Teaching Office Database Documentation: Installation & Configuration Guide

Table of Contents

1 Introduction

This document is intended to guide the user in installing the TODB software on a web server. The general assumption is that this software will be installed on an internet- or intranet-facing server. However, it is also understood that it might be evaluated on a desktop computer that has been set up to run web server software (i.e. a desktop test environment). This documentation aims to be sufficient for both types of application.

Each broad step in this process has a time indication attached to assist with budgeting time for this installation. These assume that nothing goes wrong.

Should you encounter a problem, please refer to the Troubleshooting section at the end of this document. It is suggested that after reading this document, the checklist be followed (in order) to ensure that a basic running configuration is set up.

2   System Requirements

The TODB software is not demanding. It should run successfully on a relatively low-spec machine. It does require certain software to be present on the machine. These are:

1.   A web server. The software has been tested with Apache.
2.   PHP 4.4+ (PHP 5+ preferred) installed and running on the server and configured to run when PHP files are requested via HTTP. PHP must be configured with the **MySQL extension** (not / not just MySQLi) enabled.
3.   MySQL 5.0.18+ server and a command line client program

The MySQL and PHP versions mentioned here are the oldest distributions that have been tested. It is quite possible that the TODB will function partially or wholly with older versions of these packages.

The PHP installation can be fairly standard, as the bulk of the software functions used are part of the core PHP feature set. The only additional requirement is the MySQL extension. If you do not already have Apache/MySQL/PHP installed, there are several 'LAMP/WAMP' type of packages available on the web from which the whole lot can be downloaded e.g. http://www.wampserver.com/en/ for a Windows installation.

The TODB software has been installed successfully on Windows, Unix-based machines and Macintosh computers. Please refer to Raven Authentication and Macs in University of Cambridge in the Troubleshooting section for advice on using Macs to run TODB with the University of Cambridge.

The remainder of this document assumes that an Apache/PHP/MySQL server has been installed and is functioning correctly.

3   Distribution

The software is provided as an archive in zip or tar file forms. This archive will be referred to as <AllFiles>.tar and the content names may well differ depending on the packaging process – typical names are given below and the appropriate name should be substituted throughout the procedures in this document. The <dept> names are used for existing user departments and should be ignored if you are installing the Generic version of TODB.

Reference Name	Example Names	Comment
<Code>.tar	TODB_<version>.tar e.g. `TODB_1_3g.tar` <dept>_todb_<date>.tar e.g. `biochem_todb_20100308.tar`	A number of php and inc files in a directory structure. These files need only be extracted into a web-accessible directory.
<Create Tables>.sql	Create_TODB_tables.sql <dept>_todb_<date>.sql e.g.	An SQL file that creates the database tables – now supplied as part of the code.

	biochem_todb_20100308.sql	
<Create Users>.sql	Create_TODB_users.sql mysql_<date>.sql e.g. mysql_20100308.sql	An SQL file that creates database users – now supplied as part of the code

4   Authentication

The software was designed to use the RAVEN authentication system in use at the University of Cambridge. This is provided by an Apache module, and sets an environment variable 'REMOTE_USER', which is accessed by the TODB's PHP code via the $_SERVER global variable. This is then used internally by the TODB software to determine whether or not the user can edit or just view the TODB pages. Read-only access is granted in the Apache configuration (require, allow and deny operators – see http://httpd.apache.org/docs/1.3/howto/auth.html#allowdeny).

For users without RAVEN authentication installed, Basic or Digest Apache authentication will also set the appropriate PHP script variable and the software will operate correctly.

For testing purposes: if the server and client have the IP address 127.0.0.1 ('localhost', the computer's IP address for itself), the system assumes a default username, which can be changed in config.inc, and no authentication is necessary.

5   Installation of PHP Code Files (10 minutes)

First decide where to install the files. This can be any directory. Some examples:

- On Linux, this could be /srv/www/htdocs/TODB

- On Windows, perhaps C:\www\TODB

- On Macs, perhaps /Library/WebServer/Documents/TODB/

5.1 Windows:

- Unzip the files into the directory you have chosen using WinZip or a similar ZIP application.

Note: the tar distribution can also be used under Windows; you will need to download and install 'UnxUtils' (http://unxutils.sourceforge.net/) which provides many useful Unix/Linux commands for use on Windows. You will need to add to your system path; search for "Unxutils Installing" for more help.

Depending on how the software has been packaged, you may also need to unzip the source code as a second operation

5.2 Linux:

The instructions below are for a generalised versions of TODB file names (see above Distribution) with an example shown. The names of the tar files may be different for a specific department's installation.

1. Change to the desired directory (e.g. `cd /srv/www/htdocs/TODB`)

2. Enter the following command which will unpack the Documents, SQL Scripts and a further tar file containing the source code: **`tar –xvf <AllFiles>.tar`**

    e.g. `tar –xvf TODB_allfiles.tar`

3. Enter the following command which will extract the source code:

    `tar –xvf <Code>.tar`

    e.g. `tar –xvf TODB_1_3g.tar`

    The code is thus installed. It will need to be configured before it can be meaningfully used.

5.3 Testing

Please direct the browser at `http://<hostname>/<install_dir>/test_html.html`, where:

`Hostname` is the hostname of the server on which the software has been installed. If this is the local server, this will be `127.0.0.1` or `localhost`. `Install_dir` is the installation directory under the Apache document root on the server under which the TODB is installed.

- If you can access the test_html.html page, it indicates that the files should have been installed properly.

- Now click on the 'Test PHP Installation' link. TODB will not work at all if PHP is not working on the server. Note: it might be necessary to take a look at the Apache configuration described in the User Configuration section later on in this document.

6 Database Creation (20 minutes)

Choose from Automatic or Manual Database creation:

6.1 Automatic Database Creation

This procedure will create the database and empty TODB tables and also create the code that allows TODB to connect to this database. The script SQL/create_TODB_tables.sql is used to create the tables and can be edited before running this process should you wish to change anything (do not use Notepad as an editor for this as it can cause problems).

Direct the browser at `http://<hostname>/<install_dir>/dbsetup.php`

You will need to enter the following:

Root password: Enter the root password for MySQL

Database Name: This will be the name of your database and should not match any existing database.

Read Password: This password will be used for read access to the TODB database

Write Password: This password will be used for write access to the TODB database.

Click on '`Submit Query`'. All being well, you will see lots of friendly messages, in the following example the database name entered was '`todb`' and the default passwords were used (not recommended). If things have not worked out, you will need to work out what has and hasn't worked and use Manual Database Setup to complete the process.

```
Database created todb

Read user created todb_read

Read user privileges granted GRANT SELECT, LOCK TABLES ON todb.* TO
'todb_read'@'localhost' IDENTIFIED BY 'bobread'

Write user created todb_write

Write user privileges granted GRANT ALL PRIVILEGES ON todb.* TO
'todb_write'@'localhost' IDENTIFIED BY 'bobwrite'

Database tables created

config\db.inc file successfully created
```

Now continue from Database Testing to check the database tables have been created as expected.

6.2 Manual Database Setup

You do not need to follow these steps if you have successfully done Automatic Database creation.

Please follow these steps (all commands in `courier font` are to be typed in and the <ENTER> button pressed afterwards. The quotation marks are not to be entered.   <Database> indicates that you need to substitute your database name):

Note again for a given department the database name need not be TODB_1_3g

1. Open a command prompt/terminal window and change to the code installation directory (which might need to be created): (e.g. "`cd c:\www\TODB`", or "`cd /srv/www/htdocs/TODB`").

2. Log into MySQL as root (or as a user who has database creation and user creation privileges): "`mysql -u root -p`"

Note: under Linux, if your username is configured to allow root access to MySQL, the command is simply "`mysql -u root`"

3. If MySQL prompts for a password, please enter your password.

4. When logged-in, type "**create database <Database>;**"

   e.g. "`create database TODB_1_3g;`"

   If this conflicts with another database on the system, please choose another name for the database. You will need to change the name of the database in *<installation dir>*/`config/db.inc`. This will be fully explained in the configuration section.

5. Instruct MySQL to use this database: "**use <Database>;**"

   e.g. "`use TODB_1_3g;`"

6. Create the tables by running the SQL source file:

   Linux: enter "**source <SQL directory>/<Create Tables>.sql;**"
   Windows: "**source <SQL directory>\<Create Tables>.sql;**"

   e.g. "`source SQL/create_TODB_tables.sql`";

   Review the tables that should have been created: type in "**show tables;**".
   Several tables such as 'jobs_2009_10', 'divisions', 'editlocks', etc should be displayed.

7. Choose SQL usernames and passwords for the system to use. Set passwords for the readonly and write users in **<SQL directory>/<Create Users>.sql** eg `create_TODB_users.sql` in the source `SQL` directory. Set the corresponding usernames/passwords in the file `config/db.inc`.

8. Create the database users with the following command:

   Linux: enter "**source <SQL directory>/<Create Users>.sql;**"
   Windows: "**source <SQL directory>\<Create Users>.sql;**"

   e.g. "`source SQL/create_TODB_users.sql`";

9. Type in "`flush privileges`" to make sure the new users are reloaded into the MySQL database.

10. Type "`exit`" to return to command prompt.

This will have created the database, its table structures and users.

6.3 Database Testing

Test the database by pointing the browser at the TODB test page (see Testing section in 'Installation of PHP Code Files').

- Then click on the 'Test MySQL Installation' link. This checks that the server's PHP setup can access MySQL. This is also essential to the operation of TODB. Please search on Google for help here (this can be tricky on Windows).

- If this worked, click on either of the 'Test database setup (tables)' or ' Test MySQL tables' links (these point to the same test). The system will indicate whether or not the basic set of tables is available. This is not a guarantee that the schemata will be correct, but will probably be so unless there has been deliberate tampering.

   Note: you will need to set the database connection details first. Please see Database connection details .

Database tip: For those unfamiliar with MySQL, the MySQL GUI tools, particularly the 'Query Browser', can be very useful. These are downloadable here: http://dev.mysql.com/downloads/gui-tools/5.0.html. This now seems to have been replaced by the SQL Development feature of MySQL workbench 5.2 http://dev.mysql.com/downloads/workbench/5.2.html .

7   Software Configuration

This is perhaps the most difficult aspect of running the TODB. Configuration is divided into several sections. These are:

- System configuration (to ensure that the code runs properly)

- User configuration (allowing users access to the correct sections)

- Data and display configuration (setting up appropriate categories of teaching jobs, defining the look and feel of the system, choosing which database columns to display, etc)

Various additional configuration changes can be made to 'tweak' the behaviour of certain reports and so on. Beyond this, the code can be edited in a text-editor and modified if necessary. Selected reports will be covered in more detail.

7.1 System Configuration

Part of the system configuration is done during installation.

7.1.1 Database connection details

The stored details for connecting to the MySQL database are stored in the `<installation dir>/config/db.inc` file. The username/password pairs in this file match those in the SQL script for selecting the database and logging into the database in read-only or write mode.

This file is generated automatically if you follow the Automatic Database setup and you should not need to change it unless you subsequently change the database read and write passwords used by TODB. In this case you can skip to File permissions

- Any changes to the database users or their passwords made in `db.inc` (lines 10-16) need to be similarly modified on the database, by editing the **<SQL directory>/<Create Users>.sql** file and repeating steps 1-3 and 8-9 in the installation instructions above. The read-only user name is `$readuser_name` and its password is `$readuser_pass`. The write user is `$writeuser_name` and its password is `$writeuser_pass`.

- If the name of the database is changed in db.inc (the "`$database_name`" variable on line 8), the database should be recreated with the new name, modifying steps 4 and 5 of the instructions above (Section 6) to reflect the changed name of the database.

Changing the user password details is recommended if the TODB is used on a live, public-facing internet server. Otherwise there is a small danger that a hacker could use the 'default' username and password – available by downloading and viewing this open source package – to break into the TODB on the server and possibly gain access to more sensitive information on the database server.

7.1.2 Testing database connection details

Please repeat the procedure described in Database Testing to retest the MySQL connection.

7.1.3 File permissions

In general, the standard file permissions will be acceptable. On Linux, the **static_csv** directory needs to be made writeable by the web server. Either change the owner of the directory to the webserver via the command line, e.g.:

```
chown wwwrun static_csv
```

Or give more generous write permissions to the directory (this command is needed for Mac users):

```
chmod +w static_csv
```

Test the TODB's ability to write a file to this directory by clicking on the 'Test CSV output directory' link on the test home page (see 'Testing' (section 5.3)).

7.1.4 General settings

7.1.4.1 _Email recipient_

The `$email_recipient` variable in `config.inc` in the config directory is the email address to which emails generated by the system are sent. This is primarily the facility where staff members can view their own teaching allocations and use the 'send message to teaching office' utility that is available if teaching duties are incorrect. This should be set to the email address of someone responsible for teaching administration in the department.

7.1.4.2 _Institutional domain name_

Emails sent to the teaching office from TODB will feature 'reply-to' based on the user's CRSID column in the people table and the `$institutional_domain_name` domain name – so if the CRSID is

`abc123`, and `$institutional_domain_name` is `'cam.ac.uk'`, the email will reply-to `abc123@cam.ac.uk`. In future, other messaging done through the system will follow this rule as well.

7.1.4.3 *Available years and setting up flux years*

For certain places in the TODB (e.g. display of points totals per year when a particular staff member's teaching duties are listed), the system needs to know about other years of data on the database.

Years information is modified using the 'Admin: modify year configurations' link on the main page (visible/available to admin users only). This links to view_new_year.php.

It is possible to create new years of data, remove years, set flux years and set the current year, using this facility.

- The 'yearval' is used to name the tables and identifies year data to the system. The longer description (e.g. 'Oct 2009 to 2010) is used to more verbosely inform the user which year is under examination.

- If a year is set to be in flux, jobs and units pages are inaccessible to ordinary non-flux users (see Flux users). The idea behind this is that when teaching allocations are still under debate and subject to change (i.e. in a state of 'flux'), it is inappropriate for ordinary users to see these in case they plan according to what might be ultimately changed and therefore potentially incorrect.

- The correct 'current year' (the academic year in which the user finds him/herself) should be set using this facility.

7.2 User Configuration

7.2.1 Local 'test' user (localhost test scenario)

The name of the localhost test user can be set by modifying the value of **$default_local_user** variable. This is the name of the user that will be used if the system is run on a machine and accessed from itself (i.e. with the IP address 127.0.0.1).

7.2.2 External authentication mechanisms

If the TODB is to be modified to operate with a different authentication mechanism, the environment variable 'REMOTE_USER' should be set, as this is read by the TODB to determine who is accessing the system.

7.2.3 Limiting read access to users

If the system is available on the internet or an intranet but access needs to be limited to only certain authenticated and authorised users, use the following `require user` syntax in the Apache

configuration file (for users 'user1', 'user2' and 'user3'; in practice, likely to be 'smith', 'jones' and 'brown' or something similar):

```
<Directory /srv/www/htdocs/TODB/>
   XBitHack full
   order deny,allow
   deny from all
   Require user user1 user2 user3
   Satisfy all
   <Files *.inc>
      order allow,deny
      deny from all
   </Files>
</Directory>
```

The `<Directory…> location` will depend upon where you installed the TODB code. The `<Files ...></Files>` section should be added to prevent hackers from accessing the contents of `.inc` files, one of which (`db.inc`) contains database login details.

Note: under Linux, the Apache config file is often something like `/etc/apache2/default-server.conf`; under Windows, the file is stored by default in `C:\Program Files\Apache Software Foundation\Apache2.2\conf\httpd.conf`. Apache will need to be restarted after changing the config: "`apache2ctl –k graceful`" under Linux; in Windows, left-click on the Apache icon in the system tray, click on the Apache menu that appears and select 'restart'.

7.2.3.1 *Auto-generated user lists*

Apache can be configured to grant read access to the set of users that are stored in the TODB's 'current year' people table. Access is granted based on CRSID (this is how Raven, the University of Cambridge authentication system, determines access to web-based resources).

- The TODB generates a user list (default name '**TODB_Raven_Users.txt**' which can be changed in config.inc).

- This list is **generated when a user is added, updated or removed** from the People table of the 'Current Year' (check the 'Admin: Modify year configuration' link on the home page and look in the year drop-down list).

- For this list to be generated, it needs to be **writeable by the web server** – so should be given generous write permissions or be owned by *wwwrun* or whatever user Apache runs as on your system.

- The CRSIDS of all users in the People table, as well as those of the admin and flux users, are inserted into this file and associated with a **group called 'TODB_Users'**.

- **Apache is configured** to grant access to members of this group only:

  ```
  <Directory /srv/www/htdocs/todb/>
  ...
      AuthGroupFile /srv/www/htdocs/todb/TODB_Raven_Users.txt
  ```

```
        Require group TODB_users
    </Directory>
```

Apache will need to be restarted if this change is made post installation.

7.2.4 Limiting write access to users

If a user is granted read access in the Apache configuration file and not added to the '$allowedusers' array in the config.inc file, they will not have write/edit privileges.

7.2.5 Granting admin/write access to users

If a user is to be granted access to the system to add/edit/delete data (admin user) , they will need to be added to the $allowedusers array in the config.inc file. The name entered will be the same as the name by which they are authenticated (i.e. 'test', in the case of the localhost test setup, or what REMOTE_USER contains in an external authentication scenario). At Cambridge, using Raven authentication, these will be CRSIDs.

Note: because the TODB_Raven_Users.txt is only updating when someone in the People table is edited, you will need to force an edit on the People table for the current year to make sure any new admin users get updated into TODB_Raven_Users.txt.

7.2.6 Allowing Admin User to view as a 'normal user'

For acceptance test purposes, it is useful for an admin user to be able to see what a 'normal' user would see. A switch is provided in config.inc for this purpose. Use the following setting to do this: $usertestmode = 'TRUE'; When set "View screen as a normal user ?: " will appear under the HOME button line. Check this then click the next 'operation' button (e.g. Update, Download as CSV, Filter) to refresh the screen. When the 'operation' button is pressed again the user will return to the admin display.

In normal operation $usertestmode should remain unset.

7.2.7 Flux users

So-called 'flux' users can be added to the $fluxusers array in the config.inc file. These users can view teaching data when the data are (marked as being) still subject to change. This is discussed in the Users Guide.

7.3 'Look and feel' Configuration

The appearance of the TODB site is determined to a large extent by CSS stylesheets. Fragments of HTML are also inserted into certain parts of the pages that are displayed. All of these can be modified with relative ease.

Display that affects every page:

- CSS is edited in the 'header.inc' in the config directory, and CSS files can be referenced in top.inc.

- The page heading – the images that are displayed, institution logo, etc – is edited in top.inc.

- HTML elements that appear at the bottom of every page are defined in footer.inc.

- The site name, page titles, description, department, etc are set in config.inc in the config directory. Specifically, $titletext, $description, $department and $officename should be set.

The 'look & feel' as set up by default links to Cambridge University stylesheets and is appropriate for a CU department or faculty. It is recommended that other universities link to or examine their institutional stylesheets.

7.3.1 Index page:

The index page is very important, as it provides links to every 'function' that the TODB software provides. It is also the landing page for users accessing the system, so needs to provide enough information for new users to know what to do. This page has also been used in the past for announcements related to the system.

Each 'function' or operation that the TODB provides is organised around tables that contain information. The 'View Jobs' function, for example, provides access to the jobs table, with some functionality for filtering and querying this information. The 'View People' function provides similar functionality for the people table. Each of these accesses a year version of a table – so the 'View People (08/09)' link accesses the people_2008_09 table. The year is specified as the 'yearval' GET variable. Links therefore look like this:

```
<a href="view_people.php?yearval=2008_09">View people (08/09)</a>
```

7.3.2 Configuring the index page:

Some of the information presented to users in the index page is defined in index.inc in the config directory.

The remainder of the information is in config.inc, in the config directory. The variables $index_welcome and $timetable_message are specific to the index page.

7.4 Data and Display Configuration

This is the configuration of what the user sees on-screen. There are a number of components which can be configured, and these fall into two categories:

1. How data is structured

2. How the pages look, cosmetically

The structuring of data to reflect the structure of teaching in a University department or faculty is perhaps the most challenging aspect of the configuration of this software. Installing the software and tweaking the appearance are merely technical challenges that can be solved with perseverance and Google.

7.4.1 Structuring of jobs data

The jobs table contains information necessary for effectively describing each job. This document talks about 'the jobs table'. In actual fact, there are several jobs tables, one for each academic year. They are named as follows: jobs_2008_09, jobs_2009_10, etc.

The structure of a jobs table is as follows:

Field name	Description	Data type
id	Row id	int(11)
course	Name of the course	varchar(32)
year	The year within the course for which this job is done	int(11)
paper	The paper/teaching unit/module to which this job belongs	varchar(32)
prgroup	The subject group divisions with which this job can be associated	varchar(32)
name	The name/description of the job	text
type	The type of work this job represents (lecturing, giving a seminar, etc)	varchar(32)
hours	How many hours it takes to complete this job	decimal(5,1)
term	The academic term in which the job is done	varchar(32)
venue	Where this job is given (if appropriate)	varchar(64)
timeslots	When this job is done (If the timetable output is used, this should be in the form Wk*w*: *d.t*, e.g. Wk7: Tu.11.30)	varchar(255)
uname	The name of the person doing this job	varchar(32)
points	The number of stint points earned doing this job	int(11)
note	Notes	text
deleted	Has this job been marked as deleted?	tinyint(1)
updatetime	When was this job last updated?	timestamp
formula_ref	Reference to a points calculation formula	int(11)

Columns of the jobs table (equivalent to what is viewable via the 'View Jobs' link in TODB) contain information that is either readily categorised or not. An easy way to identify these is to ask if what should be stored in the column needs to be picked from a (relatively short) list of some kind, or if a more general value ought to be entered.

'Category' columns include course, job type, term, year and so on. Non-category columns are the job name, the name of the person doing the job, the paper name/code, notes, that sort of thing. Easily-categorised columns can be identified to the system by editing the config.inc file. This instructs the software to display these categories such that the user can choose to filter the data display on-screen by category; so, for example, the user can choose to view only lecturing jobs (job type 'L' selected), or just jobs that are done in the Michaelmas and Lent terms (term is 'M' and 'L').

While it is not essential to categorise jobs in any way, the analyses that can be performed on the data to ease and improve the day-to-day administration of the department will be much richer if jobs are categorised. A job is categorised by assigning particular category values to certain fields that represent the job. Job type is particularly useful: by indicating what kind of job activity each job represents, it becomes possible to see how a person's time is divided between these activities. This could be extremely useful for keeping track of how much time is spent on administration rather than lecturing activities, for example. One of the standard reports (View Stint Summary) provided with the TODB provides an effort break-down by job type, per person.

The following is a step-by-step guide for deciding how to represent teaching activities within the TODB and then implement these via the software configuration.

7.4.1.1 *Subject group divisions*

These are subject areas or disciplines within the department or faculty.   Each piece of teaching work done is represented as a teaching 'job'. Each job is done by a single person (who ought to be defined in the People table), and which has some other attribute information associated with it. This includes a subject division, the course of which this job is a part, the type of work this job represents, and so on. If a job is allocated to one or more subject group divisions, it becomes possible to see how teaching effort is distributed across disciplines.

Some examples at the University of Cambridge:

- Divinity Faculty: 'Old Testament', 'New Testament', 'Philosophy of Religion', 'Religious Studies', etc.

- Chemistry department: 'Organic', 'Inorganic', 'Theoretical', 'Physical'

- Engineering department: Groups A-F, representing 'Thermodynamics and Fluids', 'Mechanics', 'Electronics', etc.

- English: 'Commonwealth', 'Victorian', etc.

Each subject group division is assigned a short alphanumeric code. Single letters are recommended, but these can be up to three characters in length. In Divinity, for example, Old Testament is represented with the code 'OT'. It may help the users of the system (who will use codes rather than the full names of subject areas in day-to-day use) for these division codes to be somehow meaningful (having the same start letter as the subject group name, for example).

Having decided on a set of subject group divisions and corresponding short codes, these must be entered into the configuration:

1. Open config.inc in a text editor.

   Note: Avoid using Notepad for this as it can cause problems with line-endings (Wordpad is OK).

2. Modify the following line (group codes):
   ```
   $all_categories = array('year' => array('1','2','3','4'),
                           'term' => array('M','L','E'),
   ```

```
                                'type' => array('L','C','E','A', 'P',
                                'D'),
                                'group' => array('I','O','P','T', 'X')
                                );
```
Change these items to the group codes you have decided on.

3. Now modify the subject group division names:
```
$all_captions = array('year' => array('IA - Year 1',
                                      'IB - Year 2',
                                      'II - Year 3',
                                      'III - Year 4'),
                      'term' => array('Michaelmas', 'Lent',
                                      'Easter'),
                      'type' => array('Lectures',
                                      'Coursework',
                                      'Examining',
                                      'Administration',
                                      'Preparation',
                                      'Demonstrating'),
                      'group' => array('Inorganic', 'Organic',
                                       'Physical',
                                       'Theoretical',
                                       'Multi-Disc')
                      );
```
Change the group section to match the codes above, in the same order

7.4.1.2 *Job types*

Each job can have an activity type associated with it. This might be lecturing, examining, some form of administration (e.g. sitting on a committee), taking leave/sabbatical, conducting classes and seminars, and so on. These are also represented with letter codes, e.g. 'L', 'E', 'A', etc.

These are configured by modifying `config.inc` as in 7.4.1:

1. Set the job codes in the following code section:
```
$all_categories = array('year' => ...,
                        'term' => ...,
                        'type' => array('L','C','E',
                                        'A','P','D'),
                        'group' => ...
                        );
```

2. Set the job type names/descriptions as follows:
```
$all_captions = array('year' =>...,  'term' => ...,
                      'type' => array('Lectures',
                                      'Coursework',
                                      'Examining',
                                      'Administration',
                                      'Preparation',
```

```
                                                'Demonstrating'),
                    'group' => ...);
```

Make sure that the positions of the elements in these two arrays correspond and that the type letters are distinct.

Other examples of job types have included: 'Classes', 'Practicals', 'Projects', 'Sabbatical', 'Safety', 'School Duties', 'Seminars', 'Supervisions', 'Outreach', 'Web Based Learning'.

These types are shown at the top of the view jobs screen and can be used to filter the display by job type – for example to show all Lecturing jobs (in this case where the type is L).

Note: A small amount of configuration work is needed to add a new type to the Faculty Summary view facility. Please refer to the Developer's Guide for information on how to do this.

7.4.1.3 *Terms and years*

The term and year categories are modified in a similar manner to job type and subject group.

7.4.1.4 *Other columns*

The other columns (fields) provided in jobs table of the standard distribution of TODB are considered 'non-categorised', in the sense that they contain loose data rather than data derived from a list of some kind. These columns include 'paper' (paper/unit name), 'name' (job name/description) and 'note' (any notes the admin user wishes to add). They are therefore not set up in the config.inc file to display as categories in the filtering apparatus on the View Jobs page.

Department-specific 'tweaks' to report pages might respond to a convention used in one of these; for example, in Engineering, one report assumes that if the job name contains the word 'Examiner' and the job type is 'E' (examination-related jobs), the person doing this job will be considered the examiner for this course. Another example is that if the phrase '*Leader' appears, the person doing the job is considered the module leader. This functionality is generally outside of the standard TODB release but is mentioned should a user wish to add functionality to their TODB instance.

7.4.1.5 *Choice of job columns to display to ordinary (non-admin) users*

When a user chooses the 'View Jobs' page, the jobs data is displayed in 'view' mode. The columns that are displayed to the non-admin user in this mode are determined by an array variable which can be set via the jobs configuration file.

Note: If the logged-in user is an admin user, the variables defined in the next section ($adminjobcols, etc) will be used instead of these.

Note: any fields listed in $jobcols and other arrays MUST appear in the $jobitems array.

Note: it is strongly recommended that any points columns are kept to the right so that totals will line up under them.

1. Open jobs.inc (in the config directory) in a text editor

2. Edit the following line, either adding or removing column names in the Jobs table:
   `$jobcols` = array (**"course", "year", "paper", "prgroup", "name", "type", "hours", "term", "uname", "note", "points"**);

3. What is displayed on-screen as column headers corresponding to these is set in the following line:
   `$jobcolshdr` = array (**"Course", "year", "paper", "Group", "Name", "Type", "Hours", "Term", "Person", "Notes", "Points"**);

7.4.1.6 *Choice of job columns to display to admin users*

If the logged-in user is an admin user, the following variables define fields are shown; they are set in the `jobs.inc` configuration file:

1. Open `jobs.inc` (in the config directory) in a text editor

2. Edit the following line, either adding or removing column names in the Jobs table:
   `$adminjobcols` = array (**"course", "year", "paper", "prgroup", "name", "type", "hours", "term", "uname", "note", "points"**);

3. What is displayed on-screen as column headers corresponding to these is set in the following line:
   `$adminjobcolshdr` = array (**"Course", "year", "paper", "Group", "Name", "Type", "Hours", "Term", "Person", "Notes", "Points"**);

Different columns may be displayed to admin users only to limit access to certain sensitive information (home phone number, for example), or perhaps to make available columns that are not generally of interest to casual users (`id` or `updatetime`, for example).

7.4.1.7 *Choice of fields displayed on edit screen*

When the user is in 'edit' mode and selects a job record to edit (or adds a new job), certain fields are displayed on-screen for editing. Not all fields of the underlying database table are necessarily relevant to the user. The `id` and `updatetime` columns are, for example, generated automatically by the database. The choice of columns is determined as follows:

1. Open jobs.inc (in the config directory) in a text editor

2. Edit the following line, either adding or removing column names in the Jobs table:
   `$updateableadminjobcols` = array (**"course", "year", "paper", "prgroup", "name", "type", "hours", "term", "uname", "note" "points"**);

3. What is displayed on-screen as column headers corresponding to these is set in the following line:
   `$updateableadminjobcolshdr` = array (**"Course", "year", "unit", "Group", "Name", "Type", "Hours", "Term", "Person", "Notes**, **"Pts");**

The widths of the text boxes on the edit screens can be set by modifying the corresponding entry in `$adminjobcolwidths`.

Note 1: Please ensure that the correct capitalisation is used.

Note 2: any fields added to jobcols-type variables MUST exist in the '`$jobitems`' array, otherwise the javascript breaks.

7.4.1.8 *Note on Extensibility*

If a new key is added to the `$all_x arrays` in config.inc, the filtering apparatus and queries will be updated. New (string-based) columns can be added to the `jobs_xxx` tables and the system will sort itself out to generate correct queries and so on.

7.4.1.9 *Jobs data structuring conclusion*

The structuring of data in TODB is deliberately loose. It is up to individual departments to choose how they feel job tasks ought to be categorised. A department that offers only single-year postgraduate courses might have no need for the 'year' field. Many departments dispense with the 'course' field. The selection of fields that is provided with the standard TODB distribution represents the common fields by which jobs are structured in the departments that are evaluating or using the TODB software.

Interested people are invited to read the Developer's Guide to adding report pages to the system to facilitate more effective data analyses for specific departmental uses.

7.4.2 Structuring of people data

People data is stored in tables people_yearval, where yearval represents the academic year (e.g. 2008_09, 2006_07, etc). A separate set of information is stored for each year (in a separate table), so it is necessary to make copies of these tables as additional years are added to the system. As with jobs, the values that appear on-screen to non-admin users, admin users, and in the editing screens are defined in a configuration file (`people.inc` in the config directory).

Table: Example people table fields and descriptions. More may be added by modifying the table schema and updating the variables that follow.

Field name	Description	Data type
`id`	Row id	int(11)
`uname`	Unique person name	varchar(64)
`crsid`	Cambridge CRSID	varchar(32)
`engid`	Engineering ID (a legacy of the Engineering Dept heritage of this system)	varchar(32)
`division`	Subject group divisions to which this person belongs	varchar(30)
`title`	The person's title – Ms, Mr, Dr, Prof, Rev, etc	varchar(20)
`called`	The person's first name	varchar(32)
`surname`	Their surname	varchar(64)
`initials`	Their initials	varchar(32)

`quota`	The number of points that this person ought to earn	int(11)
`room`	Room number	text
`phone`	Telephone number	varchar(32)
`job_title`	Job title	varchar(32)
`college`	College affialiation	varchar(4)
`deleted`	Whether or not this person has been marked as deleted	tinyint(1)
`OK06`	Whether or not this person has indicated that his/her teaching duties have been confirmed as correct	datetime
`updatetime`	When this record was last updated	timestamp

The detail level of information required in this table (people_2008_09, people_2009_10, etc) is variable. If this table is to be used as a central source of staff data, then it might be necessary to ensure that almost all fields are completed. If the TODB is used simply as a teaching duties points calculator, it is possible to get away with a bare minimum of uname, surname, initials and possibly crsid – enough to uniquely identify who someone is, and how to contact them.

7.4.2.1 *Choice of people data columns to display to non-admin users*

When a non-admin user chooses the 'View People' page, the people data displayed are determined by array variables which can be set via the people configuration file:

Note: values in $personcols are case-sensitive (must match the values in $personitems). Fields MUST appear in the $personitems array to be used anywhere.

Note: the hdr array has an extra column 'Points' at the end which will be filled automatically.

1. Open people.inc (in the config directory) in a text editor

2. Edit the following line, either adding or removing column names in the People table:
   `$personcols = array (`**`"uname", "crsid", "engid", "division", "title", "initials", "surname", "room", "phone", "job_title", "college", "OK06"`**`);`

3. What is displayed on-screen as column headers corresponding to these is set in the following line:
   `$personcolshdr = array (`**`"Show Jobs", "crsid", "engid", "Div", "Title", "Initials", "Surname", "Room", "Phone", "Job Title", "College", "Duties-OK", "Points"`**`);`

7.4.2.2 *Choice of People columns to display to admin users*

When an admin user is viewing the 'View People' page, the columns that are displayed to the user in are determined by array variables which can be set via the people configuration file:

1. Open people.inc (in the config directory) in a text editor

2. Edit the following line, either adding or removing column names in the People table:
   `$adminpersoncols = array (`**`"uname", "crsid", "engid",`**

```
“division”, “title”, “called”,  “initials”, “surname”, “room”,
“phone”, “job_title”, “college”, “OK06”, “updatetime”);
```

3. What is displayed on-screen as column headers corresponding to these is set in the following line:
```
$adminpersoncolshdr = array (“Unique Name”, “crsid”, “engid”,
“Div”, “Title”, “Called”, “Initials”, “Surname”, “Room”,
“Phone”, “Job Title”, “College”, “Duties-OK”, “Last updated”,
“Points”);
```

Different columns may be displayed to admin users to limit access to certain sensitive information, or perhaps to make available columns that are not generally of interest to casual users (`id` or `updatetime`, for example).

7.4.2.3 *Choice of fields displayed on edit screen*

When the user is in 'edit' mode and selects a job record to edit (or adds a new person), certain fields are displayed on-screen for editing. Not all fields of the underlying database table are necessarily relevant to the user. The choice of columns is determined as follows:

1. Open people.inc (in the config directory) in a text editor

2. Edit the following line, either adding or removing column names in the Jobs table:
```
$updateableadminpersoncols = array (“uname”, “crsid”, “engid”,
“division”, “title”, “called”,  “initials”, “surname”, “room”,
“phone”, “job_title”, “college”, “OK06”);
```

3. What is displayed on-screen as column headers corresponding to these is set in the following line:
```
$updateableadminpersoncolshdr = array (“Unique Name”, “crsid”,
“engid”, “Div”, “Title”, "Called", "Initials", "Surname",
"Room", "Phone", "Job Title", "College", "Duties-OK);
```

4. The widths of the text boxes on the edit screens can be set by modifying the corresponding entry in $adminpersoncolwidths.

If you are displaying total points for this person this must be in the last column and the column name is "sum".

Note 1: Any changes to these arrays that refer to database columns are CASE-SENSITIVE! If any element in `$updateableadminpersoncols` is incorrectly capitalised, the JavaScript that generates the pop-up edit screen will break.

Note 2: any fields added to `$updateableadminpersoncols` MUST exist in the '`$personitems`' array, otherwise the js breaks.

7.4.3 *Structuring of Units data*

Units data are stored in tables `units_yearval`, where `yearval` represents the academic year (e.g. 2008_09, 2006_07, etc). A separate set of information is stored for each year (in a separate table), so it is necessary to make copies of these tables as additional years are added to the system. As with jobs and people, the values that appear on-screen to non-admin users, admin users, and in the editing screens are defined in a configuration file (`units.inc` in the config directory).

Table: Example units table fields and descriptions. More may be added by modifying the table schema and updating the variables that follow.

Field name	Description	Data type
Id	Row id	int(11)
Uname	Unique name of this unit/paper	varchar(32)
Course	Course of which this unit is a part	varchar(32)
ordering	If a special order of units display is desired, a number can be entered in this field. 1 or 0 will take it higher in the list, and a large number will send it down the list.	int(11)
Sgrps	Subject groups to which this unit is associated (comma-separated list)	varchar(32)
Name	Name/description of this unit	text
assessmode	Assessment Mode	varchar(32)
running	1 or 0, indicating whether or not this unit is running in this academic year	tinyint(1)
Note	Notes/extra details	text
deleted	Whether or not this unit has been marked for deletion	tinyint(1)
Global	units that are interesting to everyone	tinyint(1)
updatetime	When this unit information was last edited	timestamp

7.4.3.1 *Choice of Units data columns to display to non-admin users*

When a non-admin user chooses the 'View Units' page, the units data displayed are determined by array variables which can be set via the Units configuration file:

Note: values in `$unitcols` are case-sensitive (must match the values in `$unititems`). Fields MUST appear in the `$unititems` array to be used anywhere.

1. Open `units.inc` (in the config directory) in a text editor

2. Edit the following line, either adding or removing column names in the People table:
   `$unitcols` = array ("**uname", "course", "sgrps", "name", "assessmode", "running", "global"**);

3. What is displayed on-screen as column headers corresponding to these is set in the following line:
   `$unitcolshdr` = array (**"Paper/Unit", "Course", "Subj Grps", "Full Name", "Mode", "Running", "Global Interest"**);

7.4.3.2 *Choice of Units columns to display to admin users*

When an admin user is viewing the 'View Units' page, the Units data displayed are also determined by array variables which can be set via the Units configuration file:

1. Open `units.inc` (in the config directory) in a text editor

2. Edit the following line, either adding or removing column names in the People table:
   <u>$adminunitcols</u> = array (**"uname", "course", "ordering", "sgrps", "ignorewronggroups", "name", "assessmode", "running", "global", "note", "updatetime"**);

3. What is displayed on-screen as column headers corresponding to these is set in the following line:
   <u>$adminunitcolshdr</u> = array (**"Paper/Unit", "Course", "Order", "Subj Grps", "Correct Groups", "Full Name", "Mode", "Running", "Global Interest", "Note", "Last updated"**);

Different columns may be displayed to admin users to limit access to certain sensitive information, or perhaps to make available columns that are not generally of interest to casual users (`id` or `updatetime`, for example).

7.4.3.3 *Choice of fields displayed on edit screen*

When the user is in 'edit' mode and selects a job record to edit (or adds a new unit), certain fields are displayed on-screen for editing. Not all fields of the underlying database table are necessarily relevant to the user. The choice of columns is determined as follows:

1. Open units.inc (in the config directory) in a text editor

2. Edit the following line, either adding or removing column names in the Jobs table:
   <u>$updateableadminunitcols</u> = array (**"uname", "course", "ordering", "sgrps", "ignorewronggroups", "name", "assessmode", "running", "global", "note");**

3. What is displayed on-screen as column headers corresponding to these is set in the following line:
   <u>$updateableadminunitcolshdr</u> = array (**"Paper/Unit", "Course", "Order", "Subj Grps", "Correct Groups", "Full Name", "Mode", "Running", "Global Interest", "Note");**

4. The widths of the text boxes on the edit screens can be set by modifying the corresponding entry in `$adminunitcolwidths`.

Note 1: Any changes to these arrays that refer to database columns are CASE-SENSITIVE! If any element in `$updateableadminunitcols` is incorrectly capitalised, the JavaScript that generates the pop-up edit screen will break.

Note 2: any fields added to `$updateableadminunitscols` MUST exist in the '`$unititems`' array, otherwise the javascript breaks.

7.4.4 Adding new fields

New columns or 'fields' can be added to any of the jobs, people and units tables. This can be done by accessing the MySQL database directly and issuing the appropriate commands. TODB can be instructed to display these columns by editing the jobs.inc, people.inc and units.inc tables respectively. If job fields contain data that are easily categorised, the field name can be added to the arrays defined in `config.inc` to make the jobs data filterable on these (as described in User Configuration).

7.5 Configuring Points Formulae For Automatic Points Calculations

Points formulae can be used in two ways in the TODB.

- The first approach bases the number of points on the number of students registered for a paper (unit). This is primarily useful for marking, where the effort that goes into doing the job is directly dependent on the number of students.   This facility links to information on the number of students registered for each unit. As this input information is stored, points can be recalculated at any time.

- The second approach uses a 'Points Calculator' that lets the user calculate points at the time of entering data. This is used in situations where more arbitrary criteria determine the points earned. This facility was developed for situations where one paper is marked by several people where each person is responsible for marking one question, and the number of students who choose to answer that question could vary greatly within the paper. This can only be a retrospective calculation. The same facility could be used to calculate points for a member of staff supervising a completely variable number of students, where the points earned is based on number of students supervised.
   Once the calculation is made, it can only be updated by re-editing the record. The reason for this is that information on the number of students registered per paper is stored, but not the number of students per paper and per question or supervisor.

7.5.1 Points based on numbers of registered students

Note: even if this facility is not used, the tables `points_formulae_yearval` (where yearval is the academic year, e.g. 2008_09) and `studentspercourse_yearval` MUST exist, otherwise the SQL queries break.

Not much configuration is suggested or available.   To enable formula use in jobs:

- In jobs.inc, add `'"formula_id"'` to the `$jobitems` array.

- Add `'"formula_id"'` to the `$updateableadminjobcols` array

- Add `'"Formula ID"'` to the `$updateableadminjobcolshdr` array

The list of available formulae will appear in a list box (select list) in the editing screen.

 To view/edit formulae, they must be entered using the `view_points_formulae.php` with `yearval` as a GET parameter (e.g. `view_points_formulae.php?yearval=2008_09`).

To view/edit numbers of students, use `view_student_counts.php?yearval=xxxx_xx`.

7.5.2 Points based on arbitrary number of answers or students

The same Formula list is used as in the previous approach. However, formulae are not associated with jobs. Rather, the calculator window opens when the calculator icon is clicked next to the points field in the jobs editing screen. The list of Formulae appears in the window, along with fields of information (offset and multiplier, which are filled-in when a valid formula is selected). If no pre-entered formula is available, values can be entered for the offset and multiplier fields (suitable for once-off calculations). If the 'Calculate' button is pressed, the correct number of points is calculated. The 'Done' button causes the value to be inserted into the jobs editing screen in the points field.

There is no reason why both types of formulae cannot be used in the same TODB. Care must be taken when naming formulae so that the user knows which formulae apply to numbers of students and which apply numbers of 'something else'.

Configuration:

- Open config.inc in the config directory

- Set `$show_points_calculator` to either true or false (true displays the calculator icon, false hides it).

- Describe what the independent variable is – 'answers marked' or 'students supervised', or even 'independent things'. Assign this to `$calculator_x_desc`.

Remember that values entered this way cannot be recalculated except by re-editing the job, re-entering the x-value and clicking 'Calculate.'

7.6 Configuration Of Table Admin Functions

The 'Modify Table Configurations' facility is accessible via a link which is displayed on the index page when an admin user views it. This facility allows for three functions:

1. Creation (and removal) of new 'years' of data. This is used to create the tables necessary for extra years of data to be added to the system – e.g. the next academic year, or perhaps to enter a year of historical data to analyse year-by-year changes in teaching load. Selected data is copied from the source year to the new year.

2. Setting flux status of a year. See User Guide for a discussion of flux years.

3. Setting the 'current' year.

The only configuration possible for this is the choice of columns copied from the source year to the new year. The idea behind this is to save the user the trouble of starting the job list from scratch each year. The Engineering dept choose not to copy the uname column, meaning that teaching

allocations are allocated to people from scratch, with less bias or prejudice coming from a name already allocated to a task from the previous year.

Configuration is performed by editing the 'new_years.inc' file in the config directory. The list of columns copied when the 'jobs' table is copied is determined by $copy_columns['jobs']. Similarly the list of columns copied when the 'people' table is copied is determined by $copy_columns['people'], and so on.

TODB is supplied with empty tables for the year 2009-10 but with both years 2009-10 and 2010-11 shown on the 'grid' on the index page. You will need to use 'Modify year configurations' to create data for the 2010-11 academic year.

7.7 Configuring Stint Point Summary Page

The stint point summary page is the most difficult to configure, as it requires knowledge of SQL. More information is provided in the Developer's Guide.

8   Troubleshooting

8.1 Can not connect to database

Should you get the following error:
`'Access denied for user 'gen_todb_read'@'localhost' (using password: YES)`
Make sure you have typed "flush privileges" within a mysql session. Also check the name of the database you have created against the one that TODB is connecting to (set in config/db.inc).

8.2 You are logged in as .

This message appears towards the bottom of this first page and should include you user id before the '.'. This will be followed by the 'Panic: cannot set year' message should you click on one of the page links. Should you get this, you need to make sure the Apache configuration file has been set up correctly to include you as a user – see Section 4.4

8.3 'Panic: cannot set year'

This error might appear during installation, and is caused by the system being unable to set/find the requested academic year's data. This requires database access, so may in fact be caused by the TODB read user database login not having read privileges on the tables in the database. Check this by issuing 'use mysql; show grants for '<username>'@'localhost', where <username> is the name of the database read user. This command should list a line including 'select ...'. Remedy: grant select and lock tables to the read user at the MySQL prompt.

8.4 CSV files not generating

Remember to check the CSV path in config.inc

Check that the CSV directory exists

Check that the CSV directory is writeable (owned by the webserver user or generous permissions)

8.5 Timetables not displaying

There are several possible causes for non-display of timetables:

8.5.1 Data issues

- No timetable data in jobs table

    o Remedy: Add data in correct format (e.g. Wk1 F.10, Wk3 Th.3, etc)

- Incorrect timetable data in jobs table (i.e. system cannot understand it)

    o Remedy: check format (e.g. Wk1 F.10, Wk3 Th.3, etc)

- Incomplete timetable data in jobs table: timetable data are displayed by year/part, term and timeslot.

    o Remedy: it is therefore necessary to make sure that entries appear in **all** of these columns for a job to appear in the timetable.

- List of start times not matching the start times used in timeslot entries in jobs table.

    o Remedy: set correct start times in config.inc: `$tt_start_times`. This is an array of start times, in order.

- List of years/parts not matching the parts/years in use in the jobs table.

    o Remedy: correct the years/parts list in config.inc: `$tt_yearlist`. This is an array of year values (1, 2, 3 ... or 1A, 1B, 2... or IA, IB, II, III). Ensure that '1A' or 'IA' is used consistently.

8.5.2 Non-data issues

In general, unless changes have been made to the code, it is likely a configuration or permissions issue than a coding one. Changes to the timetabling code are not usually required, and not recommended.

If the timetable does not appear via the checkbox in View People, try accessing the 'View Timetable' link, if this is possible (via the shortcut if there is one, or by the generic operation box on the home page or failing this by typing view_timetable.php at the end of the URL). It would be unusual for one to work and not the other, but might help to shed light on an underlying cause.

8.6 Problems selecting Edit

If you get 'editlocks_<year> table failed:no database selected', when you try to press the Edit button, you may have a problem with write access to the database. Double check

the entries in db.inc and for the TODB users in the db and user tables in the mysql database exactly match.

8.7 Problems running MySQL client

You may need to the appropriate path (e.g. `C:\Program Files\MySQL\MySQL Server 5.1\bin`) to your path (Control Panel> System>Advanced>Environment variables) should you find things don't work when you type `mysql` at the command prompt.

8.8 Problems when adding or updating

On some configurations, you may see 'You have attempted to post a <something> edit/addition when not in edit mode; this has been rejected. Did you click 'Finish Edit' before apply ?' after clicking 'Apply' to make an edit.

This can be caused by php code being called before the html header and can be fixed by moving the php code down.

e.g. job_popup.php

```
require('config/config.inc');    // This is wrong

<html>
<head>
<title>update job</title>
<script type=text/javascript>
<!--

<?php
require('config/config.inc');    // This is correct
require('useful.inc');
```

This applies to add<something>.php, <something>_popup.php and js_PointsCalc.php.

8.9 Problems caused by ^M

Files edited on Windows machines that are later transferred to Unix or Mac machines can be populated with ^M characters at the line endings which may cause things to break – particularly links.

You can use `cat -tv <filename>` to look for these (they may not show up in vi). Also the Unix command `file <filename>` will tell you the file type. You can then use a utility such as Dos2Unix, or a substitution e.g. from a shell: `sed -I 's/\r//g' <filename>` or from vi `:%s/^V^M//g` to remove these.

8.10 Edit Pop-up box hangs and is empty

Check the ownership and permissions of the popup_form_state.inc. They should match everything else in the directory. If necessary `chmod a+rX` will fix things.

8.11 *Raven Authentication and Macs in University of Cambridge*

The Apache module for Raven authentication has needed to be compiled separately for use on Mac computers before TODB can be used with Raven authentication. This is considered outside the remit of TODB support.

The following is an example of what to include in the Apache configuration file.

```
#       Load webauth module
         LoadModule ucam_webauth_module libexec/apache2/mod_ucam_webauth.so
         LoadModule authz_user_module libexec/apache2/mod_authz_user.so

#       TODB Stuff
         <Directory "/Library/WebServer/Documents/TODB/">
                 Options Indexes MultiViews
                 AllowOverride None
                 Order allow,deny
                 Allow from all
                  Satisfy all
                  <Files *.inc>
                          Order allow,deny
                          Deny from all
                  </Files>

#       Create Raven user access to TODB directory – only test1 and ac690
can log in at present
                 AAKeyDir "/etc/apache2/webauth_keys"
                 AACookieKey "paramoudra13"
                 AuthType Ucam-WebAuth
                 Require user test1 ac690
         </Directory>
```

9 Configuration Checklist

It is recommended that these items be 'ticked off' to complete installation:

☐ Installation of files

☐ Database name and usernames/passwords setup

☐ Database creation

☐ Database user creation

☐ Apache site and user configuration

☐ TODB Admin user configuration

☐ Jobs: configuration of job types, courses, etc

☐ Units configuration

☐ People configuration

☐ Set up years

☐ Set up fluxusers

10 Other Documentation

The following documents have been produced for TODB:

Name	Purpose
Installation and Configuration Document	This document
User Guide	Supplied in html format with the code in Documents directory – this can be accessed from the ? icon. Also supplied as a pdf.
How To	This lists the tasks a computer officer may need to undertake in supporting the Teaching Office Database and provides instructions on how to do these. It is intended to take over from the Installation and Configuration document once you have installed the system.
Developers Guide	Detailed tips and instructions for developing the code and list of code items.