

Educational Problem Manager Design Manual

Robert L. Walton

September 15, 2020

Notice

The authors have placed EPM (its files and the content of these files) in the public domain; they make no warranty and accept no liability for EPM.

Table of Contents

| | | |
|-------|--|----|
| 1 | Introduction | 3 |
| 2 | Definitions and Rules | 3 |
| 2.1 | Names | 3 |
| 2.2 | Times | 3 |
| 2.3 | Account IDs | 4 |
| 2.4 | Random IDs | 4 |
| 2.5 | Tabs and Windows | 4 |
| 2.6 | Directories | 5 |
| 2.7 | Page Initialization | 6 |
| 2.8 | Locking | 8 |
| 2.9 | Security | 10 |
| 3 | Data Files | 11 |
| 4 | Session Variables | 15 |
| 5 | Web Pages | 17 |
| 5.1 | Index Page | 17 |
| 5.1.1 | Index Page File Formats | 18 |
| 5.1.2 | Index Page Session Variables | 19 |
| 5.1.3 | Index Page Transactions | 20 |
| 5.2 | Login Page | 21 |
| 5.2.1 | Login Page File Formats | 21 |

| | | |
|-------|--|----|
| 5.2.2 | Login Page Session Variables | 24 |
| 5.2.3 | Login Page Transactions | 24 |
| 5.3 | User Page | 25 |
| 5.3.1 | User Page File Formats | 25 |
| 5.3.2 | User Page Transactions | 27 |
| 5.4 | Problem Page | 29 |

1 Introduction

This document gives design information for EPM system maintainers. This document supplements but does not reiterate documentation in the EPM Help Page for users. Comments in code files in turn supplement but do not, with the exception of parameters files, reiterate this document or the Help Page.

Instructions for setting up an EPM server are in the file

```
include/maintenance_parameters.php
```

2 Definitions and Rules

2.1 Names

1. **User chosen names** consists of letters, digits, dash(-), and underscore(_), begin with a letter, and end with a letter or digit. See `/include/parameters.php $epm_name_re`.
2. **Visible file names** have basenames that consist of letters, digits, dash(-), and underscore(_), begin with a letter or digit, and end with a letter or digit, and optional extensions that obey same rules. See `/include/parameters.php $epm_filename_re`.
3. **Visible problem file names** have basenames that end with the problem name, which may optionally be preceded by a dash(-) but not by any other character.
4. Invisible problem file and directory names begin and end with plus(+).
5. Administrative files may follow other rules. In particular, email addresses have a file with a name that is the URL encoded email address, and browser tickets have a file with a name that is the 32 hex digit ticket itself.
6. **User IDs** and **team IDs** are user chosen names. An **account ID** is either a user ID or a team ID.
7. **E-mail addresses** may not have the characters <, >, ", :, or space characters.
8. A **login name** is either an e-mail address, or an account ID followed by a : followed by an e-mail address.

2.2 Times

1. Times are formatted as per `/include/parameters.php` which:

- defines `$epm_time_format` (defaults to "%FT%T%Z", which produces times such as 2020-09-15T07:40:10EDT)
- sets the time zone using `date_default_timezone_set`

2.3 Account IDs

1. **Account IDs** (AIDs) are user chosen names (2.1.1)). They are unique to the account and used for both external and internal identification. Once assigned, they cannot be changed.
2. There are two kinds of AIDs: **user UUIDs** for individual users, and **team TIDs** for teams (2.1.6).

2.4 Random IDs

1. A **random ID** is a 32 hexadecimal digit number, or equivalently a 128-bit number. Several are generated from `/dev/random` the first time the server is used, and thereafter they are generated as a pseudo-random sequence using previously generated values to aes-128-cbc encrypt previous values. See `/include/epm_random.php`.
2. Browser TICKETS are random IDs.
3. The **\$ID** variable is a random ID used to validate both POST and GET requests from pages.

For each tab, and sometimes for the view window, the first GET for the tab or window generates the first \$ID value for the pages that will occupy the tab or window, and also generates a random key that is used to generate a sequence of \$ID values for the tab or window by encrypting each \$ID to generate the next \$ID. Thereafter each request is checked to see if it has the right \$ID value, and a new \$ID value is generated for the next tab or window contents.

\$ID values are generated and checked by `/page/index.php` which is required by all page .php files (2.5.4).

2.5 Tabs and Windows

1. Transactions that make changes to the server files are executed in tabs. The **main tab** is for non-problem specific transactions. For each account problem there is a **problem tab** for transactions on that problem.

Popup windows are used to display information about server files, without making any changes. The **help window** displays help and guide information. The **documents**

index and *downloads index* windows display indices of available documents and downloads. The *auxiliary window* displays files and other information. Pages that are loaded into the auxiliary window by default can be loaded into floating windows instead (just by holding down an ALT key when launching the window): each floating window is separate and specific to its content.

2. Pages are assigned to tabs or windows. The Login, Logout, Project, User, Manage, List Edit, and Favorites Edit Pages are assigned to the main tab; Problem, Option, and Run Pages are assigned to problem tabs; the View and Template Pages are assigned to the auxiliary window; the Documents Page is assigned to the documents index window; the Downloads Page is assigned to the downloads index window; and Help and Guide Pages are assigned to the help pop-up window.
3. **Page Rule** At any given time a tab or window has a current page. A GET can change the current page. All transactions done with POSTs are checked to be sure their page is the current page for its the tab or window type. So, for example, if you have just done a GET to the Project Page, you cannot POST to the User Page. Or if you have just done a GET to the Option Page with problem=PPPP, you cannot do a POST to the Problem Page with problem=PPPP.

This rule is checked by `index.php` which is required at the beginning of all pages in tabs or windows that access the server state.

4. **Sequence Rule** Transactions within a tab are sequenced, so that if a transaction is out of sequence the tab becomes *orphaned* and must be closed. Sequencing prevents two main tabs from existing at the same time, or two problem tabs for the same problem existing at the same time.

Sequencing is done by random sequence \$IDs that are attached to each page. The next request must contain the current \$ID else the tab is orphaned. For the main tab the Login Page initializes the tab's \$ID sequence. For problem tabs the Problem Page initializes the sequence. For the view window, the View Page initiates and uses sequencing, but other view window pages are no-post pages that do no sequencing. The help window pages are `.html` pages that do no sequencing.

This rule is checked by `index.php` which is required at the beginning of all pages in tabs or windows that access the server state.

5. **Stateless Pages** The Help Page, Guide Page, and Downloads Page are stateless. They access just read-only files in the Home Directory (2.6.1).

2.6 Directories

1. There are three main directories:

H, Home Directory: This is the `epm` directory which is loaded from `github`.

W, Web Directory: This is the directory named by the EPM server URL. It contains a symbolic link to the `index.php` file that is the first file loaded when a user initially contacts the EPM server.

D, Data Directory: This is the directory containing all the mutable data for the EPM server.

2. The following subdirectories of H contain the EPM files that are directly visible to web clients:

H/page: Loadable page files. `W/page` is symbolically linked to this directory.

H/page/downloads: Example files downloadable by the client.

3. The following subdirectories of H contain the EPM files that are not directly visible to web clients:

H/include: Files that can be ‘require’ed by loadable page files.

H/bin: Binary executables of programs called by loadable pages or used for off-line maintenance.

H/template: Templates used to compute client problem files from other client or project files.

H/doc: Off-line documentation files, including this file.

H/secure: Source code for binary executables involved with security.

H/src: Source code for binary executables not involved with security.

H/setup: Initial contents of D, the data directory, during EPM server setup.

2.7 Page Initialization

1. The web directory, W (2.6.1), contains the following:

symbolic link `W/page` \rightarrow `H/page`

symbolic link `W/index.php` \rightarrow `page/index.php`

`W/parameters.php`, modified copy of `H/include/parameters.php`

`W/maintenance_parameters.php`,

modified copy of `H/include/maintenance_parameters.php`,
(only used off-line)

2. When loaded, a page initializes by executing the following steps:

- Set `$epm_page_type` to indicate the tab or pop-up window or other type. The possible values are:
`+main+` and `+problem+` for tabs;
`+view+` for a view pop-up window that POSTs;

+no-post+ for a view pop-up window that does not POST;

+download+ for pages that download files so that <script> in /page/index.php which implements the help button is suppressed (these pages do no POSTing and have no buttons).

- If the page is the first loaded in a tab or popup-window that POSTs, then it must set \$epm_ID_init to initialize a new \$ID sequence for the tab or popup-window. Otherwise the page leaves \$epm_ID_init unset.
- The page requires /page/index.php using:

```
require __DIR__ . '/index.php'
```

3. Upon being required, /page/index.php executes the following in order:

- Compute:

```
$epm_root = ROOT
$epm_self = SELF
$epm_web = W
```

where the page currently being loaded has the URL

```
http://HOST/ROOT/SELF
```

SELF has the form page/..., or if not that, the form index.php, HOST is the EPM server host name, and ROOT is whatever is left over. Here W is the EPM server web directory (p6) and is

```
$_SERVER['DOCUMENT_ROOT'] . ROOT
```
- If SELF is either index.php or page/index.php, re-routes the request to page/login.php. The request must be a 'GET' else it is not accepted.
- Loads W/parameters.php which in turn defines H and D (2.6.1).
- Runs the following checks and aborts invalid requests:
 - Checks that the client request is using the same IP address as was used for login, unless the \$epm_check_ipaddr parameter is false.
 - Checks that the session is logged in, unless the page being loaded is /page/login.php or /page/user.php.
 - Uses EPM_ABORT (p24) to check that no other session has been started after this session using the same AID:UID login name.
- If the session has logged in, defines:

```
$aid = $_SESSION['EPM_AID']
$uid = $SESSION['EPM_UID']
$name = <login name>
$is_team = $_SESSION['EPM_IS_TEAM']
$rw = true if page is read-write; false if read-only
      (see page/index.php and include/epm_rw.php for details)
```

- Defines functions and error handlers.
- Except for pages of `+download+` and `+no-post+` type, checks for violations of the Page Rule (p5) and aborts violating requests.
- Except for pages of `+download+` and `+no-post+` type, initializes or checks the \$ID to enforce the Sequence Rule (p5), and re-routes violating requests to the `/page/orphan.html` page to declare the tab or window orphaned.
- Except for pages of `+download+` type and xhttp requests, sets up shutdown function that will write statistics into `accounts/AID/+read-write+` or `accounts/AID/+read-only+`.
- Except for pages of `+download+` type and xhttp requests, defines functions and parameters for use in creating buttons and launching widows, including `<script>` functions.
- Note that many parameters and some functions are defined in `W/parameters.php`. See that file. Also see `/page/index.php` for functions it defines and button parameters it defines.

2.8 Locking

In EPM each request is an independent transaction. Locking is needed to keep two requests from interfering with each other.

Some EPM operations consist of multiple requests. However, only the last of these requests modifies EPM server state (that is not in a working subdirectory dedicated to the operation). So the strategy is to have this last request check whether other conflicting requests happened during the operation, and if yes, the last request aborts, does not change EPM state, and produces an error message.

1. **Session Locking** At the beginning of each request the PHP `session_start()` function is called. This locks the session file (where session data is stored). As a consequence, given two requests to the same session, one must complete before the other starts.
2. **Atomic Files** Some files are shared between sessions and need to be read and written atomically, so that they maintain their format specifications, but need no other locking:
 - `.list` files containing problem lists. Only one session can write such a file, but many may read it.
 - `+priv+` files containing privileges. It's possible but rare for such a file to be writable by several sessions if it has multiple owners, but if these collide, 'last-writer-wins' is an acceptable implementation. These files can have many readers.

3. **Tab Uniqueness** A session is logged into a particular account. Each tab has a type, either ‘main’ or the name of a problem in the session account. The Sequence Rule (2.5.4) ensures that there is at most one tab of each type at a given time.

More specifically, if a second tab of a give type is opened by the user for the session, the second tab gets a new sequence of \$ID numbers for the tab type, and when the first tab makes a request, its now obsolete \$ID number is detected (by `index.php`) and the tab contents are replaced by the `orphan.php` page which announces that the tab is *orphaned* and should be closed.

A similar thing happens if two windows of ‘view’ type exist with pages that execute POSTs. ‘View’ windows make no changes in the EPM file system, but do have their own session variables.

4. **Tab Independence** The pages in each tab, for the most part, operate on different data from the pages in other tabs. A problem tab operates mostly on its problem directory in the account, and the ‘main’ tab operates mostly on everything else. Therefore, since there is at most one tab of each type, by Tab Uniqueness, most requests are independent of each other.
5. **Administrative Locking** Administrative files are those in the `admin` directory tree. Only the Login Page and User Page access most administrative files These pages both begin by getting an exclusive lock on the `admin` directory using the LOCK function in `parameters.php`.

Administrative files with other or extra considerations are:

- `admin/teams/TID/+read-write+` files: These are themselves locked by `page/index.php` and `include/epm_rw.php`.
 - `admin/users/UID/UID.info` files. These are written atomically by the User Page and read atomically by the Problem Page in order to ensure the integrity of their format.
6. **Read-Write Locking** For team member logins the `admin/teams/TID/+read-write+` file is locked at the beginning of a request. If it is determined that the account is currently read-only, this file is unlocked immediately, else it is unlocked at the end of the request.
If a read-only request attempts to become read-write, the file is re-locked and remains locked till the end of the request.
This sequences requests from read-write team logins for the same team, even if the requests are made by different team members in different sessions.
 7. **Project Problem Locking** A project problem directory is locked using the LOCK function in `parameters.php` during a push or pull involving the directory. For pushing this is an exclusive lock; for pulling it is a shared lock.

A push or pull can involve several requests: the first to compile actions and the last to execute them (there can be a request in between that simply presents information stored in the session data by the first request). If the project problem directory changes between requests, because of a push to the directory by another user, the last request could cause data inconsistency. To prevent this the exclusive LOCK time of the directory is monitored to check if some other session has exclusively locked the directory between the first and last requests of an operation. If so, the last request is aborted with an error message and does not execute.

8. **Local Problem Locking** When a file is uploaded into a problem or made from another file in the problem, a background job is executed. Similarly when a `.run` file is run, the run is a background job. The local problem directory is not modified by a background job until the job finished, at which time some files may be saved in the local problem directory.

If the problem has a parent, a shared lock is obtained on the parent using LOCK at the start of the background job, and the LOCK time is checked at the end of the job to be sure it has not changed. If it has, a push to the parent was done during the job, an error is declared for the job, and the job results are not saved in the problem local directory.

Pulls to a local problem directory can be run in the ‘main’ tab while a background job is run in the local problem directory in its problem tab. To prevent conflict, every time the problem directory is altered its `+altered+` file is touched. The modification time of this is checked to detect conflicts and abort either the execution request for a pull or the finishing of an otherwise successful background job. Local problem directories can be altered when a background job keeps files, when the Problem Page deletes files, or when the Project Page pulls to the local problem.

2.9 Security

There are two ways to breach an EPM server:

- **Session Hi-Jacking** The session is identified by the cookie which is a random number. To hi-jack a session, the hacker must intercept the cookie. A good way to protect against this is to get a certificate for the EPM server so the server uses https. As alternate protection, the `parameters.php` file contains a parameter which if set will cause the session to insist that all requests made to it come from the same IP address. This might cause problems for legitimate mobile browsers, but should prevent session hi-jacking.
- **Illegal Requesting** Since it is easy to get a user account on an EPM server, a user can try to breach the server by issuing an illegal request from a legitimate session.

Therefore each request must be checked to be sure it is legal. If not, an exit with 'UNACCEPTABLE HTTP ...' message is executed.

An EPM session is definitely not stateless. Not only is there session state, such as the current user logged into the session, but there is state in the EPM server data file system.

The Page Rule (2.5.3) and Sequence Rule (2.5.4) work together to ensure that a page POSTed to will be the same page loaded by the last GET to the tab or window doing the POST. This simplifies request checking.

Request checking is just about checking the request type and request parameters to ensure that the request is legal given the current state of the server.

3 Data Files

[xxx] means file modification time is read by xxx

| Name | Format | Description | Creators | Updaters | Readers |
|--|--------|--------------------------------------|------------|-------------|------------------|
| error.log | lines | error log p18 | (all) | (all) | |
| debug.log | lines | debugging log p19 | (all) | (all) | |
| admin | dir | administrative files | login | login user | index login user |
| admin/+blocking+ | dir | email blocking control file p21 | (editor) | (editor) | login |
| admin/motd.html | html | message of the day p22 | (editor) | (editor) | login |
| admin/+lock+ | time | administrative lock file p22 | (updaters) | login user | (updaters) |
| admin/+random+ | binary | random number data p22 | login | login index | login index |
| admin/+actions+ | lines | log of administrative actions p27 | (updaters) | user | view |
| admin/browser | dir | browser tickets | login | login | login |
| admin/browser/TICKET | 1-line | ticket info p22 | login | | login |
| admin/email | dir | email files | user | user | login user |
| admin/email/EMAIL | 1-line | email info p22 | user | login user | login user |
| admin/users admin/teams | dir | administrative user/team directories | user | user | user login |
| admin/users/UID admin/teams/TID | dir | administrative account files | user | user | user login |
| admin/users/UID/ UID.login admin/teams/TID/ UID.login | lines | log of logins p23 | (updaters) | login user | [index] |
| admin/users/UID/ UID.inactive admin/teams/TID/ UID.inactive | lines | inactive .login files p27 | user | | |
| admin/users/UID/ UID.info | json | user info p25 | user | user | user problem |
| admin/teams/TID/ TID.info | json | team info p26 | user | user | user |

| Name | Format | Description | Creators | Updaters | Readers |
|--|--------|--|------------|----------------------|----------------------|
| admin/users/UID/ +actions+ admin/teams/TID/ +actions+ | lines | log of accounts's administrative actions p27 | (updaters) | user | view |
| admin/users/UID/ manager | 1-line | teams that UID manages p27 | user | user | user |
| admin/users/UID/ member | 1-line | teams of which UID is a member p27 | user | user | user |
| admin/teams/TID/ +rw+ | UID | current read-write user p27 | +main+ | +main+ | +main+ index |
| accounts | dir | holds account subdirectories | user | user | all |
| accounts/AID | dir | account subdirectory | user | problem project | all |
| accounts/AID/ +lists+ | dir | holds account problem lists | list | list favorites | +main+ view |
| accounts/AID/ +actions+ | lines | log of account problem related actions | (updaters) | project run | view |
| accounts/AID/ +read-write+ | lines | log of account read-write mode requests p19 | index | index | [+main+] |
| accounts/AID/ +read-only+ | lines | log of account read-only mode requests p19 | index | index | [+main+] |
| accounts/AID/ PROBLEM | dir | account problem directory | project | +problem+ project | +problem+ project |
| accounts/AID/ PROBLEM/ +actions+ | lines | log of problem related actions | (updaters) | project run | view |
| accounts/AID/ PROBLEM/ +altered+ | empty | alteration indicator p?? | (updaters) | problem run | [updaters] |
| accounts/AID/ PROBLEM/ +changes+ | lines | log of changes made by pulls | project | project | |

| Name | Format | Description | Creators | Updaters | Readers |
|--|---------|----------------------------------|----------------|----------------|--|
| accounts/AID/ PROBLEM/ +work+ | dir | working directory for jobs | problem run | problem run | problem run |
| accounts/AID/ PROBLEM/ +run+ | dir | working directory for runs | run | run | run |
| accounts/AID/ PROBLEM/ ... | various | files visible to users | +problem+ | +problem+ | +problem+ |
| projects | dir | p?? | login | maint | |
| lists | dir | links to published lists | list | list | favorites list project view manage |
| default | dir | default program binaries | setup | | |
| +web-save+ | dir | backup of W | backup | backup | backup |
| +web+ | link | link to W | setup | | |

setup is setup function of epm/bin/epm
 backup is backup function of epm/bin/epm

4 Session Variables

| Name | Description | Creators | Updaters | Readers |
|--------------------------|----------------------------|---------------|----------|-------------------------|
| EPM_EMAIL | login email | login | | all pages |
| EPM_AID | account ID | login user | | all pages |
| EPM_UID | user ID | login user | | login user manage |
| EPM_IS_TEAM | true iff AID is team ID | login user | | index |
| EPM_PAGE[id_type] | current session page | index | index | index |
| EPM_IPADDR | session IP address | login | | index login user |
| EPM_TIME | session time | login | | index login user |
| EPM_ID_GEN | \$ID generation | index | index | index |
| EPM_ABORT | session abort info | login user | | index |

| Name | Description | Creators | Updaters | Readers |
|-----------------------------|--|--|------------|------------|
| EPM_MAIN | temporary data for current +main+ page | project user list favorites manage | (creators) | (creators) |
| EPM_PROJECT | permanent data for project | project | project | project |
| EPM_USER | permanent data for user | user | user | user |
| EPM_MANAGE | permanent data for manage | manage | manage | manage |
| EPM_VIEW | permanent data for view | view | view | view |
| EPM_PROBLEM[problem] | permanent data for problem | problem | problem | problem |
| EPM_WORK[problem] | data for current background task | problem | problem | problem |
| EPM_RUN[problem] | data for current background run | run | run | run |

5 Web Pages

5.1 Index Page

The Index Page is required by every other EPM .php page does initial setup for all EPM .php pages. It also functions as the initial file for accessing the EPM server and reroutes these accesses to the login page.

Index Page Requires

```
include/parameters.php
include/epm_abort.php    only if aborting
include/epm_random.php   only for 'GET's to pages setting $epm_ID_init
```

Index Page Files

```
error.log                create  append  -
debug.log                -      -      -
admin/teams/AID/+rw+     create  lock    read
accounts/AID/+read-write+ create  append  -
accounts/AID/+read-only+ create  append  -
```

Index Page Session Data

```
EPM_IPADDR              -      -      read
EPM_AID                  -      -      read
EPM_UID                  -      -      read
EPM_EMAIL                -      -      read
EPM_IS_TEAM              -      -      read
EPM_ABORT                -      -      read
EPM_TIME                 -      -      read
EPM_PAGE[$id_type*]     create  update  read
EPM_ID_GEN[$id_type*]   create  update  read
```

* \$id_type is "+main+" for main tab page,
PROBLEM name for problem tab page, and
"+view+" for view window page

Index Page Global Data

The following are global variables set just before index.php is required by another page.

| | | | |
|------------------------|---|------------|--|
| \$epm_page_type | One of: | +main+ | main tab page |
| | | +problem+ | some problem tab page |
| | | +view+ | view window page |
| | | +no-post+ | view window page that does no POSTs |
| | | +download+ | page that just downloads a file |
| \$epm_ID_init | If set re-initializes \$ID generator; see \$ID below. | | |

The following are global variables defined by index.php when it is required at the beginning of an EPM .php page, and usable by the remainder of that page. For other such parameters, see the include/parameters.php file.

| | |
|---------------------|---|
| \$epm_method | the request method, either 'GET' or 'POST' |
| \$epm_root | ROOT and SELF, where the URL used to access a page |
| \$epm_self | is HOST/ROOT/SELF and SELF either has the form /page/... or the form /index.php |
| \$epm_web | the EPM web directory W (2.6.1) |
| \$rw | true if request is being processed in read-write mode; false if in read-only mode |
| \$aid | AID (\$_SESSION['EPM_AID']) if set |
| \$uid | UID (\$_SESSION['EPM_UID']) if set |
| \$lname | login name, either AID if AID == UID, or AID:EMAIL if AID != UID |
| \$is_team | true iff AID is a team ID so login is a team member login (\$_SESSION['EPM_IS_TEAM']) |
| \$ID | the identifier which must be presented by the next re- quest (as ?id=\$ID) unless the next page requested sets \$epm_ID_init; see EPM_ID_GEN on p19 |
| \$data | same as \$_SESSION['EPM_PAGE'][\$id_type]; used for per tab or view window data: see p19 |
| \$state | same as \$data['STATE']; set to 'normal' by index.php on 'GET' |

5.1.1 Index Page File Formats

error.log:

Records all PHP error messages that would normally be in the HTTP server log, including messages generated by the ERROR and WARN functions (see index.php).

Contains lines of format: CLASS ERRNO SELF AID (TIME)

followed by a stack trace. Here

| | |
|-------|--|
| CLASS | {USER,EPM,SYSTEM}_{WARNING,NOTICE,ERROR} |
| ERRNO | PHP error number |
| SELF | page name relative to W (EPM_SELF) |
| AID | account ID, or EMAIL if account ID not available |
| TIME | session time (EPM_TIME), if available |

debug.log:

Contains lines output by the DEBUG function in parameters.php. Not actually specific to index.php or any page. See \$epm_debug and the DEBUG function in parameters.php.

admin/teams/AID/+rw+: See p27

accounts/AID/+read-only+:

accounts/AID/+read-write+:

One line is appended at the end of each http request, of the form:

BEGIN-TIME END-TIME SELF AID UID

where

| | |
|------------|--|
| BEGIN-TIME | request processing start time in seconds |
| END-TIME | request processing end time in seconds |
| SELF | page name relative to W (EPM_SELF) |
| AID | account ID |
| UID | user ID |

Times are typically to the nearest microsecond.

The +read-write+ file is written if the request is read-write at its end, or the +read-only+ file is written if the request is read-only. The modification time of the +read-write+ file is used to determine the last time the account made a read-write request.

Download and xhttp requests are not recorded.

5.1.2 Index Page Session Variables

In the following \$id_type is \$epm_page_type when the latter is '+main+' or '+view+' and the problem name when the latter is '+problem+'.

EPM_PAGE[\$id_type]:

Data specific to a particular tab or to the view window. Same as \$data global variable. Re-initialized on a 'GET' by index.php to:

['SELF' => \$epm_self, 'STATE' => 'normal']

Checked by index.php for 'POST's to be sure they target the page of the last 'GET' for the given \$id_type.

EPM_ID_GEN[\$id_type]:

The list [VALUE, KEY, IV] used to generate \$ID values. The next \$ID value is VALUE. When it is set, a new VALUE is generated by encrypting the old VALUE by KEY with IV as the initialization vector (it is always 0 and is included here as an optimization). Re-initialized for pages that set \$epm_ID_init; checked for other pages of \$epm_page_type "+main", "+problem+", or "+view+".

For session variables just read by index.php, see other EPM pages.

5.1.3 Index Page Transactions

See Page Initialization (2.7.3).

5.2 Login Page

Login Page Requires

page/index.php
include/epm_random.php

Login Page Files

| | | | |
|---------------------------|--------|--------|------|
| admin/+blocking+ | - | - | read |
| admin/motd.html | - | - | read |
| admin/+lock+ | create | update | read |
| admin/+random+ | create | update | read |
| admin/browser/TICKET | create | - | read |
| admin/email/EMAIL | - | update | read |
| admin/users/UID/UID.login | - | append | stat |
| admin/users/UID/GID.login | - | append | stat |
| admin/teams/TID/MID.login | - | append | stat |

Login Page Session Data

| | | | |
|-------------|--------|---|------|
| EPM_EMAIL | create | - | - |
| EPM_AID | create | - | read |
| EPM_UID | create | - | - |
| EPM_IS_TEAM | create | - | - |
| EPM_IPADDR | create | - | read |
| EPM_TIME | create | - | read |
| EPM_ABORT | create | - | - |

5.2.1 Login Page File Formats

admin/+blocking+:

Lines of format: SIGN RE

SIGN + to not block, - to block

RE regular expression matched to the entire email name
(e.g., .* matches any email name and .*\.edu matches
any email name ending in .edu)

- The lines are read in order and the first line with RE matching the login name EMAIL is used to not block or block the EMAIL. If no line matches, the EMAIL is not blocked.
- Blank lines and whose first non-whitespace character is # are ignored. Various forms of within-line whitespace are equivalent, and whitespace at beginning or end of a line is ignored.

admin/motd.html:

An HTML file that is included inside a `<div>...</div>` block that gives the ‘message of the day’ on the Login Page. Typically this file consists of some `<p>` paragraphs. If the file does not exist, the `<div>...</div>` block is not created.

admin/+lock+:

All transactions within the **admin** directory (i.e., all http requests that access files or subdirectories within **admin**) begin by calling the **parameters.php** LOCK function to lock the **admin** directory. This function locks the directory by creating if necessary and locking this **+lock+** file for the course of the transaction. Note there are no EPM transactions longer than a single http request.

admin/+random+:

The pseudo-random number generator in **include/epm_random.php** exclusively creates, updates, and reads this file.

admin/browser/TICKET (ticket file): T AID EMAIL

TICKET ticket proper; 32 hexadecimal digit ticket number
 T ticket type; ‘c’ for confirmation number; ‘a’ for automatic
 AID account ID:
 team ID (TID) if ticket is for team member login
 user ID (UID) if ticket is for guest login
 ‘-’ if ticket is for user login
 EMAIL Email address (identifying user account)

- When a user initially logs in to create an account, the UID is not known when the ticket is created.

admin/email/EMAIL (regular email file): UID ACOUNT ATIME

EMAIL Email address encoded with PHP rawurlencode
 UID user ID
 ACOUNT Number of auto-login periods completed so far.
 ATIME Start time of newest (incomplete) auto-login period.

admin/email/EMAIL (pre-login email file): - TID ...

EMAIL Email address encoded with PHP rawurlencode
 TID Team user ID (may be more than one)

- This form of email file is created by the User Page when a team member is assigned the given EMAIL before the member has an account or EMAIL has been added to an existing account. The TID’s list all the team IDs that might in their **TID.info** file have a member which has this EMAIL and no UID. A TID might be listed whose **TID.info** file no longer contains the EMAIL.

When the pre-login form is converted to a regular form, the list of TID's is used to convert any matching EMAIL members in `TID.info` files to `UID(EMAIL)` members.

admin/users/UID/UID.login (login log):

admin/teams/TID/UID.login (login log):

admin/users/UID/GID.login (login log):

Lines of format: TIME EMAIL IPADDR BROWSER

UID User ID

TID Team ID

GID Guest User ID

TIME Session time for login (EPM_TIME)

EMAIL Email address used for login (EPM_EMAIL)

IPADDR IP address for session (EPM_IPADDR)

BROWSER \$_SERVER['HTTP_USER_AGENT'] with '(...)'s
removed and horizontal spaces replaced by ';'s

- A login with name AID:EMAIL is valid iff the file
.../AID/UID.login exists for UID the user ID associated with
EMAIL.
- Upon login, a line is appended to the appropriate the .login file,
and then that file's name and modification time are stored in
EPM_ABORT and used to abort a session if another session logs in
with the same AID:EMAIL and appends to the file, thus changing
its modification time.

5.2.2 Login Page Session Variables

| | |
|--------------------|---|
| EPM_EMAIL | EMAIL entered by user into browser; set by Login Page when either (1) EMAIL is to be transferred to user.php for a new user, or (2) browser sends TICKET which identifies EMAIL and EPM_UID is being set. |
| EPM_AID | Account ID, either user or team; set by Login Page when a valid TICKET is received, and set by User Page for new users. This equals EPM_UID for a user login, is the team ID for team member login, and is the host user ID of the EMAIL guest for a guest login. |
| EPM_UID | User ID associated with EPM_EMAIL. Set when EPM_AID is set. |
| EPM_IS_TEAM | True iff EPM_AID is team ID; Set when EPM_AID is set. |
| EPM_IPADDR | Set to \$_SERVER['REMOTE_ADDR'] by Login Page when EPM_AID is not yet set. |
| EPM_TIME | Set to \$_SERVER['REQUEST_TIME'] formatted by \$epm_format_time by Login Page if EPM_AID is not yet set. |
| EPM_ABORT | Set to [FILE,MTIME] where MTIME is the mod time of \$epm_data/FILE and the session must abort if the mod time of this file changes. Here FILE is admin/users/AID/UID.login to which a line is appended whenever EPM_AID is set for a session. |

5.2.3 Login Page Transactions

1. If regular form admin/emails/EMAIL exists log existing user in and go to Project Page. The browser first gets a ticket which it sends to the server, and the ticket specifies the EMAIL.
 - (a) Browser can look EMAIL up in browser's local memory to get ticket to send to server, or if this ticket does not exist or is invalid,
 - (b) Browser can send EMAIL to server and get confirmation number back to use as a ticket.
2. Otherwise, if no regular form admin/emails/EMAIL exists, give the browser a confirmation number to use as ticket, and upon receiving this set EPM_EMAIL and give the browser a new automatic ticket and instruct the browser to go to User Page to create new user.

5.3 User Page

User Page Files

| | | | |
|------------------------------|--------|--------|------|
| admin/email/EMAIL | create | update | read |
| admin/users/UID/UID.info | create | update | read |
| admin/teams/TID/TID.info | create | update | read |
| admin/users/UID/UID.login | - | append | stat |
| admin/users/UID/GID.login | - | append | stat |
| admin/teams/TID/UID.login | - | append | stat |
| admin/users/UID/UID.inactive | create | - | - |
| admin/users/UID/GID.inactive | create | - | - |
| admin/teams/TID/UID.inactive | create | - | - |
| admin/users/UID/manager | create | update | read |
| admin/users/UID/member | create | update | read |
| admin/teams/TID/+rw+ | create | update | read |
| admin/users/UID/+actions+ | create | append | - |
| admin/teams/TID/+actions+ | create | append | - |
| admin/+actions+ | create | append | - |

User Page Session Data

| | | | |
|-------------|--------|--------|------|
| EPM_USER | create | update | read |
| EPM_DATA | create | update | read |
| EPM_EMAIL | - | - | read |
| EPM_AID | create | - | read |
| EPM_UID | create | - | read |
| EPM_IS_TEAM | create | - | read |
| EPM_IPADDR | - | - | read |
| EPM_TIME | - | - | read |
| EPM_ABORT | create | - | - |

5.3.1 User Page File Formats

admin/email/EMAIL: see p22

admin/users/UID/UID.info (user info file):

JSON file with the following components:

| | |
|----------------|---|
| 'uid' | UID |
| 'emails' | [EMAIL { , EMAIL } [*]] |
| 'guests' | [GID { , GID } [*]] (may be missing) |
| 'full_name' | TEXT |
| 'organization' | TEXT |
| 'location' | TEXT |

where

UID user ID (i.e., an account ID) for user; cannot be changed
 once account is created
EMAIL e-mail address for user
GID UID for guest of user
TEXT plain text (with a minimum and maximum allowed
 length)

- When a team UID.info file is created, MIDs are specified as EMAILs which are resolved if possible to PIDs.
- When a person initially creates an account, all UID.info files are searched and if any have MIDs matching the new account EMAIL, they are resolved to PIDs.

admin/users/TID/TID.info (user info file):

JSON file with the following components:

'tid' TID
'manager' MANAGER
'members' [MEMBER { , MEMBER }^{*}] (may be missing)
'full_name' TEXT
'organization' TEXT
'location' TEXT

where

TID team ID (i.e., an account ID) for team; cannot be changed
 once account is created
MANAGER UID of the manager of team
MEMBER one of: MID
 (EMAIL)
 MID(EMAIL)
MID UID of member of team
EMAIL EMAIL of member of team as of time member was added
 to team
TEXT plain text (with a minimum and maximum allowed
 length)

- A MEMBER may be specified as an EMAIL or a MID. If specified as an EMAIL, and a regular **admin/email/EMAIL** (p22) exists, the MID is added. If specified as an EMAIL, and no regular **admin/email/EMAIL** file exists, the TID is added to a pre-login **admin/email/EMAIL** (p22), which is created if it does not exist.
- When a user initially creates an account with an EMAIL for which a pre-login **admin/email/EMAIL** file exists, the TID.info files for all TIDs listed in the pre-login file are searched for any MEMBERS of the form '(EMAIL)', and when one is found, its MID is added to it.

Similarly if EMAIL is added to an existing user account.

admin/users/UID/UID.login (login log):

admin/teams/TID/UID.login (login log):

admin/users/UID/GID.login (login log): see p23

admin/teams/TID/UID.inactive:

admin/users/UID/GID.inactive:

Inactive .login file, made by renaming .login file when UID is no longer a member of TID team or GID is no longer a guest of UID. May be reactivated by renaming to .login file.

admin/users/UID/manager:

A a list of single space separated TIDs of the teams of which user UID is a manager.

admin/users/UID/member:

A a list of single space separated TIDs of the teams of which user UID is a member.

admin/teams/TID/+rw+:

Either a single UID of the team member whose login currently has read-write mode, or blank if no such. This file is locked by itself for exclusive use by team member login requests, and this is independent of any +lock+ file locking. The lock is released immediately for read-only requests, but is held to the end of read-write requests.

admin/users/UID/+actions+:

admin/teams/TID/+actions+:

Lines of format: TIME AID info KEY OP VALUE

TIME Session time for login (EPM_TIME)

AID equals UID or TID from file name

KEY .info file JSON key

OP = if non-list KEY reset, + if addition to KEY's list, - if deletion from KEY's list

VALUE value given to non-list KEY, added to KEY's list, or deleted from KEY's list

- Updates to AID.info file are logged by writting lines to admin/*/AID/+actions+ file.

admin/+actions+:

Any line writted to an admin/*/AID/+actions+ file is also written to this file.

5.3.2 User Page Transactions

1. If EPM_UID not set, get data for new user and create new user account if data acceptable. Otherwise, or after creating new user account, display .info data for all

users and all teams.

2. Allow the current user to edit their own `.info` data.
3. If the current user is the manager of a team, allow that team's `.info` data to be edited.
4. Allow the current user to create a new team of which the current user is a manager.
5. NOTE: team member and guest logins cannot edit user or team `.info` or create new teams.
6. Allow a current read-only user to force a switch to read-write.

5.4 Problem Page

Problem Page Visible Files

| | | | | |
|--|--------|--------|------|--------|
| <code>accounts/AID/PROBLEM</code> | create | update | read | delete |
| <code>accounts/AID/PROBLEM/PROBLEM.tex</code> | upload | - | read | delete |
| <code>accounts/AID/PROBLEM/PROBLEM.pdf</code> | create | - | read | delete |
| <code>accounts/AID/PROBLEM/PROBLEM.c</code> | upload | - | read | delete |
| <code>accounts/AID/PROBLEM/PROBLEM.cc</code> | upload | - | read | delete |
| <code>accounts/AID/PROBLEM/PROBLEM.java</code> | upload | - | read | delete |
| <code>accounts/AID/PROBLEM/PROBLEM.py</code> | upload | - | read | delete |
| <code>accounts/AID/PROBLEM/PROBLEM</code> | create | - | read | delete |
| <code>accounts/AID/PROBLEM/PROBLEM.jar</code> | create | - | read | delete |
| <code>accounts/AID/PROBLEM/PROBLEM.pyc</code> | create | - | read | delete |
| <code>accounts/AID/PROBLEM/YYYY-PROBLEM.c</code> | upload | - | read | delete |
| <code>accounts/AID/PROBLEM/YYYY-PROBLEM.cc</code> | upload | - | read | delete |
| <code>accounts/AID/PROBLEM/YYYY-PROBLEM.java</code> | upload | - | read | delete |
| <code>accounts/AID/PROBLEM/YYYY-PROBLEM.py</code> | upload | - | read | delete |
| <code>accounts/AID/PROBLEM/YYYY-PROBLEM</code> | create | - | read | delete |
| <code>accounts/AID/PROBLEM/YYYY-PROBLEM.jar</code> | create | - | read | delete |
| <code>accounts/AID/PROBLEM/YYYY-PROBLEM.pyc</code> | create | - | read | delete |
| <code>accounts/AID/PROBLEM/XXXX-PROBLEM.in</code> | upload | - | read | delete |
| <code>accounts/AID/PROBLEM/XXXX-PROBLEM.sin</code> | create | - | read | delete |
| <code>accounts/AID/PROBLEM/XXXX-PROBLEM.sout</code> | create | - | read | delete |
| <code>accounts/AID/PROBLEM/XXXX-PROBLEM.fout</code> | create | - | read | delete |
| <code>accounts/AID/PROBLEM/XXXX-PROBLEM.ftest</code> | create | - | read | delete |
| <code>accounts/AID/PROBLEM/XXXX-PROBLEM.dout</code> | create | - | read | delete |
| <code>accounts/AID/PROBLEM/XXXX-PROBLEM.score</code> | create | - | read | delete |
| <code>accounts/AID/PROBLEM/ZZZZ-PROBLEM.run</code> | - | - | read | - |

Special values for YYYY: **generate**, **filter**, **monitor**, **display**

Problem Page Parent Files

| | | | | |
|---|---|---|------|---|
| projects/PROJECT/PROBLEM | - | - | read | - |
| projects/PROJECT/PROBLEM/PROBLEM.pdf | - | - | read | - |
| projects/PROJECT/PROBLEM/PROBLEM.optn | - | - | read | - |
| projects/PROJECT/PROBLEM/generate-PROBLEM | - | - | read | - |
| projects/PROJECT/PROBLEM/filter-PROBLEM | - | - | read | - |
| projects/PROJECT/PROBLEM/display-PROBLEM | - | - | read | - |
| projects/PROJECT/PROBLEM/monitor-PROBLEM | - | - | read | - |
| projects/PROJECT/PROBLEM/XXXX-PROBLEM.in | - | - | read | - |
| projects/PROJECT/PROBLEM/XXXX-PROBLEM.ftest | - | - | read | - |
| projects/PROJECT/PROBLEM/ZZZZ-PROBLEM.run | - | - | read | - |

Problem Page Maintenance Files

| | | | | |
|-----------------------------------|---|---|------|--------|
| accounts/AID/PROBLEM/+parent+ | - | - | read | delete |
| accounts/AID/PROBLEM/PROBLEM.optn | - | - | read | delete |

Problem Page Session Data

| | | | |
|-------------|--------|--------|------|
| EPM_EMAIL | - | - | read |
| EPM_UID | - | - | read |
| EPM_PROBLEM | create | update | read |