MyWorkWiki

Kerbal View Project

- KerbalView : http://uilennest.net/KerbalView [http://uilennest.net/KerbalView]

The KerbalView project is a technology demonstrator to try out various web technologies in a fun way. The starting point is a user (me) playing a game of `Kerbal Space Program` on a Windows laptop. Information from that gameplay is presented (live) in a browser that connects to my `Apache2` webserver that runs on a `Linux Mint` box.

Prerequisites

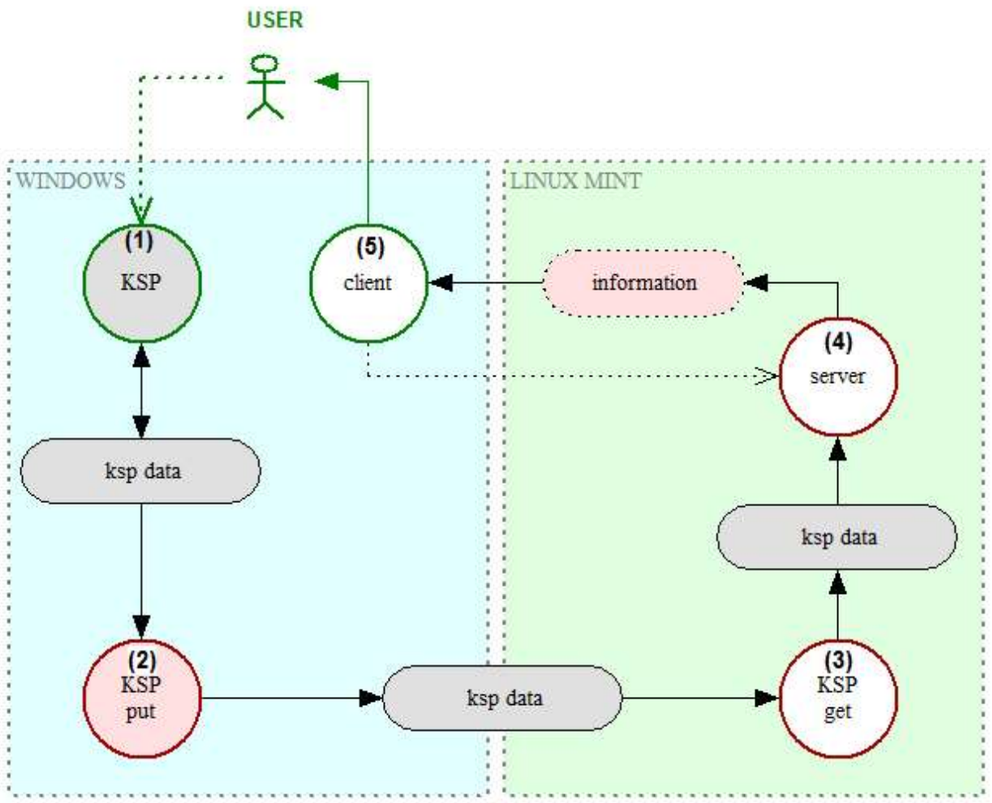
- A dropbox account for `nicomint@xs4all.nl` shares the folder `NicoMint` with dropbox user `nvermaas@xs4all.nl`
- Apache2 webserver is up and running. Folder `/home/nvermaas/www/KerbalView` is hosted and reachable from the outside.

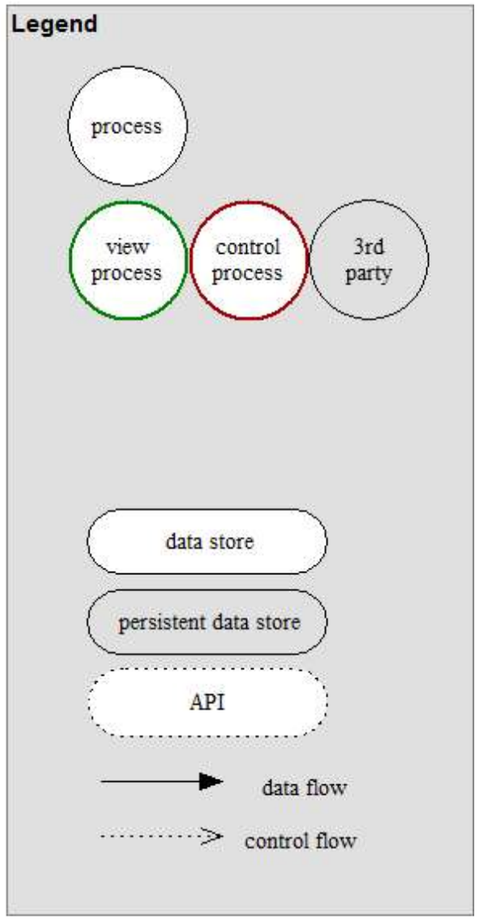
Chosen solution(s)

- I chose for 2 different methods/technogies to deliver the information to the browser
  - **(A) static/generated html**
    - a *node.js* application (`kerbal_write_html.js`) parses the kerbal data file and inserts a html fragment [http://uilennest.net/KerbalView/kerbal_html.htm] into the static "kerbals.htm" [http://uilennest.net/KerbalView/kerbals.htm] which is already served by Apache.
  - **(B) client - server**
    - a *node express* webserver runs the `kerbal_server.js` app, which responds to GET requests to deliver an API [http://uilennest.net/KerbalView/api/kerbals_json] in *json* format.
    - a frontend javascript application [http://uilennest.net/KerbalView] contacts the backend API and shows the json data.

KerbalView - Analyses

(rv: 24 aug 2017, kerbalview_analyses.sdr)





Functional Specifications:

(1) A runs KSP (plays the Kerbal Space Program game)
(2) The persistent (read only) data file is copied to the cloud when has changed.
(3) The persistent (read only) data file is copied from the cloud and handed to the server when it has changed..
(4) The server app extracts relevant info from the data file and serves that as an API to the outside world
(5) The client app requests the API from the server and shows the results to the user

---

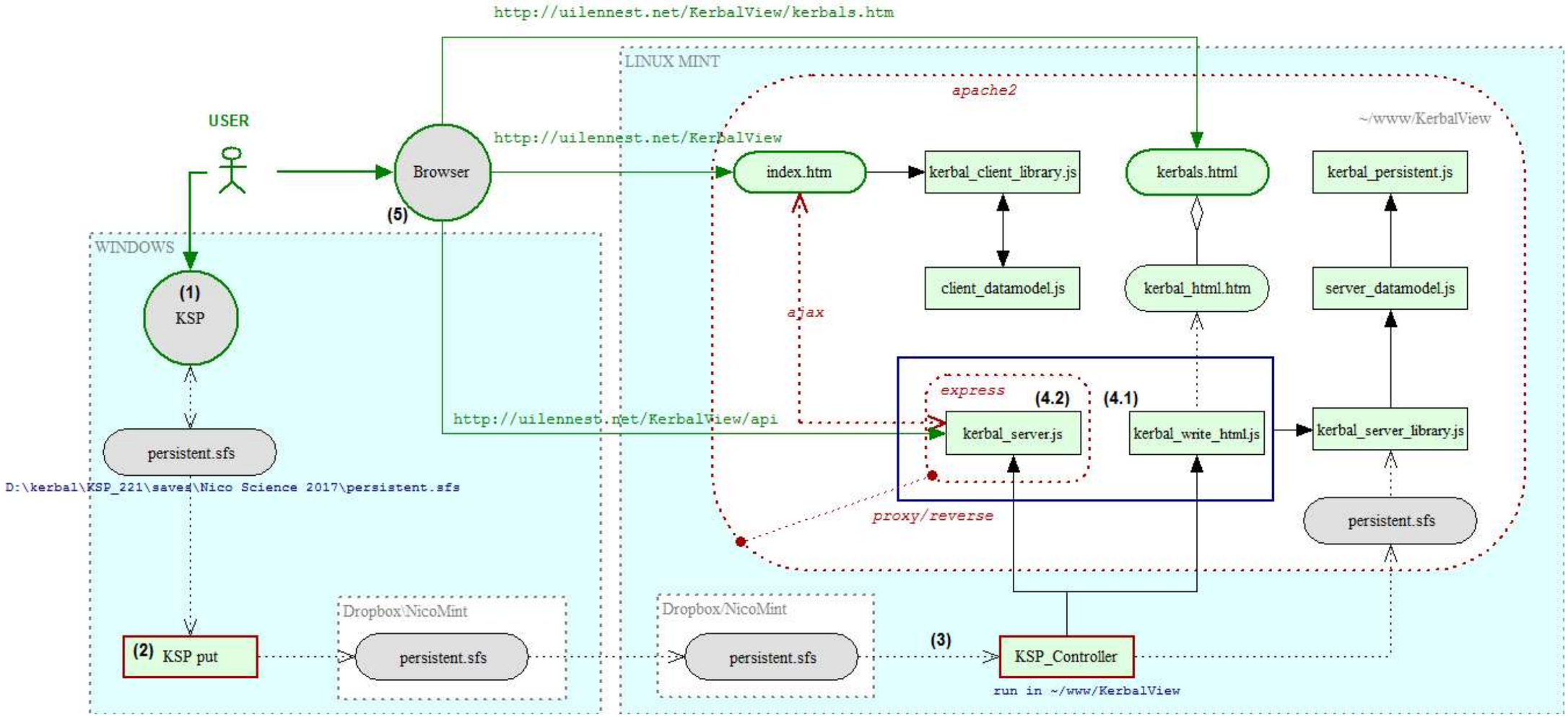
KerbalView - Technical Design & Implementation

(nv: 22 aug 2017, kerbalview_design.sdr)

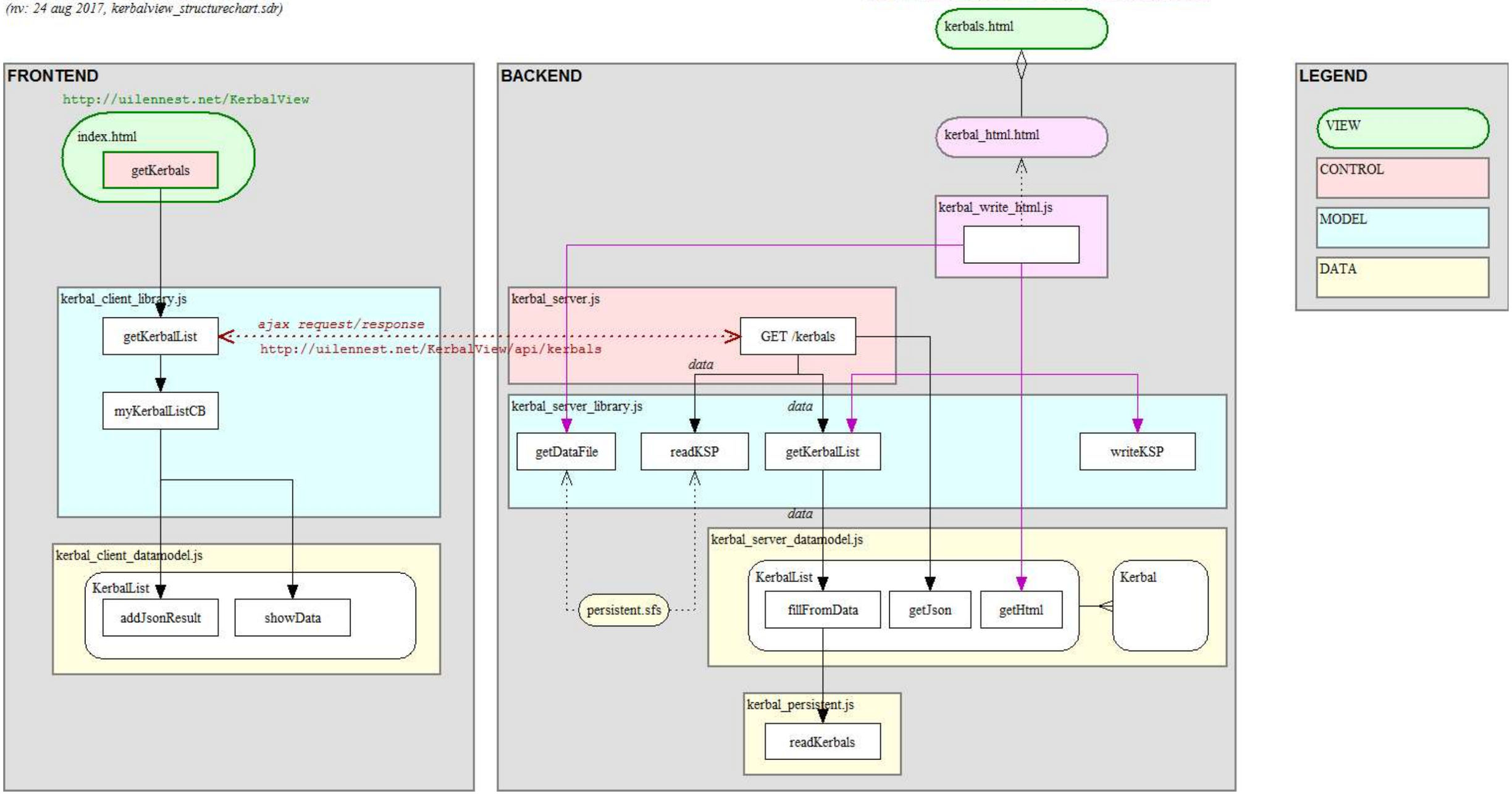
admin action | user action

to do | under construction | done

client (browser) and server (nodejs)
use a different javascript syntax
to expose functions.
(node uses 'exports').



KerbalView - Structure Chart
(nv: 24 aug 2017, kerbalview_structurechart.sdr)



Implementation

(2) KSP put

- run this command once, anywhwere (see script KSP_put.cmd)

```
mklink /J "D:\kerbal\KSP_221\saves\Nico Science 2017" "C:\Users\Vermaas\Dropbox\NicoMint\KerbalView\Nico Science 2017"
```

---

(3) KSP Controller

Run the `KSP_Controller` in `/home/nvermaas/www/KerbalView`, where it should stay active.

- sets up the link between directories
- launches the node filesystem monitoring for changes in `persistent.sfs`. If changes are detected then it executes the following commands:
    1. `cp saves/persistent.sfs .` (to make an accessible copy of the kerbal data file)
    2. `node kerbal_write_html.js`

KSB_controller

```
#!/bin/bash
# KSB_controller - Nico Vermaas - 19 aug 2017

cd /home/nvermaas/www/KerbalView
# link the dropbox folder to a local 'saves' folder to give the KerbalView app easier access
if [ ! -d saves ]; then
  ln -s '/home/nvermaas/Dropbox/NicoMint/KerbalView/Nico Science 2017' saves
fi

# start the kerbal_server.js, which serves the API on node express webserver
nohup node kerbal_server.js &

# use node filesystem monitor to detect changes in the Kerbal data file (persistent.sfs).
# If changes are detected then
# 1) copy the file to a local instance where the webserver can access it
# 2) execute the kerbal_write_html.js (node) program to update the html. (the html is included in index.html).
nohup fsmonitor -d '/home/nvermaas/Dropbox/NicoMint/KerbalView/Nico Science 2017' -s -p '+persistent.sfs' sh -c "cp saves/persistent.sfs .;node kerbal_write_html.js" &
```

shutdown and restart

Kill the 2 `node` processes that show up with the `ps -x` command. Followed by a restart and checking nohup.out for errors.

```
25887 ?        Sl     0:10 node kerbal_server.js
25888 ?        Sl     0:05 node /usr/bin/fsmonitor -d /home/nvermaas/Dropbox/NicoMint/KerbalView/Nico Science 2017 -s -p +persistent.sfs sh -c cp saves/persistent.sfs .;node kerbal_write_ht

> kill 25887 25888
> cd /home/nvermaas/www/KerbalView
> ./KSB_controller
> less nohup.out
```

---

(4) SERVER

(4.1) kerbal_write_html.js

This node js program is called by the controller to generate html that is included by `kerbals.htm`. This is a static webpage served by Apache and can be called directly with a browser. The main program is very simple, it only calls the business logic that is contained in the `kerbal_server_library.js [http://uilennest.net/KerbalView/kerbal_server_library.js]`

kerbal_write_html.js

```
// kerbal_write_html.js
// Nico Vermaas - 17 aug 2017
// node.js program to read 'description' from the Kerbal 'persistent.sfs' file and writes it to kerbal_html.htm

var ksl = require("./kerbal_server_library.js")

var fileNameOutput = 'kerbal_html.htm'

var data = ksl.getDataFile()
var my_description = ksl.getDescription(data)
var my_html = ksl.createHtml(my_description)

var myKerbalList = ksl.getKerbalList(data)
my_html += myKerbalList.getHtml()
ksl.writeKSP(fileNameOutput, my_html);
```

- SOURCE : `kerbal_server_library.js`
- SOURCE : `kerbal_server_datamodel.js`
- SOURCE : `kerbal_persistent.js`

(4.2) kerbal_server.js

This is a node js program that uses the `express` package to run as a webserver that delivers the REST API.

Deploy Node Express

- First install node express in the KerbalView directory (only once for initial setup, this step can be omitted for further updates of the kerbal_server.js).
  - `cd /home/nvermaas/www/KerbalView`
  - `npm install express -save`
  - `npm install cors -save` (to add `Access-Control-Allow-Origin` to the response header).
- see https://www.tutorialspoint.com/nodejs/nodejs_express_framework.htm [https://www.tutorialspoint.com/nodejs/nodejs_express_framework.htm]

Running

Run the command `node kerbal_server.js` in the `KerbalView` directory to start the `express` webserver and the app.
The app, and API, can then be reached *locally* with a browser on `http://127.0.0.1:8081` [`http://127.0.0.1:8081`]. **But is still invisible to the outside world**. See the next step to deploy to the outside world.

Proxy/ReverseProxy from Apache to Express

Port 80 (standard http) is open in the firewall, port 8081 is not (and will not). All traffic to port 80 is forwarded by the router to the 'nico-mint' linux machine in the network on IP = 192.166.178.37. This machine runs Apache2 webserver, which listens on port 80.

Device / Name	IP Address	Sharing	Port assigned externally IPv4
nico-mint	192.168.178.37 ::201:c0ff:fe1b:6fa1	HTTP-Server ssh MineCraft	80 22 25565

To make the `express` web server on port 8081 reachable we need to create a 'reverse proxy' in the `Apache2` webserver. This means that part of the url will serve as a trigger to forward the request somewhere else, in this case to the express webserver on port 8081. Add the following lines to the `apache2.conf` file to forward every url containing 'KerbalView/api' to our kerbal_server app on port 8081. (make a copy of apache2.conf first, just in case).

```
ProxyPass               "/KerbalView/api"  "http://127.0.0.1:8081"
ProxyPassReverse        "/KerbalView/api"  "http://127.0.0.1:8081"
```

Enable the proxy module in Apache with the following commands as root:

- `a2enmod proxy`
- `a2enmod proxy_http`
- restart with `service apache2 restart`.

- SOURCE : `kerbal_server.js`
- SOURCE : `kerbal_server_library.js`
- SOURCE : `kerbal_server_datamodel.js`
- SOURCE : `kerbal_persistent.js`

---

(5) CLIENT

(A) Generated HTML

- static website : http://uilennest.net/KerbalView/kerbals.htm [http://uilennest.net/KerbalView/kerbals.htm]
- generated part : http://uilennest.net/KerbalView/kerbal_html.htm [http://uilennest.net/KerbalView/kerbal_html.htm]

(B) Frontend + Backend API

- Frontend : http://uilennest.net/KerbalView [http://uilennest.net/KerbalView]
  - screenshot
- Backend : http://uilennest.net/KerbalView/api [http://uilennest.net/KerbalView/api]

  - http://uilennest.net/KerbalView/api/description [http://uilennest.net/KerbalView/api/description]
  - http://uilennest.net/KerbalView/api/kerbals [http://uilennest.net/KerbalView/api/kerbals]
  - http://uilennest.net/KerbalView/api/kerbals_json [http://uilennest.net/KerbalView/api/kerbals_json]
  - http://uilennest.net/KerbalView/api/kerbals_html [http://uilennest.net/KerbalView/api/kerbals_html]
  - http://uilennest.net/KerbalView/api/kerbal?name=Neil [http://uilennest.net/KerbalView/api/kerbal?name=Neil]
- SOURCE : `index.html` (frontend)
- SOURCE : `kerbal_client_library.js`
- SOURCE : `kerbal_client_datamodel.js`

Additional functionality

- Bootstrap Carousel [https://www.w3schools.com/bootstrap/bootstrap_carousel.asp] for the screenshots

Misc

- Datafile: http://uilennest.net/KerbalView/persistent.sfs [http://uilennest.net/KerbalView/persistent.sfs]
