

TestCaseDB Administration Guide



Version 3.1

1 Table of Contents

1	Table of Contents	2
2	Overview	8
3	Requirements	9
3.1	Operating System Versions	9
3.1.1	Ubuntu	9
3.1.2	Cent OS	9
3.1.3	RedHat	9
3.2	Software Packages	9
3.3	System Requirements	10
3.3.1	x86 Minimum Requirements	10
3.3.2	x86_64 Minimum Requirements	10
3.4	Dedicated Hardware	10
4	Installation	11
4.1	Download	11
4.2	Ubuntu Install	11
4.2.1	Install Operating System	11
4.2.2	SSH Server	11
4.2.3	Notes on Ubuntu Sudo	12
4.2.4	Install Required Software	12
4.2.5	Prepare MySQL Database	13
4.2.6	Install Application	13
4.2.7	Configure the Apache Host	14
4.2.8	Configure Logrotate	16
4.3	CentOS/Red Hat Install	16
4.3.1	Install Operating System	16
4.3.2	Add EPEL Repository	16

4.3.3	Install Required Software	17
4.3.4	Prepare MySQL Database	19
4.3.5	Install Application	19
4.3.6	Updating SELinux	20
4.3.7	Configure the Apache Host	21
4.3.8	Firewall Ports	22
4.3.9	Configure Logrotate	22
4.4	Final Configuration, Install Script and Initial Login	22
4.4.1	Email Configuration	22
4.4.2	Disable HTTPS	23
4.4.3	Default Session timeout	23
4.4.4	Scheduler Crontab	24
4.5	Login and Change the Password	24
4.6	Deactivate the Default Admin User	24
5	Upgrades	26
6	Backup	29
6.1	Database	29
6.2	Uploaded Files	29
7	Administration Tasks	30
7.1	Create a User	30
7.2	Change a User's Password	30
7.3	Deactivate a User	31
7.4	Unlock a User Account	31
7.5	Create Test Types	31
7.6	Create Products	32
7.7	Authorize Users for New Products	32
7.8	Create Categories	33

7.8.1	Example Category Setups	33
7.8.2	Configuring Categories	33
7.9	Create Tags	34
7.10	Create Versions	34
7.11	Creating Custom Fields.....	35
7.11.1	Defining the Field	35
7.12	Configuring Result Requirements	36
7.13	Create Devices.....	36
8	Create Test Cases, Test Plans and Stencils	38
8.1	Create Test Cases	38
8.1.1	Creating Test Cases in TestCaseDB Using Forms	38
8.1.2	Import Test Cases into TestCaseDB Using XLS.....	38
8.2	Organize Test Cases into Test Plans	40
8.2.1	Create a Test Plan.....	40
8.2.2	Adding Test Cases to Test Plans	40
8.3	Organizing Test Plans into Stencils	41
8.3.1	Create a Stencil.....	41
8.3.2	Adding Test Plans to Stencils.....	42
8.4	Locking and Versioning.....	42
8.4.1	Configuring Locking	42
8.4.2	Versioning.....	43
8.4.3	Versioning vs. Copying.....	43
9	Assign, Execute and Monitor Testing Progress	44
9.1	Assign and Execute a Test Plan or Stencil.....	44
9.2	Executing an Assignment.....	44
9.3	View Reports	45
10	Ticket System Integration.....	46

10.1	Bugzilla	46
10.1.1	Support.....	46
10.1.2	Configuration	46
10.1.3	Security Considerations.....	47
10.2	Jira	47
10.2.1	Support.....	47
10.2.2	Configuration	47
10.2.3	Security Considerations.....	48
10.3	Mantis Bug Tracker	48
10.3.1	Support.....	48
10.3.2	Configuration	48
10.3.3	Security Considerations.....	49
10.4	Redmine	49
10.4.1	Support.....	49
10.4.2	Configuration	49
10.4.3	Security Considerations.....	50
11	Automation	51
11.1	Scheduler	51
11.1.1	Enabling Global Script Execution.....	51
11.2	Operational Overview	52
11.3	Creating an Automated Test Plan	52
11.4	Monitoring Automated Plans.....	53
12	jMeter.....	54
12.1	Overview	54
12.2	Configuration	54
12.3	Operational Overview	55
12.4	Creating jMeter Test Cases	56

12.5	Analyzing results	56
13	TestCaseDB API	58
13.1	General Format	58
13.2	Automation Workflow	58
13.3	API User.....	59
13.4	API Commands	59
13.4.1	CheckVersion.....	59
13.4.2	CreateVersion.....	60
13.4.3	CreateAssignment	61
13.4.4	GetResult.....	63
13.4.5	SetResult	64
13.5	Testing the API	65
14	Importing TestLink Data	66
14.1	Requirements.....	66
14.2	What is Imported	66
14.3	Steps.....	66
15	Appendix A: User Roles	68
15.1	Permissions	68
16	Appendix B: Report Types	69
16.1	System Status	69
16.2	Release Current State.....	69
16.3	Release Current State – By User	69
16.4	Release Progress – Daily.....	69
16.5	Compare Release Results	70
16.6	Test Cases Without Steps.....	70
16.7	Open Tasks	70
16.8	Release Bug Report	70
17	Appendix C: Install Worksheet	71

17.1	Lab Setup.....	71
17.2	Production Setup	71
18	Appendix D: Install TestCaseDB in a Sub-URI	72
18.1	Symlink from Existing Site	72
18.2	Apache Configuration.....	72
18.3	Upgrades	73
18.3.1	Disable/Enable TestCaseDB.....	73
18.3.2	Symbolic Link.....	74

2 Overview

This document is intended to be a complete guide to the TestCaseDB application.

The document includes instructions for installation, planning, usage, backup and best practices.

3 Requirements

TestCaseDB is built using Ruby on Rails 3.0. While Ruby on Rails can run on a variety of operating systems, support and instructions are provided for the officially supported operating systems Ubuntu, Red Hat and CentOS.

3.1 Operating System Versions

The following operating system versions are supported.

3.1.1 Ubuntu

- Ubuntu 12.04.2 – Long Term Support
- Ubuntu 12.10

3.1.2 Cent OS

- Cent OS 5.9+
- Cent OS 6.4+

3.1.3 RedHat

- Version 5.6+
- Version 6.1+

3.2 Software Packages

The following software packages are required for use with TestCaseDB. Additional packages and Ruby Gems are required for the system. Details are given in the install instructions.

- Apache 2.x
- MySQL 5.x
- Ruby 1.9.3 (1.8.x is no longer supported)
- Rails 3.2

3.3 System Requirements

System requirements depend on the size of the team and usage level. Minimum requirements are given as a guideline. The minimum system requirements place the database and web server on the same server hardware without virtualization.

3.3.1 x86 Minimum Requirements

- 2 core processor
- 1 GB Ram
- 30 GB Hard drive space. If more than 1 GB of attachments will be stored in TestCaseDB, extra hard drive space should be allotted.

3.3.2 x86_64 Minimum Requirements

- 2 core processor
- 4 GB Ram
- 30 GB Hard drive space. If more than 1 GB of attachments will be stored in TestCaseDB, extra hard drive space should be allotted.

3.4 Dedicated Hardware

If your company plans on utilizing the Automation tool it is highly recommended that TestCaseDB runs on a dedicated server or virtual server. This step is important as the automation tool allows testers to load their own scripts on to the server. More details are available in the [Automation](#) section.

4 Installation

Installation steps are provided for [Ubuntu](#), [CentOS](#) and [RedHat](#). Detailed instructions are given for each operating system.

The installation instructions for all operating systems in this guide assume that TestCaseDB will be a unique site with its own URL. If you plan on running TestCaseDB from a Sub-URI in an existing site, for example <https://www.yoursite.com/testcasedb/>, please read [Appendix D: Install TestCaseDB in a Sub-URI](#) and utilize it while following the instructions for your specific operating system.

4.1 Download

Download the application from our website at www.testcasedb.com.

4.2 Ubuntu Install

The following instructions are for Ubuntu based installations.

4.2.1 Install Operating System

Install the operating system using standard directions from the system documentation. No special packages are required at this stage. A server install is preferred, because there is no need for a GUI desktop.

4.2.2 SSH Server

By default, Ubuntu installs do not include an SSH server. It is recommended that one be installed.

```
$ sudo apt-get install openssh-server
```

Please view the configuration of Openssh to verify that it meets company requirements. It is recommended that PermitRootLogin is disabled.

4.2.3 Notes on Ubuntu Sudo

On some versions of Ubuntu, the sudo application is compiled with the feature 'with-secure-path'. As a result, changes to the PATH variable in /etc/environment, ~/.bashrc, and others are not utilized by sudo and some command locations are unknown. To work around this issue you will need to use the complete path to access commands that are not detected when using sudo. Use the 'which' command to find the location of a file without using sudo.

4.2.4 Install Required Software

1. Update apt-get repository

```
$ sudo apt-get update
```

2. Install Ruby

```
$ sudo apt-get install ruby1.9.1-full build-essential  
libopenssl-ruby1.9.1 imagemagick libxml2 libxml2-dev libxslt1-  
dev libssl-dev
```

3. Install Apache

```
$ sudo apt-get install apache2 apache2-mpm-prefork apache2-  
prefork-dev libapache2-mod-xsendfile nodejs
```

4. Update Ruby Gems

```
$ sudo gem install rubygems-update
```

5. Install MySQL

```
$ sudo apt-get install mysql-server mysql-client libmysql-  
ruby1.9.1 libmysqlclient-dev
```

6. Install Phusion Passenger

```
$ sudo gem install passenger  
$ sudo apt-get install apache2-dev libapr1-dev libaprutil1-dev  
libcurl4-openssl-dev  
$ sudo passenger-install-apache2-module
```

7. Configure Apache to use Passenger. Add the following 3 lines to `/etc/apache2/apache2.conf`

```
LoadModule passenger_module /var/lib/gems/1.9.1/gems/passenger-3.0.19/ext/apache2/mod_passenger.so
PassengerRoot /var/lib/gems/1.9.1/gems/passenger-3.0.19
PassengerRuby /usr/bin/ruby1.9.1
```

4.2.5 Prepare MySQL Database

The steps in this section should be used to prepare a database for the TestCaseDB application. All values in the form '`<value>`' should be replaced with the items entered in your install worksheet.

1. Log into mysql

```
$ mysql -u root -p
```

2. Create the database

```
mysql> create database <database_name>;
```

3. Create the MySQL user with access to the new database. If you are installing the TestCaseDB application on a different server to the database, replace localhost with the IP of the TestCaseDB server.

```
mysql> GRANT ALL ON <database_name>.* TO
'<database_username>'@'localhost' IDENTIFIED BY
'<database_password>;
```

4.2.6 Install Application

1. Download the TestCaseDB package to the server.
2. Change directories to your install folder. For the remainder of this section, the document assumes that you are installing the application in `/home/tcdb/`.

```
$ cd /home/tcdb
```

3. Unpack the application into the install directory.

```
$ tar xzf /path/to/tarball/tcdb.vX.Y.Z.tgz
```

4. Create a symbolic link to the folder; Apache will use this link.

```
$ ln -s tcdb.vX.Y.Z tcdb
```

5. Configure the application to use the MySQL database.
 - a. Open the file `tcdb/config/database.yml` in your favourite editor.
 - b. In the 'production' section, set the values for database, username and password.
 - c. If your database is hosted on a different server, add an entry titled host and add the IP or hostname of your server. A sample is shown below.

```
production:
  adapter: mysql2
  database: <database_name>
  encoding: utf8
  username: <database_user>
  password: <database_password>
  pool: 5
  timeout: 5000
  host: <database_ip_or_hostname>
```

6. Make sure you are still in the root of the application, `/home/tcdb/tcdb` in our example and run the setup script. This will install the required gems and prepare the database.

```
$ script/setup -p
```

7. Prepare the asset pipeline.

```
$ rake assets:precompile RAILS_ENV=production
```

4.2.7 Configure the Apache Host

A virtual host must be configured to host the site. It is assumed that this will be a named virtual host. If this is the only website on the system, you may configure directly in `httpd.conf`. More details are available in Apache's documentation.

1. Create a new site configuration file using your favourite editor (example uses `vi`).

```
$ sudo vi /etc/apache2/sites-available/qa.yourdomain.com
```

2. Add the content bellowing to the file created in step 1. It is assumed that only SSL access will be allowed. Change all upper case text as appropriate.

```
<VirtualHost *:80>
  ServerName <server_name.. ex. qa.yourcompany.com>
  Redirect / <SS site URL.. ex. https://qa.yourcompany.com>
</VirtualHost>
<VirtualHost *:443>
  ServerName <server_name.. ex. qa.yourcompany.com>
  DocumentRoot /home/tcdb/tcdb/public
  SetEnv SERVER_NAME <server_name>
  RailsEnv production
  XSendFile On
  XSendFilePath /home/tcdb
  <Directory /home/tcdb/tcdb/public>
    Allow from all
    Options FollowSymLinks
    Options -MultiViews
  </Directory>
</VirtualHost>
```

3. Enable the Apache site.

```
$ sudo a2ensite qa.yourdomain.com
```

4. Restart Apache

```
$ sudo service apache2 reload
```

5. If you have an issue restarting Apache, please see the section [Error Restarting Apache](#).
6. To complete the installation, continue on to [4.4 Final Configuration, Install Script and Initial Login](#)

4.2.7.1 Error Restarting Apache

If you encounter the following error while restarting apache that says, “Invalid command 'XSendFilePath'”, changes will need to be made to the site configuration. The cause of this error is that versions of the xsendfile module below 0.10 do not recognise

the XSendFilePath command. To resolve the issue, remove the XSendFilePath token and add, "XSendFileAllowAbove On", to the configuration. Now restart Apache again.

4.2.8 Configure Logrotate

Without log rotation, the TCDB logs will grow without bound and eventually fill the hard drive. It is recommended that logrotate be configured to put a limit on the logs.

To enable logrotate, create a configuration file in /etc/logrotate.d/ called tcdb with the contents below. Remember to change the path to that of your system.

```
/home/tcdb/tcdb/log/*.log {
    daily
    missingok
    rotate 30
    delaycompress
    notifempty
    sharedscripts
    postrotate
        touch /home/tcdb/tcdb/tmp/restart.txt
    endscript
}
```

4.3 CentOS/Red Hat Install

The following instructions are for CentOS and Red Hat based installations.

4.3.1 Install Operating System

Install the operating system using standard directions from the system documentation.

No special packages are required at this stage. There is no need for a GUI desktop.

4.3.2 Add EPEL Repository

The Extra Packages for Enterprise Linux repository contains a set of packages for use with Fedora, Red Hat and Cent OS. Details are available at

<http://fedoraproject.org/wiki/EPEL>. TestCaseDB uses this repository as a source for the mod_xsendfile package.

To install the EPEL repository, run the command that corresponds to your version of the operating system.

- Version 5.x

```
$ su -c 'rpm -Uvh http://dl.fedoraproject.org/pub/epel/5/i386/epel-release-5-4.noarch.rpm'
```

- Version 6.x

```
su -c 'rpm -Uvh http://dl.fedoraproject.org/pub/epel/6/i386/epel-release-6-8.noarch.rpm'
```

4.3.3 Install Required Software

1. Update the yum repositories.

```
$ sudo yum update
```

2. Install required packages.

```
$ sudo yum install httpd httpd-devel gcc zlib zlib-devel mysql  
mysql-server mysql-devel apr-devel apr-util-devel curl curl-  
devel gcc-c++ patch readline readline-devel openssl-devel make  
bzip2 mod_xsendfile ImageMagick libxml2 libxml2-devel libxslt  
libxslt-devel wget policycoreutils-python
```

3. Install the latest libyaml

```
$ sudo yum groupinstall 'Development Tools'  
$ sudo yum install readline-devel  
$ cd ~  
$ wget http://pyyaml.org/download/libyaml/yaml-0.1.4.tar.gz  
$ tar xzf yaml-0.1.4.tar.gz  
$ cd yaml-0.1.4  
$ ./configure --prefix=/usr/local  
$ make  
$ sudo make install
```

4. Install Ruby.

```
$ cd ~
$ wget ftp://ftp.ruby-lang.org/pub/ruby/1.9/ruby-1.9.3-p392.tar.gz
$ tar xzf ruby-1.9.3-p392.tar.gz
$ cd ruby-1.9.3-p392
$ ./configure --prefix=/usr/local --enable-shared --disable-install-doc --with-opt-dir=/usr/local/lib
$ make
$ sudo make install
```

5. Set mysqld and httpd to autostart. Start MySQL.

```
$ sudo /sbin/chkconfig httpd on
$ sudo /sbin/chkconfig mysqld on
# check the changes
$ sudo /sbin/chkconfig --list httpd
$ sudo /sbin/chkconfig --list mysqld
# Start mysql and apache
$ sudo /etc/init.d/mysqld start
```

6. Update Ruby Gems.

```
$ sudo /usr/local/bin/gem install rubygems-update
$ sudo /usr/local/bin/update_rubygems # this will clean out your gems
```

7. Install Phusion Passenger. At the end of the second command, note the output that is supposed to be added to the Apache configuration.

```
$ sudo /usr/local/bin/gem install passenger
$ sudo /usr/local/bin/passenger-install-apache2-module
```

8. Configure Apache to use Passenger. Add the following 3 lines to /etc/httpd/conf/httpd.conf. Note, the output from step 8 should be used and the lines below are provided as a sample as to what should be added.

```
LoadModule passenger_module /usr/local/lib/ruby/gems/1.9.1/gems/passenger-3.0.19/ext/apache2/mod_passenger.so
PassengerRoot /usr/local/lib/ruby/gems/1.9.1/gems/passenger-3.0.19
PassengerRuby /usr/local/bin/ruby
```

4.3.4 Prepare MySQL Database

The steps in this section should be used to prepare a database for the TestCaseDB application. All values in the form '<value>' should be replaced with the items entered in your install worksheet.

1. Log into mysql.

```
$ mysql -u root -p
```

2. Create the database.

```
mysql> create database <database_name>;
```

3. Create MySQL user with access to the new database. If you are installing the TestCaseDB application on a different server to the database, replace localhost with the IP of the TestCaseDB server.

```
mysql> GRANT ALL ON <database_name>.* TO  
'<database_username>'@'localhost' IDENTIFIED BY  
'<database_password>;
```

4.3.5 Install Application

1. Download the TestCaseDB tar ball file to your server.
2. Change directories to your install folder. For the remainder of this section, the document assumes that you are installing the application in /home/tcdb/.

```
$ cd /home/tcdb
```

3. Unpack the application into the install directory.

```
$ tar xzf /path/to/tarball/tcdb.vX.Y.Z.tgz
```

4. Create a symbolic link to the folder. Apache will utilize the symbolic link and upgrades will be simplified.

```
$ ln -s tcdb.vX.Y.Z tcdb
```

5. Configure the application to use the MySQL database.
 - a. Open the file tcdb/config/database.yml in your favourite editor.

- b. In the 'production' section, set the values for database, username and password.
- c. If your database is hosted on a different server, add an entry titled host and add the IP or hostname of your server. A sample is shown below.

```
production:
  adapter: mysql2
  database: <database_name>
  encoding: utf8
  username: <database_user>
  password: <database_password>
  pool: 5
  timeout: 5000
  host: <database_ip_or_hostname>
```

6. Make sure you are still in the root of the application, /home/tcdb/tcdb in our example and run the setup script. This will install the required gems and prepare the database.

```
$ script/setup -p
```

7. Prepare the database for first use.

```
$ rake db:migrate RAILS_ENV="production"
```

8. Prepare the asset pipeline.

```
$ rake assets:precompile RAILS_ENV="production"
```

4.3.6 Updating SELinux

If you keep the TCDB application in /home/tcdb/tcdb, Apache will not be able to access the site with SELinux enabled. While we highly recommend that the user leaves SELinux disabled, the task is left to the user. To get TestCaseDB running we will place SELinux in permissive mode.

1. Open `/etc/selinux/config` in your text editor and make the change below:

```
SELINUX=enforcing
# change the line above to
SELINUX=passive
```

2. Reboot the system.

4.3.7 Configure the Apache Host

A virtual host must be configured to host the site. It is assumed that this will be a named virtual host. If this is the only website on the system, you may configure directly in `httpd.conf`. More details are available in Apache's documentation.

1. Create a `tcd` folder in the `httpd` config folder and add a configuration file.

```
$ cd /etc/httpd/conf
$ sudo mkdir tcd
$ sudo vi tcd/qa.yourdomain.com.conf
```

2. Add the content below to the file created in step 1. It is assumed that only SSL access will be allowed. Change all upper case text as appropriate.

```
<VirtualHost *:80>
  ServerName qa.yoursite.com
  Redirect / https://qa.yoursite.com/
</VirtualHost>
<VirtualHost *:443>
  ServerName qa.yoursite.com
  DocumentRoot /home/tcd/tcd/public
  SetEnv SERVER_NAME qa.yoursite.com
  RailsEnv production
  XSendFile On
  XSendFilePath /home/tcd
  <Directory /home/tcd/tcd/public>
    Allow from all
    Options -MultiViews
  </Directory>
</VirtualHost>
```

3. Add the configuration file from step 1 to the main Apache config. This is done by adding the line below to the end of `/etc/httpd/conf/httpd.conf`.

```
Include ../conf/tcdb/*.conf
```

4. Restart Apache.

```
$ sudo service httpd restart
```

4.3.7.1 Error Restarting Apache

If you encounter the following error while restarting apache that says, “Invalid command 'XSendFileAllowAbove’”, changes will need to be made to the site configuration. The cause of this error is that versions of the xsendfile module below 0.10 do not recognise the XSendFilePath command. To resolve the issue, remove the XSendFilePath token and add, “XSendFileAllowAbove On”, to the configuration. Now restart Apache again.

4.3.8 Firewall Ports

By default, ports 80 and 443 may be blocked on your system using iptables. The ports should be opened to allow traffic to reach your web server. It is recommended that an administrator in your company decide the most effective policy for your corporation.

4.3.9 Configure Logrotate

Without log rotation, the TCDB logs will grow without bound and eventually fill the hard drive. It is recommended that logrotate be configured to put a limit on the logs. To enable logrotate, create a configuration file in /etc/logrotate.d/ called tcdb with the contents below. Remember to change the path to that of your system.

4.4 Final Configuration, Install Script and Initial Login

4.4.1 Email Configuration

Before running the install script, email settings must be configured. Email can be sent using sendmail or smtp. Instructions for both are provided.

4.4.1.1 Configuring SMTP

To configure outgoing email via SMTP use these steps.

1. Open the file `tcdB/config/initializers/setup_mail.rb` in your favourite text editor.
2. Set all of the fields to match those of a user on your smtp server.
3. Save the changes.

4.4.1.2 Configure Sendmail

To configure outgoing email via Sendmail use these steps.

1. Open the file `tcdB/config/initializers/setup_mail.rb` in your favourite text editor.
2. Remove the sample data.
3. Add the lines in the box below to configuration files.

```
ActionMailer::Base.smtp_settings = {  
  :address => "localhost"  
}  
ActionMailer::Base.default_url_options[:host] =  
"localhost:3000"
```

4. Save the changes.

4.4.2 Disable HTTPS

By default, TestCaseDB is designed to be hosted using HTTPS. If you plan on hosting using the insecure protocol HTTP, one configuration change is required. The steps below describe that change.

1. Open `<testcasedb_root_directory>/config/application.rb` in you favourite text editor.
2. Change the line, `'config.ssl_enabled = true'` to, `'config.ssl_enabled = false'`.
3. Save the changes.

Warning: For security purposes, TestCaseDB recommends that you use HTTPS. By disabling HTTPS, all passwords and communication will be made in plain text with the server.

4.4.3 Default Session timeout

By default, the session timeout is one hour. TestCaseDB users are logged out once the session timeout is hit. If you would like to change the session timeout value follow the steps in this section.

1. Open <testcasedb_root_directory>/config/application.rb in your text editor.
2. Find the line, 'config.session_timeout = 60'.
3. Change the timeout value to the desired interval. The time is measured in minutes.
4. Save the changes.

4.4.4 Scheduler Crontab

If your team will be utilizing the automation tools, a cron job must be created to trigger the scheduler. The cron job should run the rake script every five minutes. A sample is provided below. The sample assumes that TestCaseDB is installed in /home/tcdb/tcdb and should be entered on a single line.

```
*/5 * * * * cd /home/tcdb/tcdb; /usr/local/bin/rake sched:start  
RAILS_ENV='production'
```

4.5 Login and Change the Password

You can now open the URL and login to the application with the default user admin/ChangeMe. Next, you must change the password for the administrator user. The administrator's username should also be changed while you are editing the password, though this is not required. Steps to change the password are below.

1. Click the My Settings link.
2. On the Edit user page, enter your new password in the Password and Password Confirmation fields.
3. Click the Update User button.

4.6 Deactivate the Default Admin User

For security, it is highly recommended that you deactivate default administrator user as described in this section.

1. Log into TestCaseDB as an administrator.
2. Click the Admin drop down in the navigation bar.
3. Click the Users link.

4. Create a new user with the role Administrator.
5. Logout of TestCaseDB.
6. Login as the new administrator.
7. Click the Admin drop down.
8. Click the Users link.
9. Click the Edit link for the user admin.
10. Deselect the Active checkbox.
11. Click Update User.

5 Upgrades

This section describes how to upgrade to the latest version of TestCaseDB software.

1. Disable the website. There are two options for this. You can either shut down Apache or simply disable the virtual host. Instructions for both are provided. Do either A or B.

- a. Shutdown Apache with the init.d script.

```
$ sudo /etc/init.d/apache2 stop
```

- b. Disable the virtual host.

- i. Ubuntu. Disable the site using the Apache command a2dissite..

```
$ sudo a2dissite qa.yourdomain.com
```

- ii. Red Hat/CentOS. Rename the configuration file to a value that is not recognized by Apache and gracefully restart Apache.

```
$ cd /etc/httpd/conf/tcdb
$ sudo mv qa.yourdomain.conf
qa.yourdomain.conf.off
$ sudo apachectl2 graceful
```

2. Backup the database.

```
$ mysqldump --add-drop-table -u root -p <database_name> >
/path/to/backup/db_tcdb_backup_$(date +%y-%m-%d).sql
$ nice -10 bzip2 /path/to/backup/db_tcdb_backup_$(date +%y-%m-%d).sql
```

3. If you do not have a backup of the install, it is recommended that you make one.

If the application folder contains a lot of data, especially logs and attachments, be sure that the server has sufficient amount of space for the backup.

```
$ cd /path/to/tcdb/folder
$ tar -czvf tcdb.$(date +%y-%m-%d).tgz tcdb
```

4. Unpack the latest version of the application. This step assumes that you used /home/tcdb as the base folder. Change directories if applicable.

```
$ cd /home/tcdb
$ tar -xvzf /path/to/new/package/tcdb.vX.Y.Z.tgz
```

5. Copy configuration files from the previous release.

```
$ cp -v tcdb/config/database.yml tcdb.vX.Y.Z/config
$ cp -v tcdb/config/initializers/setup_mail.rb
tcdb.vX.Y.Z/config/initializers/
$ cp -v tcdb/config/application.rb tcdb.vX.Y.Z/config
```

6. Move the attachments to the new application folder. For safety, a copy can be done instead, but will use more hard drive space.

```
$ mv tcdb/assests/uploads/ tcdb.vX.Y.Z/assets/
```

7. Update the symbolic link to point to the new version of the application

```
$ rm tcdb
$ ln -s tcdb.vX.Y.Z/ tcdb
```

8. Run the migration and enable a Phusion Passenger restart.

```
$ cd tcdb
$ bundle install
$ rake db:migrate RAILS_ENV="production"
$ touch tmp/restart.txt
```

9. Compile the asset pipeline.

```
$ rake assets:precompile
```

10. Re-enable the Apache host.

- a. If you shutdown Apache in step 1, start it again.

```
$ sudo /etc/inti.d/apache2 start
```

- b. If you disabled the virtual host, enable it with the instructions that suit your system.

- i. Ubuntu. Enable the TestCaseDB site using the Apache command a2ensite.

```
$ sudo a2ensite qa.yourdomain.com
```

- ii. Red Hat/CentOS. Replace qa.yourdomain.com with the name of the file that you configured.

```
$ cd /etc/httpd/conf/tcdb
$ sudo mv qa.yourdomain.conf.off
qa.yourdomain.conf
$ sudo apachectl2 graceful
```

11. Load the TestCaseDB page and verify that users can log in.

6 Backup

Backing up the system is a task that should be done regularly. When backing up TestCaseDB, there are two important items that need to be backed up.

- The Database
- Uploaded files

6.1 Database

Backups of the entire database should be kept. There are numerous options available. Speak with your database administrator to find the solution that works for you.

- mysqldump of the database
- Replicate to another MySQL server
- Copy the database files directly. There are scripts to do 'HotCopies' of live databases

6.2 Uploaded Files

All uploaded files are stored in the TestCaseDB application hierarchy. Files are located in tcdb/assets/uploads/. A backup of this directory should always be kept.

Numerous methods can be used and it is left to the customer to backup the files.

7 Administration Tasks

7.1 Create a User

1. Log into the system as an administrator.
2. Click the Admin drop down in the navigation bar.
3. Click the Users link.
4. Click the New User button.
5. Fill out all of the fields on the form. For details on the different roles, [Appendix A: User Roles](#)
6. Click the Active checkbox to activate the user.
7. Select which products the user has access to in the Authorized Products section.
8. Click the Create User button.

7.2 Change a User's Password

7.2.1.1 Change Your Password

1. Click the My Settings link.
2. On the My Settings page, enter your new password in the Password and Password Confirmation fields.
3. Click the Update User button

7.2.1.2 Change Another User's Password

1. Click the Admin drop down in the navigation bar.
2. Click the Users link.
3. Click the Edit link for your user entry.
4. On the Edit user page, enter your new password in the Password and Password Confirmation fields.
5. Click the Update User button.

7.3 Deactivate a User

When a user no longer needs access to your system their account should be deactivated.

1. Log into the system as an administrator.
2. Click the Admin drop down in the navigation bar.
3. Click the Users link.
4. Click the Edit button for the user.
5. Deselect the activate check box.
6. Click the Update User button.

7.4 Unlock a User Account

After five invalid login attempts, a user's account will be locked out for security reasons.

Administrators can unlock the account using these steps.

1. Log into the system as an administrator.
2. Click the Admin drop down in the navigation bar.
3. Click the Users link.
4. Find your user and click their Unlock link.
5. If the user requires a new password, click the edit link and change the password on the edit page.

7.5 Create Test Types

By default, the system includes test types *Manual* and *Automatic*. If you desire different types, follow these steps.

1. Log into TestCaseDB as an administrator.
2. Click the Admin drop down in the navigation bar.
3. Click the Test Types link.
4. Click the New Test Type button.
5. Give the type a name.

6. If desired, enter a description for the test type.
7. Click Create Test type.
8. The new test type is displayed. Click the back button to return to the list.
9. Repeat steps 4 – 8 as many times as desired.

7.6 Create Products

Entries for different products can be created using the steps in this section.

1. Log into TestCaseDB as an administrator.
2. Click the Admin drop down in the navigation bar.
3. Click the Product link.
4. Click the New Product button.
5. Enter the name of the product.
6. If desired, enter a description for the product.
7. Click Create Product.
8. Repeat steps 4 – 7 as many times as desired.

7.7 Authorize Users for New Products

Users can only access products that they have access to. Use these steps to authorize multiple users for a single product.

1. Log into TestCaseDB as an administrator or QA manager (the manager must have access to the product).
2. Click the Admin drop down in the navigation bar.
3. Click the Product link.
4. Click the Edit link for the product that you want to authorize users to use.
5. In the Authorized Users section, select all the users that you want to grant access to.
6. Click the Update Product button.

7.8 Create Categories

Categories need to be created for each Product in the system. Careful thought should be given to this step, as it will define how your test cases are organized. Sample categories are provided in the screenshot below to give ideas on how test cases can be organized into categories, but customers should not feel limited to this kind of organization.

Categories can break a product into features, testing methods, test tools or any other breakdown that suits your regular workflow.

7.8.1 Example Category Setups

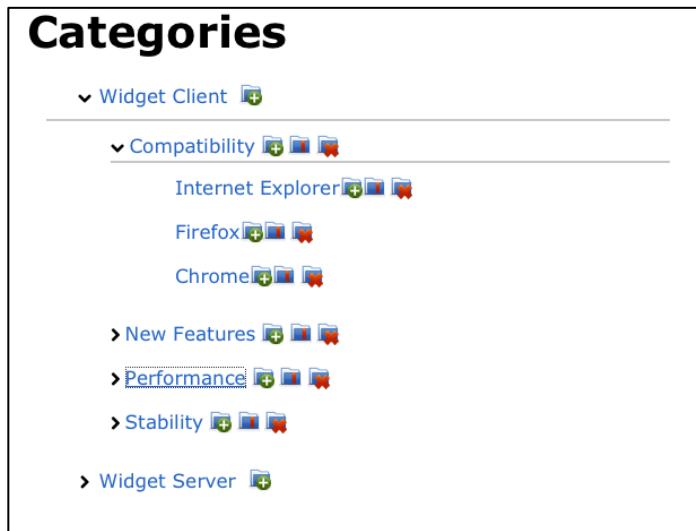



Figure 1 - Sample categories

7.8.2 Configuring Categories

1. Log into TestCaseDB as an administrator.
2. Click the Admin drop down in the navigation bar.
3. Click the Category link.
 - The category page loads. Initially, a list of products is displayed. Clicking on a product name or category name will display the list of sub-categories.

4. To add a category to a product or a sub-category to an existing category, click the  icon beside the existing product or category.
5. Enter a name for the category in the name field and then click Create Category.
 - If the list for the existing item had already been expanded, the new item will be added to the list. If the list is not visible, the new item will appear when the user expands the list.
6. Repeat steps 4 and 5 to create all desired categories.

7.9 Create Tags

In addition to categories, tags can be used to classify test cases. Unlike categories, tags are optional and test cases can have zero, one or multiple tags. QA teams can use tags in anyway they see fit. One common use is indicating applicable platforms for test cases. For example, if you are testing a mobile application, your tags could be a list of mobile operating systems and browsers.

Instructions for creating tags are given below.

1. Log into TestCaseDB as an administrator.
2. Click the Admin drop down in the navigation bar.
3. Click the Tags link.
4. Click the New Tag button.
5. Give the tag a name.
6. Click Create Tag.
7. Repeat steps 3 – 6 as many times as desired.

7.10 Create Versions

Each product has a unique set of versions. Assignments and results are tracked against versions. Initial versions can be created early on and users will continuously add new versions to the system to correspond with new versions of your software using the steps below.

1. Log into TestCaseDB as an administrator.

2. Click the Admin link in the right corner.
3. Click the Versions link.
4. Click the New Versions button.
5. Select a product that this version will belong to.
6. Enter a version in the version field.
7. Optionally, add a description.
8. Click the Create Version button.

7.11 Creating Custom Fields

It is possible to define custom fields for use on test cases, test plans and devices. Custom fields can be defined at any time, but it should be noted that fields are only added to existing items when they are created or edited. Therefore, if you define a custom field after a test case is created, that field will only appear on the test case the next time it is edited. The custom field will appear on all new test cases automatically.

7.11.1 Defining the Field

1. Log into TestCaseDB as an administrator or qa manager.
2. Click the Admin link in the right corner.
3. Click the Custom Fields link.
4. Click the New Custom Fields button.
5. Select an Item Type. This will define which type of item the field will appear on.
6. Enter a field name. This will be the label given to the field on forms and view pages.
7. Select a field type. The options are:
 - a. Check Box: This will place a check box beside the label.
 - b. String: Provides a text box for text.
 - c. Number: Provides a field to enter numbers. Integer and floating points are accepted.
 - d. Radio Button: Provides a set of radio buttons for the user to select. The radio items are defined in the Accepted values section with a comma-separated list.
 - e. Drop Down: Provides a select list. The items in the list are defined in the Accepted values section with a comma-separated list.

8. If radio button or drop down were selected in the previous step, fill in the list of accepted values.
9. Verify that the Active check box is enabled.
10. Click the Create Custom Field button.

7.12 Configuring Result Requirements

TestCaseDB allows administrators to configure what information is required to mark a test case failed or blocked. For both scenarios, you can individually select if bugs and/or comments are required. As an additional benefit, if ticket system integration is configured, the system will also verify that bug numbers can be found in the ticket system.

Configure the requirements using these steps.

1. Log into TestCaseDB as an administrator.
2. Click the Admin drop down in the navigation bar.
3. Click the Settings link.
4. Click the Edit link for one of the following:
 - Require Bug For Failed Result
 - Require Comment For
 - Require Bug For Blocked Result
 - Require Comment For Blocked Result
5. Click the enabled checkbox if you would like to enable the requirement.
6. Click Update Setting.
7. Repeat steps 4 – 6 for the remaining settings.

7.13 Create Devices

Devices serve two purposes in TestCaseDB:

- 1) The scheduler is able to provide separate queues for different resources. For use with automation, more information is available in the automation section.
- 2) They help indicate what resources; software or other item a test plan should be run against when it is used in a test plan. It is easy to provide these details using device custom fields.

Instructions for creating devices are given below.

1. Log into TestCaseDB as an administrator.
2. Click the Admin drop down in the navigation bar.

3. Click the Devices link.
4. Click the New Device button.
5. Give the tag a name and description. A detail name is recommended, as this is how stencils will refer to the device.
6. If this device will be used as an automation queue set the active field.
7. Fill out the custom fields.
8. Click Create Device.
9. Repeat steps 4 – 8 as many times as required.



8 Create Test Cases, Test Plans and Stencils

8.1 Create Test Cases

Test cases can be entered into the system manually using forms or they can be imported into the application from XLS spreadsheets. Instructions for both methods are provided in this section.

8.1.1 Creating Test Cases in TestCaseDB Using Forms

To create test cases in the application follow these steps.

1. Log into TestCaseDB.
2. Click Test Cases link in the header and then click List.
3. Clicking on product and category names, navigate to the category you would like to add the case to.
4. Click the  icon beside the category that you would like to add the test case to.
5. Fill in all of the fields on the test case form.
6. Create the steps for the test case.
 - a. To add additional steps, use the Add Step button in the steps section.
 - b. Use the delete button to remove unwanted steps.
 - c. Re-order steps by clicking and dragging the  icon.
7. Save the changes. There are two options for saving the changes.
 - a. Click the Create Test Case button to save the changes and view the case.
 - b. Click the Save and Create Additional button which will save the changes to this test case and open a new test case. The product and category on the new test case will match this test case.

8.1.2 Import Test Cases into TestCaseDB Using XLS

To import test cases into TestCaseDB follow the simple steps below.

1. Log into TestCaseDB.
2. Click the Test Cases link in the header and then click List.

3. Click the Import Test Cases button.
4. On the import page, download the sample spreadsheet.
5. Fill out the spreadsheet following the rules in [Rules for importing test cases](#).
6. If you closed the import page, re-open it.
7. Select a Product for the test cases.
8. Select a category for the test cases.
9. Select the file to download in the file section.
10. Click Import Test Cases.

8.1.2.1 Rules for Importing Test Cases

The rules for XLS test case imports are below. Reference Figure 2 for a sample spreadsheet.

1. The XLS file must include the header line.
2. Test Cases are defined on lines that include values for the Test Case Name, Description, Type and Step. Values entered for Result are optional.
3. Lines with only Step and optionally the result value are considered additional steps for the previous test case.
4. All of the test cases in a single file can only be downloaded to a single category.
5. Content must be placed on the first worksheet of the spreadsheet.

	A	B	C	D	E
1	Test Case Name	Description	Type	Step	Result
2	Sample Case 1	This is the first downloaded case	Manual	Step 1	
3				Step 2	Result 2 for case 1
4	Sample Case 2	This is another sample case	Manual	Step for case 2	Result for case 2
5				Second step for case 2	Second result for case 2
6				Third step for case 2	Third result for case 2
7				Fourth step with no result	
8				Fifth step for case 2	The fifth result for case 2

Figure 2 - Sample test case import file

8.2 Organize Test Cases into Test Plans

Test plans are collections of test cases that should be executed together. Test cases belong to a single product, but they may contain test cases from other products.


8.2.1 Create a Test Plan

Test plans can be created using the steps below.

1. Log into TestCaseDB.
2. Click the Test Plan link in the header then click New.
3. Fill in all fields on the test plan form.
4. At this point you can click Create Test plan which will show you the created test plan or you can click Save and Add Test Cases which will take you to a page that will allow you to add more test cases right away.

8.2.2 Adding Test Cases to Test Plans

1. Open the edit view of the test plan if it is not already open.
2. Browse the Available Test Cases section by clicking on the product and category names.

3. To add a test case to a plan, click the Add Case link. The case will be added to the list above and removed from the available cases list.
4. Repeat 2 and 3 until completed.
5. Cases are organized in the order they are added. If they need to be re-ordered drag and drop them using the  icon.
6. Click the Update Test Plan button to save all changes.

8.3 Organizing Test Plans into Stencils

Test Plans can be assigned to users for execution. However, there is an optional step to help further group sets of tasks called Stencils.

Stencils are groups of test plans with associated devices. Like test plans, stencils can be assigned for execution. The power of stencils is that multiple executions of the same plan can be tracked.

For example, let us assume that you have product X that is supported on three different operating systems running two different database systems. For each combination of database system and operating systems you run a simple regression for each release. However, on the most popularly used operating system you run an extended regression for each database. Stencils provide a powerful tool in assigning the extended regression for each release to make sure that each scenario is covered. To cover this scenario you would generate 6 devices to represent the different combinations of operating systems and databases. Then add 6 rows to a stencil and indicate which plan is run against which device.


8.3.1 Create a Stencil

Test plans can be created using the steps below.

1. Log into TestCaseDB.
2. Click the Stencils link in the header then click New.
3. Fill in all fields on the stencil form.

4. At this point you can click Create Stencil plan which will show you the created stencil, or you can click Save and Add Test Plans which will take you to a page that will allow you to add test plans right away.

8.3.2 Adding Test Plans to Stencils

1. Open the stencil's edit view if it is not already open.
2. Click the Add Test Plan button. A row will be added to the test plans list.
3. Select the product and then corresponding test plan that you would like run.
4. Select the device that the test plan should be run against. This will indicate to the tester what they should test against.
5. Repeat 2 through 4 until completed. Note that you can repeat test plans and devices as many times as needed.
6. Plans are organized in the order they are added. If they need to be re-ordered drag and drop them using the  icon.
7. Click the Update Stencil button to save all changes.

8.4 Locking and Versioning

TestCaseDB allows you to lock test cases, test plans and stencils once they have been executed and assigned. This prevents changes to test cases, which allows for more accurate tracking. However, it is possible that test cases may need to change over time. To deal with this feature, new versions of test cases and plans can be created.

8.4.1 Configuring Locking

If your team requires test cases to be locked once they have been executed, follow these steps.

1. Log in as an administrator.
2. Click the Admin drop down in the navigation bar.
3. Click the Settings link.

4. Click the Edit link for “Allow Test Case Edit After Execution”.
5. Click the enabled checkbox to remove the checkmark.
6. Click the Update Setting button.

If your team requires test plans and stencils to be locked once they have been assigned for execution, follow these steps.

1. Log in as an administrator.
2. Click the Admin drop down in the navigation bar.
3. Click the Settings link.
4. Click the Edit link for “Allow Test Plan Edit After Assignment”.
5. Click the enabled checkbox to remove the checkmark.
6. Click the Update Setting button.

8.4.2 Versioning

To create a new version of a test case, open the test case in the browser and click New Version. The new version will be created and will contain the items below.

- Identical name
- The new test case’s version will incremented by one
- The initial case will be set to deprecated
- All other values with the exception of comments will be copied to the new test case.

Note that when you view your deprecated test case in list views, the name of the test case will have a line through it to indicate that it is a deprecated version. To create a new version of a test plan, open the test plan and click New Version.

8.4.3 Versioning vs. Copying

In addition to creating new versions of test cases and test plans, you can copy test cases and test plans. The major difference is that a copied plan or case is not linked to the original. For copied items, the version is set to 1, the name is changed to include COPY, and the remaining items are copied except for comments.

9 Assign, Execute and Monitor Testing Progress

9.1 Assign and Execute a Test Plan or Stencil

An assignment must be created for a tester to execute a test plan. An assignment links the test plan to a product, version and user that will execute. Steps to assign a test plan.

1. If a version has not been created for this run, create it now. Instructions are available in [Create Versions](#).
2. Click Test Assignments in the header then click List.
3. Click the New Assignment button.
4. Select a product and version
5. Select either a test plan or stencil to be executed.
6. Add any notes applicable for this run.
7. In the task components select the user that will execute the test plan and set the due date.
8. Click the Create Assignment button.

Notes:

- When the assignment is created, a task is automatically created for the user and it appears on their home page when they log in.
- An email is sent to the user to notify them of the pending task.

9.2 Executing an Assignment

1. Log into TestCaseDB
2. Click the Test Assignment item in the header then click Execute button.
3. Click the Execute link for the assignment you will be executing.
4. Click the Execute button for the test case you want to execute.
5. Follow the steps in the test case.
6. If you have an attachment to add, use the Add Attachment link now.
7. Set the Result for the test case and any other required details.

8. Click Save and Execute Next to run the next test case in the test plan.

9.3 View Reports

There are several kinds of reports in the application. To create and run a report, follow the steps below. Details for each report are in [Appendix B: Report Types](#). Users should note that they can only see and run the reports that they create.

1. Log into TestCaseDB.
2. Click the Reports drop down then click List.
3. If the desired report exists, click its Run link. Otherwise, continue to step 4.
4. Click the New Report link.
5. Select the report type.
 - Based on the selected report type, different options will be displayed. Fill out the fields as required.
6. Click Create Report.
7. Click the Run link.

10 Ticket System Integration

TestCaseDB is designed to integrate with ticketing software. Integration allows TestCaseDB to verify data accuracy and improve reporting abilities. Instructions for each supported ticketing system are provided in this section.

10.1 Bugzilla

10.1.1 Support

Bugzilla is officially supported from version 4.1 and up. If issues are found with earlier versions of Bugzilla, a ticket may be submitted to support. Integration is provided using Bugzilla's webservice API which is located at `http://<bugzilla_base_url>/xmlrpc.cgi`.

10.1.2 Configuration

Configuration of access to Bugzilla requires the following items.

- Bugzilla base URL
- A user that can access bug states

The information can be used in these steps to configure the integration.

1. Log in as an administrator user.
2. Click the Admin drop down in the navigation bar.
3. Click the Settings link.
4. Click the Edit link for the Ticket System setting.
5. Select Bugzilla and click Update Setting.
6. Click the Edit link for the Ticket System Url setting.
7. Enter the base URL for your Bugzilla system, for example <https://bugzilla.yourcompany.com/> and click Update Setting.
8. Click the Edit link for the Ticket System Username setting.
9. Enter the username and click Update Setting.
10. Click the Edit link for the Ticket System Password setting.
11. Enter the password and click Update Setting.

10.1.3 Security Considerations

The Bugzilla username and password are stored in plain text in the TestCaseDB system. The password is never displayed in the application, but can be found with database access. Considering this, restrictions on the Bugzilla account should be applied.

10.2 Jira

10.2.1 Support

Jira versions 4.3.4 and up are supported by TestCaseDB. No support will be provided below this, because the Jira REST API was introduced in version 4.3.4.

10.2.2 Configuration

Configuration of access to Jira requires the following items.

- Jira API URL
- A user that can access bug states

The information can be used in these steps to configure the integration.

1. Log in as an administrator user.
2. Click the Admin drop down in the navigation bar.
3. Click the Settings link.
4. Click the Edit link for the Ticket System setting.
5. Select Jira and click Update Setting.
6. Click the Edit link for the Ticket System Url setting.
7. Enter the API URL for your Jira system. The URL is dependent on your configuration and the version of Jira that you use. For example, if your Jira instance is hosted at <http://jira.yourcompany.com/> and you are running version 4.4.1, enter the value <http://jira.yourcompany.com/rest/api/2.0.alpha1/> and Click Update Setting. Note the trailing '/'. It is important. For more details on the value for your system, items a through c below provide the required details.
 - a. For 4.x.y releases of Jira equal to or above 4.3.4 the URL is,
"http://<base_url>/rest/api/2.0.alpha1/".

- b. For 5.0.x releases, the url is, “http://<base_url>/rest/api/2/”.
 - c. For more details, take a look at <https://developer.atlassian.com/static/>
8. Click the Edit link for the Ticket System Username setting.
9. Enter the username and click Update Setting.
10. Click the Edit link for the Ticket System Password setting.
11. Enter the password and click Update Setting.

10.2.3 Security Considerations

The Jira username and password are stored in plain text in the TestCaseDB system. The password is never displayed in the application, but can be found with database access. Considering this, restrictions on the Jira account should be applied.

10.3 Mantis Bug Tracker

10.3.1 Support

Mantis versions 1.2.10 and up are supported by TestCaseDB. Issues with previous versions will be investigated though a fix is not guaranteed.

Integration with Mantis requires that MantisConnect is installed. To test if you have it installed visit http://<you_mantis_system>/api/soap/mantisconnect.php?wsdl and verify that the page loads.

10.3.2 Configuration

Configuration of access to Mantis requires the following items.

- Mantis URL
- A user that can access bug states

The information can be used in these steps to configure the integration.

1. Log in as an administrator user.
2. Click the Admin drop down in the navigation bar.
3. Click the Settings link.
4. Click the Edit link for the Ticket System setting.

5. Select Mantis and click Update Setting.
6. Click the Edit link for the Ticket System Url setting.
7. Enter the URL for your Mantis system. For example
<https://mantis.yourcompany.com/> and click Update Setting. Click the Edit link for the Ticket System Username setting.
8. Enter the username and click Update Setting.
9. Click the Edit link for the Ticket System Password setting.
10. Enter the password and click Update Setting.

10.3.3 Security Considerations

The Mantis username and password are stored in plain text in the TestCaseDB system. The password is never displayed in the application, but can be found with database access. Considering this, restrictions on the Mantis account should be applied.

10.4 Redmine

10.4.1 Support

Redmine versions 1.1.0 and up are supported by TestCaseDB. Integration is provided via Redmine's built-in REST API.

10.4.2 Configuration

Configuration of access to Redmine requires the following items.

- Redmine URL
- A user that can access bug states

The information can be used in these steps to configure the integration.

1. Log in as an administrator user.
2. Click the Admin drop down in the navigation bar.
3. Click the Settings link.
4. Click the Edit link for the Ticket System setting.
5. Select Redmine and click Update Setting.

6. Click the Edit link for the Ticket System Url setting.
7. Enter the URL for your Redmine system. For example <https://redmine.yourcompany.com/> and click Update Setting. Click the Edit link for the Ticket System Username setting.
8. Enter the username and click Update Setting.
9. Click the Edit link for the Ticket System Password setting.
10. Enter the password and click Update Setting.

10.4.3 Security Considerations

The Redmine username and password are stored in plain text in the TestCaseDB system. The password is never displayed in the application, but can be found with database access. Considering this, restrictions on the Redmine account should be applied.

11 Automation

TestCaseDB provides two mechanisms to automate software testing. The first mechanism is the API. The API allows you to integrate TestCaseDB into your existing automated testing for result tracking and analysis. The API is discussed in detail in section 13, [TestCaseDB API](#).

The second automation mechanism is the scheduler. The scheduler allows users to upload scripts to TestCaseDB, organize the scripts into test plans and schedule their execution. This section describes all the steps required to utilize the scheduler.

11.1 Scheduler

The scheduler allows teams to upload scripts to test cases for each execution at specified times.

Before continuing, it must be pointed out that allowing users to upload scripts and run them on your server has inherent risks. As such, several safety mechanisms have been included to prevent the execution of rogue scripts and improve the security of this feature. The two main safety precautions are that script execution is disabled globally by default and each test case must be approved for execution. Enabling global execution is described in this section and individual test cases are described later on.

In addition to the two precautions, it is recommended that the TestCaseDB application be run on its own dedicated server or virtual server. This step will prevent people from running scripts against your other data.

11.1.1 Enabling Global Script Execution

By default, the scheduler will not run any tests for security reasons. An administrative user must enable the script execution feature. Enabling the scheduler can be performed with the steps below.

1. Log into the system as an administrator.
2. Click the Admin drop down in the navigation bar.
3. Click the Settings link.
4. Click the Edit link for the setting Allow Attachment Execution.
5. Set the Enabled check box to true.

6. Click Update Setting.

11.2 Operational Overview

The automation scheduler works by performing the following steps.

1. Every 5 minutes, the scheduler runs to check if there are any scheduled items that need to be run for a particular device not currently in use. If there are scheduled items, the scheduler continues, otherwise it exits.
2. A queue is built for each device and threads are spawned for each device to verify that they run concurrently.
3. The first scheduled test plan is loaded and a version for the associated product is created. The version includes a time stamp to indicate when the testing was done.
4. Next, an assignment is created that combines the test plan and newly created version.
5. One by one, each test case is executed in the order specified in the test plan.
 - a. If there are multiple attachments on a test case, only the first attachment is run. (will the system go back and run the test case with the other attachments or only the first one? Otherwise, what happens to the other attachments on a test case?) Only the first one. Others are ignored!
 - b. A test case is marked as passed if the script generates an exit code of 0.
 - c. All other exit codes result in the test case being marked as failed.
 - d. All output to STDERR and STDOUT are logged in the test case result. The data is recorded in the notes section.
6. The scheduler runs any remaining test plans.
7. The scheduler records that it has run.

11.3 Creating an Automated Test Plan

Creating an automated test plan is similar to creating a regular test plan. The steps are described below.

1. A product should be created if one does not exist. It is recommended that a unique product be created for automation as a lot of data can be generated in a relatively short period of time.
2. Categories should be created for the product.

3. For each automation test, a test case should be created.
 - a. Steps can be included in the test case to explain what it does, but they are not required.
 - b. The script should be uploaded to the test case.
 - c. At this point a QA manager or administrator should open the test case and download the script. They should review that the script is safe to run and then click the Make Executable link so the scheduler can run this particular script.
4. A test plan must be created and the test cases added.
5. The test cases should be ordered in the desired execution order.
6. A scheduler item should be created.
 - a. Click the Scheduler drop down in the navigation bar then click List.
 - b. Click the New Schedule button.
 - c. Select a device that this should be queued in.
 - d. Select the product that you utilised.
 - e. Select the test plan.
 - f. Set the time and days the test plan should be run.
 - g. In the Email Recipients section, select the users you want to receive a summary of the results.

11.4 Monitoring Automated Plans

All logs can be found in <TestCaseDB_root_folder>/log/schedule.log. The logs track the full scheduler execution process. To simplify the logs, no results or output from scripts are included in the logs.

In addition, all selected recipients will receive an email after each run of a scheduled test plan. The email will include the results and links to view the data in TestCaseDB.

12 jMeter

In addition to executing normal scripts, TestCaseDB's scheduler can execute Apache jMeter test plans. This section provides details on integrating jMeter testing with TestCaseDB.

12.1 Overview

To run jMeter test cases, TestCaseDB takes jMeter test plan files stored in test cases and runs them on a remote host. This is done by loading the script to the remote host for each run, downloading the results at the end of each run, cleaning up the jMeter host and analysing the result files. More details are provided in the rest of this section.

12.2 Configuration

Additional configuration is required to utilize the jMeter integration. A listing of options is provided with a brief description.

- jMeter Host – The hostname or IP address of the target system that will execute the jMeter test plan.
- jMeter User – The username that will be used to login to the host.
- jMeter Password – The password for the user. This field is optional. It must be set if a certificate is not set. Note that this password is stored in the database in clear text.
- jMeter SSH Certificate – The path to the private certificate for SSH login. This field can be used to enter the path to a private certificate that will be used for SSH login to the jMeter host. The path should be to the location of the file on the TestCaseDB server. This field is optional. If it is not used, a password is required.
- jMeter Working Directory – The directory from which TestCaseDB will work while logged into the remote host. The jMeter test plan will be loaded here and jMeter will be launched while TestCaseDB is running from this location. An example value is `"/home/user"`.
- jMeter Application Path – The location of the jMeter jar file on the jMeter host. The location can be given as an absolute path (ex. `/home/user/apache-jmeter/bin/Apache-JMeter.bin`), or as a path relative to the working directory (ex. `Apache-jmeter/bin/Apache-JMeter.bin`).

- jMeter Max Execution Time – The maximum execution time for the script in minutes. This value should be set to a value that will allow each of your scripts to complete. Max execution is used to make sure that individual scripts do not get stuck in endless loops.

Steps are provided to configure these items.

1. Login as an admin user.
2. Click the Admin drop down in the navigation bar.
3. Click the Settings link.
4. Click the edit link for the jMeter Host setting.
5. Set the value.
6. Click Update Setting.
7. Repeat steps 4 – 6 for the remaining settings.

12.3 Operational Overview

When the TestCaseDB scheduler is running and it loads a test case that has a test type of jMeter, the following steps are performed instead of the standard automation execution.

1. The first attachment of the test case is assumed to be the jMeter test plan. The attachment is sent to the jMeter host and loaded into the configured working directory (as defined in the settings) using SCP.
2. TestCaseDB logs into the jMeter host using the credentials provided in the settings.
3. TestCaseDB starts jMeter on the jMeter host machine. It provides the test plan as the target for it to run.
4. When the jMeter run completes, TestCaseDB copies the log file and any defined target files to the TestCaseDB server and attaches them to the result for this execution.
5. TestCaseDB removes the test plan, log file and target files from the jMeter host.
6. TestCaseDB analyzes the defined target files and stores the statistics in the TestCaseDB database.
7. When the automation scheduler completes, the result of the jMeter run is included in the final email. If any of the items tested in jMeter have an average execution time that is 10% longer than its previous run, a warning is included in the email.

12.4 Creating jMeter Test Cases

Automated jMeter test cases can be created using the steps below as a guideline.

1. Create a standard jMeter test plan in jMeter using the following guidelines.
 - a. Make sure you include a Graph Result item in your test thread. Set the graph result item to save XML data and set a file name for the results.
 - b. Give your samplers clear logical names as they will be used in the reports within TestCaseDB
 - c. When saving your jMeter test plan, do not use spaces in the filename.
2. Create a test case in TestCaseDB.
 - a. Fill out all required fields.
 - b. Set the test case type to jMeter.
 - c. In the target file section enter the name of the graph result file; Note if you used an absolute path in jMeter, use it here. If it is a relative filename, use a relative value here. Be sure to set the sampler type to match the data generated by jMeter.
 - d. Attach the jMeter test plan file to the test case.
3. Create a test plan in TestCaseDB and add the test case created in the previous step to the test plan.
4. Create a scheduler item to run the test plan.

12.5 Analyzing results

After the test case is run, open the result in TestCaseDB. The result can be found easily using two methods.

- Look for the result link included in the email sent at the end of the scheduler execution.
- In TestCaseDB, click Test Assignments, find the corresponding assignment and click show. Then click view result for the desired item.

The result will show a graph displaying average time, standard deviation and number of samples for each request type. It is also shown for all samples. This is useful to see if any particular request type is slowing your server down.

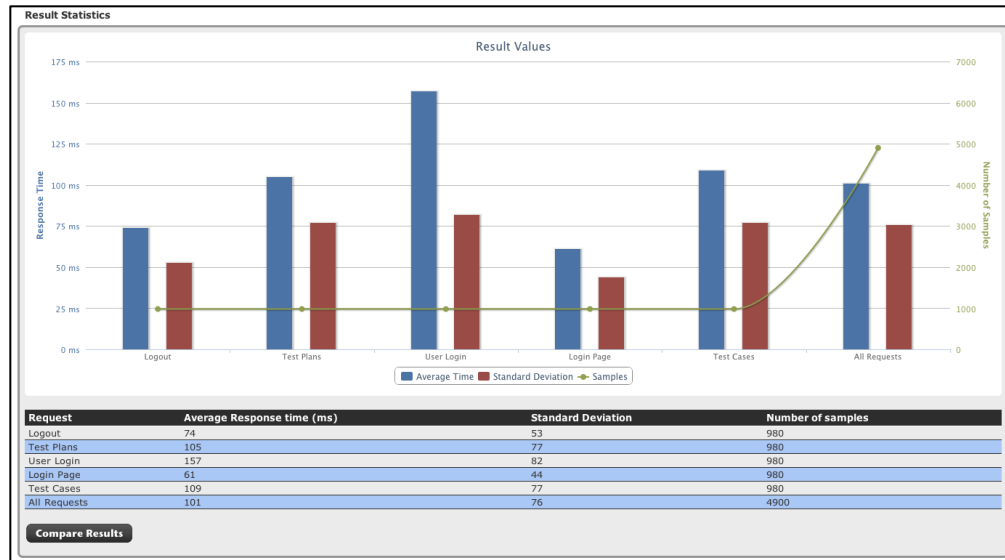


Figure 3 - Results from jMeter execution

The next step is to click the Compare Results button. This button will load a page with a graph comparing the results for each request type with previous runs of this test case (when run as part of this test plan and scheduler).

This graph is useful for comparing trends and to see if code changes have introduced performance issues. On this graph, missing statistics for request types are displayed as 0. This may be seen if a jMeter file is changed and updated on a test case.

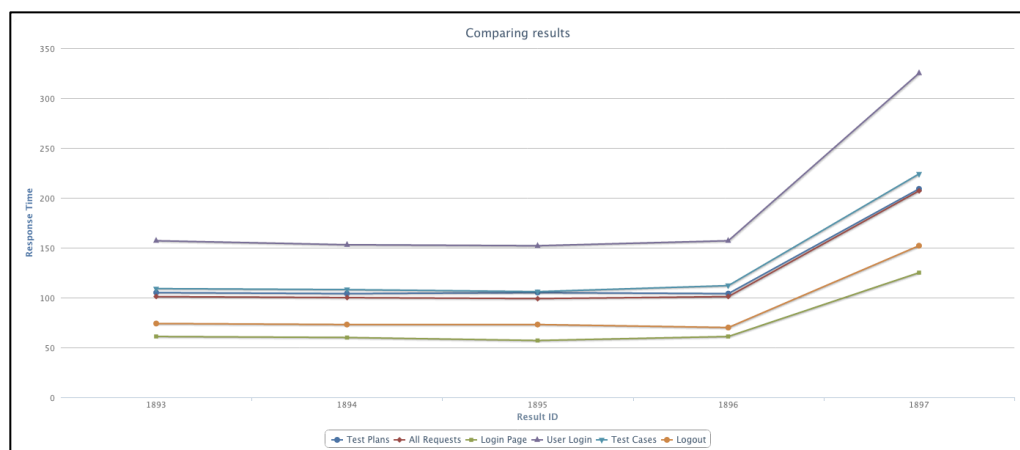


Figure 4 - Comparing results

13 TestCaseDB API

TestCaseDB provides an API for integration with automation systems. The API utilizes XML requests to allow system to add versions and results to TestCaseDB.

13.1 General Format

All requests use the format displayed below.

```
<?xml version="1.0"?>
<api_request>
<command></command>
<option1></option1>
<option2></option2>
<option3></option3>
<option4></option4>
<value></value>
</api_request>
```

TestCaseDB returns a result using the format below.

```
<?xml version="1.0" ?>
<result>
<success></success>
<error></error>
<details></details>
</result>
```

13.2 Automation Workflow

To utilize automation via the API a full understanding of the workflow is required. An overview of the workflow is provided in figure 3.

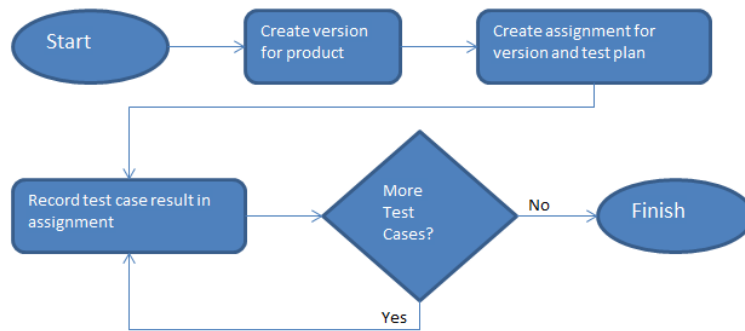


Figure 5 - Automation API workflow

The first step is to create a version for the product, because all results are associated with a version. The second step is to create an assignment for the version and test plans. The assignment is used to track results for each test case. Unlike assignments created using the web interface, assignments created via the XML API will not have associated tasks. Once the assignment is created, results for each test case can be generated.

13.3 API User

To access the API, a dedicated API user is required. To create the API user, follow the standard workflow for creating a user and select the “XML API User” role.

The API user logs in using a single access token. To find the token, open the user’s settings page and the token will be displayed at the bottom of the page.

13.4 API Commands

This section provides details for all of the available API commands.

13.4.1 CheckVersion

The check version command will verify if a version exists for a product. A list of parameters is available in the parameters table.

Parameter	Variable	Description	Required
Option1	Product Name	The name of the product	Yes
Value	Version	The version name.	Yes

A sample request where you are checking that version v1.2.3 exists for product Widget would look like the following.

```
<?xml version="1.0"?>
<api_request>
<command>CheckVersion</command>
<option1>Widget</option1>
<option2></option2>
<option3></option3>
<option4></option4>
<value>v1.2.3</value>
</api_request>
```

There are several possible responses from the command. Below is an explanation of the combination of success, error and detail values. The explanation is not included in the response and is presented in the table below for clarification.

Success	Error	Details	Explanation
Yes	No	Version found.	The version exists
No	No	Version not found for this product.	A match for this version and product name were not found
No	Yes	Invalid Product. Version not found.	The product could not be found in the system

13.4.2 CreateVersion

The create version command provides a method to create a version for a product. A list of parameters is available in the parameters table.

Parameter	Variable	Description	Required
Option1	Product Name	The name of the product	Yes
Value	Version	The version name.	Yes

A sample request where you are creating version v1.2.3 for product Widget would look like the following.

```
<?xml version="1.0"?>
<api_request>
<command>CreateVersion</command>
<option1>Widget</option1>
<option2></option2>
<option3></option3>
<option4></option4>
<value>v1.2.3</value>
</api_request>
```

There are several possible responses from the command. Below is an explanation of the combination of success, error and detail values. The explanation is not included in the response and is presented in the table below for clarification.

Success	Error	Details	Explanation
Yes	No	Version created successfully.	The version was created
No	Yes	Error creating version.	There was a general error when creating the version
No	Yes	Invalid Product. Unable to create version.	The product could not be created, because the product could not be found

13.4.3 CreateAssignment

The create assignment command provides the ability to create an assignment for a test plan and version. A list of parameters is available in the parameters table.

Parameter	Variable	Description	Required
Option1	Product Name	The name of the product	Yes
Option2	Product Version	The version that this test will be run against	Yes
Option3	Test Plan Version	The version of the test plan. Always 1	Yes

		for new test plans. The value can be seen in the test plan detail view.	
Value	Test Plan ID	The ID number of the test plan. Open the test plan in your web browser. The ID will be the integer value at the end of the URL.	Yes

A sample request where you are checking that version v1.2.3 exists for product Widget would look like the following.

```
<?xml version="1.0"?>
<api_request>
<command>CheckVersion</command>
<option1>Widget</option1>
<option2></option2>
<option3></option3>
<option4></option4>
<value>v1.2.3</value>
</api_request>
```

There are several possible responses from the command. Below is an explanation of the combination of success, error and detail values. The explanation is not included in the response and is presented in the table below for clarification.

Success	Error	Details	Explanation
Yes	No	ID number of the assignment.	When created successfully, the details value contains the ID of the assignment. The ID is required for saving results.
No	Yes	Failed to create assignment.	A general issue was encountered and the item could not be saved.

No	Yes	Assignment not created. Test Plan not found.	The assignment was not created, because the test plan could not be found.
No	Yes	Assignment not created. Version not found.	The assignment was not created, because the test plan version could not be found.
No	Yes	Assignment not created. Product not found	The assignment was not created, because the product could not be found.

13.4.4 GetResult

The get result command will pull a result from TestCaseDB. A list of parameters is available in the parameters table.

Parameter	Variable	Description	Required
Option1	Assignment ID	The ID of the assignment for the result you're looking for.	Yes
Value	Test Case ID	The ID of the test case.	Yes

A sample request where you are checking the result for test case 454 on assignment 23 would look like the following.

```
<?xml version="1.0"?>
<api_request>
<command>GetResult</command>
<option1>23</option1>
<option2></option2>
<option3></option3>
<option4></option4>
<value>454</value>
</api_request>
```

There are several possible responses from the command. Below is an explanation of the combination of success, error and detail values. The explanation is not included in the response and is presented in the table below for clarification.

Success	Error	Details	Explanation
Yes	No	The result (one of Passed, Failed, blocked, or blank)	This is the recorded result.
No	Yes	Could not find result.	The result could not be found.

13.4.5 SetResult

The set result command allows third party applications to place results into TestCaseDB.

A list of parameters is available in the parameters table.

Parameter	Variable	Description	Required
Option1	Assignment ID	The ID of the assignment for the result you're looking for.	Yes
Option2	Test Case ID	The ID of the test case.	Yes
Value	Result	The result. Must be one of, Passed, Failed or Blocked.	Yes

A sample request where you are setting the result for test case 454 on assignment 23 would look like the following.

```
<?xml version="1.0"?>
<api_request>
<command>SetResult</command>
<option1>23</option1>
<option2>454</option2>
<option3></option3>
<option4></option4>
<value>Passed</value>
</api_request>
```


There are several possible responses from the command. Below is an explanation of the combination of success, error and detail values. The explanation is not included in the response and is presented in the table below for clarification.

Success	Error	Details	Explanation
Yes	No	Result state updated.	Result was recorded.
No	Yes	Invalid result or result already set. Please consult the documentation.	The assigned test case could not be found or it was found, but already contained a result.
No	Yes	Invalid value in request. Please consult the documentation.	The value was not set to one of Passed, Failed or Blocked.

13.5 Testing the API

There are many methods available to interact with the API. For experimentation, one of the easy methods is to use the *nix command, curl. To test the commands, place the full XML request in a text file, for example, command.xml. Then run the curl request. For details on the API key field see the [API User](#) section.

```
$ curl --data-ascii @command.xml  
http://<testcase_db_base_url>/webapi/run.xml?api_key=<API_KEY>
```

14 Importing TestLink Data

TestCaseDB's import tool for TestLink is designed to work directly with the TestLink database. A direct connection to the database or a copy of the database is required. The first run of this tool should be run in a lab instance and not production. TestCaseDB allows all customers to run both a lab and production instance.

14.1 Requirements

The import tool is designed to run against TestLink schema DB1.4. Please follow the instructions included with TestLink 1.9.2 or 1.9.3 to upgrade to this database schema. A fresh install of TestCaseDB should be performed and the database should be empty, with the exception of an administrator user and settings.

14.2 What is Imported

The following items are imported.

1. Projects (as products)
2. Test Suites (as categories)
3. Test Cases
4. Test Plans
5. Results

Custom fields on the above items are not imported. In addition, TestLink allows users to run the same test case multiple times on a single release. Only the last result is imported. In addition, not all fields are imported. All import data should be reviewed.

14.3 Steps

1. Open <tcdb_root>/config/application.rb in your favourite text editor.
2. Scroll to the bottom of the file.
3. There is a section for TestLink import.
4. Change config.testlink_allowed to true.

5. Use the next 4 entries to configure the connection to the TestLink database.
6. Save the changes.
7. Restart TestCaseDB.

```
$ sudo touch <tcd_b_root>/tmp/restart.txt
```

8. Log into the TestCaseDB application as an administrator user.
9. Visit the URL https://<path_to_testcasedb>/install/testlink.
 - a. You can monitor the progress by watching the files
`<testcasedb_root_folder>/log/production.log`
10. Once complete, open `<tcd_b_root>/config/application.rb` in your favourite text editor.
11. Scroll down to the TestLink section.
12. Set `config.testlink_allowed` to `false`.
13. Save changes.
14. Restart TestCaseDB.
15. Verify that the data that was imported is correct.

15 Appendix A: User Roles

The following user roles are available in TestCaseDB.

- Administrator
- QA Manager
- Senior Tester
- Tester
- Developer
- API User

15.1 Permissions

Each role has the following permissions.

Feature	Administrator	QA Manager	Senior Tester	Tester	Developer	API User
Test Cases	All	All	All	Create, Update, Read, Comment	Read, Comment	-
Test Plans	All	All	All	Create, Update, Read, Comment	Read, Comment	-
Assignments	All	All	Create, Read, Execute	Read, Execute	Read	-
Tasks	All	All	Create, Update, Read	Create, Update, Read	Create, Update, Read	-
Reports	All	All	All	-	-	-
Products, Test Types, Versions, Categories, Tags	All	All	-	-	-	-
User Settings	All	-	-	-	-	-
Uploads	All	All	Upload, Download, delete	Upload, Download	Upload, Download	-

16 Appendix B: Report Types

TestCaseDB includes several report types. Details for each report type are below.

16.1 System Status

The system status report does not have any additional parameters. The report informs the user of the total number of products, categories, test cases and results in the system.

16.2 Release Current State

The release current state report uses two parameters. The parameters are a product and version. The report looks for all assignments related to a product and version and then shows the number of test cases. The cases are broken down into the number that have passed, failed, marked as blocked and ones that have not been run. The output is displayed in a pie graph.

16.3 Release Current State – By User

The release current state by user report uses two parameters. The parameters are a product and version. The report looks for all assignments related to a product and version breaks down the results per user. The cases are broken down into the number that have passed, failed, marked as blocked and ones that have not been run. The output is in a table with each line representing a user.

Note that this report puts a lot of stress on the database. If you want an overall summary of work for a current release, the release current state report is recommended.

16.4 Release Progress – Daily

The release progress – daily report uses four parameters, product, version, start date and end date. The start date and end date are optional. If the two items are not

entered, TestCaseDB will automatically find and populate the start and end dates. The report looks for all assignments related to a product and version. It then charts testing status over the course of the start and end date in a graph that shows the cumulative results of each day.

16.5 Compare Release Results

The compare release results report uses one parameter, product. The report calculates the result for all versions of a product and displays them in a bar graph. This is useful for comparing versions to see how much testing was done and if quality is improving.

16.6 Test Cases Without Steps

The test cases without steps report utilizes a single parameter, a product and is optional. The report generates a list of test cases that do not have steps. If a product is specified in the report, only test cases from that product are listed.

16.7 Open Tasks

The open tasks report does not have any parameters. The report generates a list of all tasks that are not in the closed state. The list includes the assignee, status, task name and due date.

16.8 Release Bug Report

The release bug report requires two parameters, a product and version. The report generates a list of all bugs found during a release. If TestCaseDB is configured to integrate with your bug system, it will also display the current status of the bugs.

17 Appendix C: Install Worksheet

17.1 Lab Setup

Hostname	_____
Username	_____
IP	_____
Install Folder	_____
TCDB FQDN	_____
MySQL Database	_____
MySQL User	_____
MySQL Password	_____

17.2 Production Setup

Hostname	_____
Username	_____
IP	_____
Install Folder	_____
TCDB FQDN	_____
MySQL Database	_____
MySQL User	_____
MySQL Password	_____

18 Appendix D: Install TestCaseDB in a Sub-URI

Some customers prefer to install TestCaseDB in a sub-directory of an existing site. For example, you may want to run TestCaseDB at <https://www.site.com/testcasedb>. This appendix provides details on what needs to be performed in order to complete this setup.

18.1 Symlink from Existing Site

TestCaseDB should not be installed in the folder that hosts your main site. Instead, perform a standard installation and create a symlink in your existing site to point at TestCaseDB. The command below assumes that your website is stored in `/var/www` and that you are keeping the TCDB application in `/home/tcdb/tcdb.vX.Y.Z/`.

```
$ cd /var/www
$ sudo ln -s /home/tcdb/tcdb.vX.Y.Z/public testcasedb
```

18.2 Apache Configuration

The Apache configuration for a Sub-URI install is slightly different. The configuration must be added to your existing virtual host configuration (see the installation section for general rules on configuring virtual hosts with your operating system).

A sample Sub URI Apache configuration is displayed below and is based on the assumptions made in this appendix.


```
<VirtualHost *:80>
    ServerName www.site.com
    Redirect / https://www.site.com/
</VirtualHost>
<VirtualHost *:443>
    ServerName www.site.com
    DocumentRoot /var/www
    <Directory /var/www>
        Allow from all
    </Directory>

    # It is assumed SSL is configured elsewhere

    # Everything below here is what is added to the config
    # file to enable SubURI TestCaseDB
    RailsEnv production
    RackBaseURI /testcasedb
    <Directory /var/www/testcasedb>
        Options -MultiViews
    </Directory>
</VirtualHost>
```

After restarting Apache you should be able to access TestCaseDB as part of your website.

18.3 Upgrades

There are few items that should be noted and that are useful when following the upgrade instructions in this guide.

18.3.1 Disable/Enable TestCaseDB

The upgrade instructions instruct the user to disable the virtual host. Disabling the complete is probably not desirable in this setup. Instead, add the token, 'PassengerEnabled off' to TestCaseDB's <Directory> section in the Apache virtual host configuration file and then gracefully restart Apache to only disable TestCaseDB. When you need to re-enable TestCaseDB, simply remove the configuration token from the configuration file and gracefully restart Apache.

18.3.2 Symbolic Link

Remember that the symbolic link is in the main website directory. The link should be deleted from the main site directory and re-created to point at the new version.