render(request,'patent-4.html',{})

def patent\_fifth\_view(request):

    return render(request,'patent-5.html',{})

def contact\_view(request):

    return render(request,'contact.html',{})

def who\_we\_are\_view(request):

    return render(request,'who\_we\_are.html',{})

def our\_vision\_view(request):

    return render(request,'our\_vision.html',{})

def result\_view(request):

    return render(request,'result.html',{})

def bert\_result\_view(request):

    return render(request,'bert\_result.html',{})

def submit(request):

    print ("---------SUBMIT REQUEST RECEIVED-----------");

    print ("reading values from GET request...")

    name   = request.GET['name']

    email  = request.GET['email']

    title  = request.GET['title']

    desc   = request.GET['desc']

    claim  = request.GET['claim']

    ipc    = request.GET['ipc']

    sector = request.GET['sector']

    print("received values from GET request are...");

    print(" -name:   ", name)

    print(" -email:  ", email)

    print(" -title:  ", title)

    print(" -desc:   ", desc)

    print(" -claim:  ", claim)

    print(" -ipc:    ", ipc)

    print(" -sector: ", sector)

    print("calling insert values...")

    similarity = insertPatent(name, email, title, desc, claim, ipc, sector)

    print("values are successfully inserted to db")

    print ("----------SUBMIT REQUEST PROCESSED-----------");

    print ("sim:" + str(similarity))

    return render(request, 'index.html', {'similarity' : str(similarity)})

def getDescriptionListFromDB():

    client = pymongo.MongoClient("mongodb://localhost:27017/")

    # Database name: testdb

    db = client["testdb"]

    # collection name : patent

    collection = db["patent"]

    result = collection.find()

    descriptions = []

    for i in result:

        descriptions.append(i)

    return descriptions

def insertPatent(name, email, title, desc, claim, ipc, sector):

    with BertClient(port=5555, port\_out=5556, check\_version=False, check\_length=False) as bc:

        client = pymongo.MongoClient("mongodb://localhost:27017/")

        import math

        query\_vec\_1 = bc.encode([desc])[0]

        valuesInDB = getDescriptionListFromDB()

        similarity = 0

        mostsimilarid = 0

        for y in valuesInDB:

            x = y['desc']

            patent\_id = y['\_id']

            query\_vec\_2 = bc.encode([x])[0]

            cosine = np.dot(query\_vec\_1, query\_vec\_2) / (np.linalg.norm(query\_vec\_1) \* np.linalg.norm(query\_vec\_2))

            local = 1 / (1 + math.exp(-100 \* (cosine - 0.95)))

            print (str(local) + "-" + x)

            if local > similarity:

                similarity = local

                mostsimilarid = patent\_id

        #Database name: testdb

        db = client["testdb"]

        #collection name : patent

        collection = db["patent"]

        #query

        query = { "name": name, "email": email, "title": title, "desc": desc, "claim":claim, "ipc":ipc, "sector":sector , "sim":similarity, 'similar\_id':mostsimilarid}

        #execute query

        collection.insert\_one(query)

        return similarity

def controlUser(email,password):

    client = pymongo.MongoClient("mongodb://localhost:27017/")

    db = client["testdb"]

    collection = db["accounts"]

    query = {"email":email, "password":password}

    collection.find(query)