Programing Assignment -3

Part - 1

Description:

For this assignment, I developed a simple web based query system which is capable of querying mashups and APIs based on different criteria. Based on the input parameters, the system is capable of displaying relevant API details/Mashup names which can help an enduser in finding a interesting mashup or api of his choice.

Technologies Overview:

To design a seamless interface as discussed the following softwares/tools have been used in aiding development:

Front end:

HTML, CSS, Bootstrap, Javascript, Socket.io

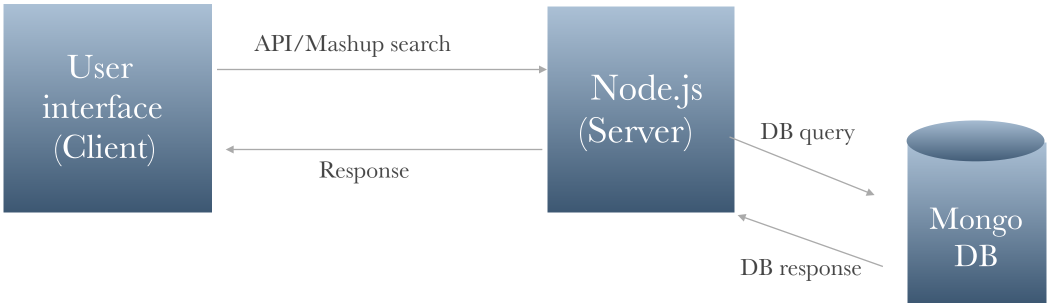
Back end:

Node.js

Database:

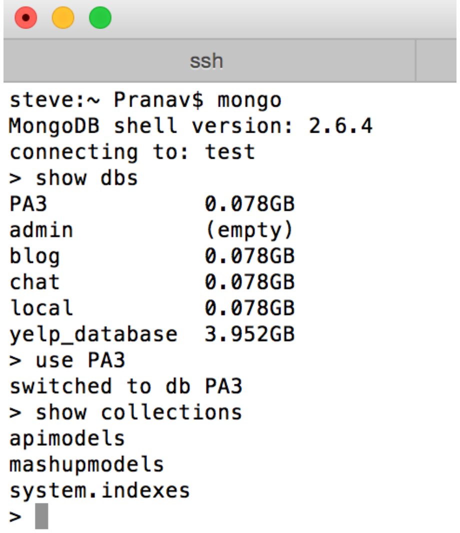
MongoDB

Design overview:



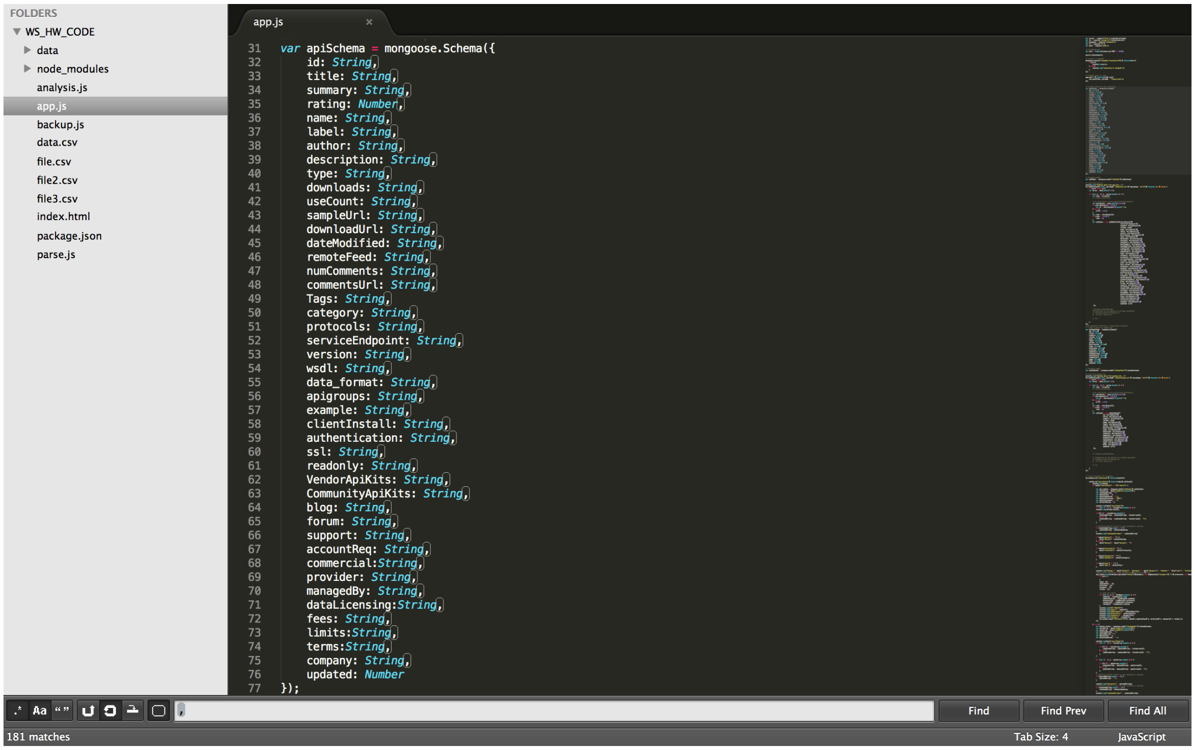
Database design:

MongoDB has been used for storing api and mashup records. A single database ‘PA3’ has been created, which consists of two different collections ‘apimodel’ and ‘mashupmodels’. MongoDB uses JSON format for storing documents. ’apimodels’ contains records of all api related documents. ‘mashupmodels’ contain records of all mashup related documents.

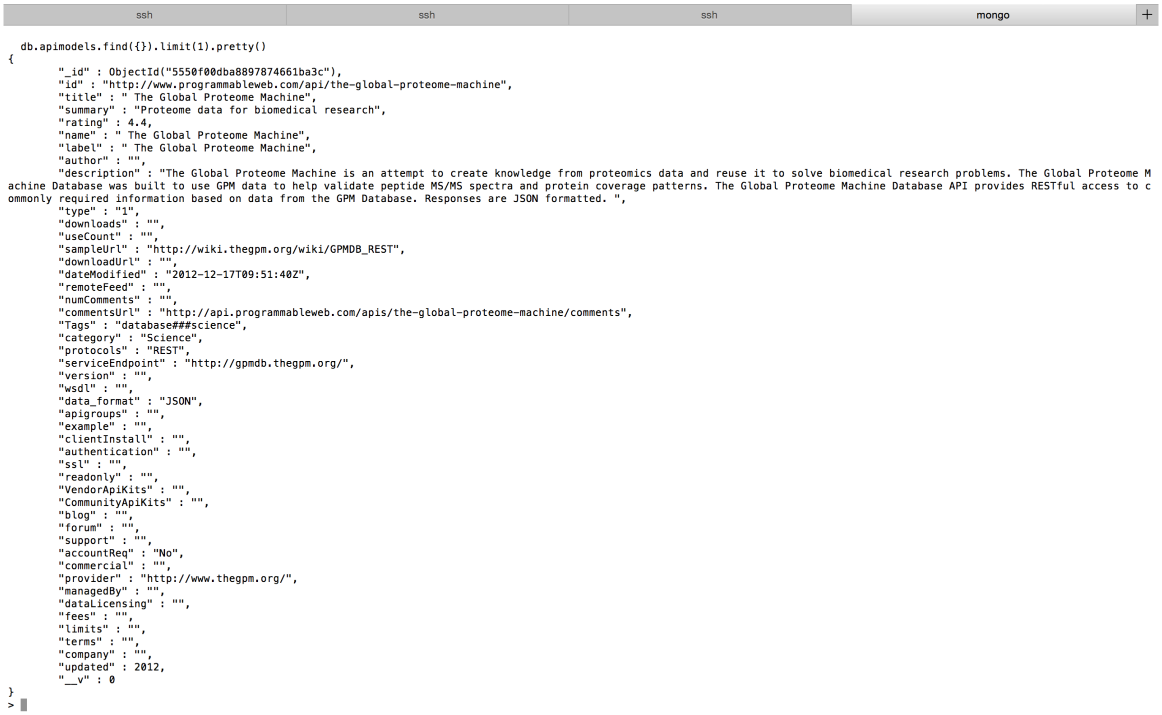


```
steve:~ Pranav$ mongo
MongoDB shell version: 2.6.4
connecting to: test
> show dbs
PA3            0.078GB
admin          (empty)
blog           0.078GB
chat           0.078GB
local          0.078GB
yelp_database  3.952GB
> use PA3
switched to db PA3
> show collections
apimodels
mashupmodels
system.indexes
>
```

For ‘apimodels’ following schema is used while inserting into the database

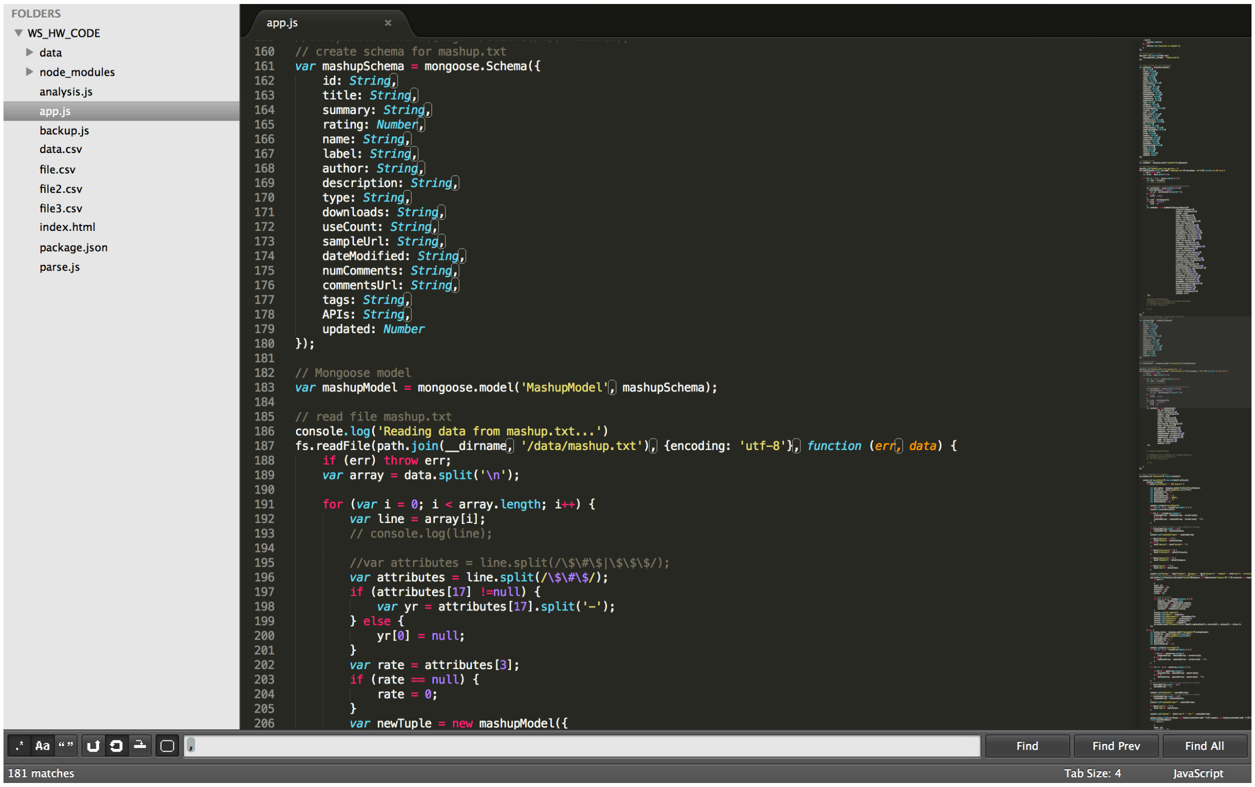


```
var apiSchema = mongoose.Schema({
    id: String,
    title: String,
    summary: String,
    rating: Number,
    name: String,
    label: String,
    author: String,
    description: String,
    type: String,
    downloads: String,
    useCount: String,
    sampleUrl: String,
    downloadUrl: String,
    dateModified: String,
    remoteFeed: String,
    numComments: String,
    commentsUrl: String,
    Tags: String,
    category: String,
    protocols: String,
    serviceEndpoint: String,
    version: String,
    wsdl: String,
    data_format: String,
    apigroups: String,
    example: String,
    clientInstall: String,
    authentication: String,
    ssl: String,
    readonly: String,
    VendorApiKits: String,
    CommunityApiKits: String,
    blog: String,
    forum: String,
    support: String,
    accountReq: String,
    commercial:String,
    provider: String,
    managedBy: String,
    dataLicensing:String,
    fees: String,
    limits:String,
    terms:String,
    company: String,
    updated: Number
});
```



```
  db.apimodels.find({}).limit(1).pretty()
{
        "_id" : ObjectId("5550f00dba8897874661ba3c"),
        "id" : "http://www.programmableweb.com/api/the-global-proteome-machine",
        "title" : " The Global Proteome Machine",
        "summary" : "Proteome data for biomedical research",
        "rating" : 4.4,
        "name" : " The Global Proteome Machine",
        "label" : " The Global Proteome Machine",
        "author" : "",
        "description" : "The Global Proteome Machine is an attempt to create knowledge from proteomics data and reuse it to solve biomedical research problems. The Global Proteome M
achine Database was built to use GPM data to help validate peptide MS/MS spectra and protein coverage patterns. The Global Proteome Machine Database API provides RESTful access to c
ommonly required information based on data from the GPM Database. Responses are JSON formatted. ",
        "type" : "1",
        "downloads" : "",
        "useCount" : "",
        "sampleUrl" : "http://wiki.thegpm.org/wiki/GPMDB_REST",
        "downloadUrl" : "",
        "dateModified" : "2012-12-17T09:51:40Z",
        "remoteFeed" : "",
        "numComments" : "",
        "commentsUrl" : "http://api.programmableweb.com/apis/the-global-proteome-machine/comments",
        "Tags" : "database###science",
        "category" : "Science",
        "protocols" : "REST",
        "serviceEndpoint" : "http://gpmdb.thegpm.org/",
        "version" : "",
        "wsdl" : "",
        "data_format" : "JSON",
        "apigroups" : "",
        "example" : "",
        "clientInstall" : "",
        "authentication" : "",
        "ssl" : "",
        "readonly" : "",
        "VendorApiKits" : "",
        "CommunityApiKits" : "",
        "blog" : "",
        "forum" : "",
        "support" : "",
        "accountReq" : "No",
        "commercial" : "",
        "provider" : "http://www.thegpm.org/",
        "managedBy" : "",
        "dataLicensing" : "",
        "fees" : "",
        "limits" : "",
        "terms" : "",
        "company" : "",
        "updated" : 2012,
        "__v" : 0
}
> 
```

For ‘mashupmodels’ following schema is used while inserting into the database



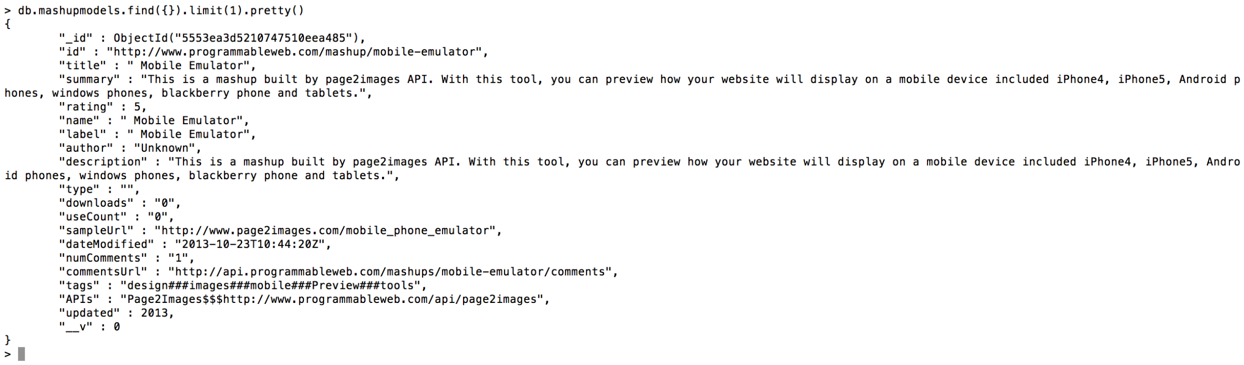
```javascript
// create schema for mashup.txt
var mashupSchema = mongoose.Schema({
    id: String,
    title: String,
    summary: String,
    rating: Number,
    name: String,
    label: String,
    author: String,
    description: String,
    type: String,
    downloads: String,
    useCount: String,
    sampleUrl: String,
    dateModified: String,
    numComments: String,
    commentsUrl: String,
    tags: String,
    APIs: String,
    updated: Number
});

// Mongoose model
var mashupModel = mongoose.model('MashupModel', mashupSchema);

// read file mashup.txt
console.log('Reading data from mashup.txt...')
fs.readFile(path.join(__dirname, '/data/mashup.txt'), {encoding: 'utf-8'}, function (err, data) {
    if (err) throw err;
    var array = data.split('\n');

    for (var i = 0; i < array.length; i++) {
        var line = array[i];
        // console.log(line);

        //var attributes = line.split(/\$\#\$|\$\$\$/);
        var attributes = line.split(/\$\#\$/);
        if (attributes[17] !=null) {
            var yr = attributes[17].split('-');
        } else {
            yr[0] = null;
        }
        var rate = attributes[3];
        if (rate == null) {
            rate = 0;
        }
        var newTuple = new mashupModel({
```



```
> db.mashupmodels.find({}).limit(1).pretty()
{
        "_id" : ObjectId("5553ea3d5210747510eea485"),
        "id" : "http://www.programmableweb.com/mashup/mobile-emulator",
        "title" : " Mobile Emulator",
        "summary" : "This is a mashup built by page2images API. With this tool, you can preview how your website will display on a mobile device included iPhone4, iPhone5, Android p
hones, windows phones, blackberry phone and tablets.",
        "rating" : 5,
        "name" : " Mobile Emulator",
        "label" : " Mobile Emulator",
        "author" : "Unknown",
        "description" : "This is a mashup built by page2images API. With this tool, you can preview how your website will display on a mobile device included iPhone4, iPhone5, Andro
id phones, windows phones, blackberry phone and tablets.",
        "type" : "",
        "downloads" : "0",
        "useCount" : "0",
        "sampleUrl" : "http://www.page2images.com/mobile_phone_emulator",
        "dateModified" : "2013-10-23T10:44:20Z",
        "numComments" : "1",
        "commentsUrl" : "http://api.programmableweb.com/mashups/mobile-emulator/comments",
        "tags" : "design###images###mobile###Preview###tools",
        "APIs" : "Page2Images$$$http://www.programmableweb.com/api/page2images",
        "updated" : 2013,
        "__v" : 0
}
>
```

Test cases scenarios :

1) Scenario

API Search

Input:

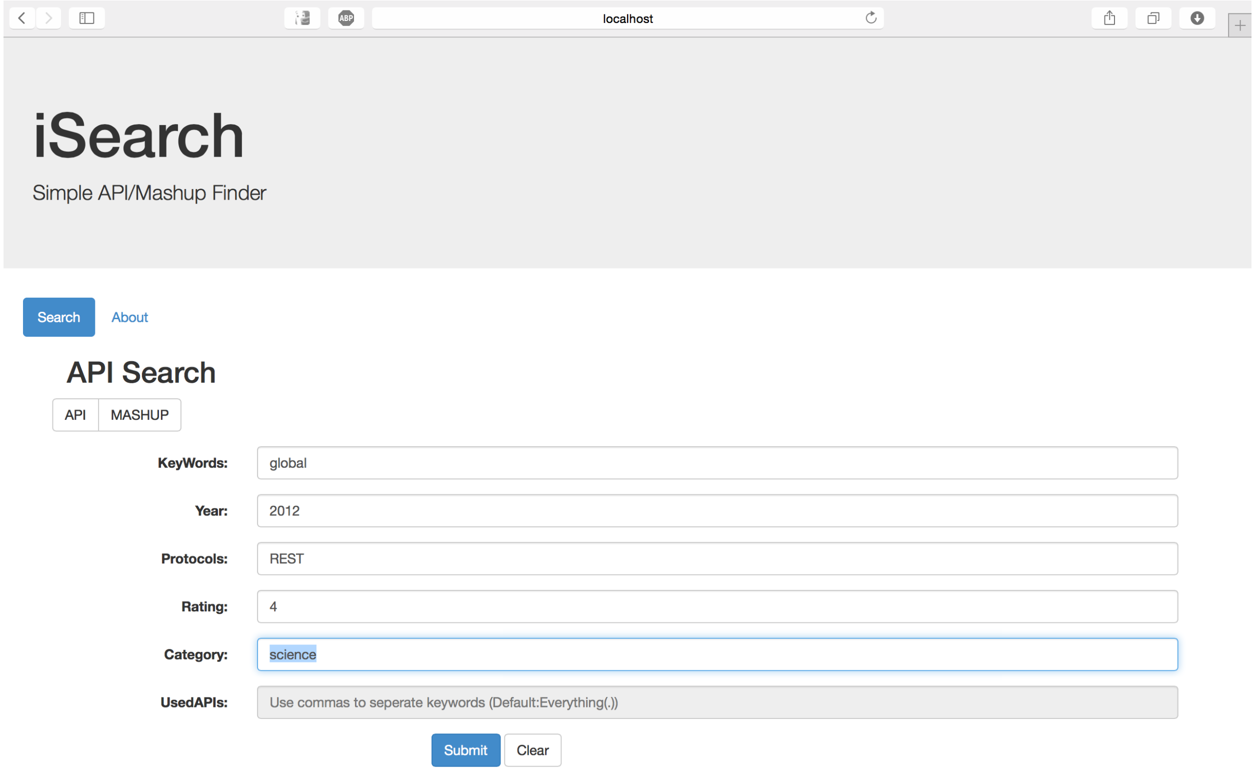
Keywords: global

Year: 2012

Protocols: 4

Rating: 4

Category: science



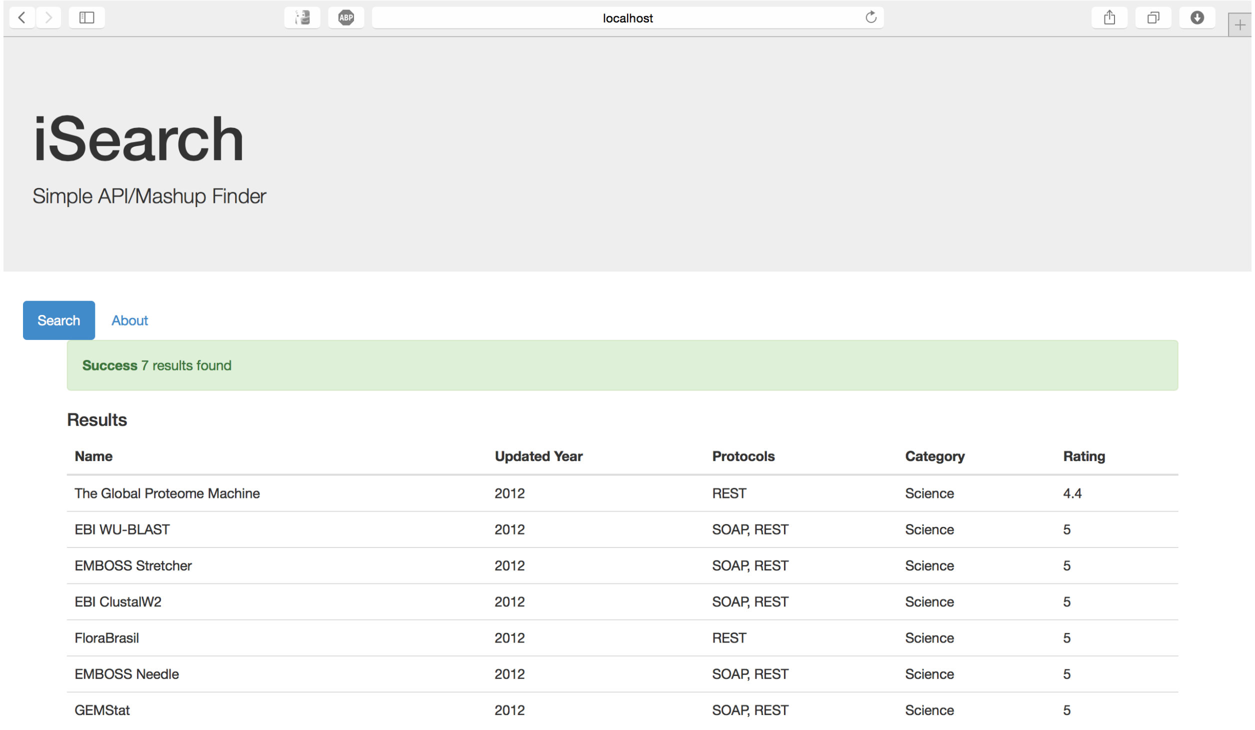
iSearch

Simple API/Mashup Finder

API Search

KeyWords:	global
Year:	2012
Protocols:	REST
Rating:	4
Category:	science
UsedAPIs:	Use commas to seperate keywords (Default:Everything(.))

Results:



iSearch

Simple API/Mashup Finder

Success 7 results found

Results

Name	Updated Year	Protocols	Category	Rating
The Global Proteome Machine	2012	REST	Science	4.4
EBI WU-BLAST	2012	SOAP, REST	Science	5
EMBOSS Stretcher	2012	SOAP, REST	Science	5
EBI ClustalW2	2012	SOAP, REST	Science	5
FloraBrasil	2012	REST	Science	5
EMBOSS Needle	2012	SOAP, REST	Science	5
GEMStat	2012	SOAP, REST	Science	5

2) Scenario

API Search

Input:

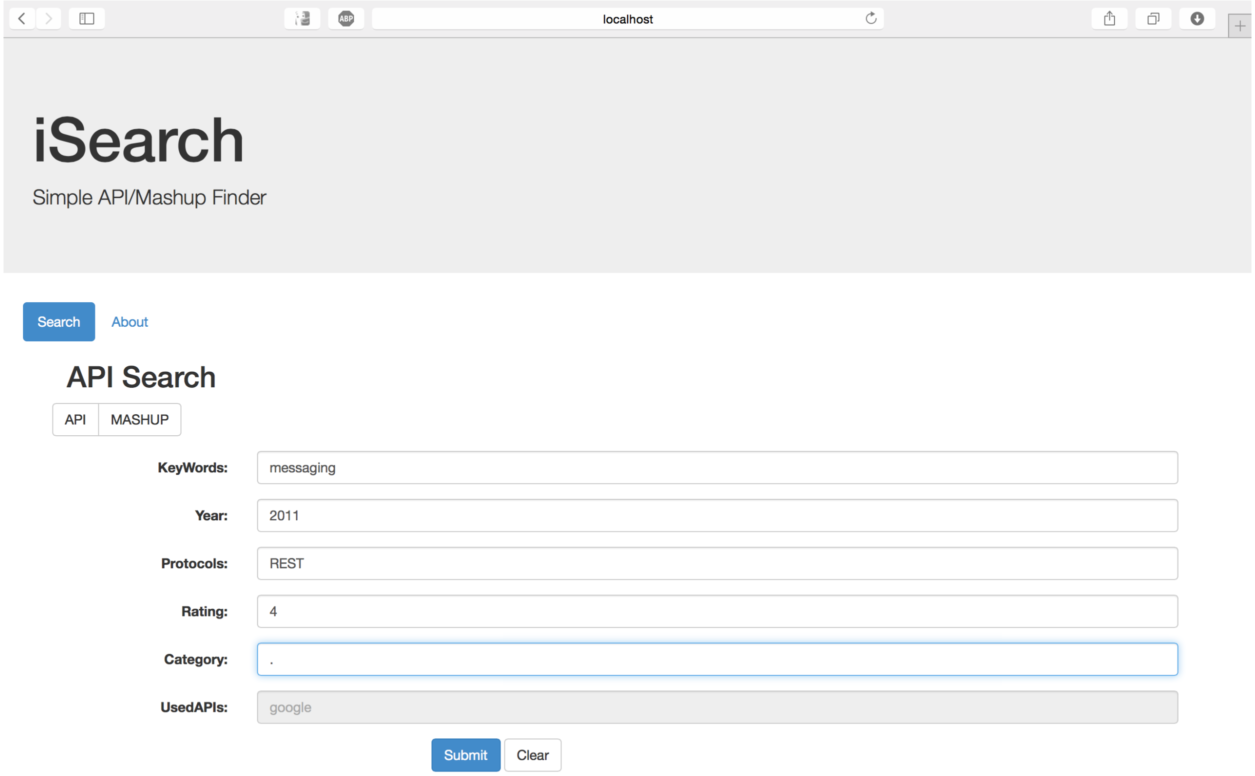
Keywords: messaging

Year: 2011

Protocols: REST

Rating: 4

Category: '.' wildcard (ALL)



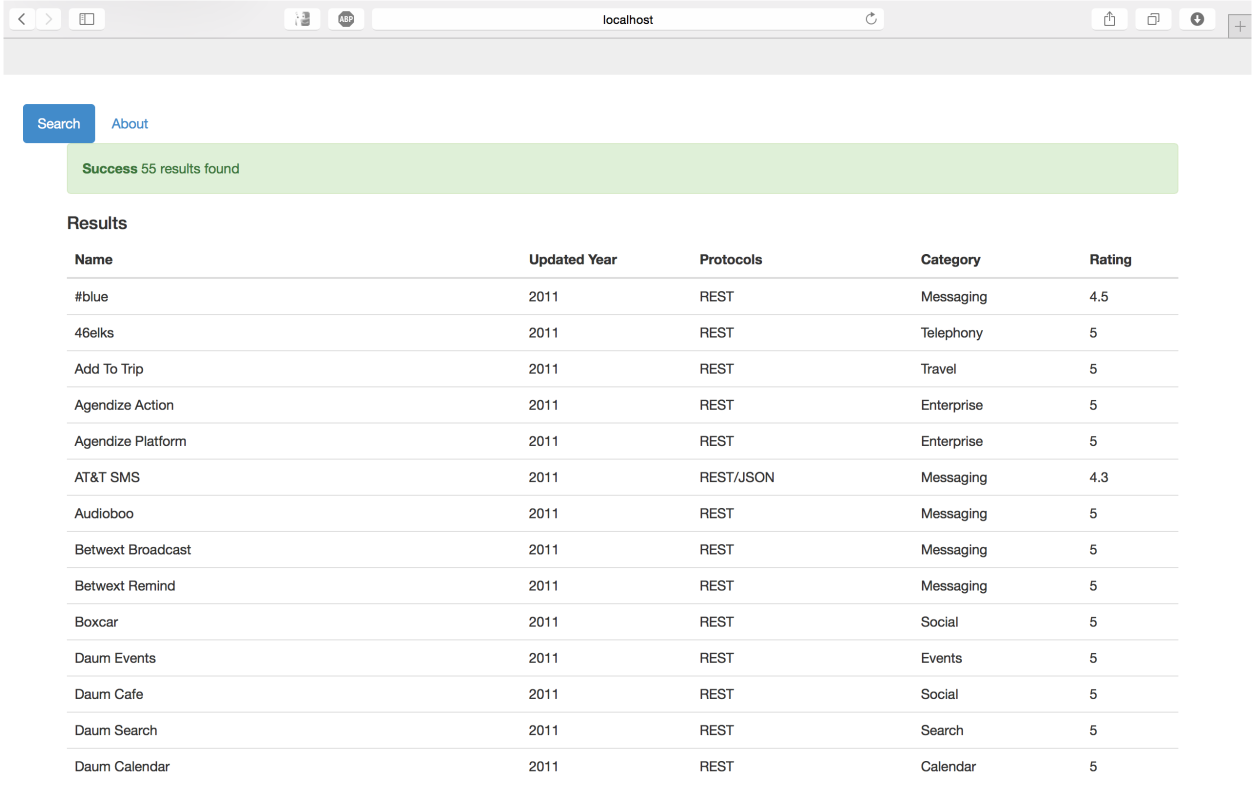
iSearch

Simple API/Mashup Finder

API Search

KeyWords:	messaging
Year:	2011
Protocols:	REST
Rating:	4
Category:	.
UsedAPIs:	google

Results:



Success 55 results found

Results

Name	Updated Year	Protocols	Category	Rating
#blue	2011	REST	Messaging	4.5
46elks	2011	REST	Telephony	5
Add To Trip	2011	REST	Travel	5
Agendize Action	2011	REST	Enterprise	5
Agendize Platform	2011	REST	Enterprise	5
AT&T SMS	2011	REST/JSON	Messaging	4.3
Audioboo	2011	REST	Messaging	5
Betwext Broadcast	2011	REST	Messaging	5
Betwext Remind	2011	REST	Messaging	5
Boxcar	2011	REST	Social	5
Daum Events	2011	REST	Events	5
Daum Cafe	2011	REST	Social	5
Daum Search	2011	REST	Search	5
Daum Calendar	2011	REST	Calendar	5

3) Scenario

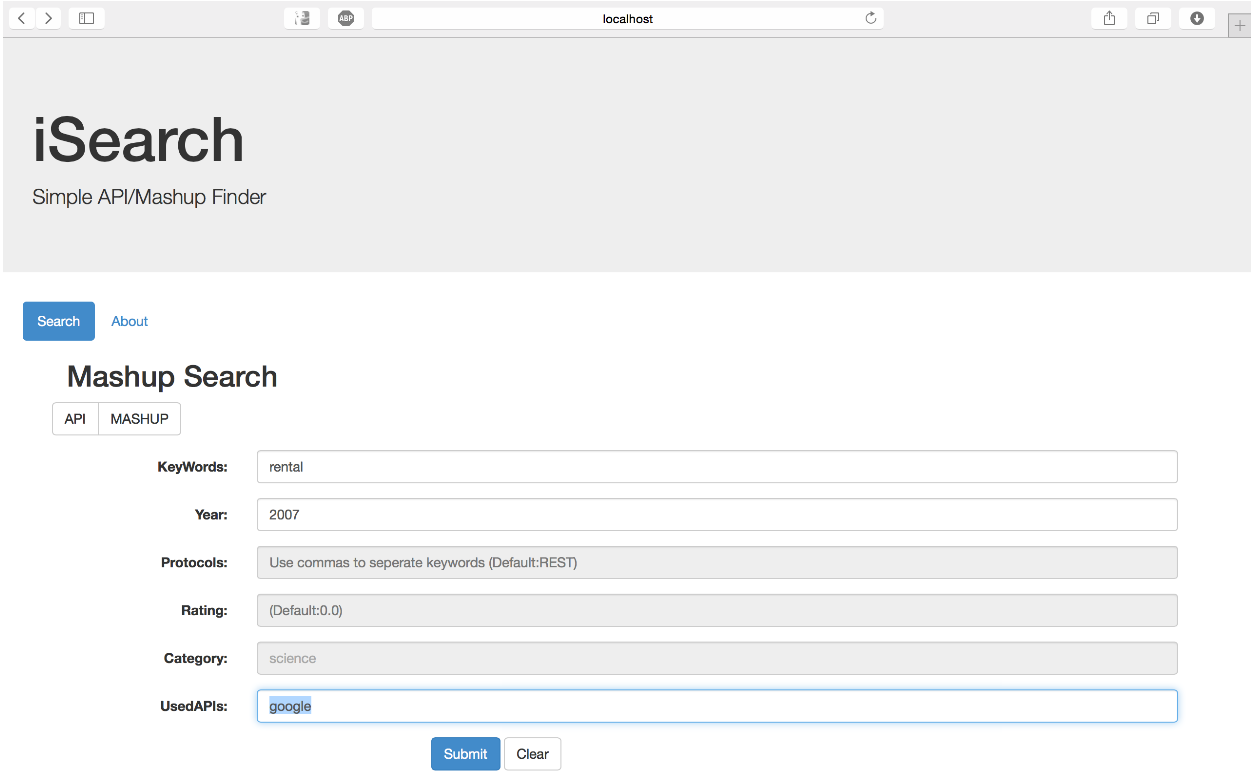
Mashup Search

Input:

Keywords: rental

Year: 2007

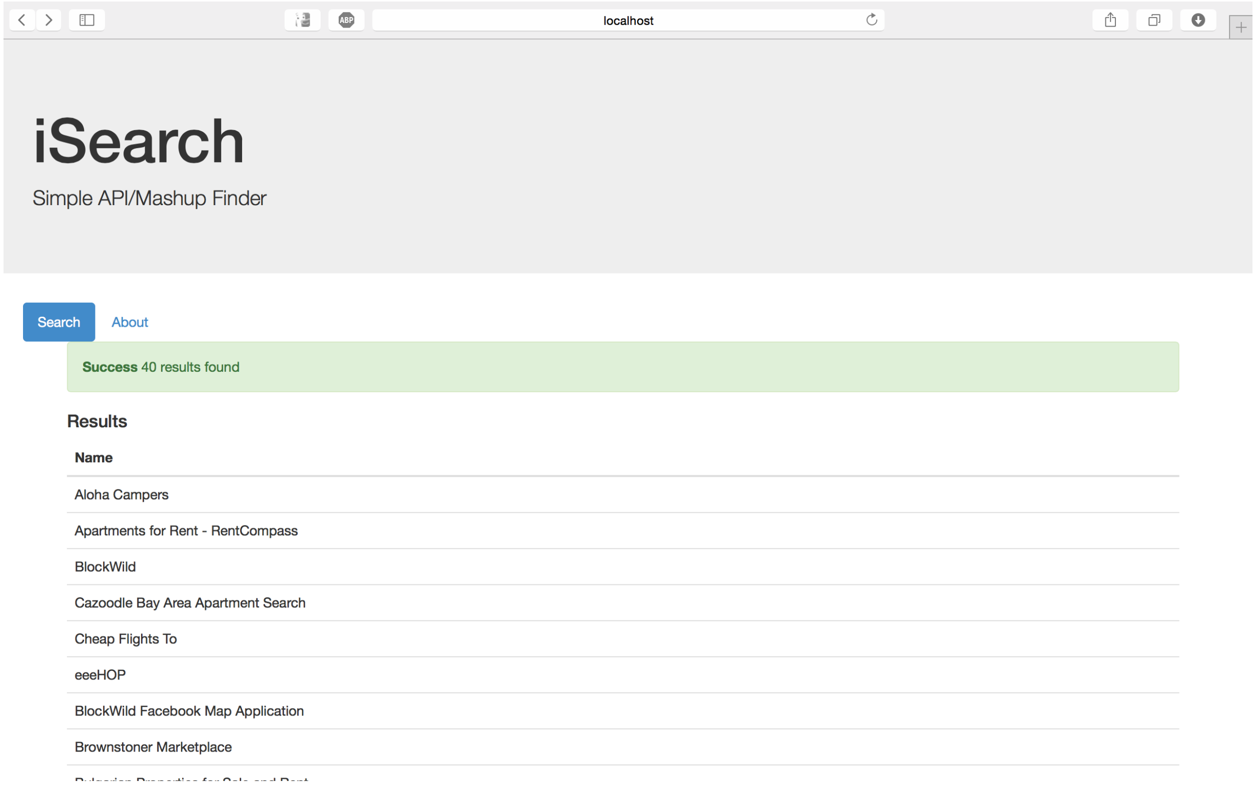
UsedAPIs: google



Mashup Search

KeyWords:	rental
Year:	2007
Protocols:	Use commas to seperate keywords (Default:REST)
Rating:	(Default:0.0)
Category:	science
UsedAPIs:	google

Results:



iSearch

Simple API/Mashup Finder

Success 40 results found

Results

Name
Aloha Campers
Apartments for Rent - RentCompass
BlockWild
Cazoodle Bay Area Apartment Search
Cheap Flights To
eeeHOP
BlockWild Facebook Map Application
Brownstoner Marketplace
Bulgarian Properties for Sale and Rent

4) Scenario

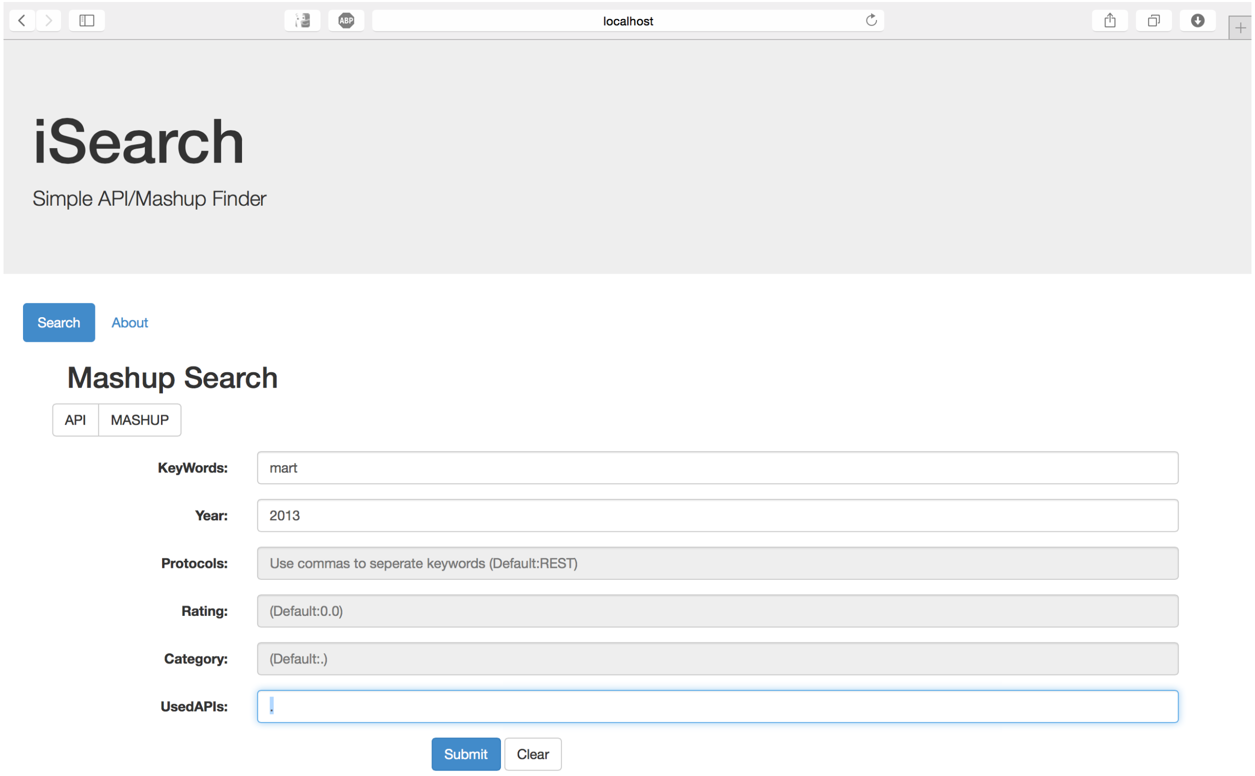
Mashup Search

Input:

Keywords: mart

Year: 2013

UsedAPIs: .



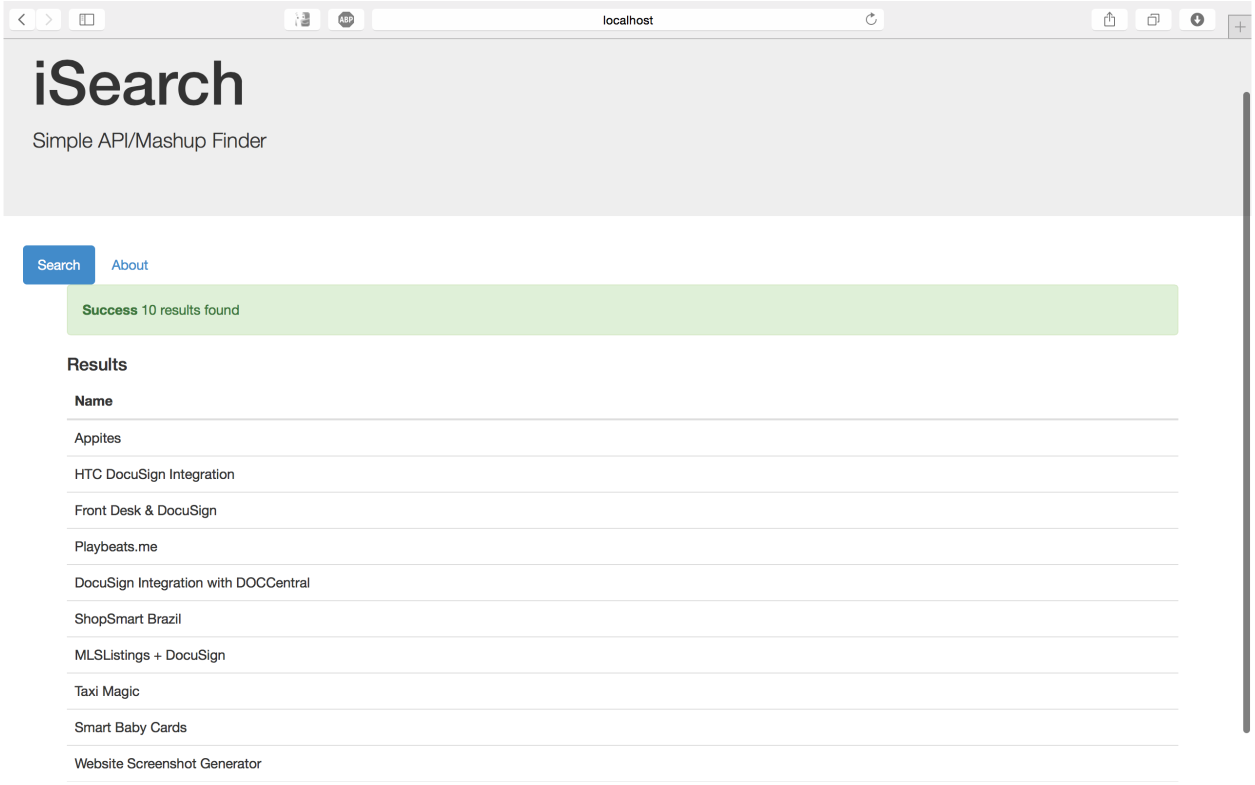
iSearch

Simple API/Mashup Finder

Mashup Search

KeyWords:	mart
Year:	2013
Protocols:	Use commas to seperate keywords (Default:REST)
Rating:	(Default:0.0)
Category:	(Default:.)
UsedAPIs:	

Results:



iSearch

Simple API/Mashup Finder

Success 10 results found

Results

Name
Appites
HTC DocuSign Integration
Front Desk & DocuSign
Playbeats.me
DocuSign Integration with DOCCentral
ShopSmart Brazil
MLSListings + DocuSign
Taxi Magic
Smart Baby Cards
Website Screenshot Generator