milliways 1.0

code manual

The UNIX principles:

“Keep it small and simple, don’t do it twice and build your work on top of others.”

Index

1. Introduction

This manual describes the programming code of milliways. It provides reasons for the choices done and structure given within the milliways Web-GUI.
This manual - from describing the current status of milliways code and reasoning how it has developed from the past - will therefore supply all kind of principles for the future of milliways.

milliways was born as a very low-level project document repository. It was based from the very beginning on a SQL database system in the background connected to a “Big-Data” file server. From the very beginning the ftp server nested in the Linux operating system configuration has proven to be a very valuable tool.

In fact the ftp-functionality was the main reason why milliways was created: the users in the field of architecture and engineering wanted to download and backup their data at any time and at any extent. All other known tools on the web (e.g. mediafire, dropbox etc.) did not provide such feature. They rather made the user feel that he is not anymore in command of his data once he uses their more or less proprietary tools.

ftp may be the oldest protocol on the internet and some may regard it old-fashioned and outdated. But in reality there is no other alternative available - taken into account not only the technological view but also the holistic view on data security and safety. milliways gives the user back his command to his own data. It seems that this feature is becoming more valuable in times of Google, Amazon and Facebook.

Looking at the current market the user still has very few alternatives when demanding a truly open source, open licensed and scalable tool for his data management in internet.
milliways - by its very nature of incorporating Linux-Operating System features - offers this to the user. It is absolutely free to use and the code is completely available. It is not restricted by any license policy except that successors and further developments should not take it to a commercial career and they must name the origin - thus ensuring the tradition of open source.

Open source seen as a commercial alternative for costly software is only in second priority. With milliways the originators intended primarily to establish an open source standard that can be spread around the users without any limit. As an open standard it does not belong to anyone. It is common property. The history of the internet and the web languages can tell us how powerful this can become. It takes its time but it will be for the good for the common stakeholders - not for only few shareholders.

The originators of milliways do not fear to have wasted time and money when developing milliways. In fact it has been proven that open source code has higher quality and is therefore more efficient and causes less work in the long run. Obeying the UNIX principles (cf. entrance page) will make you work with more fun and with better results - especially in terms of time and money!

The predecessors of milliways were some code modules already developed decades before. They have been tested and every so often redesigned in various environments (e.g. in an event database, in an address database (the very beginning), for a document storage system and a digital dictionary (hyperbuild.info comes from there)). All of these modules were always free code. All of them ran under Linux and the XAMPP environment. As a consequence all further development - and this system of milliways is a consequence - will be open source in its very sense. Therefore milliways uses the Creative Commons License CC-BY-NC-SA.

milliways is more than only a software package. It is a system build up by interrelated hard- and software-components. Therefore this code manual encompasses more than only the listing and description of the mere software code. milliways can only be understood completely knowing also the hardware. Installing milliways is only possible when also consulting the more hardware oriented installation manual. The user may decide when and where he reads in these documents. It is his data that his i managing with his version of milliways. milliways as such belongs to everybody.

2. Scope

This document is to inform on the principles and concepts of the programming code of the milliways system. It will describe

- the external structure of the complete set of milliways programming files and
- the principle internal structure of milliways program files.
- internal file structure and code in detail

The program files (software) are mostly coded in PHP. They determine the “Look and Feel” of the Graphical User Interface (GUI) which must supply sufficient functionality to the User.

All program files containing program code (HTML, PHP) are ASCII-readable and have internal comment lines at specific places.

The intention is to supply the reader with a concept that has been elaborated through many years of programming experience. The milliways concept has been developed by programming work on-the-job. It has happened that major new needs coming from the users resulted in a significant structural rebuild of the code. But it is also manifest that within the last years no major structural changes where made - thus telling us that the now founded structure seems to be a fairly good base to work on and make further developments.

This document will also explain some philosophy of milliways. It will tell why some features have been programmed thus this way and not the other ways around.

This document will also serve as a guide to possible further development in terms of programming.

3. Terms and Definitions

Please also refer to terms and definitions in the main user manual and consult hyperbuild.info.

3.1 project

defined regime of use for a milliways setup. All data within a project belong to a common data environment accessible only to a defined set of users.
Note: It is possible that data is being transferred from one project to another.

3.2 system

a technical setup of soft- and hardware providing an environment to set up a project

3.3 attribute

“column” in a SQL-dataset. a SQL-database is made up by tables. Each table is characterized by its various columns (e.g. ID, name, street, etc…). The set of columns is specific to the table and therefore attributing to its character. Therefore a column is also called an attribute. All values in one column must be of the same type (e.g. number or string - e.g. a column or attribute called “name” must be of string type). Each value in a column characterizes resp. attributes to the dataset it belongs to. A dataset is one row in a SQL-database table.

3.4 admin role

user being able to access all projects at the same time via the Web-GUI of a milliways system. Admins have access to all datasets of one milliways installation. Admins must take special care when transferring datasets across projects.

3.5 CRUD functions

process or function of Creating, Reading, Updating or Deleting data. Those are the basic functions necessary for any information management system.

3.6 ACID functionality

database functionality so that every CRUD process is guaranteed to be handled Completely (Atomicity), Consistent, Isolated (Independently) and Durable.

4. Basic Concept and Principles

The milliways system is made up by of three main elements:

1) the hardware system made up by a (Linux-)Server accompanied by a large File Storage System,

2) the software programs implemented on the (Linux-)Server mainly coded in PHP and

3) the configuration necessary to the Linux-Server to allow for ftp and other services.

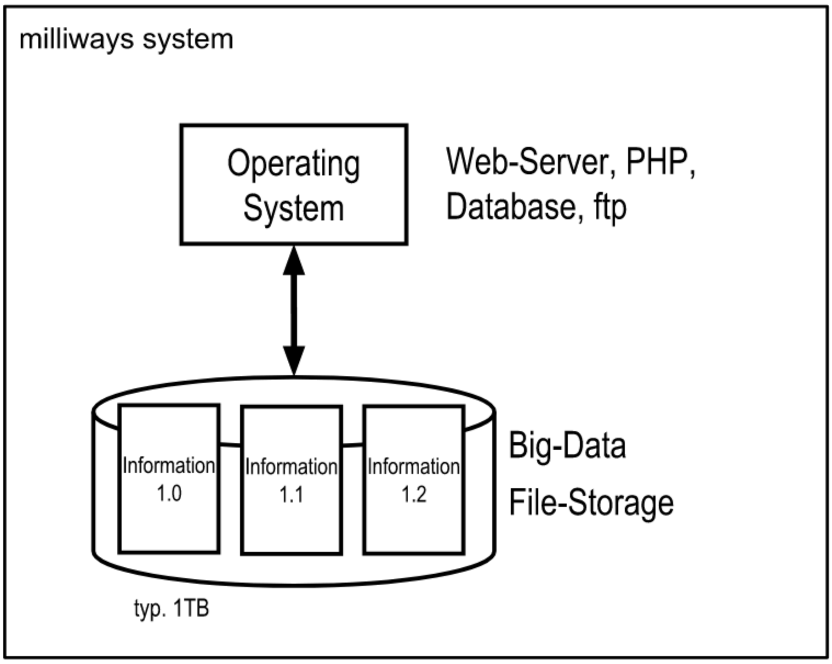


Figure: the milliways system

The first and third element are described in detail in the installation manual. Currently all milliways installations run on Linux servers. Servers with other Operating Systems are possible. But as of their non-proprietary nature it may need different routes to configure and may cause difficulties.

This document provides a complete overview to understand the interactions and relations in the milliways system. This document then focuses on the programming code providing a GUI to access some of the system resources.

4.1 PHP as main Programming Language

PHP has been chosen as main programming language code, as

- it is common to a large community of programmers thus ensuring a further development without looking for rare specialists (maximum production market share) ;
- it is still under active development (current version 7.2.10 from Sept 2018) (living product);
- it has a large command set, can handle multiple variables in very variable ways and supplies many function libraries like SQL (maximum application range);
- it is server-side thus ensuring a low-level client base. Acording to Wikipedia 83% of all web servers use PHP (maximum technology base);
- its newer version include command sets to handle object-oriented data thus supplying an alternative to Java (maximum satisfaction to market requirements);

Remembering the UNIX principles (cf. entrance Page) milliways tries to use a minimum set of commands offered by PHP.

With few exceptions milliways makes use of only basic PHP commands so that a future transfer into another language is possible without major hindrances.

This also ensures a maximum of downward compatibility in case the chosen technical configuration is using previous versions of PHP (Downward Compatibility is less than 100% with PHP 7).

Being under continuous and active development PHP may also support new Standards like SPARQL.

PHP is a server-sided. This can causes a high Server-Workload. But due the contemporary standard of high-end Server Supply with multiple Cores and RAM this will not affect milliways performance.

4.2 XAMPP as Development Server Base

XAMPP is a package supplied by the Open Source community Apache Friends. The combination of these letters stand for the main components (Web-Server, Database-Server, Programming Code Parser, ftp-Server) of the package:

- X stands for various the Operating Systems (Linux, Solaris, macOS, Windows) the package can be installed on. The package is suited and named to the specific type of Operating System (e.g. LAMPP for Linux only..)
- A stands for the Apache Web-Server integrated in the package. Apache is the far most commonly used Web-Server in Internet. Is is often already integrated

in many Linux Distributions. Therefore installing the XAMPP package can cause double installation (cf. Section Known Bugs in Install Manual).

- M stands for the Maria Database System and formerly for MySQL Database System (which is the less advanced Oracle Fork of Maria DB).
- P (the first one) stands for PHP or Perl as Programming Code that can be used to access Webpages to the Servers.
- P (the second one) stands for ProFTPD as the ftp-Server. If the basic server system (e.g. a debian Linux installation) is located outside the physical range of the server administrator or the accessing user (e.g. it is a virtual server hosted in some data center) there is no direct access to the file system of the operating system of basic server. Thus the server admin nor any other user will be able to physically transfer files to and from the server. Therefore a file transport service via internet connection is necessary. This is provided by a file transfer server installation and a local client using the file-transfer-protocol (ftp). XAMPP provides the Pro FTP Daemon (ProFTPD) for this case as one of the mostly used ftp-servers in internet. As file transfer is one of the most important functions to configure, maintain and manage a server system this functionality is one of the first being used with or right after setup the Linux server (cf. Install Manual)

> milliways recommends to make use of the XAMPP package of Apache friends as this package provides the four main base features (Apache Web-Server, Maria DB, PHP Parser and ProFTPD by easy installation procedures.

4.3 System Setup to provide Web-Services

To setup of the whole system he following steps will be normally be necessary when using a remote (virtual) server in a datacenter.

1) setup a Server System with an Operating System (e.g. debian Linux)
2) access this Server System over internet (e.g. by SSH client and Terminal Emulator like PuTTY)
3) download and install the XAMPP package and supporting tools (e.g. Midnight Commander)
4) configure the Operating System, the ProFTPD-Server and the Apache Web-Server of the package (cf. milliways Installation Manual)

When all this steps have been performed correctly (cf. milliways installation manual) uploading HTML- or PHP-files into the server directory via ftp-services will make them available to the Web.

The configuration of ProFTPD-Server, the suitable configuration of the User Accounts on the base Operating System and suitable code configuration and programming in milliways enables users to up- and download files in parallel to the up- and download features provided by the GUI of milliways. This direct file access by ftp-client to the milliways file storage area provides a powerful and convenient backup possibility.

milliways supplies ftp-Services parallel to the milliways Web-GUI by the setup of the milliways system as whole.

5. The Graphical User Interface (GUI)

The Graphical User Interface (GUI) is a web-based working environment for the user of milliways. This ensures an accessibility of all data handled and stored in the milliways system from almost any device that is connected to the Internet. milliways is using a web-based GUI working with every Standard-Web-Browser and on any device that has Web-Browser installed.

milliways is completely servers-sided - all code runs on the server when any request is initiated by the user to the system.This way the client side, the desktop computer of the client or the User Interface of the mobile device accessing the system by a browser request, needs no further resources or installed application programs.

milliways works with any Web-browser

All Systems accessible via an Web-GUI usually start with an index.html or index.php as first program code file.

milliways starts with an index.html that defines frames and starts the login process- After successful login the actual milliways system offers the available applications.

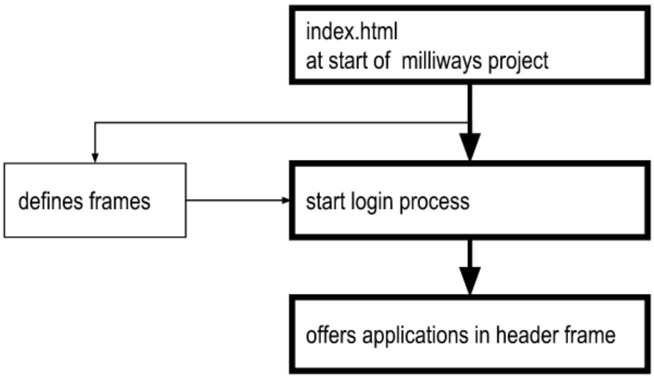

index.html
at start of milliways project
defines frames
start login process
offers applications in header frame

Figure: The main milliways process steps

5.1 The main elements of the GUI

By running the index.html the GUI of milliways separates the screen into four main elements called frames:

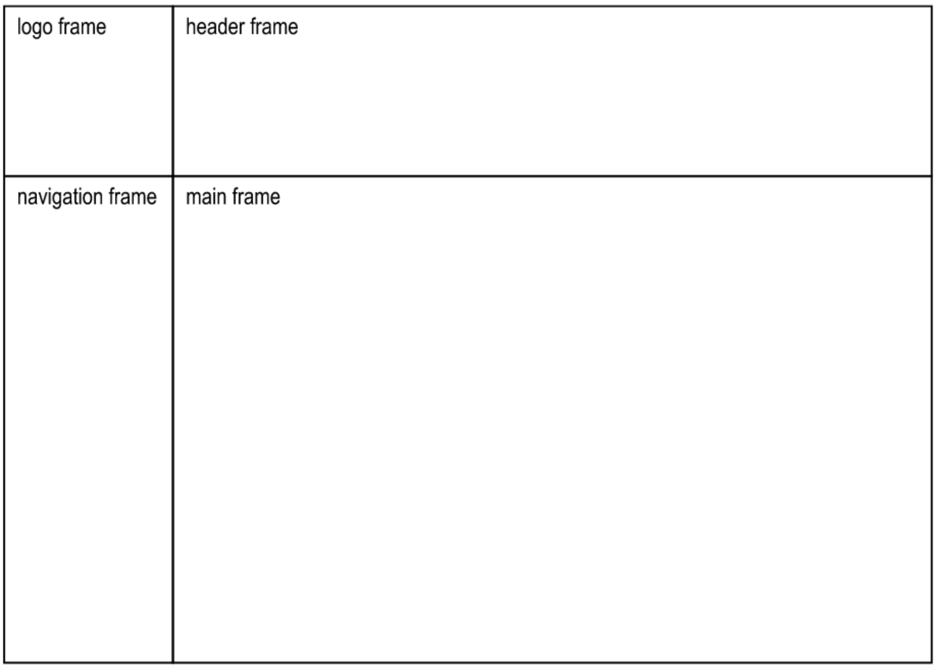


Figure: the milliways framed GUI

The placement of these elements follow the logical work trace coming from top to down and from left to right.

1) The Header Frame (top right)

The “entrance” area of milliways. Here the user chooses his working application or program from the milliways collection of applications. He is then guided to

2) The Navigation Frame (down left)

The “selection area” determines what information will be made accessible in the

3) The Information Content or Main Frame (down right)

All Information made available here can be managed (created, read, updated or deleted).

4) The Logo Frame (top left)

The Logo Frame is not necessary for the functionality of the milliway. It is more an “add-on” to place a logo and to show corporate identity. A logo placed here may represent a link to another Webpage (e.g. the public Webpages of the Company)

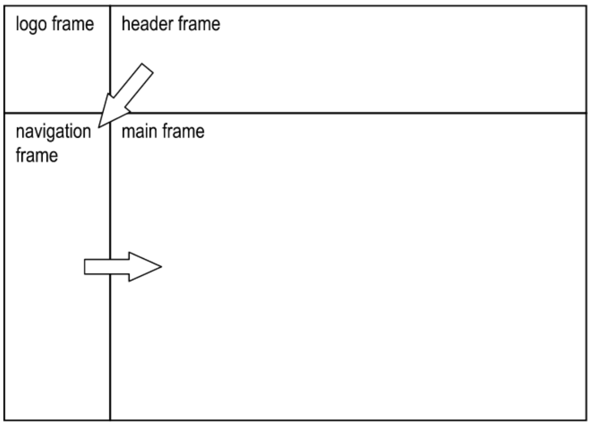


Figure: Frame Sequence

Frames are currently classified in HTML5 as “non-conforming feature” as some of their known disadvantages (e.g. non-compatible to search engine behaviour). But for the purpose of milliways they have advantages that are not covered by the successors like iframe. As of this there is no current need to leave the frame concept.

The Frame concept of milliways structure the Web-GUI in

- the top header has always the same visible content. It is the “anchor” to the user.
- the navigation sidebar changes its content according to the currently used application. It is the “filter” to the information required
- the main frame can extend endlessly and can show any information type like lists or 2/3/4-D-graphics. It is “viewer” and “editor” simultaneously.
- the logo frame is an add-on that may be used for corporate ID.

index.html

The size of logo, header and navigation frame is restricted by a pixel value. It is set to a minimum in the very first file being addressed when calling milliways by its Installation Web-address (URL of installed milliways system) - the index.html

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">

<html>
<!--
when this code is executed $_SERVER['PHP_AUTH_USER'] is
already given  by weblogin and can be used in the following
```

```
php routines
$_SERVER['PHP_AUTH_USER']determines $Systemproject by
SWITCH routine in include_setSystemvariables.php
logo.php displays logo according to $SystemProject
logins_query.php starts personal login and register
routines
after successful login normal header.php is presented
navigation.php is empty page with time stamp
blankpage.html is empty page
-->

<frameset rows="180,*" frameborder="0" framespacing="0"
border="0">
    <frameset cols="280,*" frameborder="0" framespacing="0"
border="0">
       <frame src="logo.php" name="logo">
       <frame src="logins_query.php" name="header">
    </frameset>
    <frameset cols="280,*" frameborder="0" framespacing="0"
border="0">
       <frame src="navigation.php" name="navigation">
       <frame src="blankpage.html" name="main">
    </frameset>
</frameset>
</html>
<!-- changed 25.01.17 -->
```

180 pixels for navigation frame width,

280 pixels for header frame height

code listing of index.html - calling logo.php, navigation.php, blankpage.html and logins_query.php

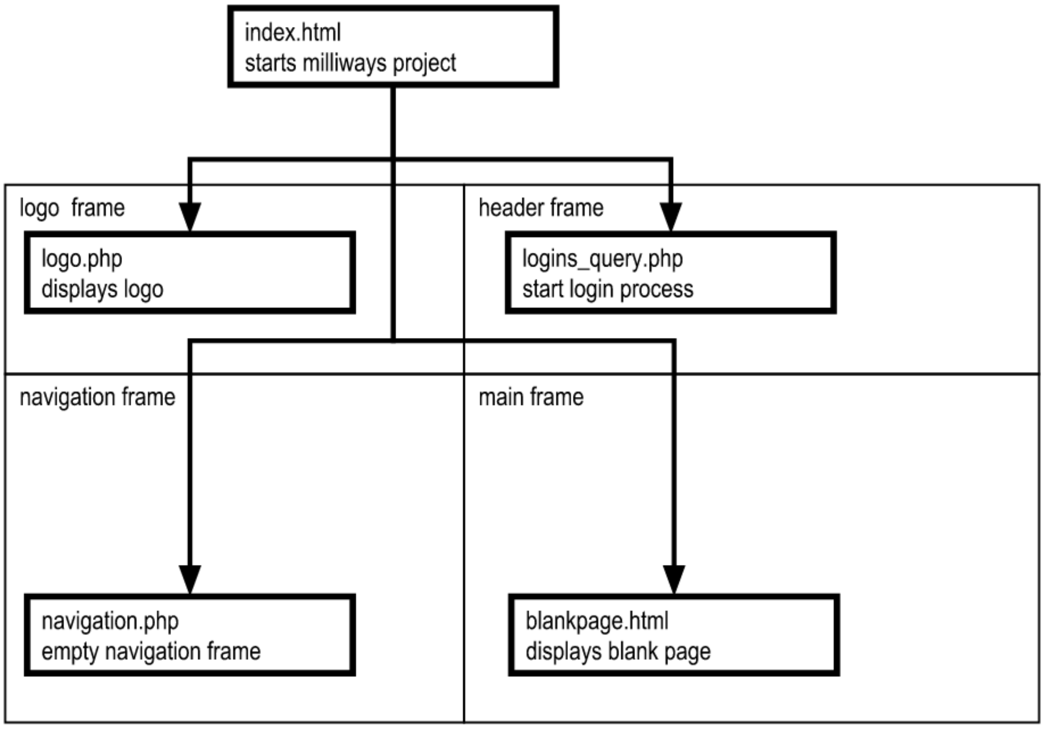


Figure: index.html calls the four program code files and sets the frame dimensions

The logo.php is a stand-alone program code file. It is a “dead-end” in terms of milliways functionality. The logo displayed by logo.php in the logo frame may supported with a link to other websites - thus leaving the milliways system.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
<html>
<head>
<?php
include 'include_setSystemvariables.php';
include 'include_setSystemconstants.php';
?>
<meta content="text/html; charset=ISO-8859-1"
http-equiv="content-type">
<title><?php echo $SystemProject; ?> database
system</title>
</head>
<body>
<div align="center"><a><img src="<?php echo
$SystemLogo; ?>" alt="no_logo"></a></div>
</body>
</html>
<!-- vk 01.02.18 10:00 -->
```

so that the variables $SystemProject and $SystemLogo are known

The navigation.php program code ensures to have an clean navigation frame from the beginning.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">
<html>
<head>
<?php
include 'include_setSystemvariables.php';
include 'include_setSystemconstants.php';
?>
<meta content="text/html; charset=ISO-8859-1"
http-equiv="content-type">

<title><?php echo $SystemProject; ?> database
system</title>

<style>
*
{font-family: Arial, Verdana, sans-serif; font-size:
large; background-color:<?php echo $SystemColor; ?>; }
input,select,option,textarea
{font-size:8pt; background-color:#FFFFFF;}
</style>

<link rel="stylesheet" type="text/css"
href="formats.css">
</head>

<body>
<div style="font-size: 8px;">vk 14.01.15 05:10</div>
</body>
</html>
```

so that the navigation frame has the chossen Project color from the very beginning

The blankpage.html program code provides a white blank page in main frame from the beginning.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Frameset//EN"
   "http://www.w3.org/TR/html4/frameset.dtd">
<html>
<head>
<title>not defined yet</title>
</head>
</html>
```

The logins_query.php is the first real program code starting the milliways system. The login_query starts the login process. For program code details refer to special process section below.

general structure of program code files

Every program code file has the same sequential structure:

- header according to current standards (e.g. < !DOCTYPE ….)
- html body
- php content integrated into the html body (exceptionally not in this case of index.html)
- comment line with timestamp of last change and initials of last changer

Comments of any type (HTML or PHP style) are located at any place in the code according to the need of further explaining the code content and/or structure.

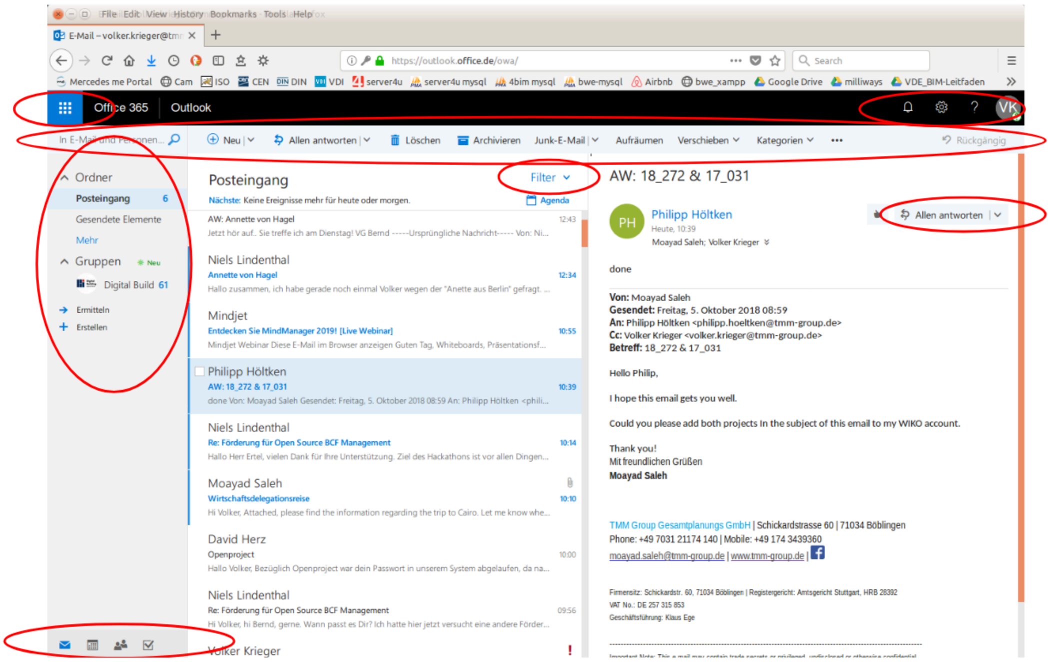
Comments in HTML code areas start with <!- - and end with - ->.

Comments in PHP code area start with /* and end with */. Line end comments start with //.

structure of frames

The values of the height of the header frame and of the width of the navigation frame are minimized and adjusted so that the attribute names in navigation and the amount of application names fit into the frame.
milliways is set up on the base of a minimalistic principle. Modern programming languages allow for many interaction features with the user via the GUI. milliways tries to reduces that to a minimum. This is ensured when

- the GUI command structure is a very flat hierarchy (possibly only one branching)
- the GUI commands are placed and grouped at a very restricted (frame) area in the GUI (not to confuse the user with multiple menu bars at all kind of GUI locations)



Nächste: Keine Ereignisse mehr für heute oder morgen.

AW: Annette von Hagel 12:43
Jetzt hör auf.. Sie treffe ich am Dienstag! VG Bernd -----Ursprüngliche Nachricht----- Von: Ni...

Niels Lindenthal
Annette von Hagel 12:34
Hallo zusammen, ich habe gerade noch einmal Volker wegen der "Anette aus Berlin" gefragt. ...

Mindjet
Entdecken Sie MindManager 2019! [Live Webinar] 10:55
Mindjet Webinar Diese E-Mail im Browser anzeigen Guten Tag, Whiteboards, Präsentationsf...

Philipp Höltken
AW: 18_272 & 17_031 10:39
done Von: Moayad Saleh Gesendet: Freitag, 5. Oktober 2018 08:59 An: Philipp Höltken <phili...

Niels Lindenthal
Re: Förderung für Open Source BCF Management 10:14
Hallo Herr Ertel, vielen Dank für Ihre Unterstützung. Ziel des Hackathons ist vor allen Dingen...

Moayad Saleh
Wirtschaftsdelegationsreise 10:10
Hi Volker, Attached, please find the information regarding the trip to Cairo. Let me know whe...

David Herz
Openproject 10:00
Hallo Volker, Bezüglich Openproject war dein Passwort in unserem System abgelaufen, da na...

Niels Lindenthal
Re: Förderung für Open Source BCF Management 09:56
Hi Volker, hi Bernd, gerne. Wann passt es Dir? Ich hatte hier jetzt versucht eine andere Förder...

Volker Krieger !

AW: 18_272 & 17_031

PH Philipp Höltken
Heute, 10:39
Moayad Saleh; Volker Krieger

done

Von: Moayad Saleh
Gesendet: Freitag, 5. Oktober 2018 08:59
An: Philipp Höltken <philipp.hoeltken@tmm-group.de>
Cc: Volker Krieger <volker.krieger@tmm-group.de>
Betreff: 18_272 & 17_031

Hello Philip,

I hope this email gets you well.

Could you please add both projects In the subject of this email to my WIKO account.

Thank you!
Mit freundlichen Grüßen
Moayad Saleh

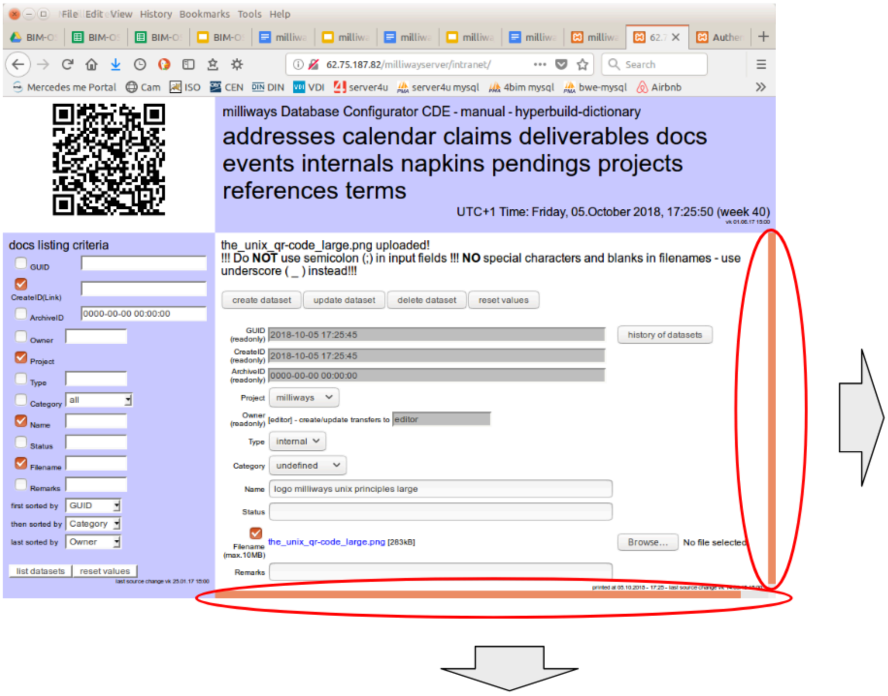
TMM Group Gesamtplanungs GmbH | Schickardstrasse 60 | 71034 Böblingen
Phone: +49 7031 21174 140 | Mobile: +49 174 3439360
moayad.saleh@tmm-group.de | www.tmm-group.de |

Firmensitz: Schickardstr. 60, 71034 Böblingen | Registergericht: Amtsgericht Stuttgart, HRB 28392
VAT No.: DE 257 315 853
Geschäftsführung: Klaus Ege

Figure: GUI as it should not be - with seven colorful menu bar locations all around the content

Therefore in milliways each frame has a dedicated task not changing its task and location throughout the program occurrence to the user.

The information content or main frame covers most of the screen space - as it contains most of the information. As width and height of Header, Logo and Navigation Frame are fixed by set pixel values (cf. Section Code) all other screen space is used to the main frame. The larger the screen the better the user can make use of it.



milliways Database Configurator CDE - manual - hyperbuild-dictionary

addresses calendar claims deliverables docs events internals napkins pendings projects references terms

UTC+1 Time: Friday, 05.October 2018, 17:25:50 (week 40)

docs listing criteria

- □ GUID
- ☑ CreateID(Link)
- □ ArchiveID 0000-00-00 00:00:00
- □ Owner
- ☑ Project
- □ Type
- □ Category all
- ☑ Name
- □ Status
- ☑ Filename
- □ Remarks

first sorted by GUID

then sorted by Category

last sorted by Owner

list datasets | reset values

last source change vk 25.01.17 15:00

the_unix_qr-code_large.png uploaded!
!!! Do **NOT** use semicolon (;) in input fields !!! **NO** special characters and blanks in filenames - use underscore (_) instead!!!

create dataset | update dataset | delete dataset | reset values

GUID (readonly): 2018-10-05 17:25:45 — history of datasets

CreateID (readonly): 2018-10-05 17:25:45

ArchiveID (readonly): 0000-00-00 00:00:00

Project: milliways

Owner (readonly): [editor] - create/update transfers to editor

Type: internal

Category: undefined

Name: logo milliways unix principles large

Status:

☑ Filename (max.10MB): the_unix_qr-code_large.png [283kB] — Browse... No file selected

Remarks:

printed at 05.10.2018 - 17:25 - last source change vk

Figure: While navigation frame width and header frame height are fixed the main frame can extend endlessly. The screen limitation is overcome by scrolling using the usual side bars to move the content in that frame.

milliways main frame contains the information and is located right down in the GUI screen - analogue to the browser principle.

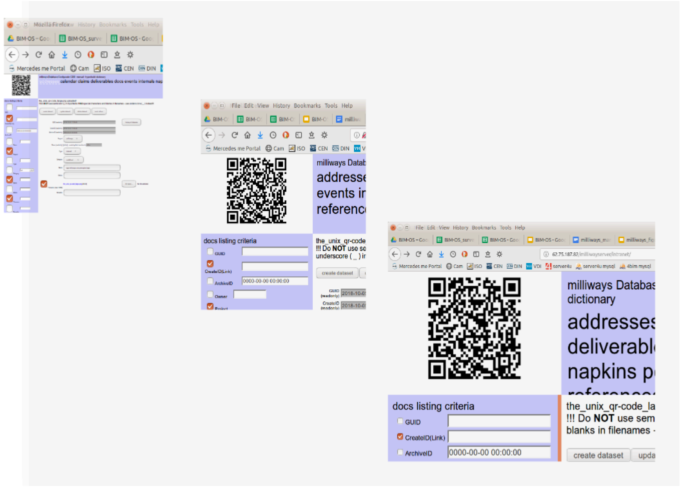


Figure: Adjusting readability of the GUI by standard browser commands STRG + and STRG -

The user can adjust screen resolution for the GUI to his convenience using

standard browser features by pressing Strg + and Strg -.

frames and workflow

milliways reserves a dedicated frame for each type of its process steps.

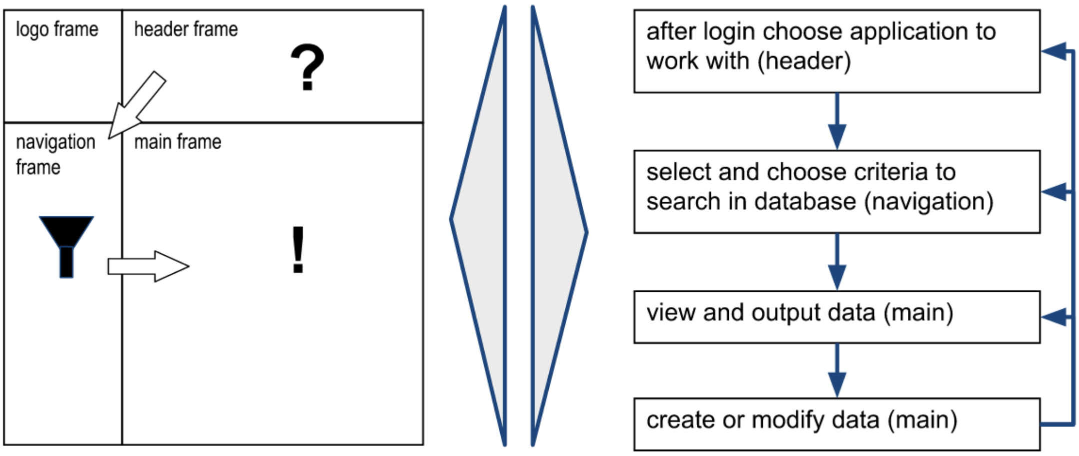

logo frame
header frame
?
navigation frame
main frame
!
after login choose application to work with (header)
select and choose criteria to search in database (navigation)
view and output data (main)
create or modify data (main)

Figure: Frames and Workflow of milliways GUI

The header frame is reserved for the login and for the “first level of choice”. As there is no other level of choosing applications the user is assured to always find its root entrance in header frame. The header frame will not vanish during the use of milliways.

The navigation frame is a filter. It can be preset and/or freely configured. It filters (restricts) the output to the main frame. The navigation frame will also not vanish during the use of milliways. But it changes appearance depending on the application being called.

The main frame has two consecutive process functions. It is the output view area (Read) and the single data creation area (Create. Update, Delete). Create, Update and Delete can only be accessed via the Read modus. But the user can always go back to any process.

The above principles will be found in the application programming files as well.

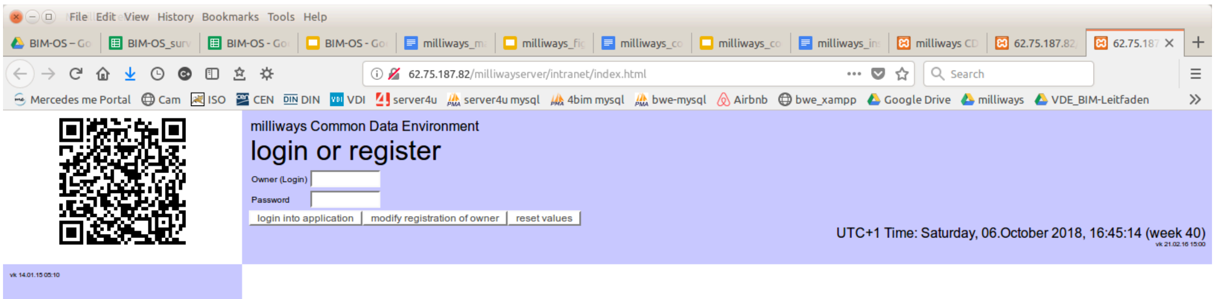
milliways is not making use of pop-up windows and drop-down submenus for program function or features. It has a strict but simple frame-process policy.

5.2 The header

The header guides the user through two consecutive processes:

1) login to the system or project
2) always make all first-level choices available to use milliways applications

The Login Process is being described in detail in a separate section (cf. Login Process). It is restricted to the header frame and partially to the main frame. The main frame is only used to create, modify or delete new user accounts to the system or project.



milliways Common Data Environment

login or register

Owner (Login)

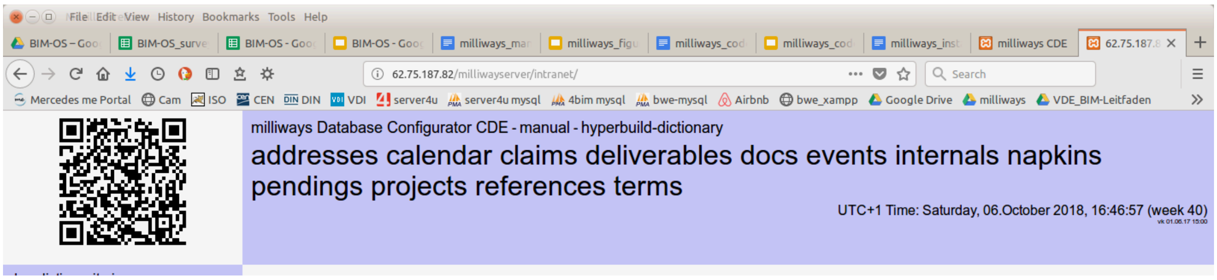
Password

UTC+1 Time: Saturday, 06.October 2018, 16:45:14 (week 40)

vk 21.02.16 15:00

vk 14.01.15 05:10

Figure: header frame at login process



milliways Database Configurator CDE - manual - hyperbuild-dictionary

addresses calendar claims deliverables docs events internals napkins pendings projects references terms

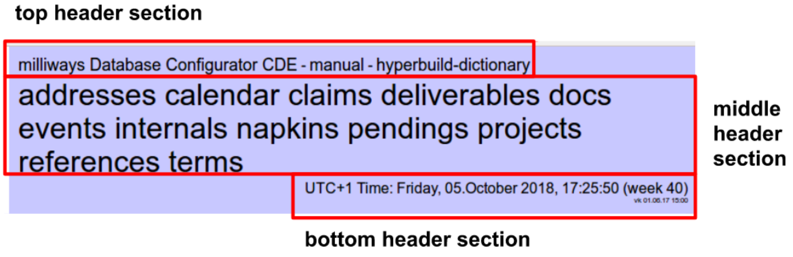
UTC+1 Time: Saturday, 06.October 2018, 16:46:57 (week 40)

vk 01.06.17 15:00

Figure: header frame after successful login process

Once the login process is successfully finished the header offers all applications available in the project. The applications available are defined in the include program code file include_SetSystemvariables.php. For more detail refer to section include code.

The operational header has three sections with different functions:



milliways Database Configurator CDE - manual - hyperbuild-dictionary

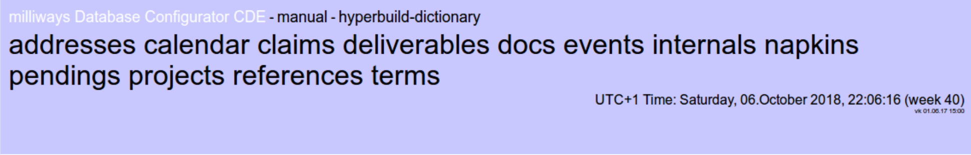
addresses calendar claims deliverables docs events internals napkins pendings projects references terms

UTC+1 Time: Friday, 05.October 2018, 17:25:50 (week 40)

vk 01.06.17 15:00

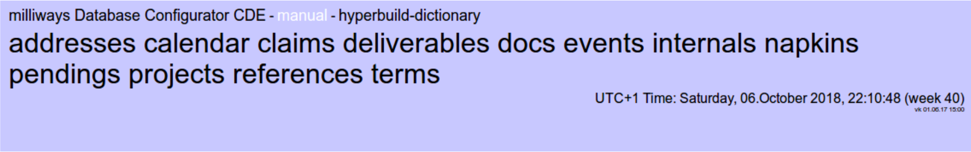
The top header section

The top header section shows links directing out the actual tab window.
The first link is below the project name and creates another instance of milliways in the same project but in a new window tab.



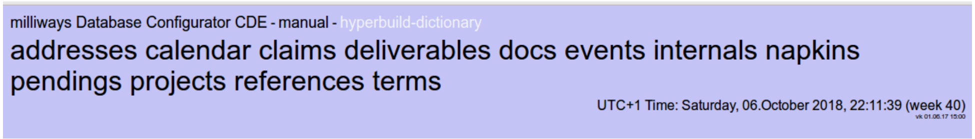
milliways Database Configurator CDE - manual - hyperbuild-dictionary
addresses calendar claims deliverables docs events internals napkins pendings projects references terms
UTC+1 Time: Saturday, 06.October 2018, 22:06:16 (week 40)
vk 01.06.17 15:00

The second link point to the online manual



milliways Database Configurator CDE - manual - hyperbuild-dictionary
addresses calendar claims deliverables docs events internals napkins pendings projects references terms
UTC+1 Time: Saturday, 06.October 2018, 22:10:48 (week 40)
vk 01.06.17 15:00

The third link directs to another Web-based system called hyperlink.hyperlink is a terminology database with similar Web-GUI.



milliways Database Configurator CDE - manual - hyperbuild-dictionary
addresses calendar claims deliverables docs events internals napkins pendings projects references terms
UTC+1 Time: Saturday, 06.October 2018, 22:11:39 (week 40)
vk 01.06.17 15:00

```
<?php
echo '<table border="0" cellpadding=0">';
echo '<tr><td class="whitelink" align="left"
valign="center" style="font-size:16px">';
echo '<a href="index.html"
target="_blank">'.$SystemProject.' CDE</a>';
echo ' - <a href="https://docs.google.com/……"
target="_blank">manual</a>';
echo ' - <a href="http://www.hyperbuild.info"
target="_blank">hyperbuild-dictionary</a>';
echo '</td>';
echo '</tr>';
...
```

clicking on the projectname $SystemProject causes a second call of index.html in another window

clicking on manual calls the manual

clicking on hyperbuild calls a dictionary

The middle header section

The middle header section displays the available applications. Each time one application is chosen the respective *applicationname*_navigation.php is called and the respective criteria to choose from are displayed in the navigation frame. This may be done at any time.

```
…
echo '<tr>';
echo '<td class="whitelink" >';
if (strstr($SystemApps,'media'))
  {echo '<a href="media_navigation.php"
target="navigation"><span style="font-size:32px">media >
</span></a>';}

if (strstr($SystemApps,'furniture'))
  {echo '<a href="furniture_navigation.php"
target="navigation"><span style="font-size:32px">furniture
+ </span></a>';}

if (strstr($SystemApps,'equipment'))
  {echo '<a href="equipment_navigation.php"
target="navigation"><span style="font-size:32px">equipment
</span></a>';}

…

if (strstr($SystemApps,'projects'))
   {echo '<a href="projects_navigation.php"
target="navigation"><span style="font-size:32px">projects
</span></a>';}

if (strstr($SystemApps,'references'))
   {echo '<a href="refs_navigation.php"
target="navigation"><span
style="font-size:32px">references </span></a>';}

if (strstr($SystemApps,'terms'))
   {echo '<a href="terms_navigation.php"
target="navigation"><span style="font-size:32px">terms
</span></a>';}

echo '</td>';
echo '</tr>';
echo '</table>';
```

The middle section of the header only offers application whose names are listed in the system variable $SystemsApps. The system variable $SystemApps is set in include_SetSystemvariables.php. Please refer for more detail to the section of the include program code.

The bottom header section

The bottom section of the header displays the current time stamp.

```
<div align="right" style="font-size:16px" id="clock"></div>
<div align="right" style="font-size:8px">vk 06.10.18
15:00</div>
```

The current program code displays UTC+1.
The program code is outside the <php? …?> code area. It is based on javascript and from other sources listed in the code. Please refer to the original code in header.php.

5.3 Other instances

As described before milliways offers the possibility to start another instance of itself in a new browser tab window. Calling another instance is done by clicking on the project name displayed in the top left corner of the header.
The additional instance will not leave the browser session - thus it will not change the project. Calling another instance may be helpful when the user wants to work in multiple applications at the same time. All CRUD-actions taken will effect the same database and therefore consistent according to ACID.

5.4 The admin role

There are multiple projects possible to run on the same milliways system. Each of them must be configured according to the guidelines in the installation manual.
Please refer in the installation manual and in the login process section of this manual for more details.
Users being logged in as admin can choose the project in the navigation (filter) process.
They may list datasets of all projects or only from a single project.
Users being logged in as admin can determine to which project the save process is directed. Special care must be taken especially deciding whether a create, update or delete process is chosen.
The admin role is characterized by an empty value of the variable $Systemproject of the session.

```
echo '<tr>';
echo '<td><input type="checkbox" name="list_DocOwner"
value="yes" >Owner</td>';
echo '<td><input type="text" name="partDocOwner" size="8"
maxlength="40" value="'.$partDocOwner.'"></td>';
echo '</tr>';

// echo 'This is control code for SystemProject
'.$SystemProject.'<br>';
// at this point all project CDEs with empty SystemProject
(e.g. admin etc.) will show Project drop down
// CDEs with non-empty System Project stay only in their
regime list additionally all public type datasets
// but they can choose the list_AppProject option for listing
// the distinction towards $SystemProject is necessary to
avoid all projects in listing
```

```
echo '<tr>';
echo '<td><input type="checkbox" name="list_DocProject"
value="yes" checked>Project</td>';
    if (empty($SystemProject))
      {
      echo '<td><select type="text" name="partDocProject"
size="1">';
      echo '<option value="" selected>all</option>';
             foreach ($SystemProjectArray as $Project) {echo
'<option>'.$Project.'</option>';}
      echo '</select>';
      }
      else
      { echo '<input type="hidden" name="partDocProject"
value="'.$SystemProject.'">';}
echo '</tr>';
```

5.5 The color, CSS and the links

There is not much formatting due to the stringent simplicity of milliways in terms of colors and web design (UNIX principle *“keep it small and simple”*). The milliways GUI is designed to carry only one color besides the white background. This color may be defined by a corporate identity. Thus it may be darker or lighter.

Only to make the Web-GUI less excitedly milliways uses CSS for links. Only “link, visited, focus, hover and active” attributes of CSS are defined in the program code file format.css.

Almost all program code files refer to format.css. The .bluelink version in format.css should be used for white backgrounds, the .whitelink for colored background. This is the only case where CSS is used in milliways.

```
@charset "ISO-8859-1";
/* CSS Document */

/* for white background */
.bluelink a:link     {color:navy;   text-decoration:none; }
.bluelink a:visited     {color:blue;   text-decoration:none; }
.bluelink a:focus     {color:blue;   text-decoration:none; }
.bluelink a:hover     {color:green; text-decoration:none; }
.bluelink a:active     {color:lime;   text-decoration:none; }

/* for colored background */
```

```
.whitelink a:link     {color:black; text-decoration:none; }
.whitelink a:visited     {color:black; text-decoration:none; }
.whitelink a:focus     {color:white; text-decoration:none; }
.whitelink a:hover     {color:white; text-decoration:none; }
.whitelink a:active     {color:black; text-decoration:none; }


/* last source change vk 31.10.2012 10:15 */
```

All other formats are defined in the header section of each program code files. They differ depending on the process step the application is representing.

5.6 The include code files

Program code that is the same and is being used by more than one Program code files should be centrally available. This is done by “include” files. “include” files are a common PHP feature.

milliways uses the following centrally available include files in almost all program code:

- include_SetSystemvariables.php
- include_SetSystemconstants.php
- include_dbconnect.php

milliways uses the following include files specifically in applications code:

- include_set*Applicationname*constants.php
- include_*applicationname*_postvariables.php

The include_SetSystemvariables.php

The include file for the Systemvariables holds all system variables for all milliways projects possible on that server.
It also defines the variable $dataftpfolder. $dataftpfolder is necessary to store all data files in the correct folder. So that they can be accessed by ftp as well.Please refer to installation manual and to the *applicationname*_save.php section below for more details.

```
<?php
```

```
/*
Systemvariables and Systemconstants apply to all application
routines in the System
Applicationconstants apply only to specific application routine
the following Systemvariables are defined
*/

/*
to differ between and set system dataftp folder
*/
$dataftpfolder = "milliwayserver";

/*
weblogin determines SystemProject
valid weblogins are listed in .htaccess file
password for valid weblogins are in dedicated passwordfiles

SystemProject determines ownership of dataset
SystemProject buildigital and public accesses all projects thus
$SystemProject is empty for all projects
SystemProject public has restricted $SystemType public
*/

switch ($_SERVER['PHP_AUTH_USER'])
   {
   case "milliways”:             // "online"
     $SystemProject="milliways"; // all projects displayed
     $SystemType="";             // all types displayed
     $SystemColor="#CCCCFF";     //
     $SystemLogo="the_unix_qr-code.png";
     $SystemApps="addresses calendar claims deliverables docs
events internals napkins pendings projects references terms";
      break;

   case "public": //weblogin public shows all projects but only
datasets in public type, login routine is bypassed
      $SystemProject="";        // all projects displayed
      $SystemType="public";   // only public types displayed
      $SystemColor="#CCCCFF"; //
      $SystemLogo="the_unix_qr-code.png";
      $SystemApps="addresses docs events internals pendings";
      break;
}

// last change vkrieger 01.10.2018

?>
```

The include_SetSystemconstants.php

The include file for the Systemconstants holds all system constants for all milliways projects possible on that server.
It also defines the name and login constants to access the database. Please refer to installation manual and to the *applicationname*_save.php section below for more details.

```
<?php

/*
Systemvariables and Systemconstants apply to all application
routines in the System
Applicationconstants apply only to specific application routine
the following Systemconstants are defined
*/

$selection     ='';

// DB constants

    $dbserver = "localhost";
    $dbname   = "wiki4bim";
    $username = "anybody";
    $password = "none";

// Arrays of SystemProject constants

// Project is owner of dataset
$SystemProjectArray = [
'milliways',
];

// Category classifies accessibility of dataset
$SystemTypeArray = [
'internal',
'project',
'public',
'other',
];

// last change vkrieger 13.07.2018

?>
```

The include_dbconnect.php

The include file for the database connection…..

```php
<?php

/*
addresssing mysql-server by placing login valuse in variables
*/

$dbserver = "localhost";
$dbname   = "wiki4bim";
$username = "anybody";
$password = "none";

$sqlconnect =
mysqli_connect($dbserver,$username,$password,$dbname) or die
("not connected to mysql server and database. login or password
wrong?"); // new for php7

// last change by vk at 2018-03-13 17:00
?>
```

The include_set*Applicationname*constants.php

The include file for setting constants typical for an application…..

```
<?php

?>
```

The include_*applicationname*_postvariables.php

The include file for the transfer of contents of variables

```
<?php

?>
```

6. The Applications

Each milliway application is based on an autonomous process to handle information and has an separate autonomous datatable in the connected database. Data or dataset transfer from one application to the other is possible with special program code.

Currently milliways supports the following applications:

applicationname shown in header	features	status
addresses	similar to vCard format	matured and tested in international environment, yet no vcf import on Web-GUI level
calendar	basic day calendar with allocation an user-, task- and geographic-level, ASCII readable	premature
claims	RACI-function- and target oriented tracking of processes	work-in-progress, incomplete
deliverables	highly configurable Big-Data Information-Container System, multi language capable, ftp-service	highly matured, can be configured to use in accordance to ISO19650 and PAS 1192
docs	basic information-container repository, single-container	matured
events	highly configurable event database, multi-project capable	matured, needs low-level version
internals	information-container repository for internal user groups, multi-container capable	matured, may be combinable with napkins and/or docs
napkins	information-container repository for multi-container	matured, may be combinable with internals and/or docs
pendings	RACI-function- and target	work-in-progress, may be

	oriented tracking of processes	combinable with claims
projects	information container repository for administration to keep tracking projects and contracts	tested
references	information container repository for administration and marketing to supply reference data in various formats, pdf output according to design given requirements	tested
terms	local instance of hyperbuild database, multi-language capable as dictionary, terminology and translation database	matured - can be replaced by direct use of hyperbuild

All application have in common that any listing of their data can be downloaded as file in CSV or MS-XLS Format.

Although the different applications support different kind of user processes and handle therefore different type of datasets the underlying programming code files follow the same structure.
Each application is supported by four process steps and the following corresponding four programming code files:

1) *applicationname*_navigation.php
managing the listing criteria, recording the criteria input in the navigation frame und supplying access to the data listing in the mainframe

2) *applicationname*_list.php
displaying the content of the database according to what was chosen in *applicationname*_ navigation.php

3) *applicationname*_modify.php
displaying and modifying the content of single datasets chosen in *applicationname*_list.php

4) *applicationname*_save.php
saving the changes being made in *applicationname*_modify.php to the database

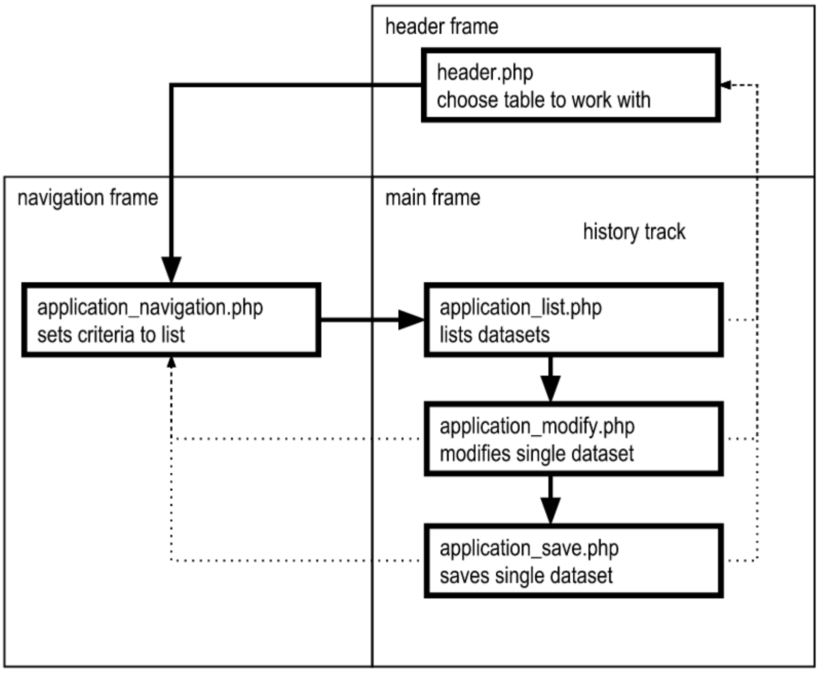


Figure: Frames, Program Code Files and main route through milliways

The user may return to any of the process steps or may repeat any of the process steps as often as wanted or needed. Changes will only take effect when the *applicationname*_save-php is executed. The *applicationname*_save.php is executed every time the user completes a data modification. For more details refer to the specific Application section below.

The general structure and code of each of these program files is described below.

6.1 The general structure of the *applicationname*_navigation.php

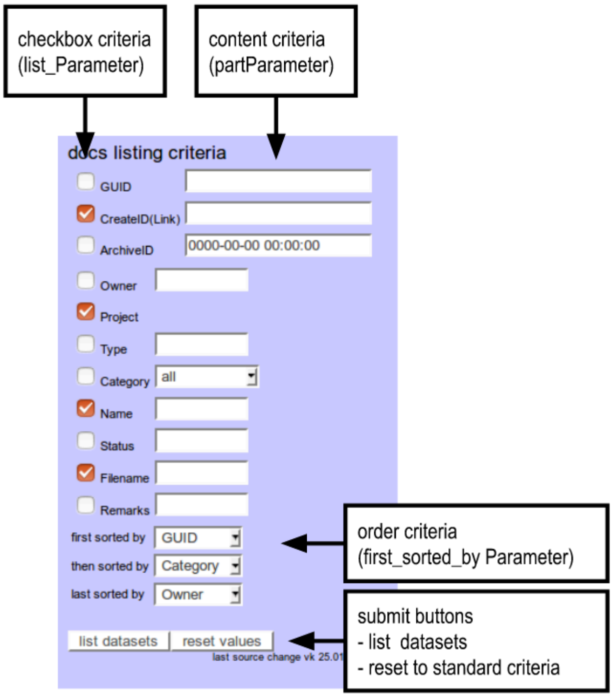
The typical *applicationname*_navigation.php sets the criteria for the listing and modification of data. Below docs_navigation.php is used for representative code.

Three criteria categories are available:

- checkbox to choose the attributes (columns) of a dataset to be listed
- text entry fields to place search strings for attribute values in datasets
- order criteria as dropdown lists of possible attributes

The bottom part of the navigation frame is made up by submit buttons causing different actions:

- resetting checkboxes and text entry field contents to predefined values. These predefined values are given in the program code.
- calling the *applicationname*_list.php to list all datasets in the main frame according the chosen criteria



checkbox criteria
(list_Parameter)

content criteria
(partParameter)

docs listing criteria

□ GUID
☑ CreateID(Link)
□ ArchiveID 0000-00-00 00:00:00
□ Owner
☑ Project
□ Type
□ Category all
☑ Name
□ Status
☑ Filename
□ Remarks

first sorted by GUID
then sorted by Category
last sorted by Owner

order criteria
(first_sorted_by Parameter)

last source change vk 25.01

submit buttons
- list datasets
- reset to standard criteria

The typical *applicationname*_navigation.php consists therefore of the following code blocks

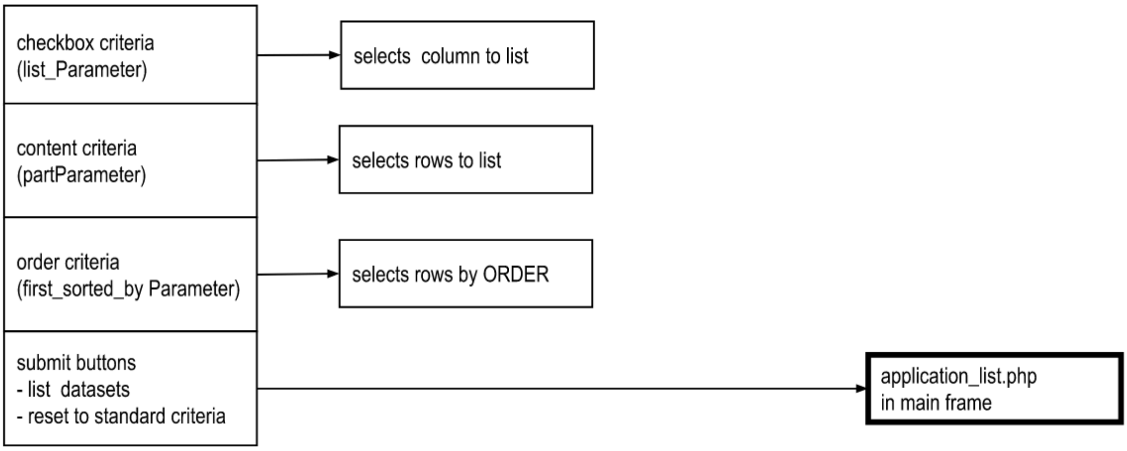


Figure: Code blocks in typical navigation.php

The *applicationname*_navigation.php uses the “form” directive calling the *applicationname*_list.php when the submit buttons are pressed.

```
<form method="post" enctype="multipart/form-data" action="docs_list.php"
target="main">
...
</form>
```

There is a change from PHP to HTML in the current version of *applicationname*_navigation.php. The checkbox and text entry fields lines are coded in PHP. The order criteria is still in HTML.

The content chosen in *applicationname*_navigation.php ist transported into the other milliways application by using two sets of variables:

- $list_*ApplicationnameAttributename*
  These variables of type checkbox determine wether an attribute will be listed
- $part*ApplicationnameAttributename*
  These variables of type string contain search patterns to be used when datasets are selected

These variables must be made available to the other milliways application by calling special include files recovering the content of these variables by the $_POST[..] command.

```
if (!empty($_POST['list_DocName'])) {$list_DocName =$_POST['list_DocName'];}
else {$list_DocName ='';}
```

See include code files section above for more details.

There are three variables determining the order of listing:

- $first_sorted_by

- $then_sorted_by
- $last_sorted_by

All of them are transported into the list_*applicationname*.php by the include_*applicationname*_postvariables*.php*

The full code listing of *applicationname*_navigation.php is listed below.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
<html>

<head>
      <?php include 'include_setSystemvariables.php'; include
'include_setSystemconstants.php'; ?>
      <meta content="text/html; charset=ISO-8859-1"
http-equiv="content-type">
      <title><?php echo $SystemProject; ?> database
system</title>
    <style>
    *
{font-size:16px ; font-family: Arial, Verdana, sans-serif;
background-color:<?php echo $SystemColor; ?>;}
    input                          {font-size:12px ;
font-family: Arial, Verdana, sans-serif;
background-color:#FFFFFF; }
    select,option,textarea                {font-size:12px ;
font-family: Arial, Verdana, sans-serif;
background-color:#FFFFFF; }
    table,tr,td
{font-size:10px ; font-family: Arial, Verdana, sans-serif; }
    </style>
    <link rel="stylesheet" type="text/css"
href="formats.css">
</head>

<body>

docs listing criteria<br>

<?php

include 'include_setDocconstants.php';
include 'include_docs_postvariables.php';

if (!isset($_SESSION)) { session_start();}

$listdatasets = "";

if ($listdatasets == "" AND $_SESSION['LoginType'])
    {
echo '<form method="post" enctype="multipart/form-data"
action="docs_list.php" target="main">';
```

typical header

calling of include files

start of form

```
$partDocArchiveID ='0000-00-00 00:00:00'; // to firstly
display only non-archived datasets when navigation is called

    echo '<table>';
    echo '<tr>';
    echo '<td><input type="checkbox" name="list_DocGUID"
value="yes" >GUID</td>';
```

start of checkbox and text entry field input

```
    echo '<td><input type="text" name="partDocGUID" size="20"
maxlength="40" value="'.$partDocGUID.'"></td>';
    echo '</tr>';
    echo '<tr>';
    echo '<td><input type="checkbox" name="list_DocCreateID"
value="yes" checked>CreateID(Link)</td>';
    echo '<td><input type="text" name="partDocCreateID"
size="20" maxlength="40" value="'.$partDocCreateID.'"></td>';
    echo '</tr>';
    echo '<tr>';
    echo '<td><input type="checkbox" name="list_DocArchiveID"
value="yes" >ArchiveID</td>';
    echo '<td><input type="text" name="partDocArchiveID"
size="20" maxlength="40"
value="'.$partDocArchiveID.'"></td>';
    echo '</tr>';
    echo '</table>';

    echo '<table>';
    echo '<tr>';
    echo '<td><input type="checkbox" name="list_DocOwner"
value="yes" >Owner</td>';
    echo '<td><input type="text" name="partDocOwner" size="8"
maxlength="40" value="'.$partDocOwner.'"></td>';
    echo '</tr>';

// echo 'This is control code for SystemProject
'.$SystemProject.'<br>';
// at this point all project CDEs with empty SystemProject
(e.g. admin etc.) will show Project drop down
```

sometimes old control code is transferred into a comment

```
// CDEs with non-empty System Project stay only in their
regime list additionally all public type datasets
// but they can choose the list_AppProject option for listing
// the distinction towards SystemProject is necessary to
avoid all projects in listing

echo '<tr>';
echo '<td><input type="checkbox" name="list_DocProject"
value="yes" checked>Project</td>';
    if (empty($SystemProject))
       {
       echo '<td><select type="text" name="partDocProject"
size="1">';
             echo '<option value="" selected>all</option>';
             foreach ($SystemProjectArray as $Project) {echo
'<option>'.$Project.'</option>';}
             echo '</select>';}
       else { echo '<input type="hidden"
name="partDocProject" value="'.$SystemProject.'">';}
    echo '</tr>';

// echo 'This is control code for partDocProject
'.$partDocProject.'<br>';

echo '<tr>';
echo '<td><input type="checkbox" name="list_DocType"
value="yes" >Type</td>';
    if ($SystemType == "public")
       {echo '<td><input style="background-color:#C0C0C0"
type="text" name="partDocType" size="8" maxlength="40"
value="public" readonly></td>';}
       else
       {echo '<td><input type="text" name="partDocType"
size="8" maxlength="40" value="'.$partDocType.'"></td>';}
echo '</tr>';

echo '<tr>';
echo '<td><input type="checkbox" name="list_DocCategory"
value="yes" >Category</td>';
    echo '<td><select type="text" name="partDocCategory"
size="1">';
             echo '<option value="" selected>all</option>';
             foreach ($DocCategoryArray as $Category) {echo
'<option>'.$Category.'</option>';}
             echo '</select>';
echo '</td>';
echo '</tr>';

echo '<tr><td><input type="checkbox" name="list_DocName"
value="yes" checked>Name</td><td><input type="text"
name="partDocName" size="8" maxlength="40"
value="'.$partDocName.'"></td></tr>';

echo '<tr><td><input type="checkbox" name="list_DocStatus"
value="yes" >Status</td><td><input type="text"
name="partDocStatus" size="8" maxlength="40"
value="'.$partDocStatus.'"></td></tr>';

echo '<tr><td><input type="checkbox" name="list_DocFilename"
value="yes" checked>Filename</td><td><input type="text"
name="partDocFilename" size="8" maxlength="40"
value="'.$partDocFilename.'"></td></tr>';

echo '<tr><td><input type="checkbox" name="list_DocRemarks"
value="yes" >Remarks</td><td><input type="text"
name="partDocRemarks" size="8" maxlength="40"
value="'.$partDocRemarks.'"></td></tr>';
```

```
echo '</table>';

?>


<!-- sorting part -->
<table>
<tr><td>first sorted by</td>
<td><select name="first_sorted_by" size="1">
      <option value="DocGUID" selected>GUID</option>
      <option value="DocCreateID">CreateID</option>
      <option value="DocArchiveID">ArchiveID</option>
      <option value="DocProject">Project</option>
      <option value="DocOwner" >Owner</option>
      <option value="DocType" >Type</option>
      <option value="DocCategory">Category</option>
      <option value="DocName" >Name</option>
      <option value="DocStatus" >Status</option>
      <option value="DocFilename" >Filename</option>
      <option value="DocRemarks" >Remarks</option>
      </select>
      </td>
      </tr>

<tr><td>then sorted by</td>
<td><select name="then_sorted_by" size="1">
   <option value="DocGUID">GUID</option>
   <option value="DocCreateID">CreateID</option>
   <option value="DocArchiveID">ArchiveID</option>
   <option value="DocProject" >Project</option>
   <option value="DocOwner" >Owner</option>
   <option value="DocType" >Type</option>
   <option value="DocCategory" selected>Category</option>
   <option value="DocName" >Name</option>
   <option value="DocStatus" >Status</option>
   <option value="DocFilename" >Filename</option>
   <option value="DocRemarks" >Remarks</option>
   </select>
   </td>
   </tr>

<tr><td>last sorted by</td>
<td><select name="last_sorted_by" size="1">
   <option value="DocGUID">GUID</option>
   <option value="DocCreateID">CreateID</option>
   <option value="DocArchiveID">ArchiveID</option>
   <option value="DocProject" >Project</option>
   <option value="DocOwner" selected>Owner</option>
   <option value="DocType" >Type</option>
   <option value="DocCategory" >Category</option>
   <option value="DocName" >Name</option>
   <option value="DocStatus" >Status</option>
   <option value="DocFilename" >Filename</option>
```

end of checkbox and text entry field part

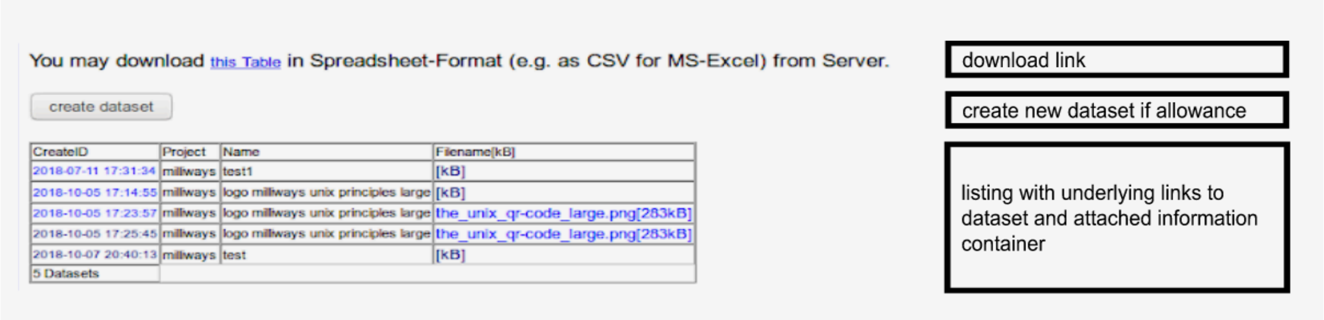
code change from PHP to HTML

start of listing order criteria part

```
   <option value="DocRemarks" >Remarks</option>
   </select>
   </td>
   </tr>

</table>

<br>

<table>
<input type="submit" name="listdatasets" value="list
datasets">
<input type="reset" value="reset values">
</table>                                                         end of
                                                                 listing
</form>                                                          order
                                                                 criteria
<?php
} elseif ( $listdatasets );
                                                                 submit
echo '<div align="right" style="font-size: 8px;">last source     buttons
change vk 25.01.17 18:00</div>';

?>
</body>
</html>                                                          end of form
```

6.2 The general structure of the *applicationname*_list.php

The typical *applicationname*_list.php lists datasets according to the criteria set by *applicationname*_navigation.php . Below docs_list.php is used for representative code.


You may download this Table in Spreadsheet-Format (e.g. as CSV for MS-Excel) from Server.

create dataset

CreateID	Project	Name	Filename[kB]
2018-07-11 17:31:34	milliways	test1	[kB]
2018-10-05 17:14:55	milliways	logo milliways unix principles large	[kB]
2018-10-05 17:23:57	milliways	logo milliways unix principles large	the_unix_qr-code_large.png[283kB]
2018-10-05 17:25:45	milliways	logo milliways unix principles large	the_unix_qr-code_large.png[283kB]
2018-10-07 20:40:13	milliways	test	[kB]
5 Datasets			

download link
create new dataset if allowance
listing with underlying links to dataset and attached information container

The listing of *applicationname*_list.php provides two extra features:

- a CSV download
- creation of a new single dataset (e.g. if missed in the listing)

The listing of *applicationname*_list.php also provides the link to every dataset for CRUD operation. This is only available if the checkbox of the “CreateID” attribute is checked.

The typical *applicationname*_list.php consists of the following code blocks

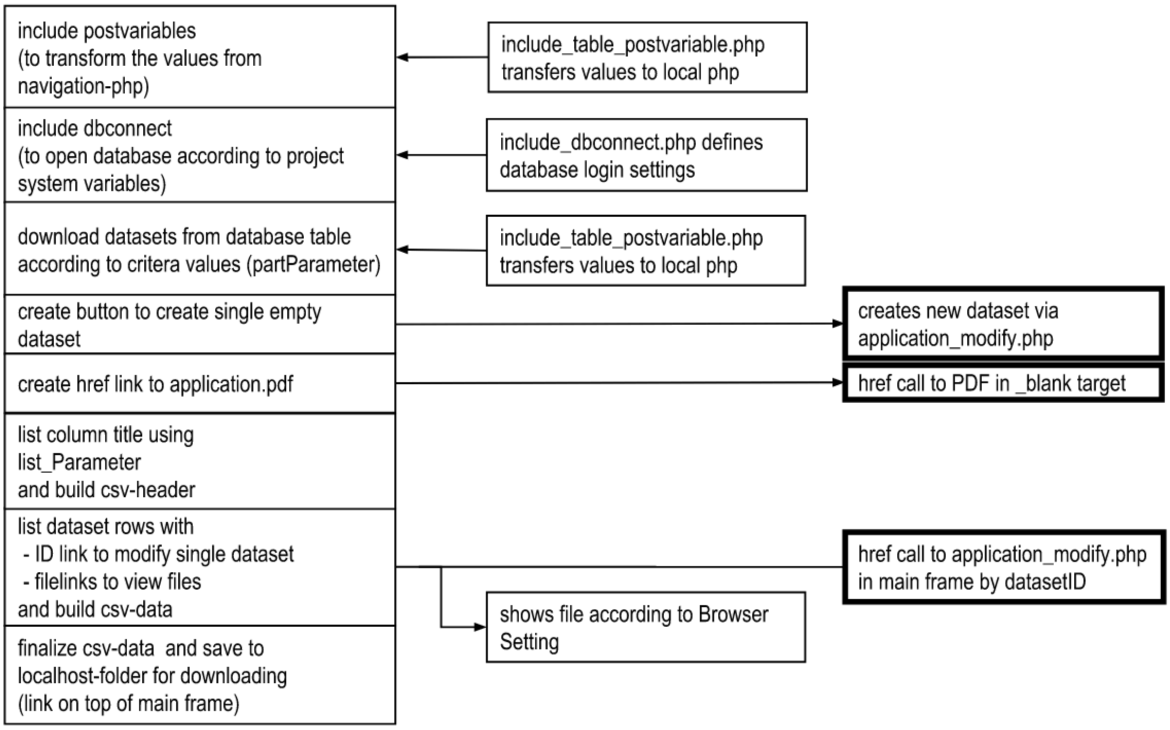

include postvariables (to transform the values from navigation-php)
include dbconnect (to open database according to project system variables)
download datasets from database table according to critera values (partParameter)
create button to create single empty dataset
create href link to application.pdf
list column title using list_Parameter and build csv-header
list dataset rows with - ID link to modify single dataset - filelinks to view files and build csv-data
finalize csv-data and save to localhost-folder for downloading (link on top of main frame)
include_table_postvariable.php transfers values to local php
include_dbconnect.php defines database login settings
include_table_postvariable.php transfers values to local php
creates new dataset via application_modify.php
href call to PDF in _blank target
href call to application_modify.php in main frame by datasetID
shows file according to Browser Setting

Figure: Code blocks in typical list.php

The download feature of files in CSV-format is achieved by simultaneously adding data to a pre-existing file *applicationname*.xls when the output to the main window is generated by consecutive echo commands. The link is presented on top of the listing.

The creation of a new dataset is achieved by calling *applicationname*_save.php. *applicationname*_save.php has a routine code section to *applicationname*_modify.php thus allowing to enter the values of the new dataset.

Please find the full listing of a typical *applicationname*_list.php below

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
<?php include 'include_setSystemvariables.php'; include
'include_setSystemconstants.php'; ?>
<meta content="text/html; charset=ISO-8859-1"
http-equiv="content-type">
      <title><?php echo $SystemProject; ?> database
system</title>

<style>
    *
{font-size:16px ; font-family: Arial, Verdana, sans-serif; }
    input,a                          {font-size:12px ;
font-family: Arial, Verdana, sans-serif; }
    select,option,textarea                  {font-size:12px ;
font-family: Arial, Verdana, sans-serif; }
    table,tr,td
{font-size:10px ; font-family: Arial, Verdana, sans-serif; }
</style>

<link rel="stylesheet" type="text/css" href="formats.css">
</head>

<body>

You may download <a href="docslist.xls">this Table</a> in
Spreadsheet-Format (e.g. as CSV for MS-Excel) from Server.
<br><br>                                                        link code
                                                                for
<?php                                                           download
                                                                listing
include 'include_setDocconstants.php';
include 'include_docs_postvariables.php';
$link = mysqli_connect($dbserver,$username,$password,$dbname);  usual
                                                                includes
if (!isset($_SESSION)) { session_start();}
```

```
// <form></form> must enclose all <input> to transfer to next
app
echo '<form method="post" enctype="multipart/form-data"
action="docs_save.php">';

if ($_SESSION['LoginType']=='admin' OR
$_SESSION['LoginType']=='supereditor' OR
$_SESSION['LoginType']=='editor')
    {
    echo '<input type="submit" name="createdataset"
value="create dataset">';
    }

echo '<br><br>';
```

only if user is of type admin or editor new dataset can be created

```
/*
select datasets according to criteria
add datasets that are from all other Projects but of Type
public
*/


      $dbquery = "SELECT * FROM docs WHERE

LOCATE('$partDocGUID', DocGUID)>0 AND
LOCATE('$partDocCreateID', DocCreateID)>0 AND
LOCATE('$partDocArchiveID', DocArchiveID)>0 AND
(LOCATE('$partDocProject', DocProject)>0 OR LOCATE('public',
DocType)>0) AND
LOCATE('$partDocOwner', DocOwner)>0 AND
LOCATE('$partDocType', DocType)>0  AND
LOCATE('$partDocCategory', DocCategory)>0 AND
LOCATE('$partDocName', DocName)>0 AND
LOCATE('$partDocStatus', DocStatus)>0 AND
LOCATE('$partDocFilename', DocFilename)>0 AND
LOCATE('$partDocRemarks', DocRemarks)>0

ORDER BY $first_sorted_by, $then_sorted_by, $last_sorted_by "
;
```

select datasets according to navigation criteria

```
      $dbresult = mysqli_query($link,$dbquery);

      $header =""; //empty header for excel file (dataset
structure)
      $data    =""; //empty $data variable for excel file (all
rows in one variable)

echo '<table border="1" cellspacing="0">';
```

place order of datasets according to navigation criteria

```
echo '<tr>';
```

start listing

```
if ($list_DocGUID =='yes'){echo '<td>GUID</td>';
$header.="GUID". "\t";}
```

start with

```
if ($list_DocCreateID =='yes'){echo '<td>CreateID</td>';
$header.="CreateID". "\t";}
if ($list_DocArchiveID =='yes'){echo '<td>ArchiveID</td>';
$header.="ArchiveID". "\t";}
if ($list_DocProject =='yes'){echo '<td>Project</td>';
$header.="Project". "\t";}
if ($list_DocOwner =='yes'){echo '<td>Owner</td>';
$header.="Owner". "\t";}
if ($list_DocType =='yes'){echo '<td>Type</td>';
$header.="Type". "\t";}
if ($list_DocCategory =='yes'){echo '<td>Category</td>';
$header.="Category". "\t";}
if ($list_DocName =='yes'){echo '<td>Name</td>';
$header.="Name". "\t";}
if ($list_DocStatus =='yes'){echo '<td>Status</td>';
$header.="Status". "\t";}
if ($list_DocFilename =='yes'){echo '<td>Filename[kB]</td>';
$header.="Filename[kB]". "\t";}
if ($list_DocRemarks =='yes'){echo '<td>Remarks</td>';
$header.="Remarks". "\t";}

    echo '</tr>';
    $data .= "\n"; //end of dataset in excel file (marks row
end in data)



/*
outputs datasets and fills $data variable
*/

$DatasetCount = 0;

// control code echo $SystemProject.'<br>';

      while($dbrow =
mysqli_fetch_array($dbresult,MYSQLI_BOTH))
       {
    $DatasetCount += 1;

    // output to main window and selected data to $data excel
variable
       echo '<tr>';
if ($list_DocGUID
=='yes'){echo'<td>'.$dbrow['DocGUID'].'</td>';
$data.='"'.$dbrow['DocGUID'].'"'."\t";}
if ($list_DocCreateID=='yes')
    {
    echo '<td class="bluelink"><a style="font-size:xx-small"
href="docs_modify.php?DocCreateID='.$dbrow['DocCreateID'].'">'
.$dbrow['DocCreateID'].'</a></td>';
    $data .= '"' . $dbrow['DocCreateID'] . '"' . "\t";
    }
if ($list_DocArchiveID
```

column names

list datasets row by row

create link to each dataset

```
=='yes'){echo'<td>'.$dbrow['DocArchiveID'].'</td>';
$data.='"'.$dbrow['DocArchiveID'].'"'."\t";}
if ($list_DocProject
=='yes'){echo'<td>'.$dbrow['DocProject'].'</td>';
$data.='"'.$dbrow['DocProject'].'"'."\t";}
if ($list_DocOwner
=='yes'){echo'<td>'.$dbrow['DocOwner'].'</td>';
$data.='"'.$dbrow['DocOwner'].'"'."\t";}
if ($list_DocType
=='yes'){echo'<td>'.$dbrow['DocType'].'</td>';
$data.='"'.$dbrow['DocType'].'"'."\t";}
if ($list_DocCategory
=='yes'){echo'<td>'.$dbrow['DocCategory'].'</td>';
$data.='"'.$dbrow['DocCategory'].'"'."\t";}
if ($list_DocName
=='yes'){echo'<td>'.$dbrow['DocName'].'</td>';
$data.='"'.$dbrow['DocName'].'"'."\t";}
if ($list_DocStatus
=='yes'){echo'<td>'.$dbrow['DocStatus'].'</td>';
$data.='"'.$dbrow['DocStatus'].'"'."\t";}

if($list_DocFilename=='yes')
    {

$ProjectFilePath='/dataftp/'.$dataftpfolder.'/'.$dbrow['DocPro
ject'].'/';
    //echo $dbrow['DocFilename']."<br>";
    //echo strpos($dbrow['DocFilename'],"http")."<br>";
    //if (isset(strpos($dbrow['DocFilename'],'http')))
{$ProjectFilePath="";} // allows external URL to be included -
URL must be changed directly and manually in dataset
    //echo $ProjectFilePath."<br>";
    echo    '<td class="bluelink">'.'<a
href="'.$ProjectFilePath.$dbrow['DocFilename'].'">'.$dbrow['Do
cFilename'].'['.$dbrow['DocFilesize'].'kB]'.'</a>';

$data.='"'.$dbrow['DocFilename'].'['.$dbrow['DocFilesize'].'kB
]'.'"'."\t";
    }

if ($list_DocRemarks
=='yes'){echo'<td>'.$dbrow['DocRemarks'].'</td>';
$data.='"'.$dbrow['DocRemarks'].'"'."\t";}

    echo '</tr>';
    $data .= "\n"; //end of dataset in excel file (marks row
end in data)
    }
echo '<tr>';
echo '<td>'.$DatasetCount.' Datasets</td>';
echo '</tr>';
echo '</table>';
```

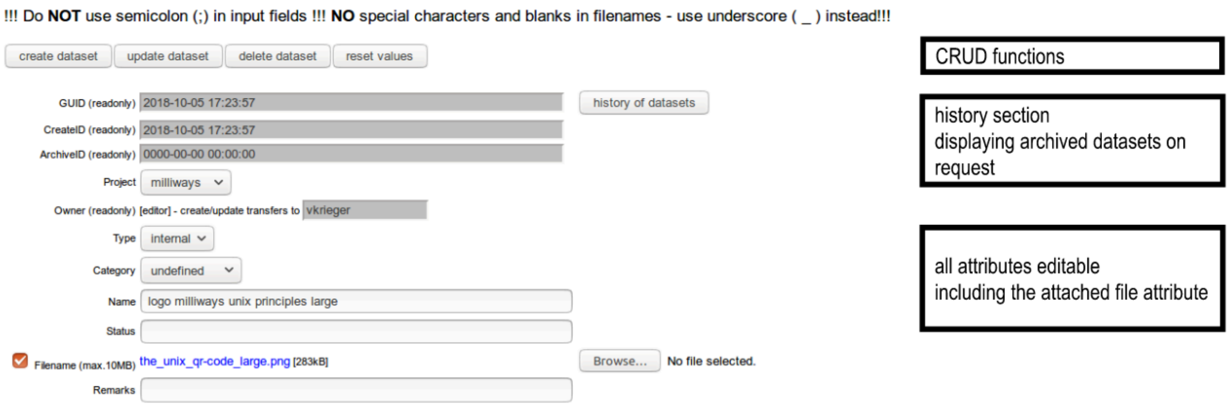
special link to data area of milliways system

```
echo '</form>';

/*
export selected table to .xls onto local server folder to
download as offered from there
*/

$fp = fopen('docslist.xls','w');
fwrite($fp,$header);
fwrite($fp,"\n");
fwrite($fp,$data);
fclose($fp);

echo '<div align="right" style="font-size: 8px;">printed at
'.$datum = date("d.m.Y - H:i").' - last source change vk
14.03.18 18:00</div>';4

?>
</body>
</html>
```

writing $data to download file

6.3 The general structure of the *applicationname*_modify.php

The typical *applicationname*_modify.php offers the possibility to create, update or delete a single datasets chosen from the output of *applicationname*_list.php. Below docs_modify.php is used for representative code.



!!! Do **NOT** use semicolon (;) in input fields !!! **NO** special characters and blanks in filenames - use underscore (_) instead!!!

create dataset | update dataset | delete dataset | reset values

GUID (readonly) 2018-10-05 17:23:57 | history of datasets

CreateID (readonly) 2018-10-05 17:23:57

ArchiveID (readonly) 0000-00-00 00:00:00

Project milliways

Owner (readonly) [editor] - create/update transfers to vkrieger

Type internal

Category undefined

Name logo milliways unix principles large

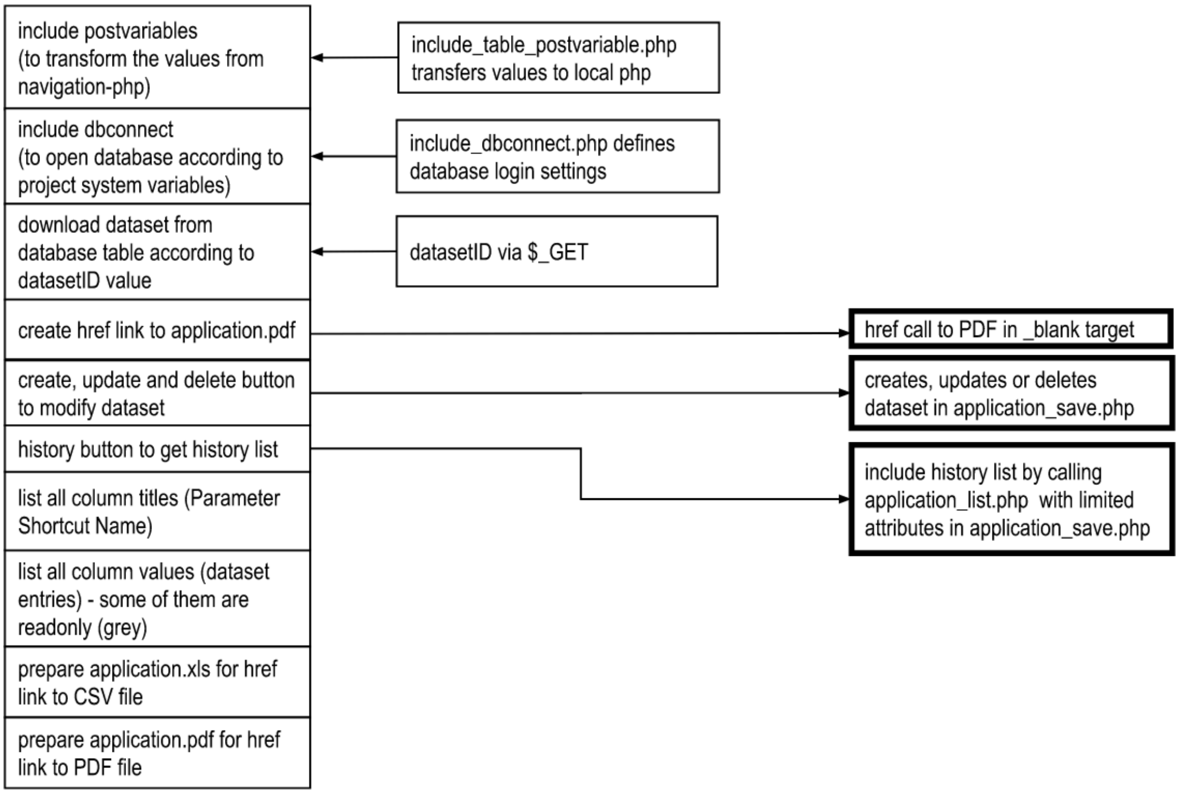
Status

☑ Filename (max.10MB) the_unix_qr-code_large.png [283kB] | Browse... No file selected.

Remarks

CRUD functions
history section displaying archived datasets on request
all attributes editable including the attached file attribute

The typical *applicationname*_modify.php consists of the following code blocks



include postvariables (to transform the values from navigation-php) ← include_table_postvariable.php transfers values to local php

include dbconnect (to open database according to project system variables) ← include_dbconnect.php defines database login settings

download dataset from database table according to datasetID value ← datasetID via $_GET

create href link to application.pdf → href call to PDF in _blank target

create, update and delete button to modify dataset → creates, updates or deletes dataset in application_save.php

history button to get history list → include history list by calling application_list.php with limited attributes in application_save.php

list all column titles (Parameter Shortcut Name)

list all column values (dataset entries) - some of them are readonly (grey)

prepare application.xls for href link to CSV file

prepare application.pdf for href link to PDF file

Figure: Code blocks in typical modify.php

history of datasets

An important requirement of Information Management according to ISO 19650 is that no more data is deleted in reality - instead it is being archived. Whenever the “delete dataset” process is initiated by pressing the respective button in the milliways Web-GUI it results in a transfer of the dataset into the archive and an apparent disappearance from the usual milliways Web-GUI. The “deleted” dataset can be retrieved from the archive.
To satisfy this “no-delete” requirement and yet keep the milliways Web-GUI reasonably simple the “history of datasets” functionality is implemented in the modify_*applicationname*.php.

Please find the full listing of a typical *applicationname*_modify.php below

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN">
<html>
<head>
      <?php include 'include_setSystemvariables.php';
include 'include_setSystemconstants.php'; ?>
      <meta content="text/html; charset=ISO-8859-1"
http-equiv="content-type">
       <title><?php echo $SystemProject; ?> database
system</title>
      <style>
    *
{font-size:16px ; font-family: Arial, Verdana, sans-serif;
}
    input,a                          {font-size:12px ;
font-family: Arial, Verdana, sans-serif; }
    select,option,textarea
{font-size:12px ; font-family: Arial, Verdana, sans-serif;
}
    table,tr,td
{font-size:10px ; font-family: Arial, Verdana, sans-serif;
}
    </style>
    <link rel="stylesheet" type="text/css"
href="formats.css">
</head>

<body>

!!! Do <b>NOT</b> use semicolon (;) in input fields !!!
<b>NO</b> special characters and blanks in filenames - use
underscore ( _ ) instead!!!
<br><br>

<?php
```

```
include 'include_setDocconstants.php';
include 'include_docs_postvariables.php';
$link =
mysqli_connect($dbserver,$username,$password,$dbname);

if (!isset($_SESSION)) { session_start();}

$DocCreateID = $_GET["DocCreateID"];

$dbquery = " SELECT * FROM docs WHERE LOCATE
('$DocCreateID', DocCreateID) >0 ";
$dbresult = mysqli_query($link,$dbquery);
$dbrow = mysqli_fetch_array($dbresult,MYSQLI_BOTH);

    $updatedataset="";
    $createdataset="";
    $deletedataset="";
    $historydataset="";

if ($updatedataset == "" AND $createdataset == "" AND
$deletedataset == "" AND $historydataset =="")
    {
    echo '<form method="post" action="docs_save.php"
enctype="multipart/form-data" >';
    if ($_SESSION['LoginType']=='admin' OR
$_SESSION['LoginType']=='supereditor' OR
$_SESSION['LoginType']=='editor')
       {
       echo '<input type="submit" name="createdataset"
value="create dataset">';
       echo '<input type="submit" name="updatedataset"
value="update dataset">';
       echo '<input type="submit" name="deletedataset"
value="delete dataset">';
             echo '<input type="reset" value="reset
values">';
             echo '<br><br>';
      }

    echo '<table>';
    echo '<tr>';
    echo '<td align="right">GUID (readonly)</td>';
    echo '<td><input style="background-color:#C0C0C0"
type="text" name="DocGUID" size="60" maxlength="100"
value="'.$dbrow['DocGUID'].'" readonly></td>';
    echo '<td><input type="submit" name="historydataset"
value="history of datasets"></td>';
    echo '</tr>';
    echo '<tr>';
    echo '<td align="right">CreateID (readonly)</td>';
    echo '<td><input style="background-color:#C0C0C0"
type="text" name="DocCreateID" size="60" maxlength="100"
```

```
value="'.$dbrow['DocCreateID'].'" readonly></td>';
    echo '</tr>';
    echo '<tr>';
    echo '<td align="right">ArchiveID (readonly)</td>';
    echo '<td><input style="background-color:#C0C0C0"
type="text" name="DocArchiveID" size="60" maxlength="100"
value="'.$dbrow['DocArchiveID'].'" readonly></td>';
    echo '</tr>';
    echo '<input type="hidden" name="DocInitialProject"
value="'.$dbrow['DocProject'].'">';
    echo '<input type="hidden" name="DocProject"
value="'.$dbrow['DocProject'].'">';
    echo '<tr>';
    echo '<td align="right">Project</td>';
    // if SystemProject is not empty this should preselect
the selected project
    // but if this is called directly from listing no
project is selected
    // then the SystemProject should be preselected
    // if SystemProject is empty this should preselect the
selected project by the Project value of the selected
dataset
    // but if this is called directly from listing public
project is selected
    // SystemTypeArray and SystemRegimeArray are stored in
SystemConstants
    /*
    from     list        modify
    empty   undef(public)  dbrow
    project system/dbrow   dbrow
    */
    echo '<td><select name="DocProject" size="1">';
       if (!empty($SystemProject))
             {foreach ($SystemProjectArray as $Project)
{echo '<option'; if ($SystemProject==$Project) {echo '
selected';} echo '>'.$Project.'</option>';}}
       else
             {foreach ($SystemProjectArray as $Project)
{echo '<option'; if ($dbrow['DocProject']==$Project) {echo
' selected';} echo '>'.$Project.'</option>';}}
       echo '</select></td>';
    echo '</tr>';
    echo '<tr>';
    echo '<td align="right">Owner (readonly)</td>';
    echo '<td>'.'['.$dbrow['DocOwner'].'] - create/update
transfers to '.'<input style="background-color:#C0C0C0"
type="text" name="DocOwner" size="15" maxlength="100"
value="'.$_SESSION['LoginLogin'].'" readonly></td>';
    echo '</tr>';
    echo '<tr>';
    echo '<td align="right">Type</td>';
    echo '<td><select name="DocType" size="1">';
             foreach ($SystemTypeArray as $Type) {echo
'<option'; if ($dbrow['DocType']==$Type) {echo '
selected';} echo '>'.$Type.'</option>';}
             echo '</select></td>';
    echo '</tr>';
    echo '<tr>';
    echo '<td align="right">Category</td>';
    echo '<td><select name="DocCategory" size="1">';
             foreach ($DocCategoryArray as $Category)
{echo '<option'; if ($dbrow['DocCategory']==$Category)
{echo ' selected';} echo '>'.$Category.'</option>';}
             echo '</select></td>';
    echo '</tr>';

echo '<tr><td align="right">Name </td><td><input
type="text" name="DocName" size="60" maxlength="255"
value="'.$dbrow['DocName'].'"></td></tr>';
echo '<tr><td align="right">Status </td><td><input
type="text" name="DocStatus" size="60" maxlength="255"
value="'.$dbrow['DocStatus'].'"></td></tr>';

/*
- existor has value "File" when $dbrow['DocFilename'] is
given
- $_FILES['DocFilename']['tmp_name'] is given
- when new DocFilename is uploaded existor should be
checked
- when existor is unchecked DocFilename should be emptied
*/
// hidden input transfers dbrow[] to FileName and FileSize
// SystemProject would be empty in case of public and
admin login

// must be dataftp instead of data/ftp and make use of
Alias in http.conf !!!
$ProjectFilePath='/dataftp/'.$dataftpfolder.'/'.$dbrow['Do
cProject'].'/';

echo '<tr>';
echo '<td align="right">';
echo '<input type="checkbox" name="existor" value="File"';
if ($dbrow['DocFilename']) {echo ' checked';} echo '>
Filename (max.10MB)</td>';
echo '<input type="hidden" name="DocFilename"
value="'.$dbrow['DocFilename'].'">';
echo '<input type="hidden" name="DocFilesize"
value="'.$dbrow['DocFilesize'].'">';
echo '<td align="left" class="bluelink"><a
href="'.$ProjectFilePath.$dbrow['DocFilename'].'">'.$dbrow
['DocFilename'].'</a> ['.$dbrow['DocFilesize'].'kB]</td>';
echo '<td><input type="file" name="DocFilename"
size="15"></td>';
echo '</tr>';
echo '<tr><td align="right">Remarks </td><td><input
type="text" name="DocRemarks" size="60" maxlength="255"
value="'.$dbrow['DocRemarks'].'"></td></tr>';

    echo '</table>';

echo '</form>';

} elseif ( $updatedataset OR $createdataset OR
$deletedataset OR $historydataset)

echo '<div align="right" style="font-size: 8px;">printed
at '.$datum = date("d.m.Y - H:i").' - last source change
vk 13.03.18 18:00</div>';

?>
</font>
</body>
</html>
```

6.4 The general structure of the *applicationname*_save.php

The typical *applicationname*_save.php completes the CRUD process selected in *applicationname*_modify.php by writing a single dataset via INSERT or UPDATE SQL-command into the database. After that it returns to *applicationname*_modify.php. Below docs_save.php is used for representative code.

The typical *applicationname*_save.php consists of the following code blocks

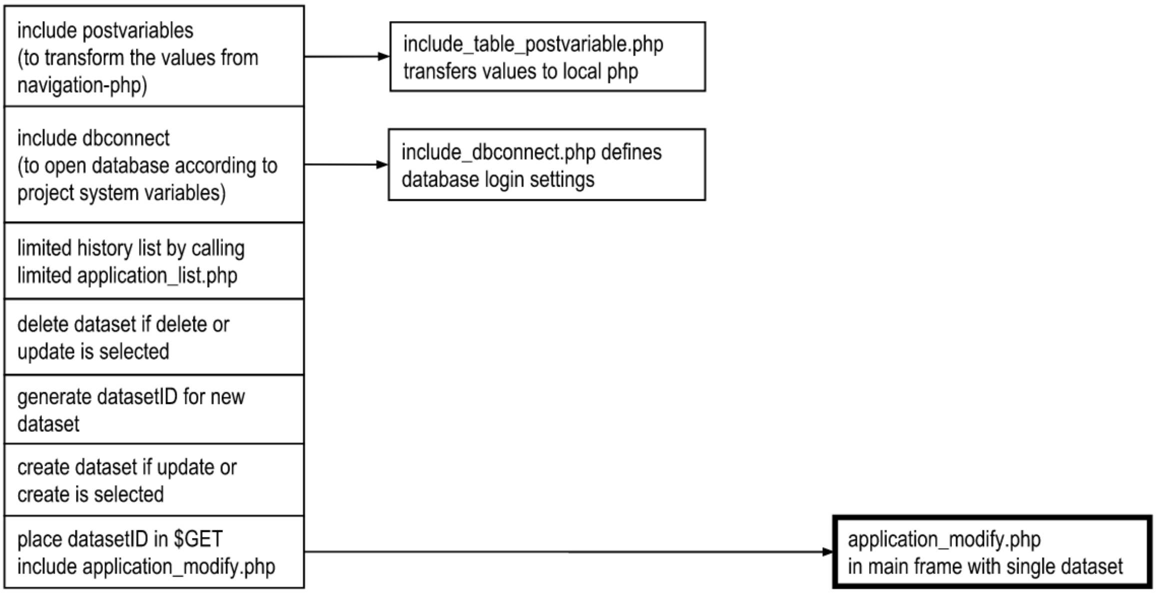

include postvariables (to transform the values from navigation-php)
include dbconnect (to open database according to project system variables)
limited history list by calling limited application_list.php
delete dataset if delete or update is selected
generate datasetID for new dataset
create dataset if update or create is selected
place datasetID in $GET include application_modify.php
include_table_postvariable.php transfers values to local php
include_dbconnect.php defines database login settings
application_modify.php in main frame with single dataset

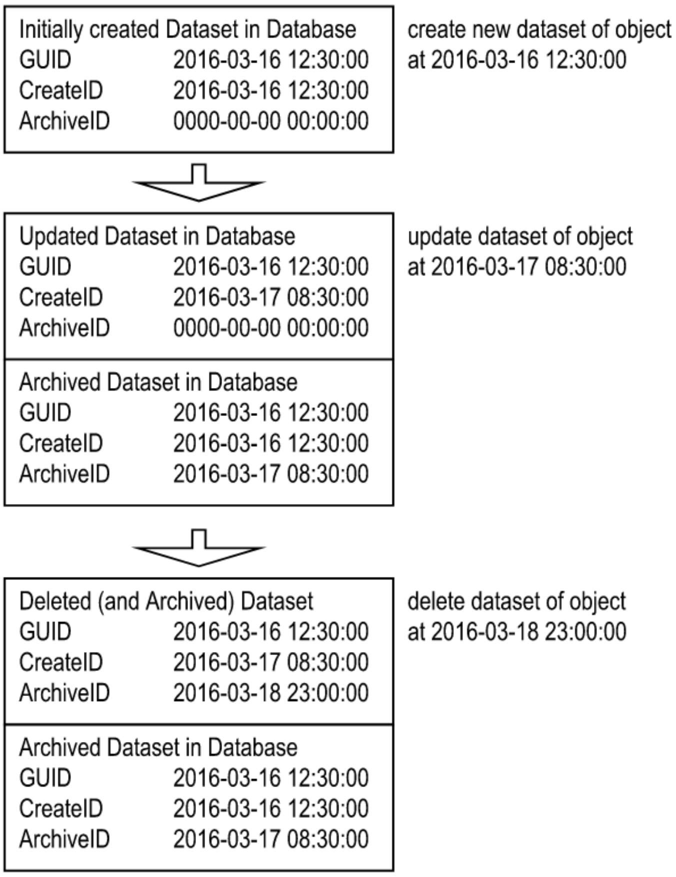
Figure: Code blocks in typical save.php

The `$_GET['DocCreateID']=$DocCreateID;` takes care that the same dataset is called when returning to *applicationname*_modify.php

*applicationname*_save.php does not output main frame unless there is a file upload.

versioning and dataset history

The CRUD process is changing the IDs of the dataset according to the following schema



Initially created Dataset in Database
GUID 2016-03-16 12:30:00
CreateID 2016-03-16 12:30:00
ArchiveID 0000-00-00 00:00:00

create new dataset of object
at 2016-03-16 12:30:00

Updated Dataset in Database
GUID 2016-03-16 12:30:00
CreateID 2016-03-17 08:30:00
ArchiveID 0000-00-00 00:00:00

Archived Dataset in Database
GUID 2016-03-16 12:30:00
CreateID 2016-03-16 12:30:00
ArchiveID 2016-03-17 08:30:00

update dataset of object
at 2016-03-17 08:30:00

Deleted (and Archived) Dataset
GUID 2016-03-16 12:30:00
CreateID 2016-03-17 08:30:00
ArchiveID 2016-03-18 23:00:00

Archived Dataset in Database
GUID 2016-03-16 12:30:00
CreateID 2016-03-16 12:30:00
ArchiveID 2016-03-17 08:30:00

delete dataset of object
at 2016-03-18 23:00:00

Figure: Different stages of dataset IDs.

The “active” dataset has always the timestamp “0000-00-00 00:00:00” in ArchiveID.
The CreateID identifies the current dataset and changes over time. It is changed everytime the dataset is changed and updated.
The common GUID identifies the sequential set of datasets. The common GUID is identical with the CreateID at initial creation. Therefore a new GUID is only created when a new dataset is created.
Find the respective listing in the *applicationname*_save.php

```
if (!empty($_POST["updatedataset"]) OR
!empty($_POST["deletedataset"]))
    {
    $dbchange = "UPDATE docs SET DocArchiveID =
'$CurrentTimeStamp' WHERE DocCreateID = '$DocCreateID'";
    $dbquery = mysqli_query($link,$dbchange) or die ("not
updated!");
    }

if (!empty($_POST["updatedataset"]) OR
!empty($_POST["createdataset"]))
    {
    if (!empty($_POST["createdataset"])) {$DocGUID =
$CurrentTimeStamp;}
    $DocCreateID = $CurrentTimeStamp;
    $DocArchiveID='0000-00-00 00:00:00';
```

Please also refer user manual for more details.

Displaying the dataset history is achieved by a <form></form> command in *applicationname*_modify.php. Executing this command results in listing all archived datasets by their IDs, Owners, Names and Status. Clicking on one of the CreateIDs in this history recalls this dataset in *applicationname*_modify.php - thus allowing the user to create a new dataset from this archived dataset. This way an archived dataset can be transformed into the active dataset - leaving the archived version though untouched.

Please find the full listing of a typical *applicationname*_save.php below

```
<!DOCTYPE html public "-//W3C//DTD HTML 4.0 //EN">
<html>
<head>
      <?php include 'include_setSystemvariables.php';
include 'include_setSystemconstants.php'; ?>
      <meta content="text/html; charset=ISO-8859-1"
http-equiv="content-type">
       <title><?php echo $SystemProject; ?> database
system</title>
      <style>
    *
{font-size:16px ; font-family: Arial, Verdana, sans-serif;
}
    input,a                         {font-size:12px ;
font-family: Arial, Verdana, sans-serif; }
    select,option,textarea                 {font-size:12px
; font-family: Arial, Verdana, sans-serif; }
    table,tr,td
{font-size:10px ; font-family: Arial, Verdana, sans-serif;
}
    </style>
    <link rel="stylesheet" type="text/css"
href="formats.css">
</head>

<body>
<?php

include 'include_setDocconstants.php';
include 'include_docs_postvariables.php';
$link =
mysqli_connect($dbserver,$username,$password,$dbname);

if (!isset($_SESSION)) { session_start();}

if (!empty($_POST["historydataset"]))
    {
    $_POST['partDocGUID']=$DocGUID;
    $_POST['partDocArchiveID']='';
    $_POST['list_DocGUID']='yes';
    $_POST['list_DocCreateID']='yes';
    $_POST['list_DocArchiveID']='yes';
    $_POST['list_DocOwner']='yes';
    $_POST['list_DocName']='yes';
    $_POST['list_DocStatus']='yes';
    include 'docs_list.php';
    }

$CurrentTimeStamp = date("Y-m-d H:i:s");

// deletedataset: puts current dataset in archive status by
entering current datetime into ArchiveID
// updatedataset: puts current dataset in archive status by
entering current datetime into ArchiveID and then
createdataset

if (!empty($_POST["updatedataset"]) OR
!empty($_POST["deletedataset"]))
    {
    $dbchange = "UPDATE docs SET DocArchiveID =
'$CurrentTimeStamp' WHERE DocCreateID = '$DocCreateID'";
    $dbquery = mysqli_query($link,$dbchange) or die ("not
updated!");
    }
// updatedataset: creates new dataset with same GUID and
with current datetime in CreateID
// createdataset: creates new dataset with new GUID and
with current datetime in CreateID, leaving the current
untouched

if (!empty($_POST["updatedataset"]) OR
!empty($_POST["createdataset"]))

    {
    if (!empty($_POST["createdataset"])) {$DocGUID =
$CurrentTimeStamp;}
    $DocCreateID = $CurrentTimeStamp;
    $DocArchiveID='0000-00-00 00:00:00';


$ProjectFilePath='/data/ftp/'.$dataftpfolder.'/'.$DocProjec
t.'/';

$ProjectInitialFilePath='/data/ftp/'.$dataftpfolder.'/'.$Do
cInitialProject.'/';

    if (!empty($_FILES['DocFilename']['tmp_name']))
      {if (
move_uploaded_file($_FILES['DocFilename']['tmp_name'],
$ProjectFilePath.basename
($_FILES['DocFilename']['name'])))
             { echo $DocFilename = basename
($_FILES['DocFilename']['name']); echo ' uploaded!<br>';
$DocFilesize = round($_FILES['DocFilename']['size']/1000);}
             else { echo ' error uploading!<br>'; }
       } else {if (!$existor){$DocFilename='';
$DocFilesize='';}}

    if ((!empty ($DocFilename)) AND
empty($_FILES['DocFilename']['tmp_name']))
       {
       $DocFilenameExist = $DocFilename;
       copy
($ProjectInitialFilePath.$DocFilenameExist,$ProjectFilePath
.$DocFilename);
       }

    $dbchange = "INSERT INTO docs SET

       DocGUID = '$DocGUID',
       DocCreateID = '$DocCreateID',
       DocArchiveID = '$DocArchiveID',
       DocProject = '$DocProject',
       DocOwner = '$DocOwner',
       DocType = '$DocType',
       DocCategory = '$DocCategory',
       DocName = '$DocName',
       DocFilename = '$DocFilename',
       DocFilesize = '$DocFilesize',
       DocStatus = '$DocStatus',
       DocRemarks = '$DocRemarks'
       ";
    $dbquery = mysqli_query($link,$dbchange) or die ("not
created!");
    }

$_GET['DocCreateID']=$DocCreateID;
include 'docs_modify.php';

echo '<div align="right" style="font-size: 8px;">printed at
'.$datum = date("d.m.Y - H:i").' - last source change vk
14.03.18 18:00</div>';

?>
</body>
</html>
```

6.5 How to match program code with database

milliways is designed to be as modular as possible. Whereas each module and the system as a whole is open source. As a result milliways has two relatively independent but strongly interacting modules:

- the database holding all the information except the uploaded information containers
- the PHP program code modules with variables exactly having to match the above

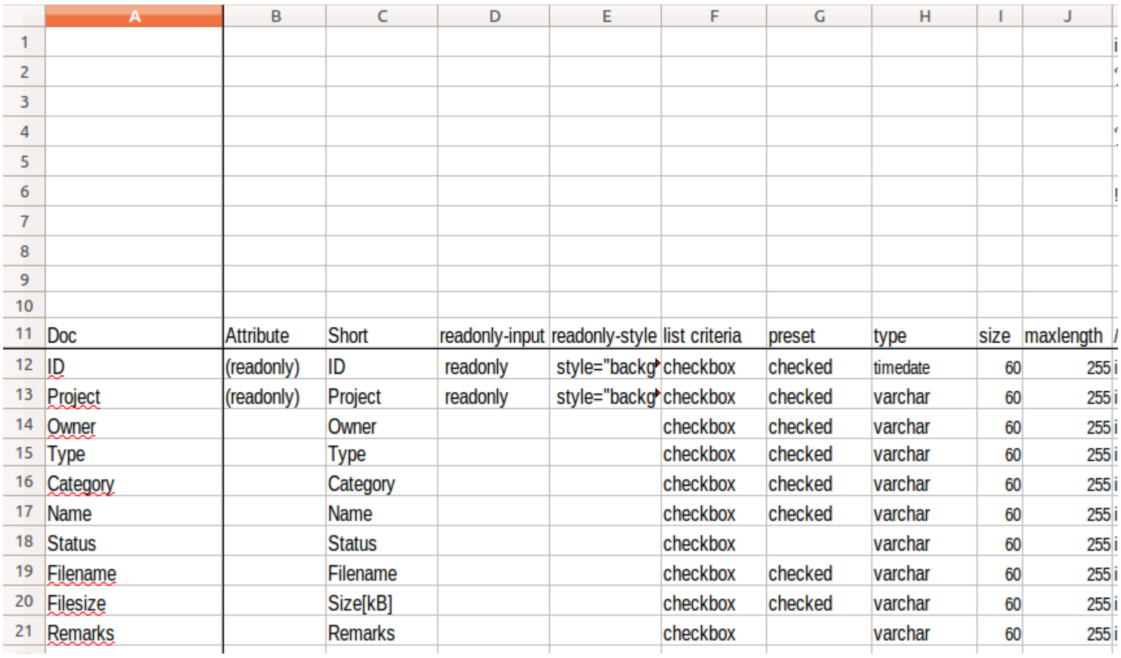
To ensure this compatibility it is recommended to make use of calculation sheet.

1) make list of attributes (variables in PHP) in sheet as strings
2) make list of code snippets for each code section for each application as string
3) combine strings to complete code lines

This way creating, updating or deleting dataset structures is done by changing the list of attributes. If done correctly the complete code lines can be copied from the sheet into the code program file.
To check compatibility between database and program code only the primary listing of attributes (variables in PHP) in the sheet must be compared with the structure listing of the corresponding database table.

making use of calculation sheets for programming code

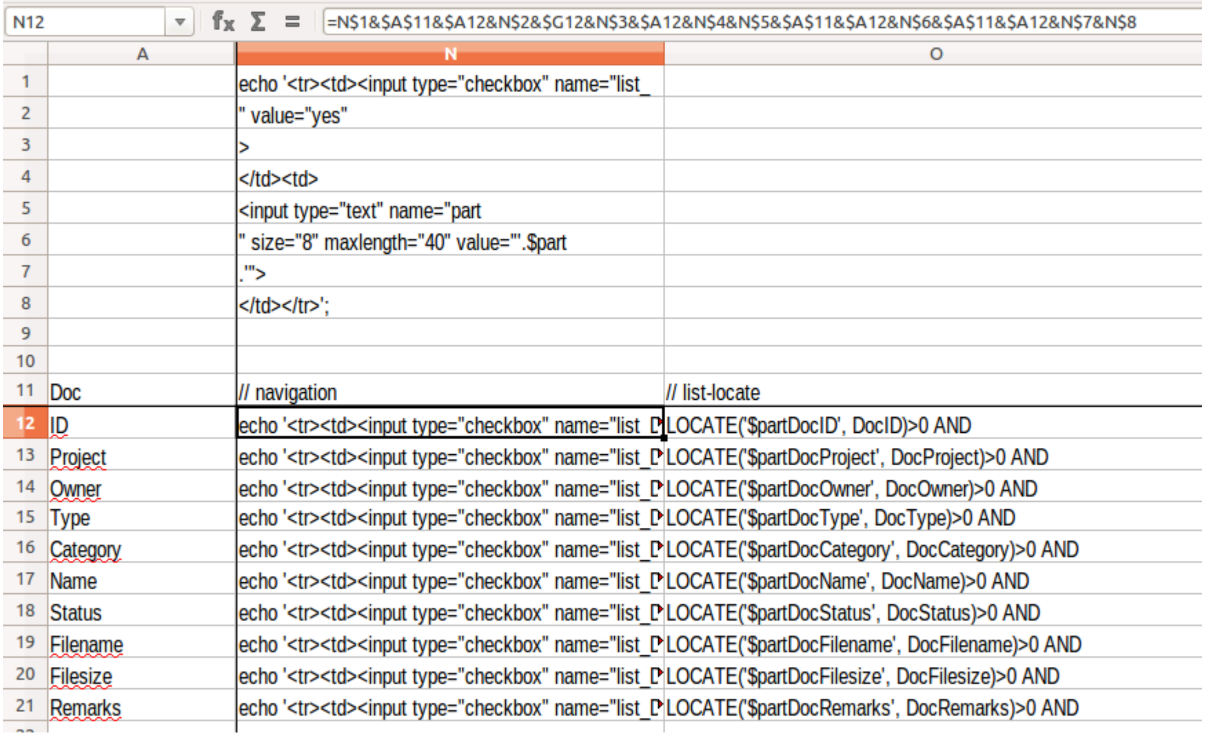
Primarily all variables/attribute for specific application are listed in a sheet. Also some additional “attributes of the attributes” are place in the sheet:



	A	B	C	D	E	F	G	H	I	J	
1											i
2											'
3											
4											'
5											
6											!
7											
8											
9											
10											
11	Doc	Attribute	Short	readonly-input	readonly-style	list criteria	preset	type	size	maxlength	/
12	ID	(readonly)	ID	readonly	style="backg	checkbox	checked	timedate	60	255	i
13	Project	(readonly)	Project	readonly	style="backg	checkbox	checked	varchar	60	255	i
14	Owner		Owner			checkbox	checked	varchar	60	255	i
15	Type		Type			checkbox	checked	varchar	60	255	i
16	Category		Category			checkbox	checked	varchar	60	255	i
17	Name		Name			checkbox	checked	varchar	60	255	i
18	Status		Status			checkbox		varchar	60	255	i
19	Filename		Filename			checkbox	checked	varchar	60	255	i
20	Filesize		Size[kB]			checkbox	checked	varchar	60	255	i
21	Remarks		Remarks			checkbox		varchar	60	255	i

figure: list of attributes of doc application

From this source the respective code lines are calculated by string-addition in the sheet for each application.



N12 =N$1&$A$11&$A12&N$2&$G12&N$3&$A12&N$4&N$5&A11&$A12&N$6&A11&$A12&N$7&N$8

	A	N	O
1		echo '<tr><td><input type="checkbox" name="list_	
2		" value="yes"	
3		>	
4		</td><td>	
5		<input type="text" name="part	
6		" size="8" maxlength="40" value="'.$part	
7		.'">	
8		</td></tr>';	
9			
10			
11	Doc	// navigation	// list-locate
12	ID	echo '<tr><td><input type="checkbox" name="list_D	LOCATE('$partDocID', DocID)>0 AND
13	Project	echo '<tr><td><input type="checkbox" name="list_D	LOCATE('$partDocProject', DocProject)>0 AND
14	Owner	echo '<tr><td><input type="checkbox" name="list_D	LOCATE('$partDocOwner', DocOwner)>0 AND
15	Type	echo '<tr><td><input type="checkbox" name="list_D	LOCATE('$partDocType', DocType)>0 AND
16	Category	echo '<tr><td><input type="checkbox" name="list_D	LOCATE('$partDocCategory', DocCategory)>0 AND
17	Name	echo '<tr><td><input type="checkbox" name="list_D	LOCATE('$partDocName', DocName)>0 AND
18	Status	echo '<tr><td><input type="checkbox" name="list_D	LOCATE('$partDocStatus', DocStatus)>0 AND
19	Filename	echo '<tr><td><input type="checkbox" name="list_D	LOCATE('$partDocFilename', DocFilename)>0 AND
20	Filesize	echo '<tr><td><input type="checkbox" name="list_D	LOCATE('$partDocFilesize', DocFilesize)>0 AND
21	Remarks	echo '<tr><td><input type="checkbox" name="list_D	LOCATE('$partDocRemarks', DocRemarks)>0 AND

N12 generates the code for the navigation application with reference to the variable ID.

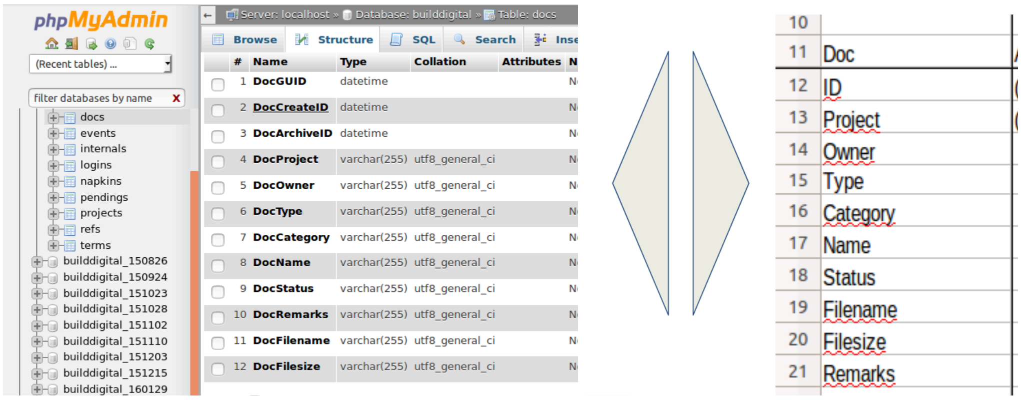
```
echo '<tr><td><input type="checkbox" name="list_DocID" value="yes" checked>ID</td><td><input type="text"
name="partDocID" size="8" maxlength="40" value="'.$partDocID.'"></td></tr>';
```

figure: program code line of doc application made by combination of strings

By drag and drop of multiple cells of the sheet into the respective program code at the correct place a fast and correct programming code is generated.

check the database tables with the code variables

Currently the check between the database and the milliways code must be done manually. In principle the configuration of the SQL-database could be also done by the calculation sheet method generating correct SQL code. This could be a future feature.



#	Name	Type	Collation	Attributes	N
1	DocGUID	datetime			N
2	DocCreateID	datetime			N
3	DocArchiveID	datetime			N
4	DocProject	varchar(255)	utf8_general_ci		N
5	DocOwner	varchar(255)	utf8_general_ci		N
6	DocType	varchar(255)	utf8_general_ci		N
7	DocCategory	varchar(255)	utf8_general_ci		N
8	DocName	varchar(255)	utf8_general_ci		N
9	DocStatus	varchar(255)	utf8_general_ci		N
10	DocRemarks	varchar(255)	utf8_general_ci		N
11	DocFilename	varchar(255)	utf8_general_ci		N
12	DocFilesize	varchar(255)	utf8_general_ci		N

10	
11	Doc
12	ID
13	Project
14	Owner
15	Type
16	Category
17	Name
18	Status
19	Filename
20	Filesize
21	Remarks

figure: manual check between SQL-database table structure and variable list in calculation sheet

> milliways database and milliways code are technically independent.
> Their compatibility must be ensured manually.
> Calculation sheets can help generating of repeating lines of code.
> The milliways database may be downloaded at any time by standard SQL commands.

6.6 The structure of the addresses application

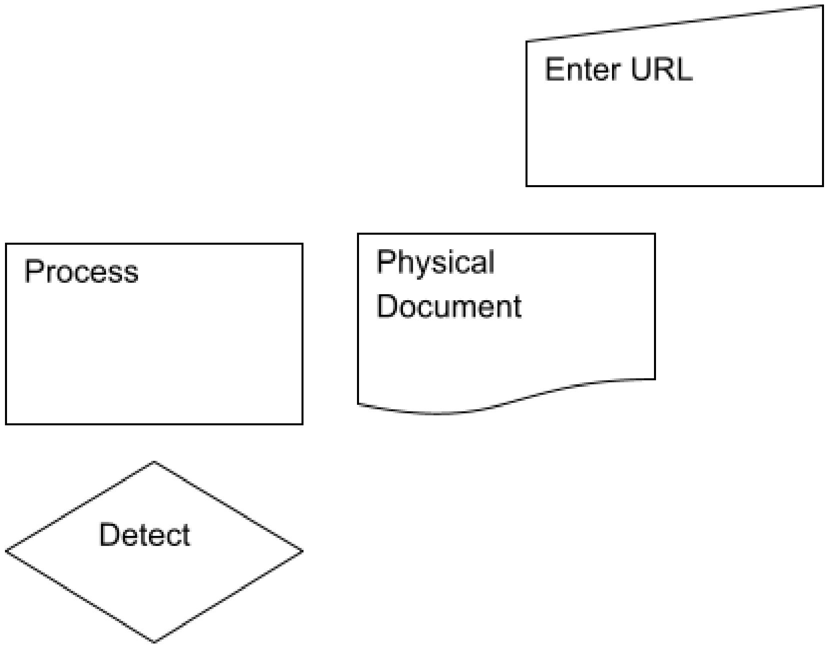
tbd

6.7 The structure of the docs application

tbd

7. Special Processes

…



7.1 The login process

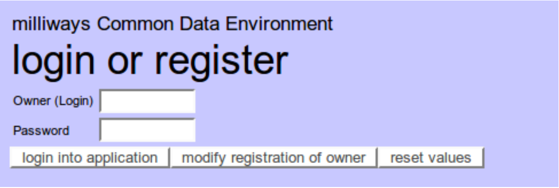
The login process is currently a two-step process. It should be simplified.
The current login process uses

1) htaccess web login features installed on operating system level.
   Login and password for this process step are implemented at operating system level. The web login defines the $Systemproject variable that can be referred to by PHP - thus defining the project the user has logged in for and works in the milliways system
2) PHP programmed code for personal login
   Login and password for the personal login process step refer to a database table in milliways. Thus they can be seen in plain text format by specially authorized database accounts.

The personal login process - after successful web login and thereby setting the project domain - has five files available. Two of them make up the login routine. The additional three routines are - similar to the other, normal applications - to create, modify or delete new users to the project.

- logins_query.php
- logins_check.php
  and the additional routines to change the database table “logins”
- logins_list.php
- logins_modify.php
- logins_save.php

The personal login process starts with logins_query.php coming from index.html.



milliways Common Data Environment

login or register

A “public” web login overrides the personal login, but results in restricted access to only “public” datasets.

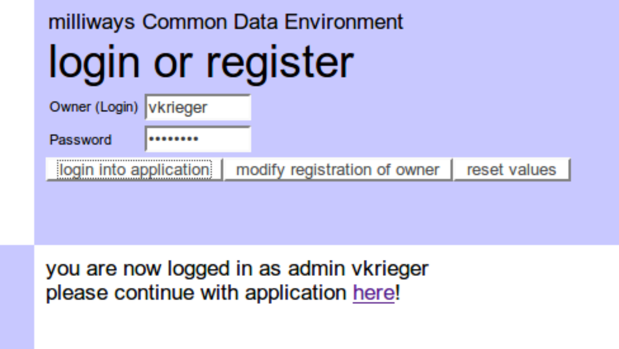
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
"http://www.w3.org/TR/html4/frameset.dtd">
<html>
<head>
…
</head>
```

```
<body>
<?php
include 'include_logins_postvariables.php';
include 'include_setSystemvariables.php';

if ($SystemType == "public")
    {
    session_start();
    $_SESSION['LoginType']="viewer";
    $_SESSION['LoginLogin']="public";
    echo 'You are now logged in as '.$_SESSION['LoginLogin'].'
'.$_SESSION['LoginType'].'<br>';
    echo 'Please continue with application <a href="header.php"
target="header">here</a>!<br>';
    }
    else
    {
    echo '<table border="0" cellpadding=0">';
    echo '<tr><td class="whitelink" align="left" valign="center"
style="font-size:16px">';
    echo '<a href="index.html" target="_blank">'.$SystemProject.'
Common Data Environment</a>';
    echo '</td></tr>';
    echo '<tr><td><span style="font-size:32px">login or
register</span></td></tr>';
    echo '</table>';

    $login='';
    $register='';

    if ($register == "" AND $login == "")
       {
       echo '<form method="post" action="logins_check.php"
target="main">';
       echo '<table>';
       echo '<tr><td>Owner (Login)</td><td><input type="text"
name="LoginLogin" size="8" maxlength="40"
value="'.$LoginLogin.'"></td></tr>';
       echo '<tr><td>Password</td><td><input type="password"
name="LoginPassword" size="8" maxlength="40"
value="'.$LoginPassword.'"></td></tr>';
       echo '</table>';
       echo '<input type="submit" name="login" value="login into
application">';
       echo '<input type="submit" name="register" value="modify
registration of owner">';
             echo '<input type="reset" value="reset values">';
       echo '</form>';
       }
       elseif
       ($register OR $login);
    }
```

```
?>
</body>
</html>
```

The logins_check.php routine follows the logins_query.php from the <form></form> code. The logins_check.php routine checks the entered login values against the entries of the logins table of the database. It displays in main frame

- on positive login a link to the main milliways program
- on refused login a respective message in main frame



milliways Common Data Environment

login or register

Owner (Login) vkrieger

Password ••••••••

you are now logged in as admin vkrieger
please continue with application here!

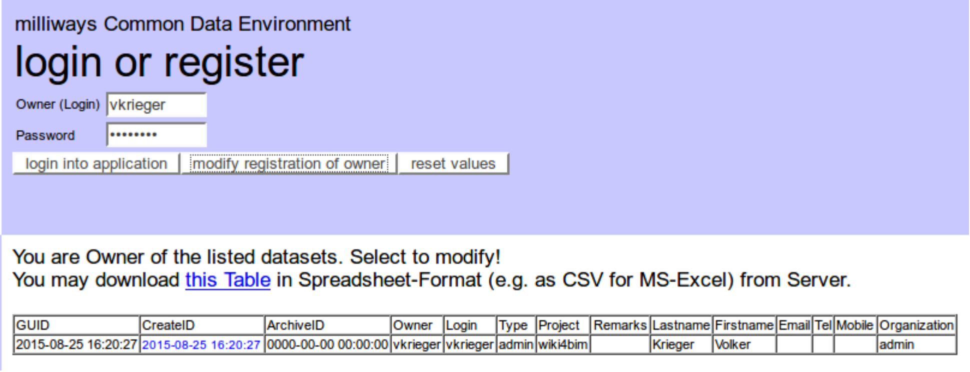
```
<!doctype html public "-//W3C//DTD HTML 4.0 //EN">
<html>
<head>
…
</head>

<body>
<?php

include 'include_logins_postvariables.php';
include 'include_setSystemvariables.php';
$link = mysqli_connect($dbserver,$username,$password,$dbname);

if (!empty($_POST["register"]))
    {include 'logins_list.php';}

if (!empty($_POST["login"]))
    {
    // compare owner and password given with database login and
password
    // missing check for project !!!!!
    $dbquery = "SELECT * FROM logins WHERE
('$LoginLogin'=LoginLogin) AND('$LoginPassword'=LoginPassword)";
      $dbresult = mysqli_query($link, $dbquery);  echo
mysqli_error($link);
      $dbrow = mysqli_fetch_array($dbresult);
    // if identical offer link to application with set access
type
```

```
    if ($dbrow['LoginLogin'] AND $dbrow['LoginPassword'])
      {
      session_start();
      $_SESSION['LoginType']=$dbrow['LoginType'];
      $_SESSION['LoginLogin']=$dbrow['LoginLogin'];
       echo 'you are now logged in as '.$_SESSION['LoginType'].'
'.$_SESSION['LoginLogin'].'<br>';
       echo 'please continue with application <a
href="header.php" target="header">here</a>!<br>';
       }
// if not in database
    else {echo 'sorry, the credentials you have supplied do not
match the login database. please try again!<br>';}
    }

//last source change vk 13.03.18 18:00
?>
</body>
</html>
```

clicking on the “here” link results in the normal header described in the header section above.

modify registration of owner

The login process offers the possibility - depending of your login credentials (admin, supereditor, editor, superviewer, viewer) - to create, update or delete new users. This is done by the additional login routines similar to the processes described already with the normal applications routines (list, modify, save).



milliways Common Data Environment

login or register

Owner (Login) vkrieger

Password ••••••••

You are Owner of the listed datasets. Select to modify!
You may download this Table in Spreadsheet-Format (e.g. as CSV for MS-Excel) from Server.

GUID	CreateID	ArchiveID	Owner	Login	Type	Project	Remarks	Lastname	Firstname	Email	Tel	Mobile	Organization
2015-08-25 16:20:27	2015-08-25 16:20:27	0000-00-00 00:00:00	vkrieger	vkrieger	admin	wiki4bim		Krieger	Volker				admin

8. Bibliography

…