COMP 490-492: Scalable Web App Front-End for the MARFCAT App I-II

Supervised by Dr. Serguei Mokhov

mokhov@cs.concordia.ca

Developed by Mehdi Mokhtar Jamai

mehdi.m.jamai@gmail.com

Concordia University

Summer 1 2015 COMP 490
Summer 2 2015 COMP 492

Outline

Background

MARFCAT UI is a responsive,light and real-time application, consuming a REST API built in JAVA. The latter is a Web Service that was built during the Winter 2014 SOEN 487 which itself is using MARFCAT a standalone java application developed by professor Serguei Mokhov.

ADd more

Technologies

--Chrome: Google browser

--Open ESB: The REST API Web Service is running on GlassFish 4 and the whole service files were built in the Open ESB IDE [1]

--Yeoman: The stack used by the application is MEAN which stands for MongoDB,ExpressJS,AngularJS and NodeJS.The fullstack was generated by one the Yeoman Generators, https://github.com/DaftMonk/generator-angular-fullstack [2] [3] [12]

--Bitbucket: The version controlling system used for this project is git, with a private repository at bitbucket

--Chrome Developer Tools: For debugging the client side of the application, the use of the Dev Tools is very interesting, it's Chrome built in feature that helps front end developers to inspect the source code, XMLHttpRequests, HTML, CSS and JS errors

--WebStorm JetBrains: The IDE used for developer the User Interface [4]

--SourceTree: GUI that helps communicating with the bitbucket repository [5]

--Command Line 'λ cmdr' : Powerful command line for Windows OS [6]

--Postman REST client: Chrome extension that help testing REST API endpoints [7]

--Mongoose: Object modeling package for Node, it is similar to an Object Relational Mapper. It has the ability to access MongoDB CRUD commands. [8]

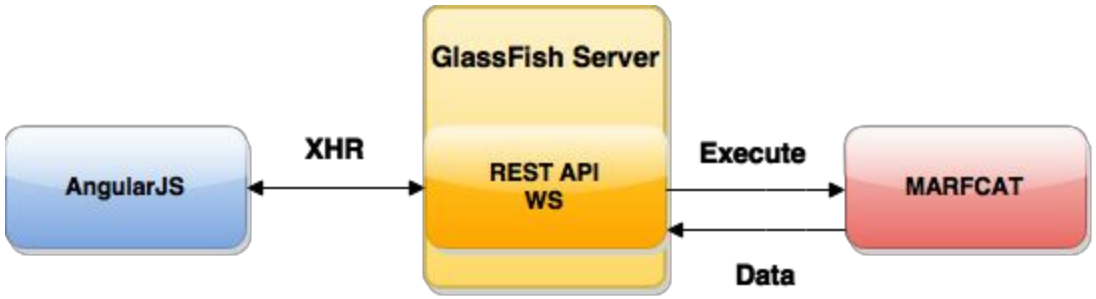
--Express: "Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications." [9]

--AngularJS: AngularJS is a structural framework for dynamic web apps. It lets you use HTML as your template language and lets you extend HTML's syntax to express your application's components clearly and succinctly. Angular's data binding and dependency injection eliminate much of the code you would otherwise have to write. [10]

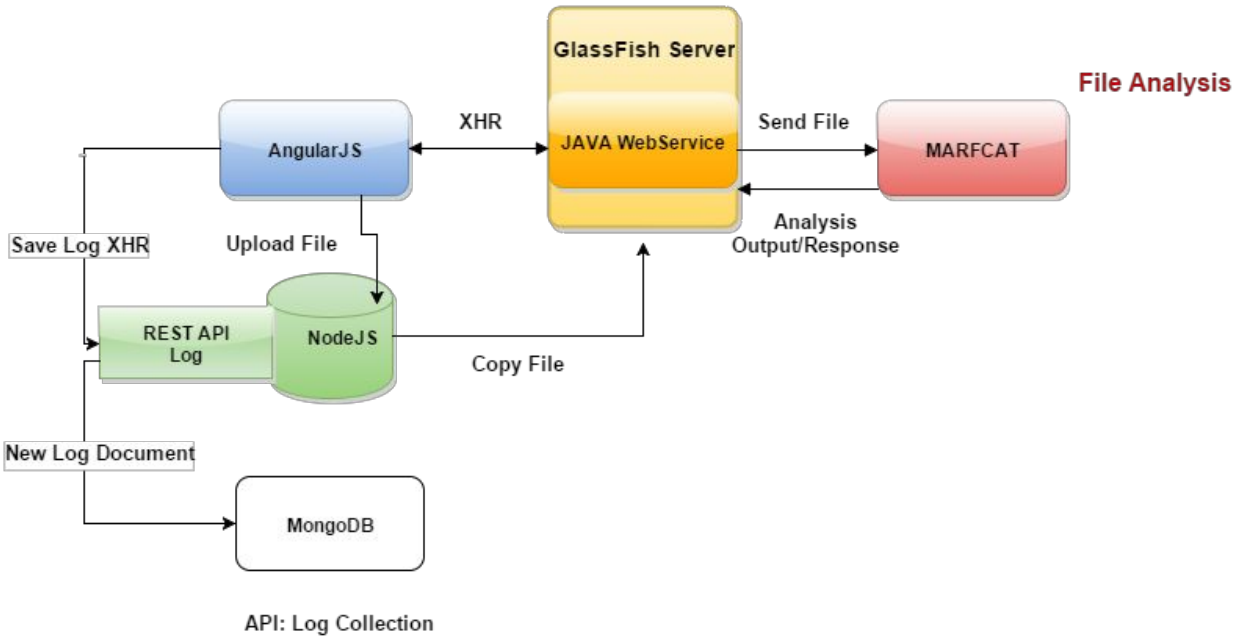
--FileUpload Dependency:

Deployment

Architecture



MARFCAT UI Stack



MARFCAT App Full Stack

Description:
Diagram above shows the communication pattern between the Front-End, the web service and MARFCAT

Prerequisites (User Manual)

Install NodeJS [12]

Install Git

As we are using a private bitbucket repository to control our source code and documentation, download the source code from the bitbucket account or use the repository url also provided at the repository account to clone through the command line.

From command line

cd to folder/src/app

Run npm install (To install node packages)

Run bower install (To install dependencies)

Run grunt serve (Deploy the application)

A new tab in chrome should open and the application will be deployed on the port 9000 (*localhost:9000*).

*The steps above apply on Linux OS or MAC OS X

MongoDB Setup

Download the latest release of the mongodb installation package

Create a folder in your system files ie “C/db/data”

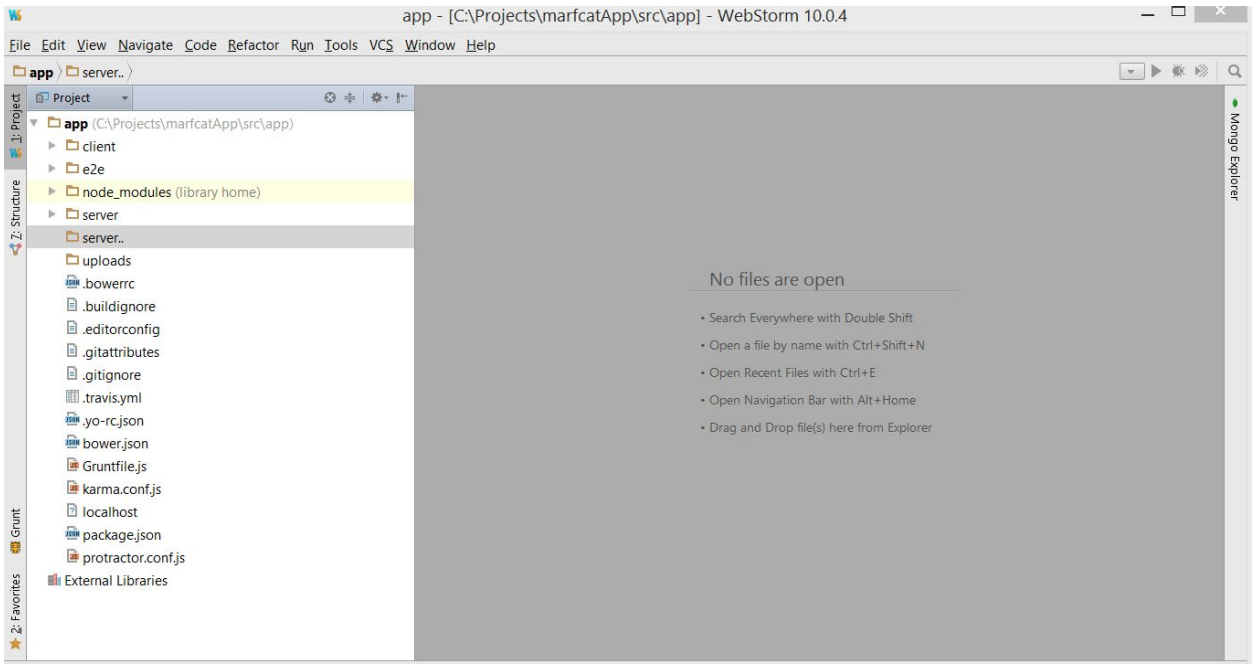
Run mongod from command line

Local MongDB will be running on 127.0.0.1:27017

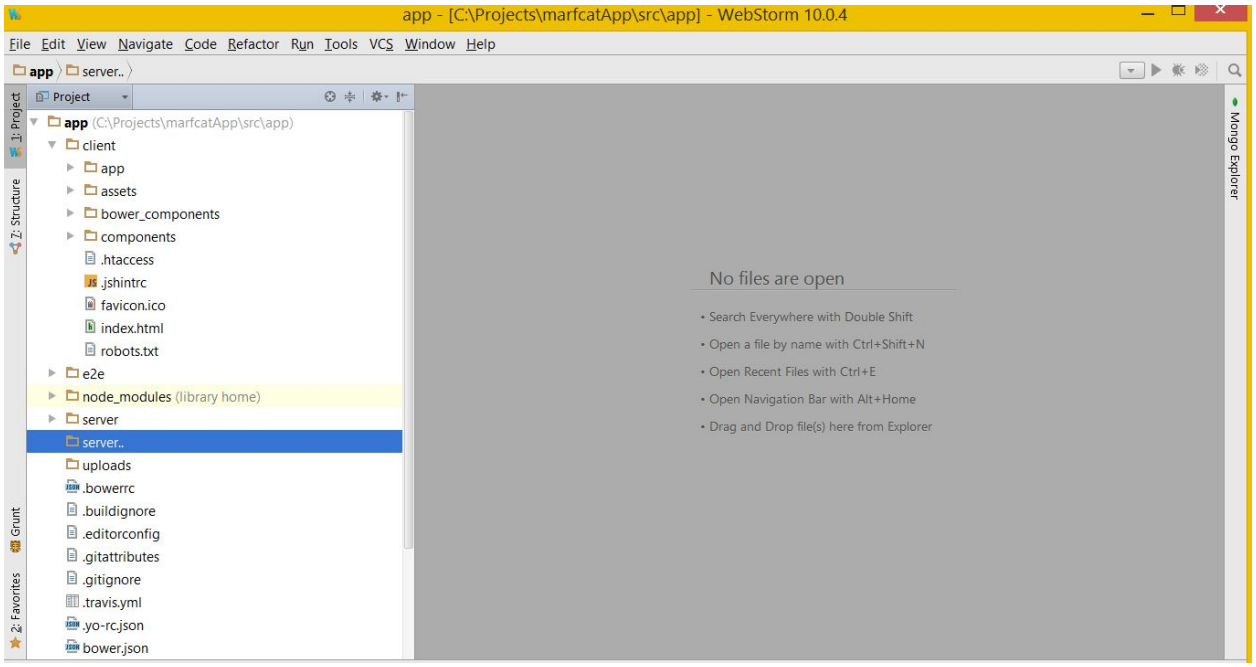
Run mongo in another command line to activate mongodb shell and manipulate data

Webstorm IDE & Application Skeleton

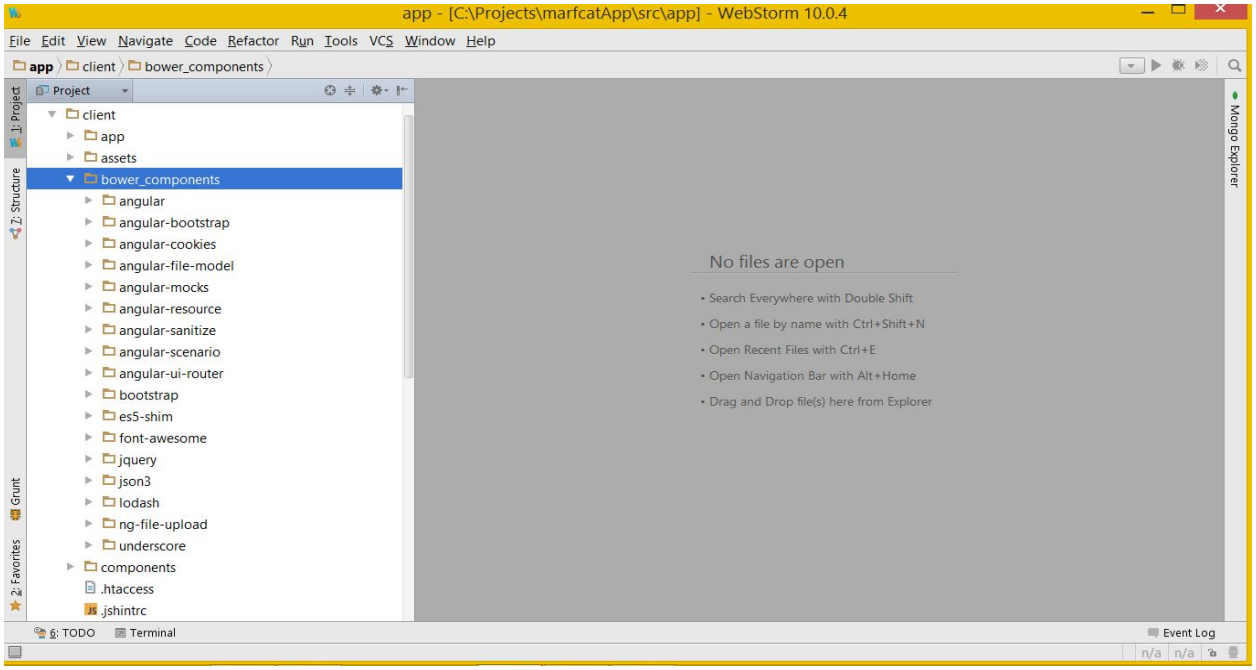
“WebStorm is a lightweight yet powerful IDE, perfectly equipped for complex client-side development and server-side development with Node.js.” [4]



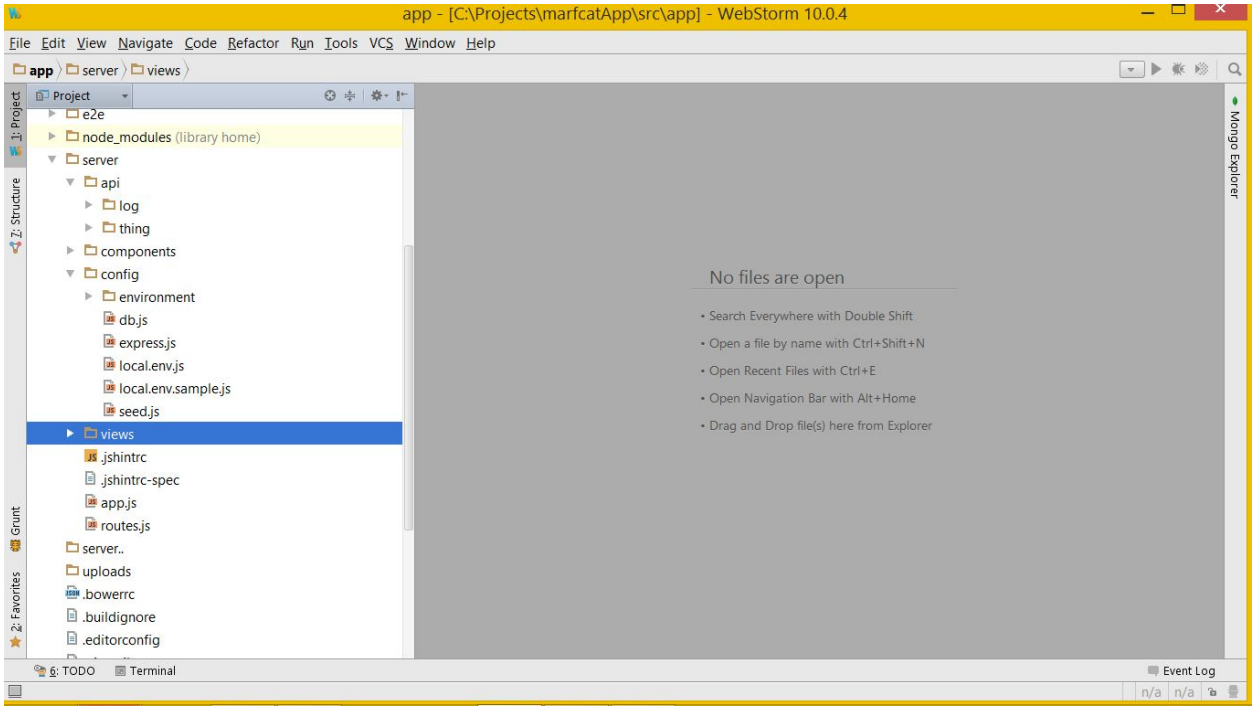
Skeleton of Complete Stack



Client Side



Bower Components



Server/Database side

The use of webstorm IDE in this projects is because of the first class support for Javascript, NodeJS, HTML and CSS. It has also a free plugin for MongoDB that provides a MongoDB shell to manipulate the data.

Features

The features developed are primarily consuming REST API by AngularJS

Posting and Reading files through REST API:

Reading files is achieved by making a GET XHR to the Web service, the request is then processed at the service side.

Posting files is achieved by choosing a file (ie: *.java) as well as configuration options

There are 6 categories of configurations to choose from(Loaders, Classification, NLP/Ngrams/Smoothing, Output Format, Feature Extraction, Preprocessors), they are displayed

in the pop up modal as 6 dropdown lists. Once an option is chosen for a specific category the appropriate code for it is saved as it is the one sent to Web Service.

Reading Weakness(s) through REST API:
Requesting Weaknesses generated by MARFACT by either passing a specific ID for a specific weakness or query all of them by requesting the report related to the analysis.

Reading Report through REST API:
Requesting the generated report from the file analysis.

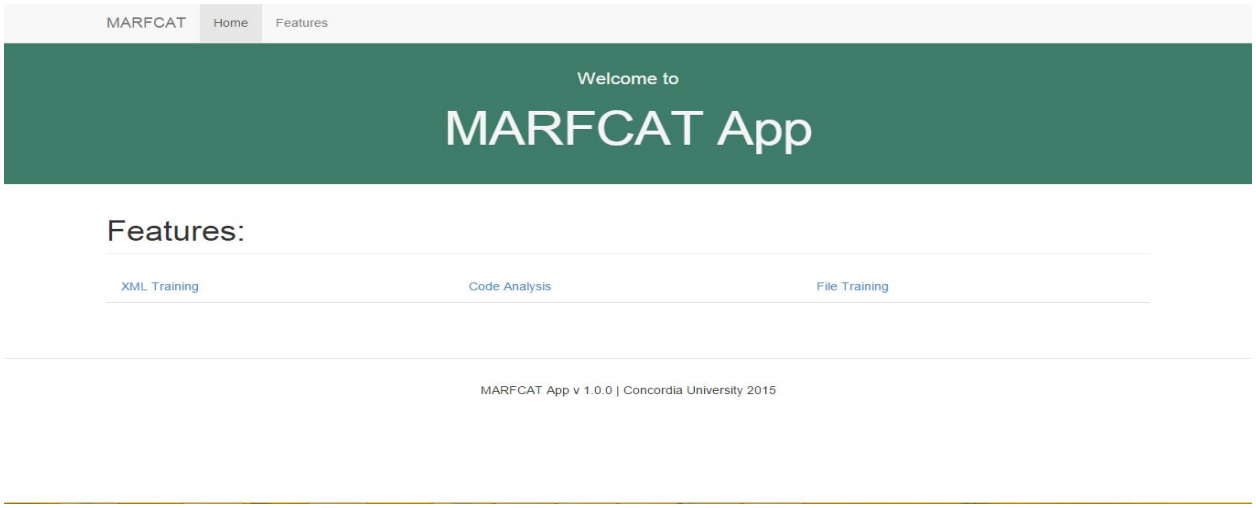
Upload file to NodeJS:
While posting a file for analysis, the chosen file gets uploaded first to NodeJS server then it is copied to GlassFish server. When the user is trying to upload a file through

Steps Diagram

Storing File Analysis Report:
Once a report is generated by MARFCAT it is the choice of the user to store it to the database.

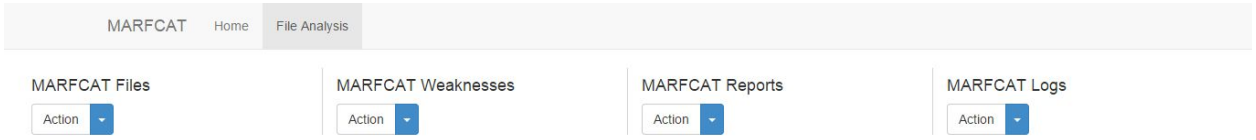
Querying Stored Reports:
MARFCAT UI has a view that help the user query of view the current reports in database of search through them.

Results



Home Page

Description:
MARFCAT UI homepage with static navbar
(This is an aold view, updated view is below)

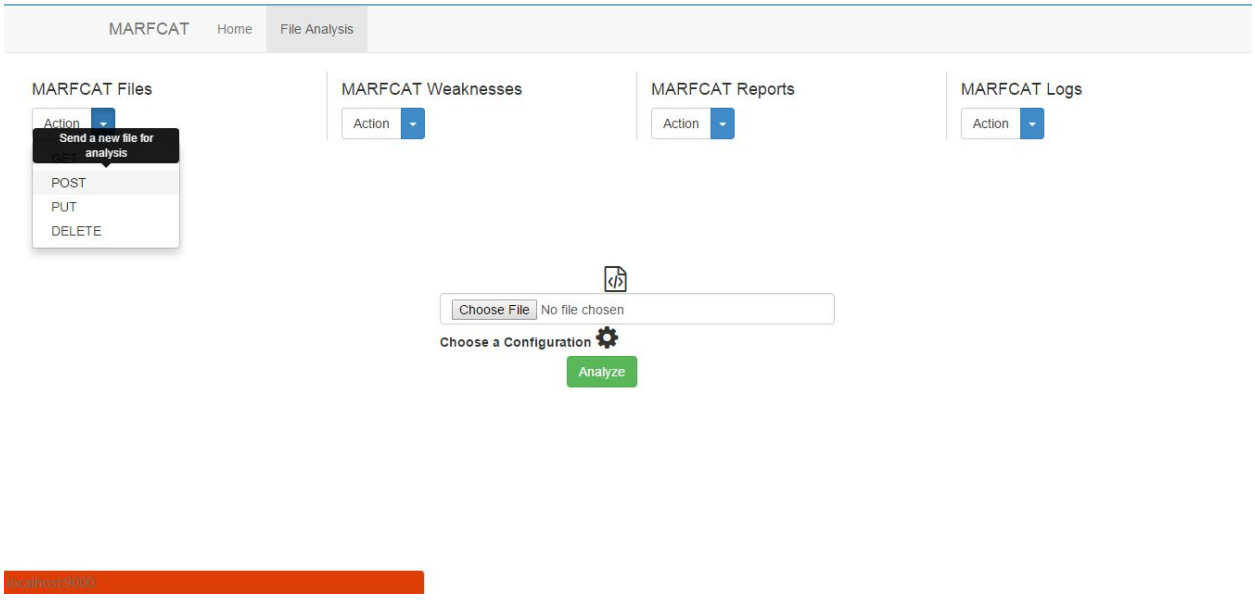


File Analysis REST API

Description
File analysis view showing MARFCAT tools

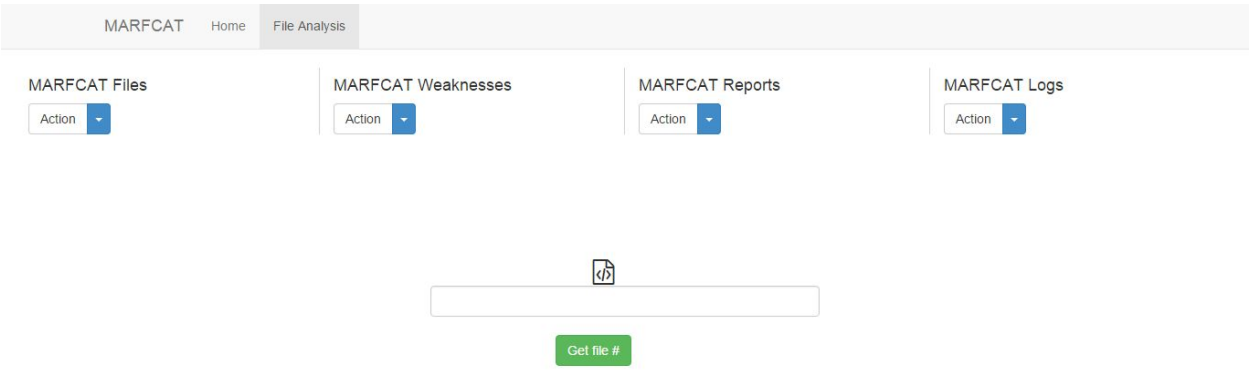
CRUD Files

Description:
MARFCAT Files dropdown displaying various actions that can be performed with WS



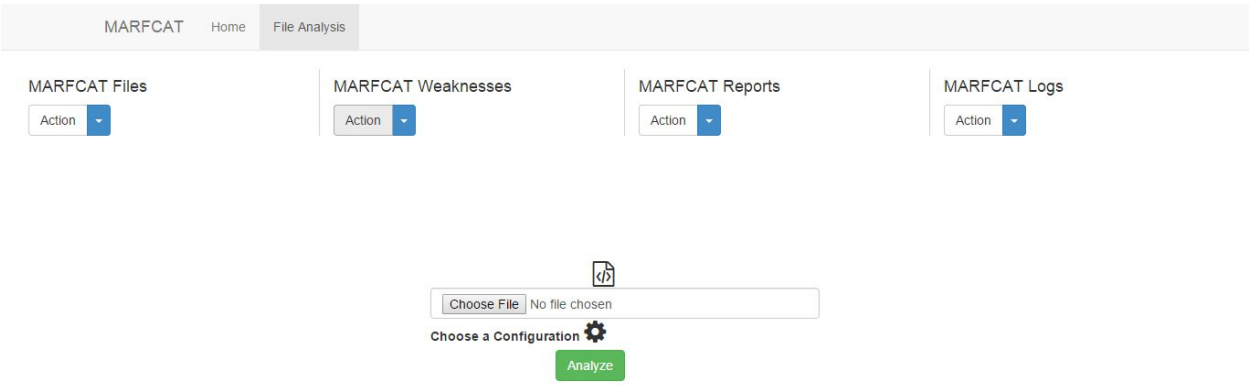
Explicit Tooltip

Description:
User friendly tooltip describing the actions
(This is an old view, updated view is below)



GET File by id

Description:
File request form
(This is aan old view, updated view is below)

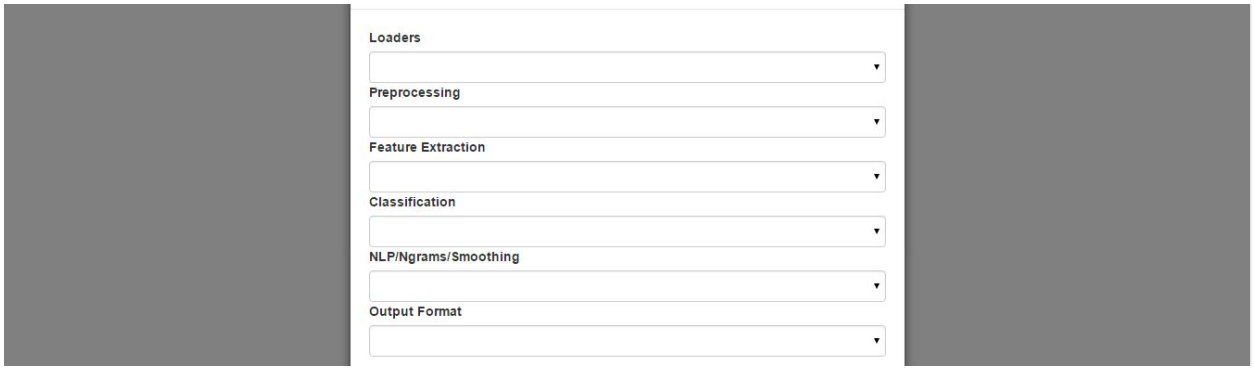


POST File & Configurations

Description:
File analysis form and configuration option icon targeting modal below
(This is an old view, updated view is below)

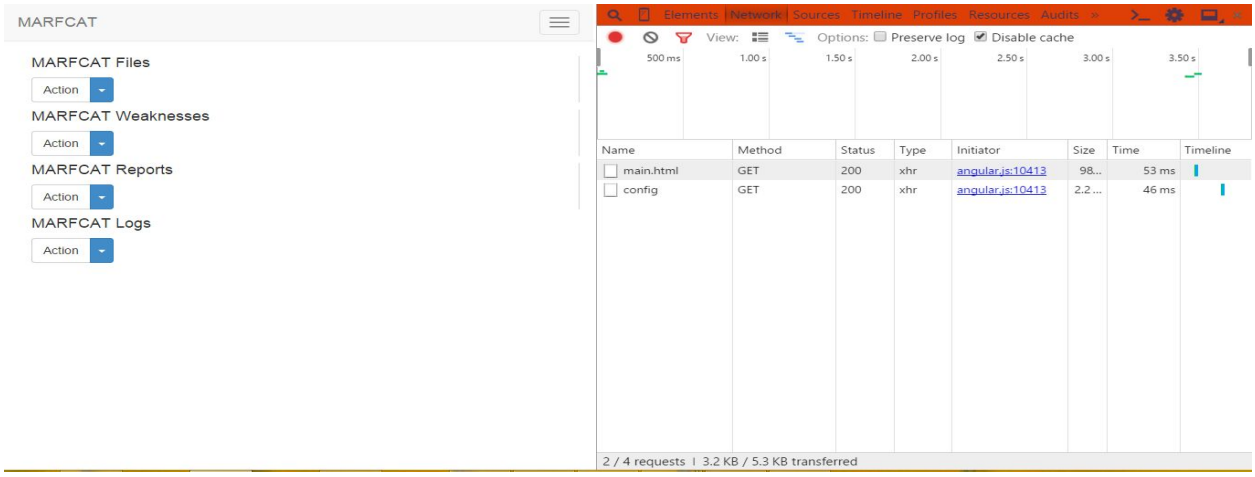
Configuration Modal with Pre-Selected Options

Description:
Modal with configuration options pre-selected for the user



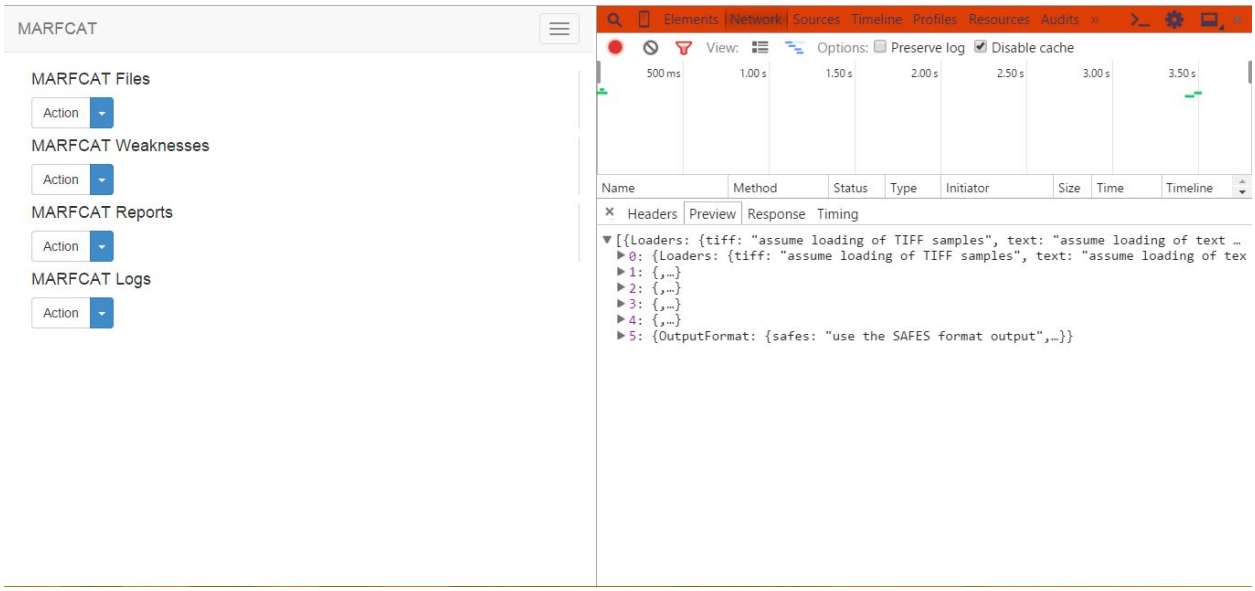
Configuration Options Dropdowns (Modal)

Description:
Pop up modal shows various dropdowns for options of each configuration algorithm
(Updated modal is below)



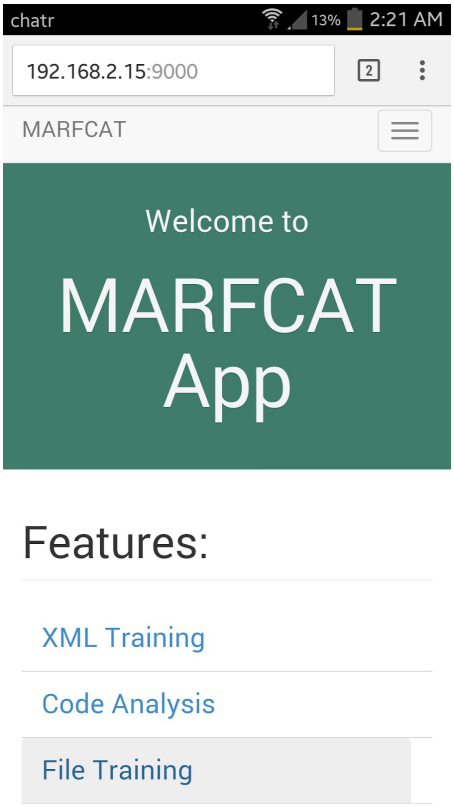
GET Configuration options from MARFCAT

Description:
‘config’ XHR is successful with 200 OK. Array of JSON objects containing configuration objects is received



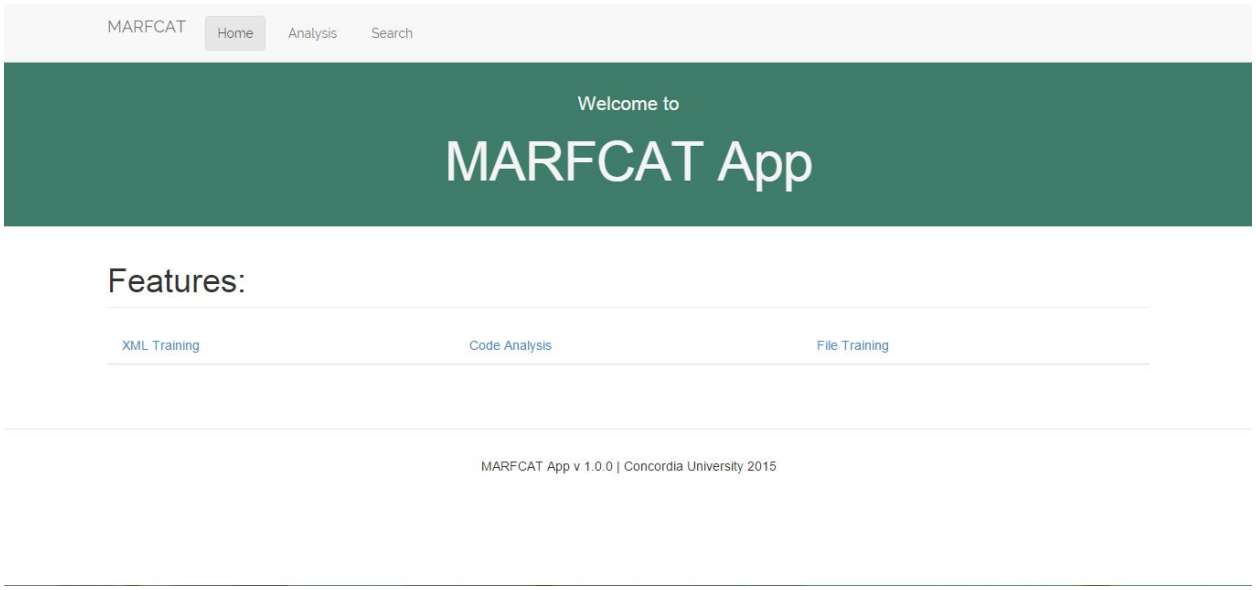
Configuration Options XHR Response

Description:
Response of ‘config’ XHR, showing the array of json objects that was built in WS



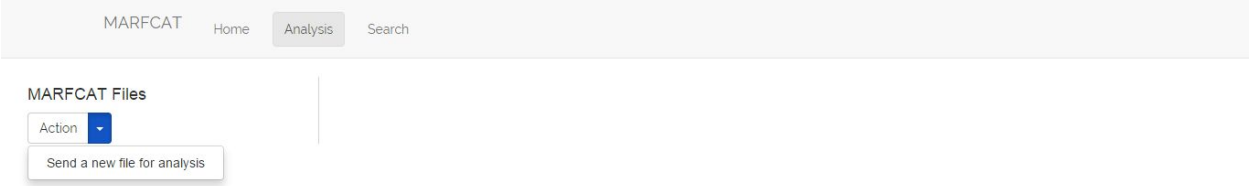
Responsive Design

Description:
The application has a responsive design on mobile devices



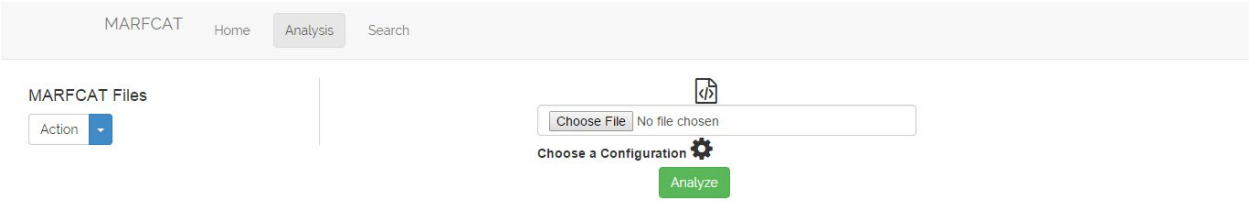
Update Navbar & Font

Description:
Updated view including navbar and family font, it is still the home page of the application, from here one can navigate to both analysis or search views



Analysis View

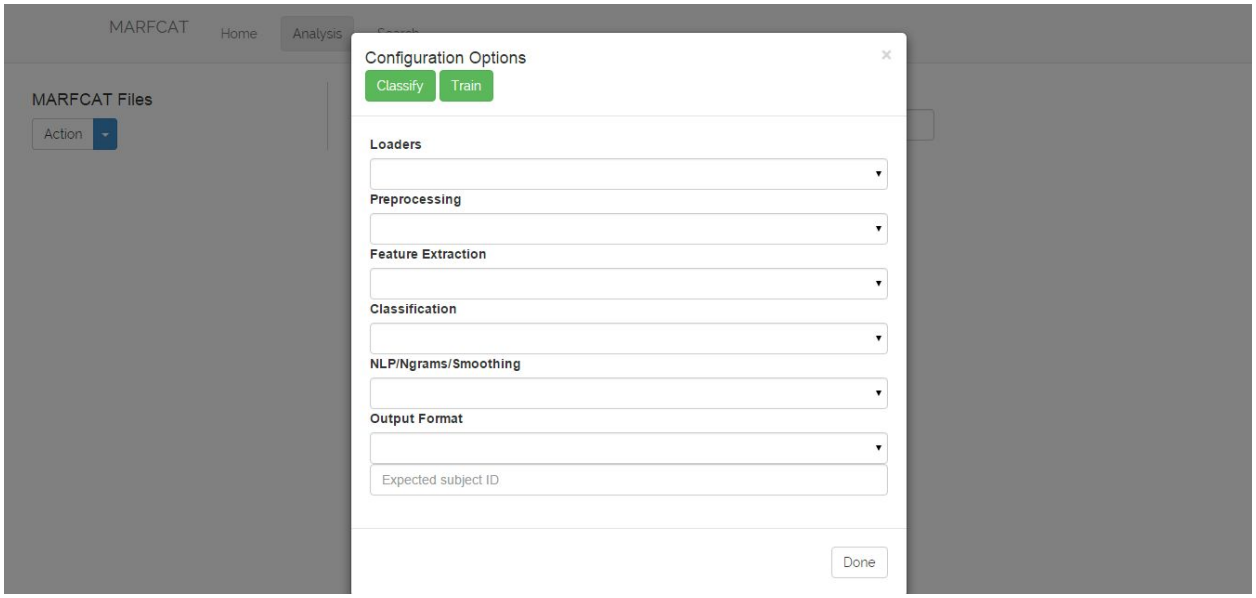
Description:
Analysis view showing a drop down list for manipulating MARFCAT files



File analysis form

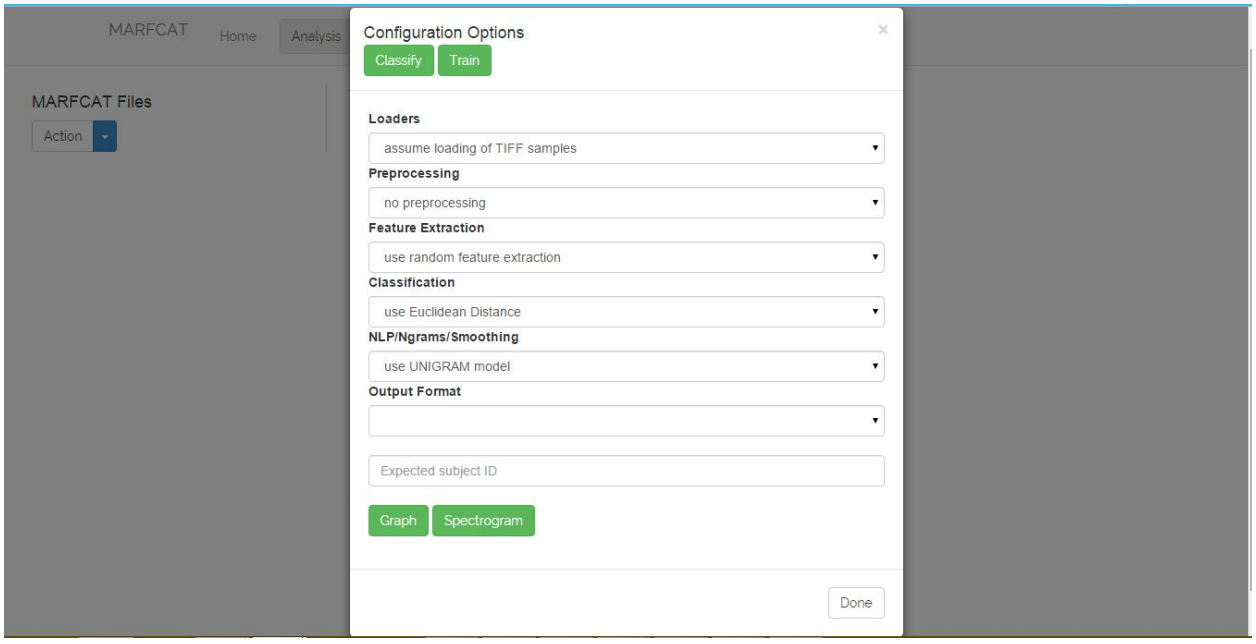
Description:
Analysis view displaying file analysis form and configuration button

=



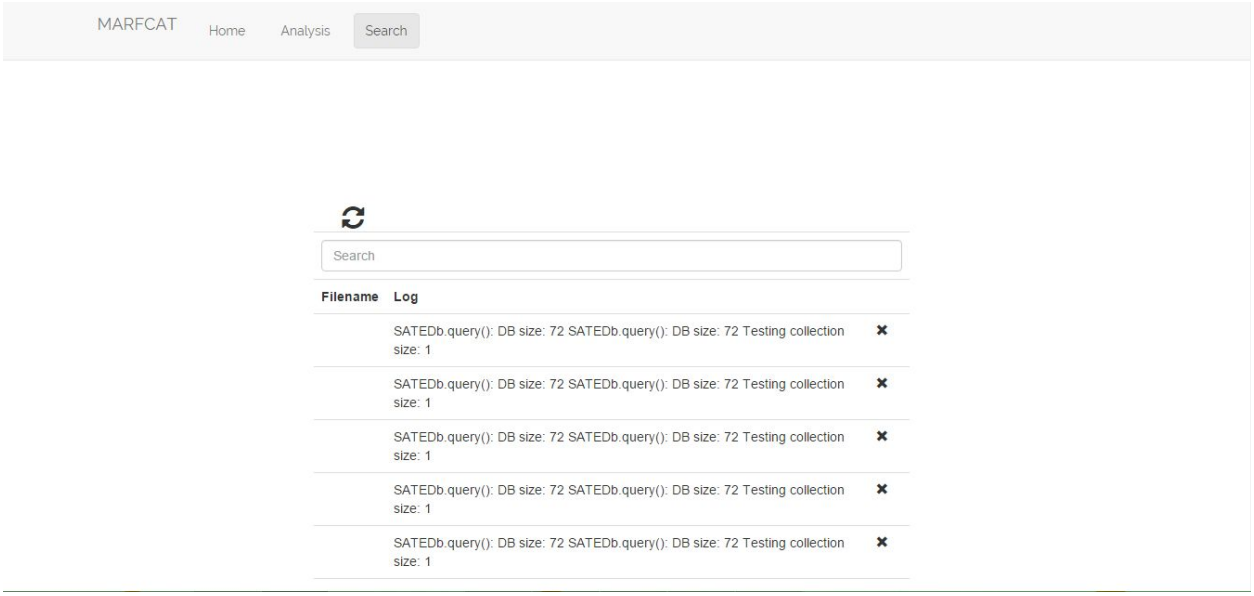
Configuration Modal

Description
Modal that is triggered after the icon click which shows the configuration options



Configuration Modal Default Options

Description:
Update of the configuration modal, the latter has pre-selected options for the user to be able to skip this step faster.



Search View

Description:
Search view showing a table with reports' details and a search bar to filter the displayed results
It will be used to make search queries and other data manipular as deleting logs or edit them.

Testing

Postman [7]
Chrome extension used to test file upload against localhost:9000
Once the application is running one can test the upload file feature through postman.
The output of our test is displayed in the command console

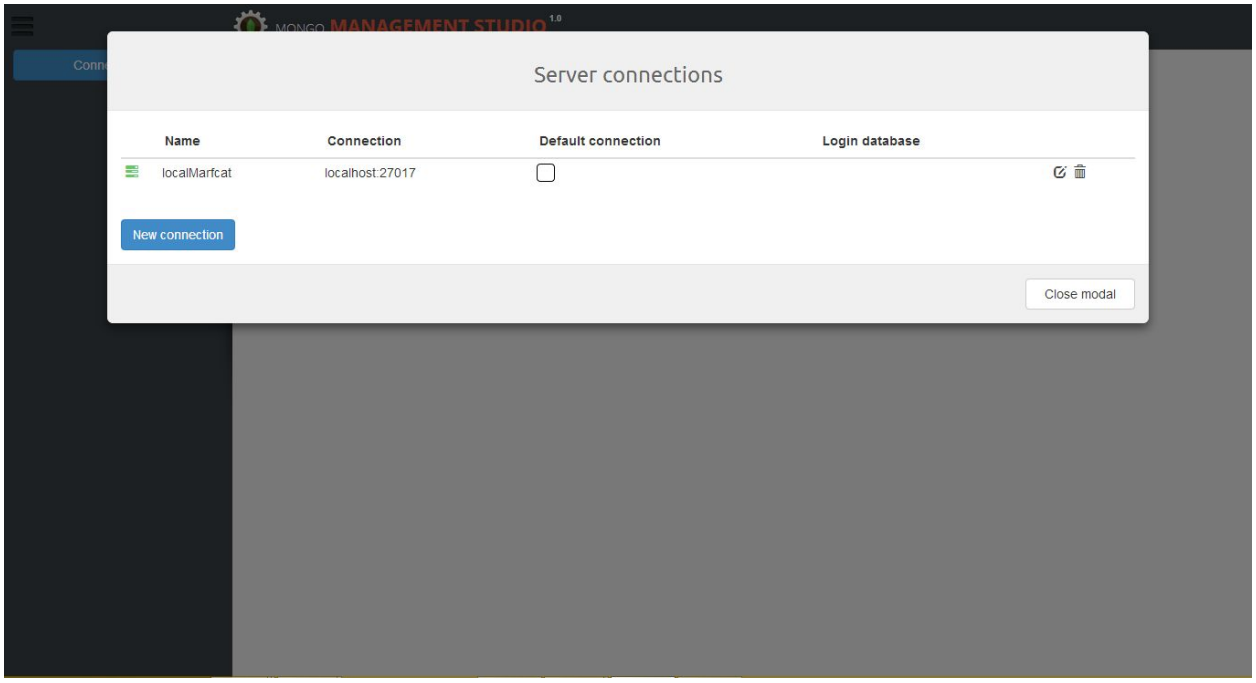
Postman features include:

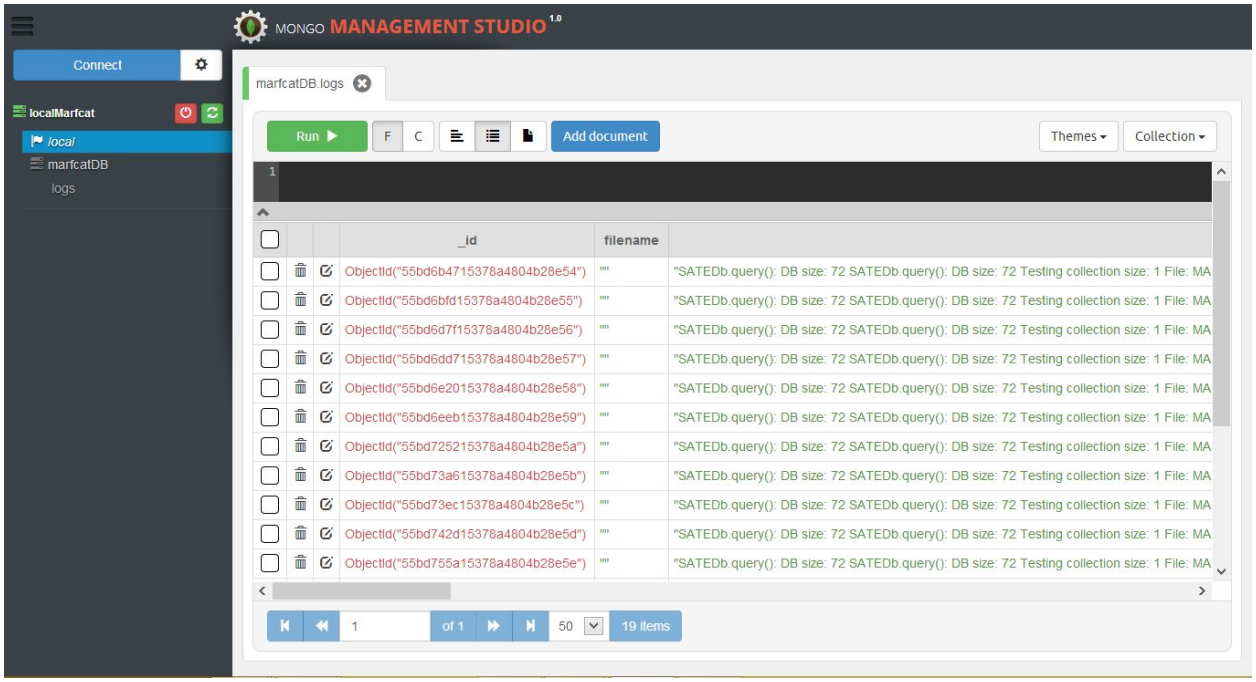
History of sent requests
Create requests quickly
Built-in authentication helpers

Mongo Management Studio [11]

Mongo Management Studio is the best way to work with MongoDB the easy way. Because of the clean and light user interface, you can execute the typical MongoDB commands fast and effective, without using the MongoDB shell. It is suitable to assists your local development and

test process as well as working with your Mongo databases on remote or production servers. By using a basic but strict user and rights management you have full control of the access rights of every single user. Of course you are also able to utilize the integrated user management of MongoDB with Mongo Management Studio.





Description:

Above is the Mongo Management Studio GUI, the first screenshot is about the connect step that helps the administrator to connect to the right database. The second screenshot represent the actual administration part of the tool used.

Limitations

The Web service (REST API) is deployed on Glassfish 4 which is running on port 8080 and the FrontEnd is running on a different port 9000. The fact that both are running on different ports makes difficult to make XHRs from port 9000 to 8080. This phenomenon is related to resource sharing and is fixed by CORS which stands for Cross Origin Resource Sharing. The Web Service was modified by adding headers to its methods (GET, POST) so it can accept request calls from a different port.

Configuration options that the Web Service receives from MARFACT are represented by one big string, which is parsed and converted to a JSON object. The latter will facilitate the iteration between its (key,value) pairs on the client side.

Current Status

Acknowledgements

SOEN 487 Team, Winter 2014

Mehdi Mokhtar Jamai

Paolo Ascura

Dong Li

Cheng Cheng

References

1: OpenESB Community. (n.d.). Retrieved September 7, 2015.

2:The web's scaffolding tool for modern webapps | Yeoman. (n.d.). Retrieved September 7, 2015.

3: DaftMonk/generator-angular-fullstack. (n.d.). Retrieved September 7, 2015.

4: The smartest JavaScript IDE. (n.d.). Retrieved September 7, 2015.

5: A free Git & Mercurial client for Windows or Mac. (n.d.). Retrieved September 7, 2015.

6: Cmder | Console Emulator. (n.d.). Retrieved September 7, 2015.

7: Postman. (n.d.). Retrieved September 7, 2015.

8: Mongoose. (n.d.). Retrieved September 7, 2015.

9: Express - Node.js web application framework. (n.d.). Retrieved September 7, 2015.

10: HTML enhanced for web apps! (n.d.). Retrieved September 7, 2015.

11: Mongo Management Studio - The simple MongoDB GUI. (n.d.). Retrieved September 7, 2015.

12: Node.js. (n.d.). Retrieved September 9, 2015.

13: Git and Mercurial code management for teams. (n.d.). Retrieved September 9, 2015.

14: MARFCAT literature

15: Mokhov, S.A.; Paquet, J.; Debbabi, M., "MARFCAT: Fast code analysis for defects and vulnerabilities," in Software Analytics (SWAN), 2015 IEEE 1st International Workshop on , vol., no., pp.35-38, 2-2 March 2015
doi: 10.1109/SWAN.2015.7070488