

TDP003 Projekt: Egna datormiljön

Systemdokumentation

Författare

Arturas Aleksandrauskas
Joakim Johansson

Innehåll

1	Revisionshistorik	2
2	A simple description	2
3	Installationsmanual	2
4	User Funktions	3
5	Folder structur with description of the files	4
5.0.1	Folder: doc	4
5.0.2	Folder: static	4
5.0.3	Folder: templates	4
5.0.4	File: README.md	4
5.0.5	File: data.json	4
5.0.6	File: data.py	5
5.0.7	File: data test.py	5
5.0.8	File: myFlaskProject.py	5
5.0.9	arty.log	5
6	Overview of the system	6
6.1	Example Senario - Frontpage	7
7	Description of the system components	8
7.1	Client Browser	8
7.2	Flask Webserver	8
7.2.1	hello()	8
7.2.2	techniques()	8
7.2.3	list()	8
7.2.4	project()	8
7.2.5	error()	8
7.3	Datalager API	8
7.3.1	load()	8
7.3.2	get project count()	8
7.3.3	get project()	8
7.3.4	search()	8
7.3.5	get techniques()	8
7.3.6	get technique stats()	9
7.4	JSON Databas	9
8	Others	10
8.1	Logging	10
8.2	Error handling	10
8.3	Tools and programs	10

1 Revisionshistorik

Ver.	Revisionsbeskrivning	Datum
1.3	Lagt till allt.	161013
1.2	Back in buissnies	161013
1.1	Mysterium	161012
1.0	Dokumentet skapas och enkel information har lagt in som en start	161012

2 A simple description

This project is a simple webpage that uses a python backend that serves the pages through a module called Flask.

The purposes of this website is to present and list all projects the user has previously been involved in. A visitor of the site can scroll through all, or search for a specific project.

3 Installationsmanual

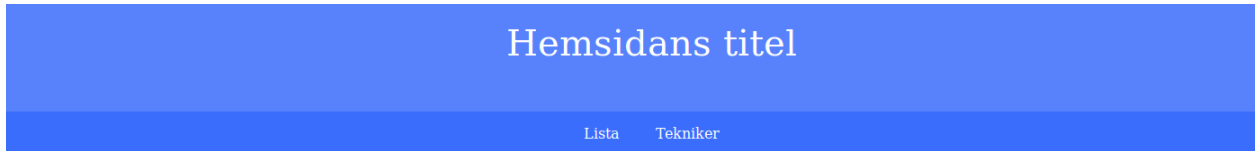
If you got this systemdocumentation file, you have probably also got a separate pdf file that is the Installationmanual. Use that if you are going to install the system on a new computer.

In case you haven't got the PDF, here is the link to an online version:

<https://gitlab.ida.liu.se/antsu07/tdp003-installationsmanual-2016/blob/master/installationsmanual.pdf>

4 User Funktionen

The user will be able to navigate around on four different pages.



On the top of the webpage there is a header with the name of the website, which the user always will be able to click to come back to the frontpage really fast.

Under the header, there is a navigation bar with two links. The List page and the Technologies page. This navigation bar will be accessible from all pages on the site.

The first page is a simple frontpage that displays useful about-information about the website's author. The page also displays all the three latest projects. Visitors can of course click on the projects to show more information.

The second page is the List page, which the user can get to by clicking in the navigation bar or searching from the 404 error page. The list page is made up of a long list of all projects in the database. There is also a box of tools to filter the list. Visitors can search with free-text, choose required techniques, and choose how the list should be ordered. And of course there is a search button to apply the filters to the list.

The third page is the Technologies page. This is more of an overview over all projects with the techniques in focus. This page is great for visitors to quickly see what type of techniques the author uses the most. But it is also really handy when a visitor looks at a project that uses ex. python and wants to know other projects the author has done where he also has used python.

The fifth page is a page that only shows a specific project. This page is hidden in the menu and you can only get to the page if you click on a project somewhere. This page shows a lot of information about the project, ex. the techniques used, start and end date, people in the project and more. If the visitor clicks on a technique, he will be directed to the technique page and moved down to the right place for the clicked tech.

The last page we have is an error page that is displayed if the user tries to go to a page that doesn't exist. Simply as that. The page has a title, a short text and a search box so that the user can continue trying to find the right page.

5 Folder structur with description of the files

```
.
Root/
  doc/
  static/
    images/
    style/
      style.css
  templates/
    index.html
    list.html
    project.html
    technologies.html
    error.html
  README.md
  data.json
  data.py
  data_test.py
  myFlaskProject.py
  arty.log
.
```

5.0.1 Folder: doc

A folder with all documents related to this project but isn't actual code, like this document for example.

You can also find, the project plan document, lofi prototype and the developers diary if you wanna follow the development, in this folder.

5.0.2 Folder: static

A folder with all the static files, like css, images and fonts.

We have no images yet, but we have added a folder incase we wanna do future graphical enhancements to the site. We also have a folder with the name style with a single css file inside it that is used to style all pages.

5.0.3 Folder: templates

A folder for all the HTML templates.

We use five templates. One for each page on the site. index.html for the front page, list.html for the list page, project.html for the page that shows one project, technologies.html for the page with all the projects sorted by tech and error.html for the error page.

5.0.4 File: README.md

A description file with info about the project and good information for outside people incase they open up this project and don't know what it is.

5.0.5 File: data.json

Our main database name with all data from all the projects. Used by the API.

5.0.6 File: data.py

This is the API-side of the project. Gets imported by the flask file and is used on almost every HTTP request.

5.0.7 File: data test.py

This is a test file that is used to test the data.py file so that it is ok on all the requirements made up by the university.

5.0.8 File: myFlaskProject.py

This is the main flask file that is active running on the server and waiting for HTTP request to handle.

5.0.9 arty.log

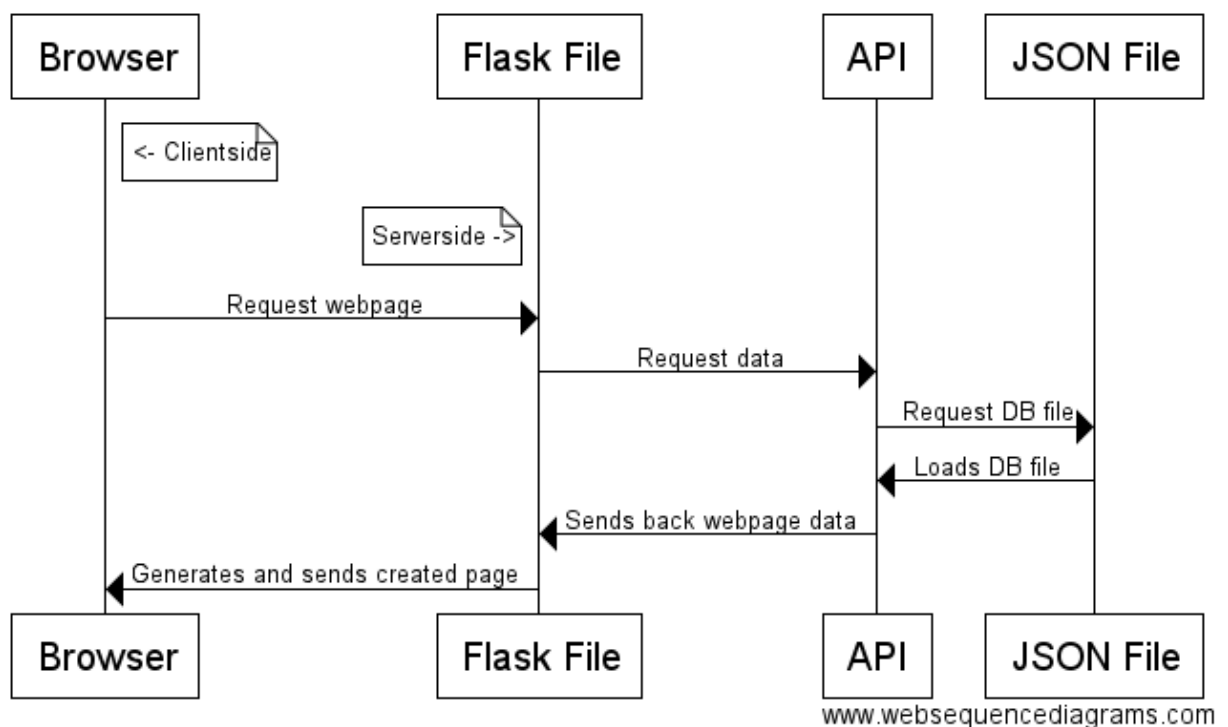
This is the log file that is created by flask and contains all logs and actions that flask has done.

6 Overview of the system

The system is a basic system for handling some webpage requests and some data.

It's made up of a client side and a server side. The clientside is simply the users browser, with the functions of requesting a page using the GET method.

The server side is made up of two main processes, the python flask webserver file and the API file that handles all the data. The API file uses a JSON file to store all the information.

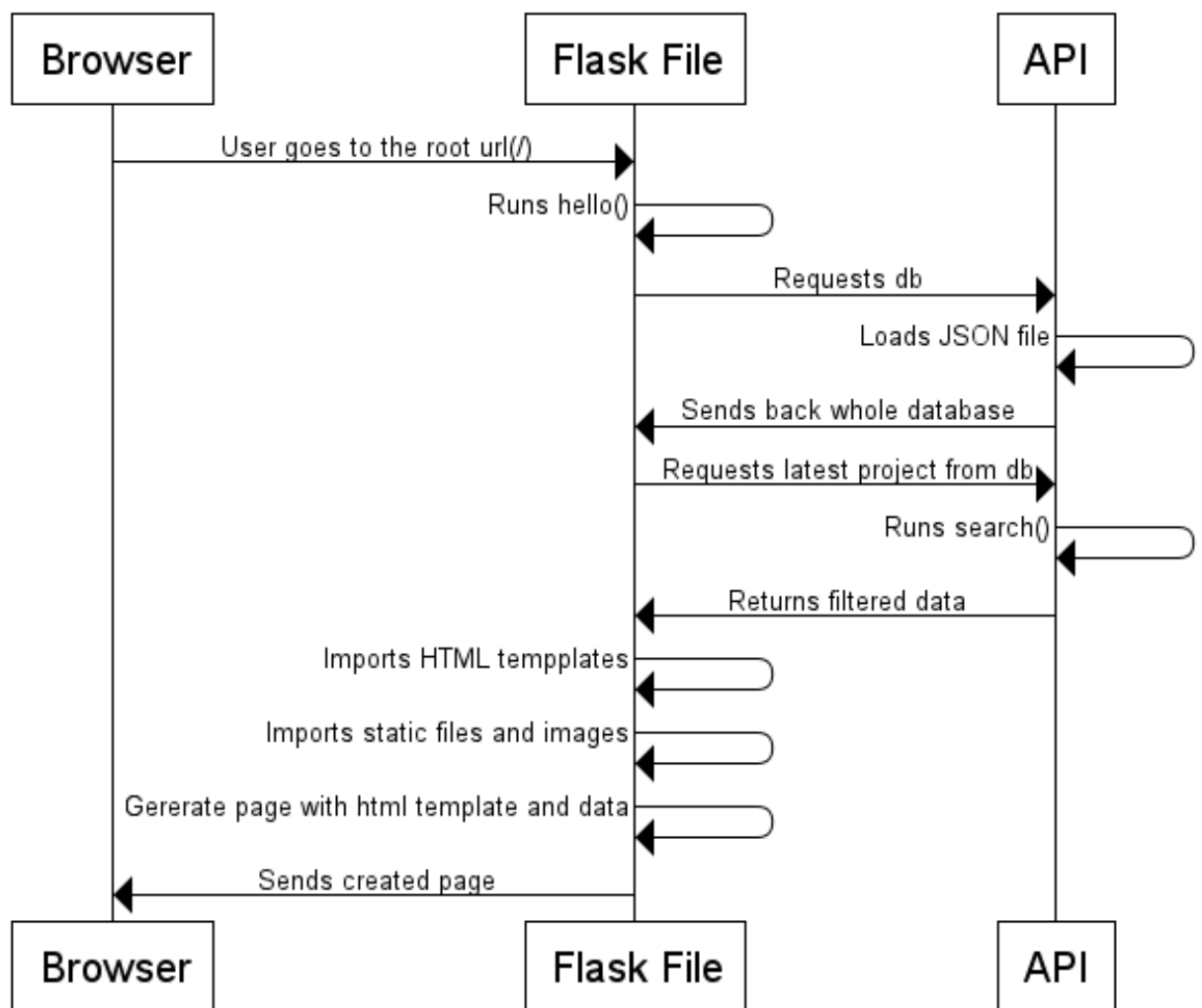


6.1 Example Senario - Frontpage

When the user goes to the root of the webpage, the browser sends a request to the the server where flask is waiting and have a function bind to requests for the root directory. In this case, the hello() funktion is run. Hello() does two things, requests the database file from the API and then sends it back to the API layer with the request that it should search and filter the file. In this case we only the sends the database parameter to get all the latest projects with the newest projects first, but we could specifiy multiply parameters to tell the search function how it should filter the projects.

When the Flask layer get the filtered data, it imports the HTML templates, css files and other static files like images. Then it generates the HTML-files from the templates and the filtrered data, and finally sends it back to the users browser that displays the webpage to the user.

User requests Frontpage page (at url "/")



www.websequencediagrams.com

7 Description of the system components

7.1 Client Browser

Client Browser is styled with CSS3 and built with HTML5. HTML files are stored in /templates and CSS files are stored in static/style.

7.2 Flask Webserver

7.2.1 hello()

Specific Root page

7.2.2 techniques()

Specific Techniques page

7.2.3 list()

Specific List page

7.2.4 project()

Specific Project page

7.2.5 error()

Specific Error page

7.3 Datalager API

7.3.1 load()

Loading the file and returns it if found else returns None ¹:2016

7.3.2 get project count()

Takes in the database and returns amount of project in the database ²:2016

7.3.3 get project()

Takes in the database and returns the project with the selected ID ³:2016

7.3.4 search()

Takes in the database, sort by, sort order, techniques, search, search fields and searches for all the specifications that the indata has ⁴:2016

7.3.5 get techniques()

Takes in the database and returns a list of all projects for the database ⁵:2016

¹http://www.ida.liu.se/~TDP003/current/portfolio-api_python3/

²http://www.ida.liu.se/~TDP003/current/portfolio-api_python3/

³http://www.ida.liu.se/~TDP003/current/portfolio-api_python3/

⁴http://www.ida.liu.se/~TDP003/current/portfolio-api_python3/

⁵http://www.ida.liu.se/~TDP003/current/portfolio-api_python3/

7.3.6 `get technique stats()`

Takes in the database and returns a list of all techniques and its projects (id and name)⁶:2016

7.4 JSON Databas

JSON is a file with a list of dictionaries that contains all the info.

⁶http://www.ida.liu.se/~TDP003/current/portfolio-api_python3/

8 Others

8.1 Logging

Flask is setup in a way that it should log every action that happens in a logfile in the root directory named `arty.log`.

8.2 Error handling

The website redirects the user to the error page incase they tries to go to a url that is not valid.

If you want to enable debug mode in flask, simply add `debug=true` as a parameter in the `app.run()`. You will then be able to see every action in the terminal and you will also be able to edit the files and see the change on the page without restarting the server.

8.3 Tools and programs

To compile all the documents in the doc folder, you have to use a latex compiler to export a PDF from the code. On linux, we recomend using the command `latexmk -pvc -xelatex` and then the filename of the `.tex` file of the wanted document.