comment installer-redmine sur-debian-12

Redmine is a free and open-source project management and issue-tracking tool. It's web-based and mainly written in Ruby on Rails. It's cross-platform and supports multiple databases and multiple languages.

Redmine is flexible and can be used for different types of organizations and projects, from small, medium, or large organizations. It allows you to create and manage multiple projects, and each project has its own Wiki, Forums, issue tracking, etc. Also, it allows you to create custom roles based on your organization's needs, and many more.

Redmine is released under the GNU GPL v2 license and can be installed on any operating system such as Linux, Windows, or macOS. It supports different types of databases, including PostgreSQL, MySQL, and SQLite (default).

Follow this step-by-step guide to install the Redmine project management and issue-tracking tool on Debian 12 Server. By following this, you will install Redmine with MariaDB as the database server and Apache2 as the web server.

Prerequisites

To begin the process, ensure you have secured:

- A Debian 12 Server.
- A non-root user with administrator privileges.
- A domain name pointed to the server IP address.

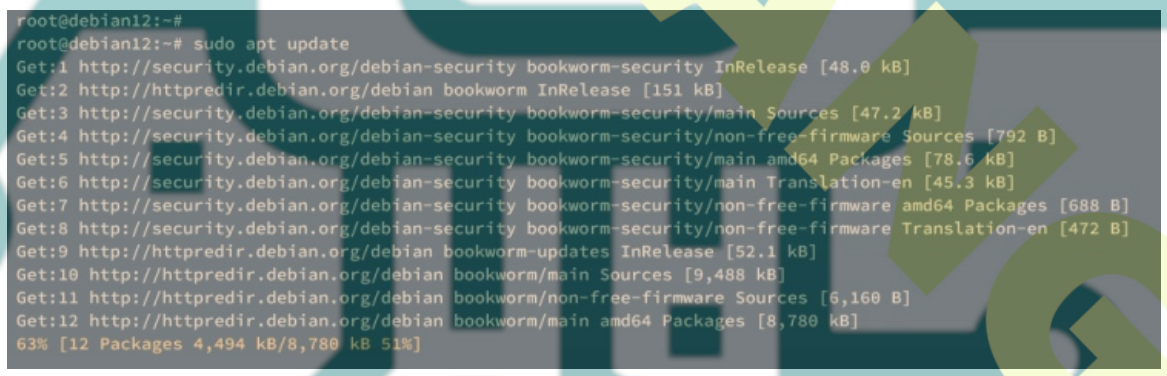
Installing Dependencies

Redmine is a web-based project management written in Ruby on Rails. To install Redmine, you must first install the following packages:

- **Apache web server**: this will be used as the web server for Redmine.
- **MariaDB server**: Redmine can be run with databases such as MySQL/MariaDB and PostgreSQL. This guide will be using the MariaDB server.
- **Ruby**: at the time of this writing, the Redmine stable version 5.0.6 can be installed with Ruby 3.1.
- Additional packages: **Certbot** for generating SSL/TLS certificates, **build-essential** for compiling Ruby code, and **Subversion** as the version control system.

Before installing dependencies, update and refresh your Debian repository using the following command.

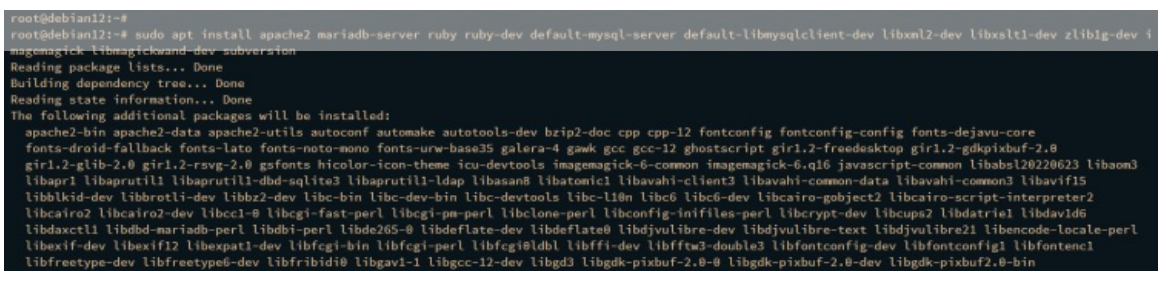
```
sudo apt update
```



Now run the apt install command below to install dependencies for Redmine, which includes *Apache2, MariaDB, Ruby, ImageMagick, Certbot,* and *Subversion*.

```
sudo apt install apache2 libapache2-mod-passenger mariadb-server certbot python3-certbot-apache ruby ruby-dev build-essential
default-mysql-server default-libmysqlclient-dev libxml2-dev libxslt1-dev zlib1g-dev imagemagick libmagickwand-dev subversion
```

Type y to proceed with the installation.



After dependencies are installed, verify each dependency by executing the following command.

Verify the *apache2* service to ensure that the service is running and enabled.

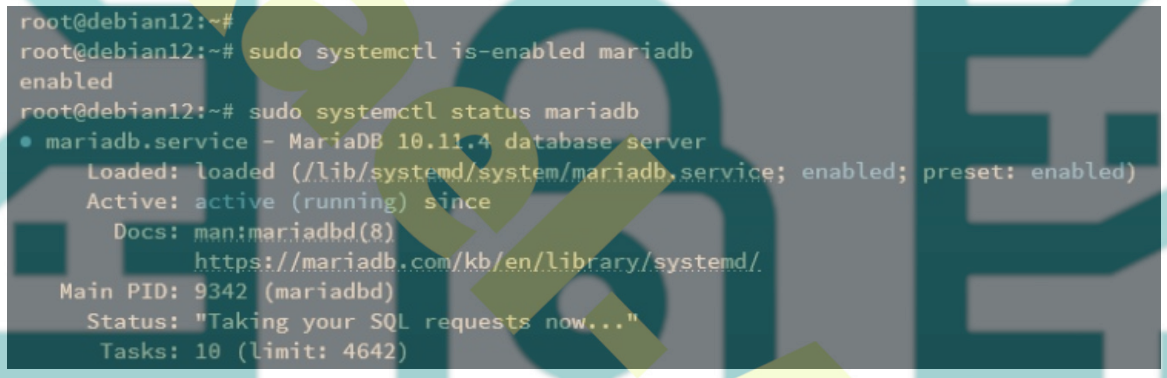
```
sudo systemctl is-enabled apache2
sudo systemctl status apache2
```

The displayed output below confirms that *apache2* is enabled and running.

Now verify the *mariadb* service by executing the following command.

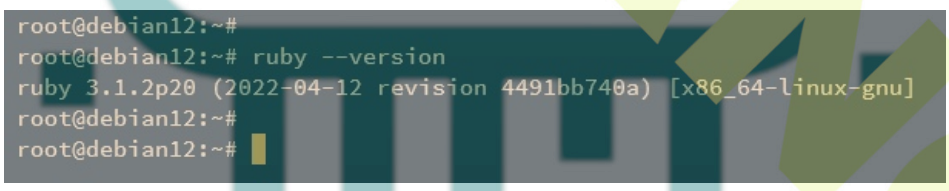
```
sudo systemctl is-enabled mariadb
sudo systemctl status mariadb
```

The output should be similar to the apache2 service, which confirms that the service is running and enabled.



Next, verify the Ruby version on your system using the following command. You should see Ruby **3.1.2** is installed on your Debian machine.

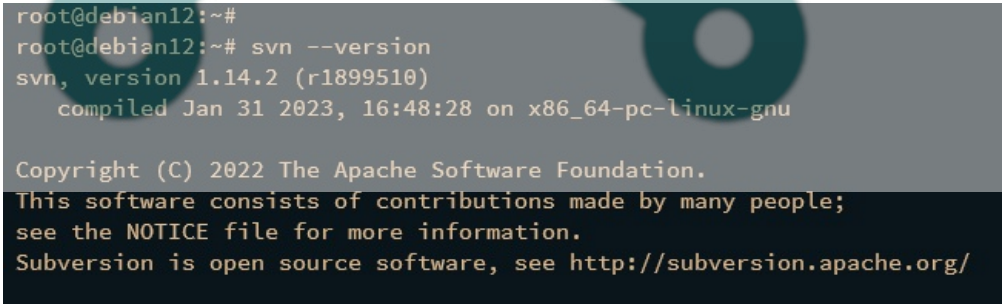
```
ruby --version
```



Lastly, verify the Subversion using the command below. This will ensure that Subversion is installed.

```
svn --version
```

The displayed output should be similar to this:



Configuring MariaDB Server

After installing dependencies, you will configure your MariaDB server installation via the *mariadb-secure-installation* utility and create a new database and user that Redmine will use.

Execute the following command to secure your MariaDB Server installation.

```
sudo mariadb-secure-installation
```

During the process, type **Y** to confirm and apply the changes or **n** for **No** to reject it. Below are some of the MariaDB Server configurations that you will be asked for:

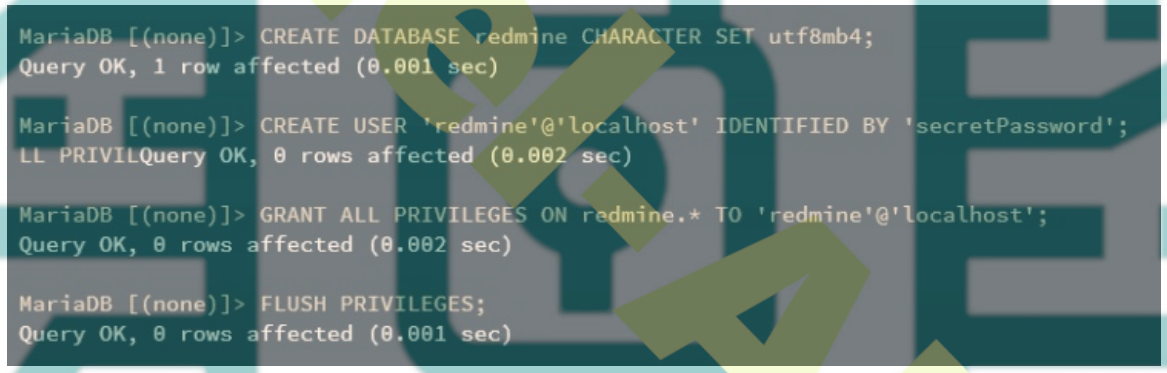
- Switch to unix_socket authentication?. Input n and press ENTER. The default MariaDB root user is already protected. optionally, you can also enable it by typing y for yes.
- Change the root password?. Input y to confirm and set up your new MariaDB root password.
- Remove anonymous user?. Input y to confirm.
- Disallow root login remotely? Input y to confirm. Only local connection will be allowed if you are using the MariaDB root user.
- Remove test database and access to it?. Input y to confirm and remove the default database 'test'.
- Lastly, input y again to reload all tables privileges on your MariaDB server and apply new changes.

After configuring mariaDB Server, log in to the MariaDB Server via the *mariadb* client command below. Type your MariaDB root password when prompted.

```
sudo mariadb -u root -p
```

Now execute the following queries to create a new database **redmine**, a new user **redmine**, with the password **secretPassword**. The following database details will be used by Redmine, and be sure to change the password.

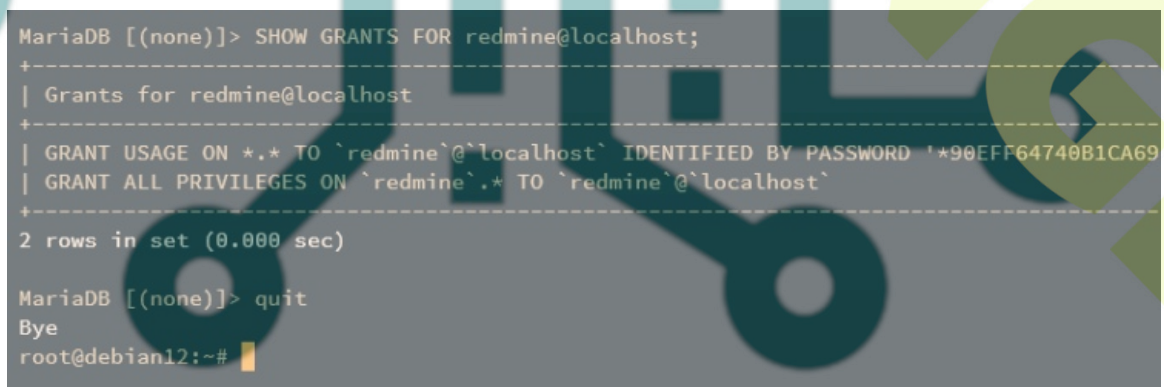
```
CREATE DATABASE redmine CHARACTER SET utf8mb4;
CREATE USER 'redmine'@'localhost' IDENTIFIED BY 'secretPassword';
GRANT ALL PRIVILEGES ON redmine.* TO 'redmine'@'localhost';
FLUSH PRIVILEGES;
```



Next, run the following query to verify the privileges for user redmine.

```
SHOW GRANTS FOR redmine@localhost;
```

The following output will be shown, which confirms that the user **redmine** can access the database **redmine**.



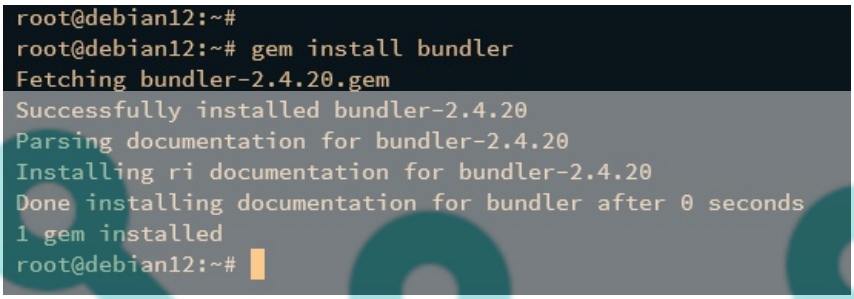
Type **quit** to exit from the MariaDB Server.

Downloading and Installing Redmine

In the following section, you will download and install Redmine on your Debian machine. You will download Redmine source code via subversion, configure Redmine with the MariaDB database server, and then install Ruby dependencies via bundler.

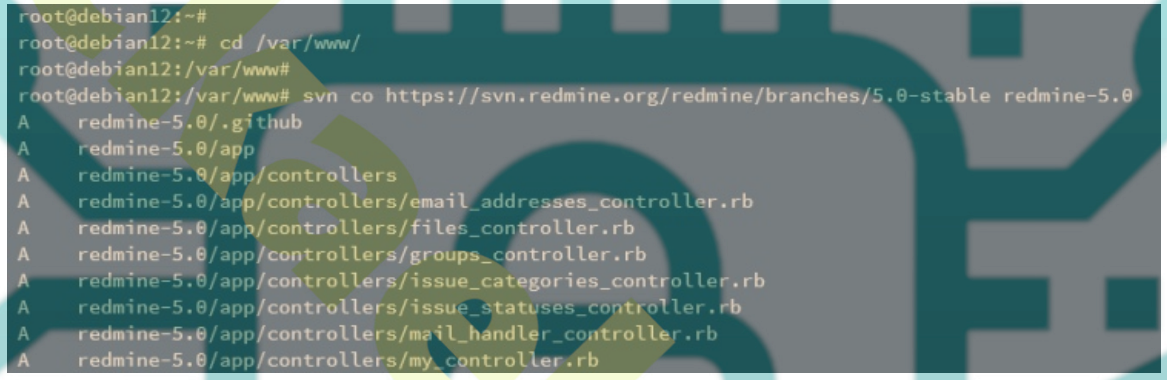
Before you get started, install a bundler to your system by executing the following command.

```
gem install bundler
```



Move to the */var/www* directory and download the Redmine source code via the *svn* command below. In this example, you will download Redmine stable 5.0 to the directory **redmine-5.0**, so your Redmine installation directory should be **/var/www/redmine-5.0**.

```
cd /var/www
svn co https://svn.redmine.org/redmine/branches/5.0-stable redmine-5.0
```



Go to the */var/www/redmine-5.0* directory and copy the default database configuration to *config/database.yml*.

```
cd /var/www/redmine-5.0
cp config/database.yml.example config/database.yml
```

Open the Redmine database configuration *config/database.yml* using the following nano editor command.

```
nano config/database.yml
```

In the **production** section, check the database configuration with the following. Be sure to change the database name, user, and password.

```
production:
  adapter: mysql2
  database: redmine
  host: localhost
  username: redmine
  password: "secretPassword"
  # Use "utf8" instead of "utfmb4" for MySQL prior to 5.7.7
  encoding: utf8mb4
```

Save and close the file when finished.

Next, run the following bundle command to disable development and test, then install Ruby dependencies for Redmine.

```
bundle config set --local without 'development test'
bundle install
```

During the process, the displayed output should be similar to this:

```
root@debian12:/var/www#
root@debian12:/var/www# cd /var/www/redmine-5.0
root@debian12:/var/www/redmine-5.0#
root@debian12:/var/www/redmine-5.0# cp config/database.yml.example config/database.yml
root@debian12:/var/www/redmine-5.0# nano config/database.yml
root@debian12:/var/www/redmine-5.0#
root@debian12:/var/www/redmine-5.0# bundle config set --local without 'development test'
root@debian12:/var/www/redmine-5.0# bundle install
Don't run Bundler as root. Installing your bundle as root will break this application for al
Fetching gem metadata from https://rubygems.org/.........
Resolving dependencies...
Fetching concurrent-ruby 1.2.2
Fetching minitest 5.20.0
Installing minitest 5.20.0
Fetching zeitwerk 2.6.12
Installing concurrent-ruby 1.2.2
Fetching builder 3.2.4
Installing zeitwerk 2.6.12
Fetching erubi 1.12.0
Installing builder 3.2.4
Fetching racc 1.7.1
Installing erubi 1.12.0
Fetching crass 1.0.6
Installing racc 1.7.1 with native extensions
Installing crass 1.0.6
Fetching rack 2.2.8
Installing rack 2.2.8
Fetching nio4r 2.5.9
Installing nio4r 2.5.9 with native extensions
```

Now generate the secret token and migrate the database by executing the following command.

```
bundle exec rake generate_secret_token
RAILS_ENV=production bundle exec rake db:migrate
```

During the database migration, the output below will be displayed.

```
root@debian12:/var/www/redmine-5.0#
root@debian12:/var/www/redmine-5.0# bundle exec rake generate_secret_token
root@debian12:/var/www/redmine-5.0# RAILS_ENV=production bundle exec rake db:migrate
== 1 Setup: migrating =========================================================
-- create_table("attachments", {:options=>"ENGINE=InnoDB", :force=>true, :id=>:integer})
   -> 0.0237s
-- create_table("auth_sources", {:options=>"ENGINE=InnoDB", :force=>true, :id=>:integer})
   -> 0.0252s
-- create_table("custom_fields", {:options=>"ENGINE=InnoDB", :force=>true, :id=>:integer})
   -> 0.0227s
-- create_table("custom_fields_projects", {:options=>"ENGINE=InnoDB", :id=>false, :force=>true})
   -> 0.0169s
-- create_table("custom_fields_trackers", {:options=>"ENGINE=InnoDB", :id=>false, :force=>true})
   -> 0.0243s
-- create_table("custom_values", {:options=>"ENGINE=InnoDB", :force=>true, :id=>:integer})
   -> 0.0234s
-- create_table("documents", {:options=>"ENGINE=InnoDB", :force=>true, :id=>:integer})
   -> 0.0181s
-- add_index("documents", ["project_id"], {:name=>"documents_project_id"})
   -> 0.0268s
-- create_table("enumerations", {:options=>"ENGINE=InnoDB", :force=>true, :id=>:integer})
   -> 0.0189s
-- create_table("issue_categories", {:options=>"ENGINE=InnoDB", :force=>true, :id=>:integer})
   -> 0.0199s
-- add_index("issue_categories", ["project_id"], {:name=>"issue_categories_project_id"})
   -> 0.0260s
```

Lastly, run the following command to load the default data to your Redmine installation.

```
RAILS_ENV=production REDMINE_LANG=en bundle exec rake redmine:load_default_data
```

If successful, you should get the output "**Default configuration data loaded**".

```
root@debian12:/var/www/redmine-5.0#
root@debian12:/var/www/redmine-5.0# RAILS_ENV=production REDMINE_LANG=en bundle exec rake redmine:load_default_data
Default configuration data loaded.
root@debian12:/var/www/redmine-5.0#
```

Configuring Apache2 Virtual Host

After you've downloaded and installed Redmine, the next step is to create a new Apache2 virtual host that will be used to run Redmine and generate SSL/TLS certificates via Certbot and Letsencrypt. So before going further, ensure that you have a domain name pointed to the server IP address.

Create a new virtual host configuration */etc/apache2/sites-available/redmine.conf* using the following nano editor command.

```
sudo nano /etc/apache2/sites-available/redmine.conf
```

Insert the following configuration and be sure to change the domain name within the **ServerName** line.

```
<VirtualHost *:80>

    ServerName redmine.hwdomain.io
    RailsEnv production
    DocumentRoot /var/www/redmine-5.0/public

    ErrorLog ${APACHE_LOG_DIR}/redmine.hwdomain.io.error.log
    CustomLog ${APACHE_LOG_DIR}/redmine.hwdomain.io.access.log combined

    <Directory "/var/www/redmine-5.0/public">
        Allow from all
        Require all granted
    </Directory>

</VirtualHost>
```

Save and close the file when you're done.

Next, run the following command to activate the **rewrite** module on the Apache2 web server, then enable the virtual host file **redmine.conf**.

```
sudo a2enmod rewrite
sudo a2ensite redmine.conf
```

After that, verify your Apache2 syntax by executing the following command. If you've proper syntax, the output "**Syntax OK**" will be displayed.

```
sudo apachectl configtest
```

Next, run the following systemctl command to restart the apache2 service and apply the changes.

```
sudo systemctl restart apache2
```

```
root@debian12:~#
root@debian12:~# sudo nano /etc/apache2/sites-available/redmine.conf
root@debian12:~#
root@debian12:~# sudo a2enmod rewrite
Enabling module rewrite.
To activate the new configuration, you need to run:
  systemctl restart apache2
root@debian12:~#
root@debian12:~# sudo a2ensite redmine.conf
Enabling site redmine.
To activate the new configuration, you need to run:
  systemctl reload apache2
root@debian12:~#
root@debian12:~# sudo apachectl configtest

AH00558: apache2: Could not reliably determine the server's fully qualified d
s this message
Syntax OK
root@debian12:~#
root@debian12:~# sudo systemctl restart apache2
root@debian12:~#
```

Lastly, generate new SSL/TLS certificates for your Redmine installation using the following certbot command. Be sure to change the domain name and email address with your information.

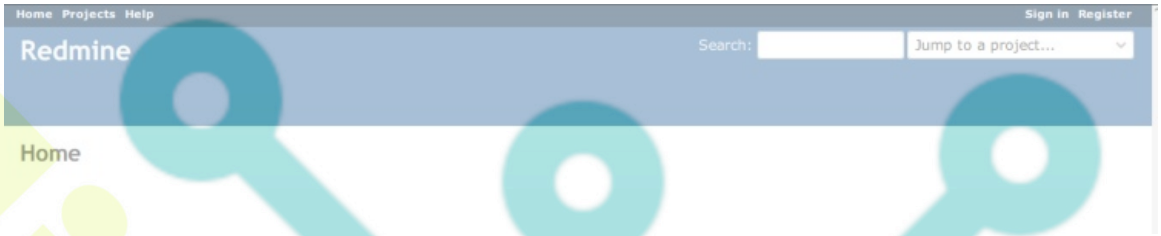
```
sudo certbot --apache --agree-tos --no-eff-email  --redirect --hsts --staple-ocsp --email admin@hwdomain.io -d redmine.hwdomain.io
```

After the process is finished, your SSL/TLS certificates will be generated to the

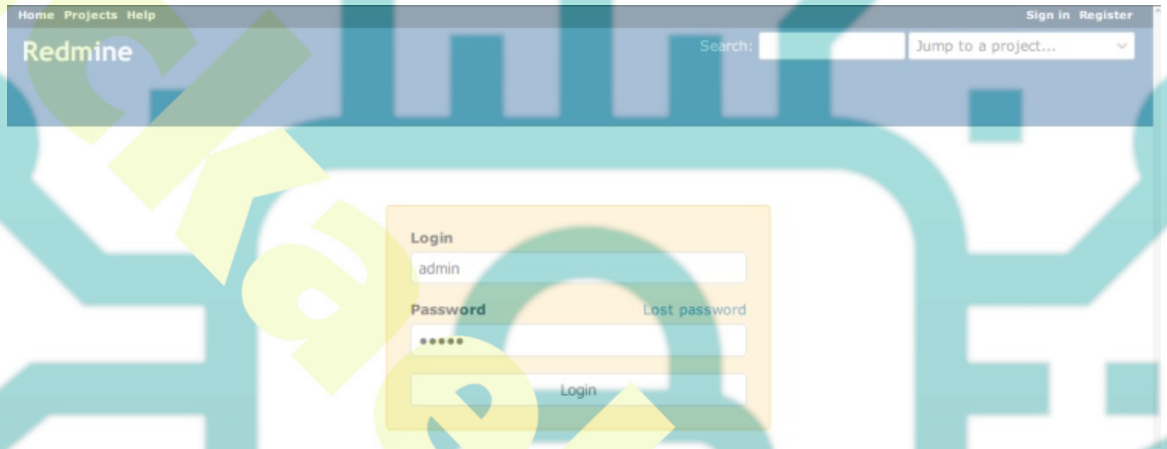
/etc/letsencrypt/live/redmine.hwdomain.io/ directory. Also, your virtual host file redmine.conf will automatically configured with HTTPS via the Certbot Apache plugin.

Accessing Redmine Installation

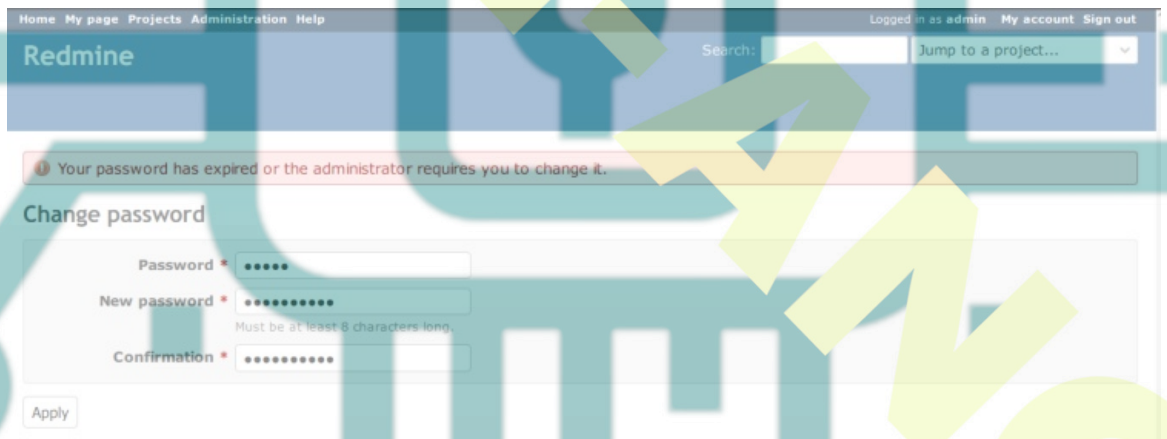
Launch your web browser and visit your Redmine domain name, such as https://redmine.hwdomain.io. If your installation is successful, the following Redmine home page will be displayed.



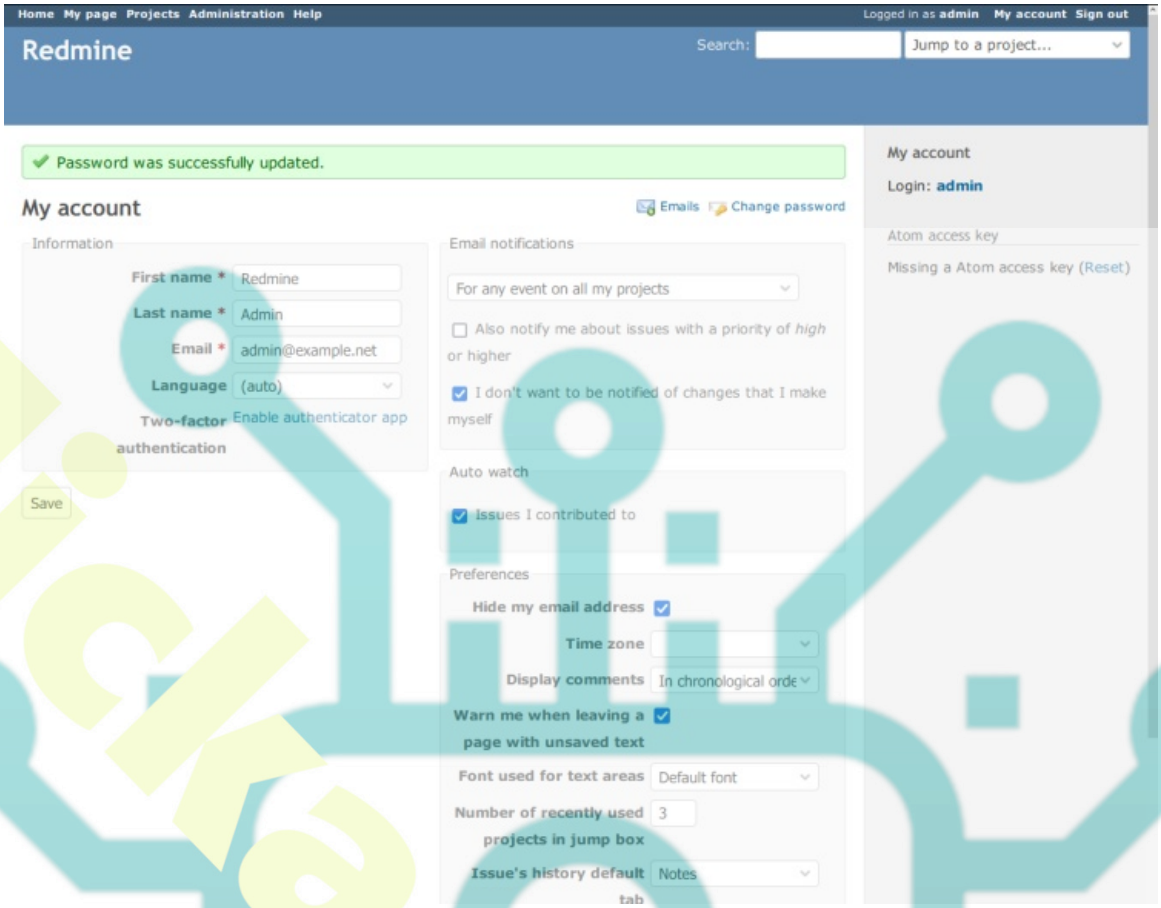
Now click the **Sign In** link on the top right to access the Redmine login page. Then, input the default user admin with password admin, then click **Login**.



First, you will be asked to change the default admin password. Input the old password admin, then input your new password and repeat, then click **Apply** to confirm the changes.



Now you will be redirected to your admin profile, and you should get the message "**Password was successfully updated**". From here, you can also change the details of your admin user, then click **Save** to confirm.



Lastly, click on Administration > Informations to get details information about your Redmine installation. The following page will be displayed, and from there confirm that Redmine **5.0.6** stable is installed with Ruby **3.1.2**, Rails 6.1, **Mysql2** database driver, and Subversion **1.14**.

Conclusion

In summary, you've successfully installed the Redmine project management and issue-tracking tool on the Debian 12 server step-by-step. You've installed Redmine with an Apache2 web server and MariaDB database server and secured your Redmine installation with SSL/TLS certificates from Letsencrypt. For here, you can now add an SMTP server to Redmine and install additional extensions and themes for your Redmine project management web application.