



**HAROKOPIO UNIVERSITY**

SCHOOL OF DIGITAL TECHNOLOGY

DEPARTMENT OF INFORMATICS AND TELEMATICS

POSTGRADUATE PROGRAMME “INFORMATICS AND TELEMATICS”

COURSE “WEB TECHNOLOGIES AND APPLICATIONS”

**Software repository aggregator for Linux distributions**

Master Thesis

**Ioannis Papadopoulos**

Athens, 2022



# **ΧΑΡΟΚΟΠΕΙΟ ΠΑΝΕΠΙΣΤΗΜΙΟ**

**ΣΧΟΛΗ ΨΗΦΙΑΚΗΣ ΤΕΧΝΟΛΟΓΙΑΣ**

**ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΜΑΤΙΚΗΣ**

**ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ “ΠΛΗΡΟΦΟΡΙΚΗ ΚΑΙ  
ΤΗΛΕΜΑΤΙΚΗ”**

**ΚΑΤΕΥΘΥΝΣΗ “ΤΕΧΝΟΛΟΓΙΕΣ ΚΑΙ ΕΦΑΡΜΟΓΕΣ ΙΣΤΟΥ”**

**Αποθετήριο πακέτων λογισμικού διανομών Linux**

Μεταπτυχιακή εργασία

**Ιωάννης Παπαδόπουλος**

Αθήνα, 2022



# **HAROKOPIO UNIVERSITY**

SCHOOL OF DIGITAL TECHNOLOGY

DEPARTMENT OF INFORMATICS AND TELEMATICS

POSTGRADUATE PROGRAMME “INFORMATICS AND TELEMATICS”

COURSE “WEB TECHNOLOGIES AND APPLICATIONS”

## **Examining Committee**

**Tsadimas Anargyros (Supervisor)**

**Teaching Laboratory Staff, Department of Informatics and Telematics,  
Harokopio University of Athens**

**Kamalakis Thomas (Examiner)**

**Professor, Department of Informatics and Telematics, Harokopio University of  
Athens**

**Michalakelis Christos (Examiner)**

**Assistant Professor, Department of Informatics and Telematics, Harokopio  
University of Athens**

## **Ethics and Copyright Statement**

I, Ioannis Papadopoulos, hereby declare that:

- 1) I am the owner of the intellectual rights of this original work and to the best of my knowledge, my work does not insult persons, nor does it offend the intellectual rights of third parties.
- 2) I accept that the Library and Information Centre of Harokopio University may, without changing the content of my work, make it available in electronic form through its Digital Library, copy it in any medium and / or any format and hold more than one copy for maintenance and safety purposes.
- 3) I have obtained, where necessary, permission from the copyright owners to use any third-party copyright material reproduced in the master thesis while the corresponding material is visible in the submitted work.

*Knowledge itself is power,*  
Sir Francis Bacon

## **Acknowledgements**

I would like to thank Mr. Tsadimas for giving me the opportunity to conduct a self-proposed thesis subject and for the excellent communication we have had during the whole thesis implementation.

## TABLE OF CONTENTS

Περίληψη.....	8
Abstract.....	9
List of figures.....	10
List of illustrations.....	11
Abbreviations.....	12
1. Introduction.....	13
1.1 Contributions of this thesis.....	13
1.2 Thesis outline.....	14
1.3 Linux packages.....	14
1.4 Linux package managers.....	15
2. Background.....	17
2.1 Package Information.....	17
2.2 Django REST Framework.....	19
2.3 SQLite.....	21
2.4 React JavaScript library.....	22
2.5 Cron Job.....	24
3. Related work.....	25
4. PKGman.....	26
4.1 Package information Collection.....	28
4.2 Back-end.....	30
4.3 Front-end.....	36
5. Conclusions and future work.....	40
5.1 Conclusions.....	40
5.2 Future Work.....	40

## Περίληψη

Η παρούσα διπλωματική εργασία έχει ως στόχο την ανάπτυξη μιας διαδικτυακής εφαρμογής λογισμικού ("PKGman"), η οποία συλλέγει περιοδικά μια σειρά από χρήσιμες πληροφορίες για πακέτα λογισμικού δημοφιλών διανομών Linux (π.χ. όνομα, διαθέσιμες εκδόσεις, μέγεθος), τις αποθηκεύει σε μια βάση δεδομένων και επίσης υποστηρίζει έναν ιστότοπο με φίλτρα αναζήτησης για την παρουσίαση τους.

Το όλο έργο αποτελείται από 3 ξεχωριστά μέρη: το μηχανισμό (python scripts εντός Docker containers) που εξάγει τις πληροφορίες, το back-end (Django REST Framework) που είναι υπεύθυνο για την εισαγωγή των εισαχθέντων χαρακτηριστικών στη βάση δεδομένων, την αναζήτηση σε αυτή με βάση συγκεκριμένα κριτήρια, καθώς και για την παροχή των αντίστοιχων αποτελεσμάτων στο front-end. Το τελευταίο κομμάτι αποτελεί το front-end (React.js), του οποίου ο ρόλος είναι να λαμβάνει τα αποτελέσματα αναζήτησης που παρέχονται από το back-end και να τα παρουσιάζει με έναν καλά δομημένο και διαισθητικό τρόπο.

Το τελικό αποτέλεσμα είναι μια διαδικτυακή εφαρμογή που μπορεί να ωφελήσει τόσο τους επαγγελματίες πληροφορικής όσο και τους απλούς χρήστες Linux. Ένας dev ops μηχανικός θα εκτιμήσει σίγουρα χαρακτηριστικά όπως το μέγεθος πακέτου και την άδεια χρήσης, ενώ σε έναν απλό χρήστη θα αρέσει η εύκολη αναζήτηση πακέτων και η λήψη εκτελέσιμων αρχείων (deb, rpm) μέσω απευθείας συνδέσμων. Συνολικά έχουν συγκεντρωθεί περισσότερα από 240.000 πακέτα λογισμικού. Το έργο υποστηρίζει επί του παρόντος τα Ubuntu 20.04, Debian 11, Kali 2021.4, Fedora 34 και CentOS 8.4.2105, αλλά μπορεί εύκολα να επεκταθεί για να συμπεριλάβει ακόμη περισσότερες διανομές που χρησιμοποιούν τον apt ή τον dnf/yum package manager. Ολόκληρη η εφαρμογή είναι containerized και επομένως μπορεί να γίνει εύκολα deploy, μέσω ενός αρχείου Docker-Compose.

**Λέξεις κλειδιά:** Διανομές Linux, πακέτα λογισμικού, αποθετήριο, συλλογή πληροφοριών, διαδικτυακή εφαρμογή



## Abstract

The purpose of the particular thesis is the development of a software web application (“PKGman”) that periodically collects a number of useful attributes about popular Linux distributions’ software packages (e.g. name, available versions, size), stores them in a database and also provides a website with search filters for their presentation.

The whole project consists of 3 distinguished parts: the mechanism (python scripts within Docker containers) which extracts the information, the back-end (Django REST framework) which is responsible for populating the database with the extracted attributes, querying it based on specific search criteria and feeding the front-end with the corresponding results. The last part is the front-end (React.js), whose role is to consume the queried results provided by the back-end and present them in a well structured and intuitive way.

The end result is a web application that can benefit IT professionals and regular linux users alike. A dev ops engineer will surely appreciate attributes such as package size and license, while a simple user will definitely like the easy package searching and binary file (i.e. deb, rpm) downloading via direct link. More than 240,000 software packages have been collected, in total. The project currently supports Ubuntu 20.04, Debian 11, Kali 2021.4, Fedora 34 and CentOS 8.4.2105, but it can easily be extended to include even more distributions that utilize the apt or dnf/yum package manager. The whole application is containerized and thus easily deployable via a Docker-Compose file.

**Keywords:** Linux distributions, software packages, repository, information collection, web application

## LIST OF FIGURES

Fig.1: installing a software package via command line, on Ubuntu.....	σ.18
Fig.2: installing a software package via the GUI application, on Ubuntu.....	σ.18
Fig.3: apt package manager - package info presentation.....	σ.19
Fig.4: dnf package manager - package info presentation.....	σ.20
Fig.5: pacman package manager - package info presentation.....	σ.21
Fig.6: crontab syntax.....	σ.25
Fig.7: Ubuntu package website - info for nano package.....	σ.25
Fig.8: Fedora package website - info for nano package.....	σ.25
Fig.9: cron job file for executing the run_all_collector_scripts.sh script.....	σ.29
Fig.10: run_all_collector_scripts.sh.....	σ.29
Fig.11: apt/dnf collector scripts pseudo code.....	σ.31
Fig.12: salsa.debian.org API's JSON response.....	σ.31
Fig.13: JWT format.....	σ.33
Fig.14: OAuth 2.0 Access / Refresh tokens flow.....	σ.34
Fig.15: Django REST serializer for Package information.....	σ.34
Fig.16: Django model for Package information.....	σ.35
Fig.17: Back-end API endpoints.....	σ.36
Fig.18: PKGman landing page with the highest rated packages in descending order.....	σ.37
Fig.19: PKGman collapsible package results with corresponding versions.....	σ.38
Fig.20: PKGman search filters.....	σ.38
Fig.21: PKGman selected table columns.....	σ.39
Fig.22: PKGman selected packages (versions).....	σ.39
Fig.23: PKGman selected packages information extracted in JSON format.....	σ.40
Fig.24: PKGman generated Dockerfile with user's selected packages (versions).....	σ.40
Fig.25: PKGman log in/sign up forms.....	σ.41

## LIST OF ILLUSTRATIONS

III.1: whole app's architecture.....	σ.28
III.2: database schema.....	σ.31

## Abbreviations

CSV	Comma Separated Values
JSON	JavaScript Object Notation
JWT	JSON Web Token
DNF	Dandified YUM
YUM	Yellow-Dog Updater Modified
API	Application Programming Interface
REST	Representational state transfer
HTTP	Hypertext Transfer Protocol
HTML	HyperText Markup Language
XML	Extensible Markup Language
JSX	JavaScript XML
DOM	Document Object Model
URL	Uniform Resource Locator
UI	User Interface
I/O	Input/output
distro	Linux DIstribution
repo	Repository
admin	Administrator
RFC	Request for Comments
RSA	Rivest–Shamir–Adleman
HMAC	Keyed Hash Message Authentication Code
ECDSA	Elliptic Curve Digital Signature Algorithm
SWHID	Software Heritage Identifier

# 1. Introduction

## 1.1 Contributions of this thesis

PKGman as an open source project itself could offer benefits to many and diverse members of the linux community.

IT professionals, especially dev ops engineers often want to compare software packages from different linux distributions in order to choose one for production environments. Packages' availability, size, license are particularly important factors when making such a decision. With PKGman they can easily have access to thousands of packages' information, relieving them from the necessity of installing multiple linux distributions locally. Not only that, but PKGman has a functionality that allows users to generate and download Dockerfiles with their selected packages and even with specific versions of their selected packages. Additional export methods, to JSON/CSV formats, are also available.

Furthermore, software developers can take advantage of PKGman, since it provides direct links to the packages' code repositories, official websites with documentation and maintainers' contact details for possible questions, new feature requests and bug reporting.

However even casual linux users can benefit from PKGman, as they may just want to find a package to install for their everyday needs, like a good video player. PKGman gives users the opportunity to search for packages with keywords or within a chosen category, such as multimedia. Then they can conveniently download the corresponding selected binary file (i.e. deb, rpm) via direct link. Last but not least, the rating system it supports can help users to avoid installing bad quality software packages.

## 1.2 Thesis outline

First of all, we'll start with a presentation of Linux software packages and their corresponding package managers. We'll continue by mentioning the existing related work in gathering Linux packages. After that, we'll make a documentation of the technology stack used in this project, whereas crucial technical, implementation details and decisions will be explained. Finally, we'll sum up and note possible extensions and future work to be done.

## 1.3 Linux packages

Most modern Linux operating systems offer a centralized mechanism for finding and installing software. Software is usually distributed in the form of packages, kept in repositories. Package repositories help to ensure that code has been vetted for use on your system, and that the installed versions of software have been approved by developers and package maintainers.(Bearnese, 2016)

A package delivers and maintains new software for Linux-based machines. Just as Windows-based computers rely on executable installers, the Linux ecosystem depends on packages that are administered through software repositories. These files govern the addition, maintenance, and removal of programs on the computer. (Haas, 2020)

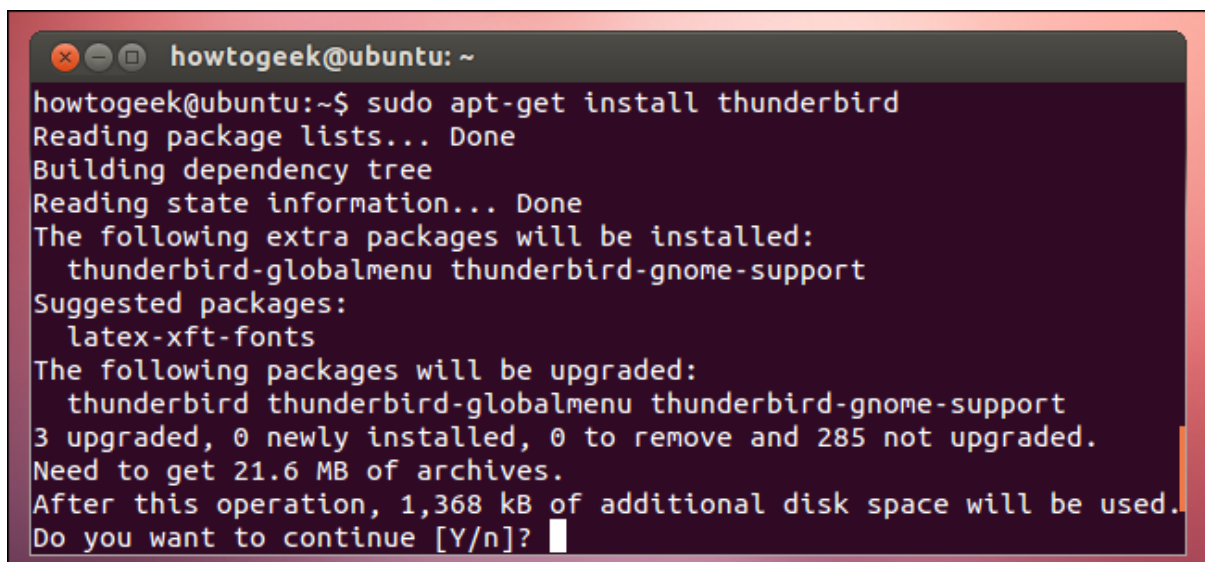
A package consists of a collection of files that perform a task. A package file is usually an archive which contains compiled binaries and other resources making up the software, along with installation scripts. Packages also contain valuable metadata, including their dependencies, a list of other packages required to install and run them. (Bearnese, 2016)

Each Linux distribution offers thousands of packages through their official repositories. Each package can have its own different versions, as well.

## 1.4 Linux package managers

Working with packages is known as package management. Linux supports several major different types of package managers. Each performs the same basic functions but uses a slightly different under-the-hood architecture and different user interfaces to perform the package-manager's core tasks. Generally, each one allows users to perform actions such as installing and managing new packages, removing unnecessary packages, upgrading the already-installed packages, searching for specific packages and updating the system to the latest available version.

For example for package installation/uninstall, regardless of the specific package manager, the user launches a software catalog that reads from one or more repositories (archives of software optimized for a given platform). Then he picks-and-chooses which software to install or uninstall through the graphical catalog, or uses a shell session to execute the commands manually. (Haas, 2020)

A terminal window titled 'howtogeek@ubuntu: ~' showing the command 'sudo apt-get install thunderbird' being executed. The output shows the package lists being read, the dependency tree being built, and the state information being read. It then lists the extra packages to be installed (thunderbird-globalmenu and thunderbird-gnome-support), suggested packages (latex-xft-fonts), and the packages to be upgraded (thunderbird, thunderbird-globalmenu, and thunderbird-gnome-support). It also shows the disk space requirements and asks for confirmation to continue.

```
howtogeek@ubuntu: ~  
howtogeek@ubuntu:~$ sudo apt-get install thunderbird  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
The following extra packages will be installed:  
  thunderbird-globalmenu thunderbird-gnome-support  
Suggested packages:  
  latex-xft-fonts  
The following packages will be upgraded:  
  thunderbird thunderbird-globalmenu thunderbird-gnome-support  
3 upgraded, 0 newly installed, 0 to remove and 285 not upgraded.  
Need to get 21.6 MB of archives.  
After this operation, 1,368 kB of additional disk space will be used.  
Do you want to continue [Y/n]? █
```

Fig.1: installing a software package via command line, on Ubuntu

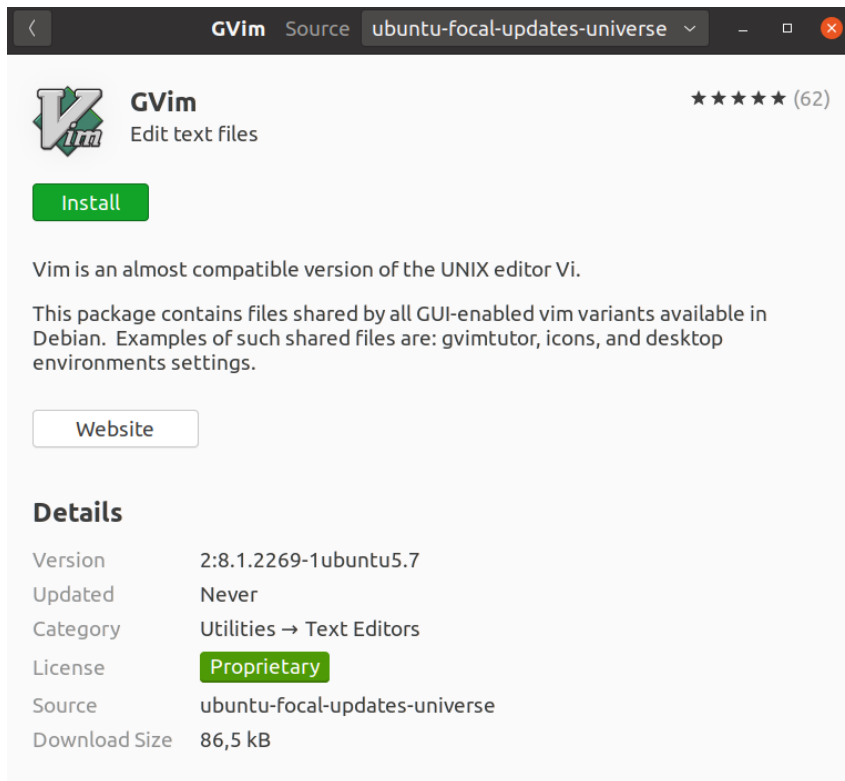


Fig.2: installing a software package via the GUI application, on Ubuntu

Common package-management systems include:

- **DPKG:** The base package manager for Debian-based distributions.
- **Apt:** A front-end for the DPKG system, found in Debian-based distributions, such as Ubuntu, Linux Mint, and Elementary OS.
- **Apt-get:** A more feature-rich front-end for the DPKG system, found in Debian-based distributions.
- **RPM:** The base package manager found in Red Hat-based distributions, such as Red Hat Enterprise Linux, CentOS, and Fedora.
- **Yum:** A front-end for the RPM system, found in Red Hat-based distributions.
- **Dnf:** A more feature-rich front-end for the RPM system.
- **ZYpp:** Found in SUSE and OpenSUSE.
- **Pacman:** The package manager for Arch Linux-based distributions.



## 2. Background

### 2.1 Package Information

The particular project is focused on packages' attributes collection. The several package managers can provide information about a number of package characteristics.

For example the apt package manager lists the following package characteristics:

```
$ apt show nano
Package: nano
Version: 2.8.6-3
Priority: standard
Section: editors
Origin: Ubuntu
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Original-Maintainer: Jordi Mallach <jordi@debian.org>
Bugs: https://bugs.launchpad.net/ubuntu/+filebug
Installed-Size: 766 kB
Depends: libc6 (>= 2.14), libncursesw5 (>= 6), libtinfo5 (>= 6)
Suggests: spell
Conflicts: pico
Breaks: nano-tiny (<< 2.8.6-2)
Replaces: nano-tiny (<< 2.8.6-2), pico
Homepage: https://www.nano-editor.org/
Task: standard, ubuntu-touch-core, ubuntu-touch
Supported: 9m
Download-Size: 222 kB
APT-Manual-Installed: yes
APT-Sources: http://in.archive.ubuntu.com/ubuntu artful/main amd64 Packages
Description: small, friendly text editor inspired by Pico
```

Fig.3: apt package manager - package info presentation (Subramanian, 2018)

The dnf package manager lists the following package characteristics:

```
$ dnf info tilix
Last metadata expiration check: 27 days, 10:00:23 ago on Wed 04 Oct 2017 06:41:00 CEST
Installed Packages
Name           : tilix
Version        : 1.6.4
Release        : 1.fc26
Arch           : x86_64
Size           : 3.6 M
Source         : tilix-1.6.4-1.fc26.src.rpm
Repo           : @System
From repo      : @commandline
Summary        : Tiling terminal emulator
URL            : https://github.com/gnunn1/tilix
License        : MPLv2.0 and GPLv3+ and CC-BY-SA
Description    : Tilix is a tiling terminal emulator with the following features:
```

Fig.4: dnf package manager - package info presentation (Subramanian, 2018)

The pacman package manager lists the following package characteristics:

```
$ pacman -Qi bash
Name           : bash
Version        : 4.4.012-2
Description     : The GNU Bourne Again shell
Architecture   : x86_64
URL            : http://www.gnu.org/software/bash/bash.html
Licenses       : GPL
Groups         : base
Provides       : sh
Depends On     : readline>=7.0 glibc ncurses
Optional Deps  : bash-completion: for tab completion
Required By    : autoconf automake bison bzip2 ca-certificates-utils db
                dhcpcd diffutils e2fsprogs fakeroot figlet findutils
                flex freetype2 gawk gdbm gettext gmp grub gzip icu
                iptables keyutils libgpg-error libksba libpcap libpng
                libtool lvm2 m4 man-db mkinitcpio nano neofetch nspr
                nss openresolv os-prober pacman pcre pcre2 shadow
                systemd texinfo vte-common which xdg-user-dirs xdg-utils
                xfsprogs xorg-mkfontdir xorg-xpr xz
Optional For   : None
Conflicts With : None
Replaces       : None
Installed Size  : 7.13 MiB
Packager       : Jan Alexander Steffens (heftig)
Build Date     : Tue 14 Feb 2017 01:16:51 PM UTC
Install Date   : Thu 24 Aug 2017 06:08:12 AM UTC
Install Reason  : Explicitly installed
Install Script : No
Validated By   : Signature
```

Fig.5: pacman package manager - package info presentation (Subramanian, 2018)

## 2.2 Django REST Framework

Django REST Framework was used for the development of the application's back-end. It is a powerful and flexible toolkit for building RESTful Web APIs. Django is a high-level Python web framework that enables rapid development of secure and maintainable websites. Built by experienced developers, Django takes care of much of the hassle of web development, so that the developer can focus on writing his app without needing to reinvent the wheel. It is free and open source, has a thriving and active community and great documentation. (*Django Introduction - Learn Web Development | MDN, 2022*)

Django helps writing software that is:

- **Complete**

Django follows the "Batteries included" philosophy and provides almost everything developers might want to do "out of the box". Because everything a developer needs is part of the one "product", it all works seamlessly together, follows consistent design principles, and has extensive and up-to-date documentation. Additionally, it provides an optional administrative create, read, update and delete interface that is generated dynamically through introspection and configured via admin models.

- **Versatile**

Django can be (and has been) used to build almost any type of website, from content management systems and wikis, through to social networks and news sites. It can work with any client-side framework, and can deliver content in almost any format (including HTML, JSON, XML, etc). Internally, while it provides choices for almost any functionality a developer might want (e.g. several popular databases, templating engines, etc.), it can also be extended to use other components if needed.

- **Secure**

Django helps developers avoid many common security mistakes by providing a framework that has been engineered to "do the right things" to protect the web app automatically. For example,

Django provides a secure way to manage user accounts and passwords, avoiding common mistakes like directly storing passwords rather than a password hash. Django enables protection against many vulnerabilities by default, including SQL injection, cross-site scripting, cross-site request forgery and clickjacking.

- **Scalable**

Django uses a component-based "shared-nothing" architecture (each part of the architecture is independent of the others, and can hence be replaced or changed if needed). Having a clear separation between the different parts means that it can scale for increased traffic by adding hardware at any level: caching servers, database servers, or application servers. Some of the busiest sites have successfully scaled Django to meet their demands (e.g. Instagram and Disqus, to name just two).

- **Maintainable**

Django code is written using design principles and patterns that encourage the creation of maintainable and reusable code. In particular, it makes use of the "Don't Repeat Yourself" (DRY) principle so there is no unnecessary duplication, reducing the amount of code. Django also promotes the grouping of related functionality into reusable "applications" and, at a lower level, groups related code into modules.

- **Portable**

Django is written in Python, which runs on many platforms. That means that it is not tied to any particular server platform, and can run its applications on many flavors of Linux, Windows, and Mac OS X. Furthermore, Django is well-supported by many web hosting providers, who often provide specific infrastructure and documentation for hosting Django apps.

## 2.3 SQLite

SQLite was used as the application's database. SQLite is an in-process library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine. The code for SQLite is in the public domain and is thus free for use for any purpose, commercial or private. SQLite is the most widely deployed database in the world, including several high-profile projects.

SQLite is an embedded SQL database engine. Unlike most other SQL databases, SQLite does not have a separate server process. SQLite reads and writes directly to ordinary disk files. A complete SQL database with multiple tables, indices, triggers, and views, is contained in a single disk file. The database file format is cross-platform (one can freely copy a database between 32-bit and 64-bit systems or between big-endian and little-endian architectures). These features make SQLite a popular choice as an Application File Format.

SQLite is a compact library. With all features enabled, the library size can be less than 750KiB, depending on the target platform and compiler optimization settings. 64-bit code is larger. And some compiler optimizations such as aggressive function inlining and loop unrolling can cause the object code to be much larger. There is a tradeoff between memory usage and speed. SQLite generally runs faster the more memory it has been given. Nevertheless, performance is usually quite good even in low-memory environments.

SQLite is very carefully tested prior to every release and has a reputation for being very reliable. Most of the SQLite source code is devoted purely to testing and verification. An automated test suite runs millions and millions of test cases involving hundreds of millions of individual SQL statements and achieves 100% branch test coverage. SQLite responds gracefully to memory allocation failures and disk I/O errors. Transactions are ACID even if interrupted by system crashes or power failures. All of this is verified by the automated tests using special test harnesses which simulate system failures. (*About SQLite*, n.d.)

## 2.4 React JavaScript library

React was utilized in order to build the application's front-end. It is a JavaScript library, created by Facebook. It is a tool for building interactive UI components.

Instead of manipulating the browser's DOM directly, React creates a virtual DOM in memory, where it does all the necessary manipulating, before making the changes in the browser DOM. It basically creates an in-memory data-structure cache, computes the resulting differences, finds out what changes have been made, and changes only what needs to be changed.

It is Declarative, meaning that the developer can design simple views for each state in an application, and React will efficiently update and render just the right components when the data changes. Declarative views make the code more predictable and easier to debug.

It is also Component-Based, meaning that it gives a developer the opportunity to build encapsulated components that manage their own state, then compose them to make complex UIs. Since component logic is written in JavaScript instead of templates, rich data can be easily passed through the app and state can be kept out of the DOM.

React can be used as a base in the development of single-page or mobile applications, as it is optimal for fetching rapidly changing data that needs to be recorded. However, fetching data is only the beginning of what happens on a web page, which is why complex React applications, like the particular one, usually require the use of additional libraries for state management, routing, and interaction with an API: Redux, React Router and axios are examples of such libraries.

React components are typically written using JSX, or JavaScript XML, although they do not have to be (components may also be written in pure JavaScript). JSX is an extension to the JavaScript language syntax. Similar in appearance to HTML, JSX provides a way to structure component rendering using syntax familiar to many developers.

## Lifecycle Methods

Lifecycle methods are custom functionality that gets executed during the different phases of a component. There are methods available when the component gets created and inserted into the DOM (mounting), when the component updates, and when the component gets unmounted or removed from the DOM.

Most widely used lifecycle methods include:

- **shouldComponentUpdate** allows the developer to prevent unnecessary re-rendering of a component by returning false if a render is not required. In that way, the application can be optimized.
- **componentDidMount** is called once the component has "mounted" (the component has been created in the user interface, often by associating it with a DOM node). This is commonly used to trigger data loading from a remote source via an API.
- **componentWillUnmount** is called immediately before the component is torn down or "unmounted". This is commonly used to clear resource demanding dependencies to the component that will not simply be removed with the unmounting of the component (e.g., removing any `setInterval()` instances that are related to the component, or an "eventListener" set on the "document" because of the presence of the component).
- **render** is the most important lifecycle method and the only required one in any component. It is usually called every time the component's state is updated, which should be reflected in the user interface.

## 2.5 Cron Job

A Cron Job was utilized to implement the periodic collection of package information. Cron is a utility program that lets users input commands for scheduling tasks repeatedly at a specific time. Tasks scheduled in cron are called cron jobs. Users can determine what kind of task they want to automate and when it should be executed.

Cron is a daemon i.e. a background process executing non-interactive jobs. A daemon is always idle, waiting for a command to request it to perform a particular task. The command can be input on any computer on the network.

A cron file is a simple text file that contains commands to run periodically at a specific time. The default system cron table or crontab configuration file is `/etc/crontab`, located within the crontab directory `/etc/cron.*/`.

With cron jobs, users can automate system maintenance, disk space monitoring, and schedule backups. Because of their nature, cron jobs are great for computers that work 24/7, such as servers. (*Cron Job: A Comprehensive Guide for Beginners 2022*, 2022)

The crontab syntax consists of five fields with the following possible values:

- **Minute:** The minute of the hour the command will run on, ranging from 0-59.
- **Hour:** The hour the command will run at, ranging from 0-23 in the 24-hour notation.
- **Day of the month:** The day of the month the user wants the command to run on, ranging from 1-31.
- **Month:** The month that the user wants the command to run in, ranging from 1-12, thus representing January-December.
- **Day of the week:** The day of the week for a command to run on, ranging from 0-6, representing Sunday-Saturday. In some systems, the value 7 represents Sunday.



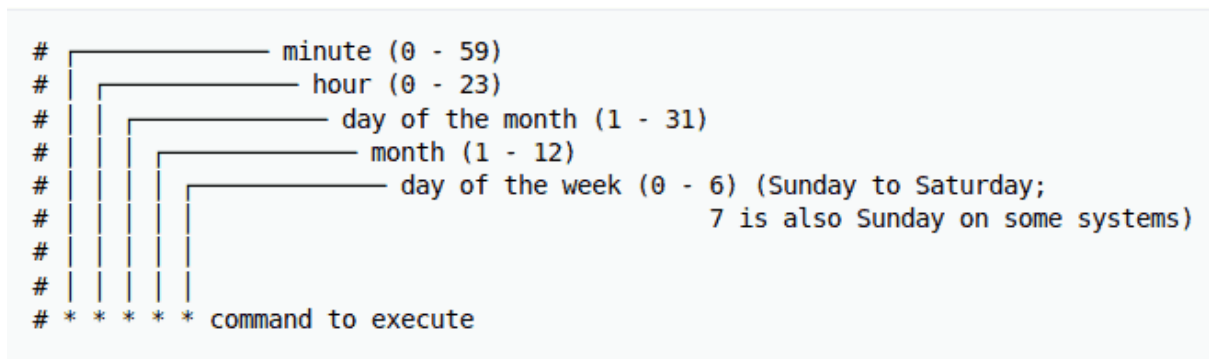


Fig.6: crontab syntax

### 3. Related work

Whereas almost every Linux distribution has its own packages website there are no centralized ones that include package information from various different distributions. Another issue is that most of these websites do not provide all the core information that a package manager can show for a software package. Additionally, usually there is no consistency among them, in the information they present.

ubuntu® packages

package names Search

Ubuntu Packages • bionic (18.04 LTS) • editors • nano

[Source: nano] [bionic] [focal] [hirsute] [impish] [jammy]

### Package: nano (2.9.3-2)

small, friendly text editor inspired by Pico

Other Packages Related to nano

- depends
- recommends
- suggests
- enhances

- libc (>= 2.14) [amd64]
- GNU C Library: Shared libraries
- also a virtual package provided by libc-udeb
- libc (>= 2.17) [amd64, ppc64el]
- libc (>= 2.4) [armhf, i386, s390]
- libncursesw6 (>= 6)
- shared libraries for terminal handling (wide character support)
- libonig2 (>= 6)
- shared low-level terminfo library for terminal handling
- spell
- GNU Spell, a clone of Unix 'spell'

#### Download nano

Architecture	Package Size	Installed Size	Files
amd64	226.0 kB	760.0 kB	[list of files]
arm64	213.3 kB	760.0 kB	[list of files]
armhf	213.4 kB	680.0 kB	[list of files]
i386	237.4 kB	804.0 kB	[list of files]
ppc64el	244.3 kB	900.0 kB	[list of files]
s390x	221.4 kB	796.0 kB	[list of files]

#### Links for nano

#### Ubuntu Resources:

- Bug Reports
- Ubuntu Changelog
- Copyright File

#### Download Source Package nano:

- [nano\_2.9.3-2.dsc]
- [nano\_2.9.3-2.orig.tar.gz]
- [nano\_2.9.3-2.debian.tar.xz]
- [nano\_2.9.3-2.debian.tar.xz]

#### Maintainer:

- Ubuntu Core Developers (Mail Archive)

Please consider filing a bug or asking a question via Launchpad before contacting the maintainer directly.

#### Original Maintainers (usually from Debian):

- Jordi Mallach
- Steve Langasek

It should generally not be necessary for users to contact the original maintainer.

#### External Resources:

- Homepage [www.nano-editor.org]

Fig.7: Ubuntu package website - info for nano package (left)

fedora PACKAGES

Search Search

### nano

A small text editor

BUILDS Updates Bugs Sources Crash Reports KOSCHEL

GNU nano is a small and friendly text editor.

#### Releases Overview

Release	Stable	Testing
Fedora Rawhide	6.2-1.fc37	-
Fedora 36	6.0-2.fc36	-
Fedora 35	5.8-4.fc35	-
Fedora 34	5.8-3.fc34	5.6.1-1.fc34

#### Package Info

- Upstream: <https://www.nano-editor.org>
- License(s): GPLv2+
- Maintainers: dnmw2, kdudka, svashisht

#### Related Packages

- default-editor
- nano-default-editor

You can contact the maintainers of this package via email at [nano-dash-maintainers@fedoraproject.org](mailto:nano-dash-maintainers@fedoraproject.org).

Fig.8: Fedora package website - info for nano package (right)

## 4. PKGman

PKGman is a web application that periodically collects a number of useful attributes about popular Linux distributions' software packages, stores them in a SQLite database and also provides a website with search filters for their presentation.

The application currently supports the following Linux distributions:

- **Ubuntu 20.04**
- **Debian 11**
- **Kali 2021.4**
- **Fedora 34**
- **CentOS 8.4.2105**

The collected package attributes are the following:

- **name**
- **Linux distribution** [Ubuntu, Debian, Kali, Fedora, CentOS]
- **type** [deb, rpm]
- **category**
- **license**
- **maintainer**
- **description**
- **homepage website URL**
- **code repository URL**

For every package version the following attributes are also collected:

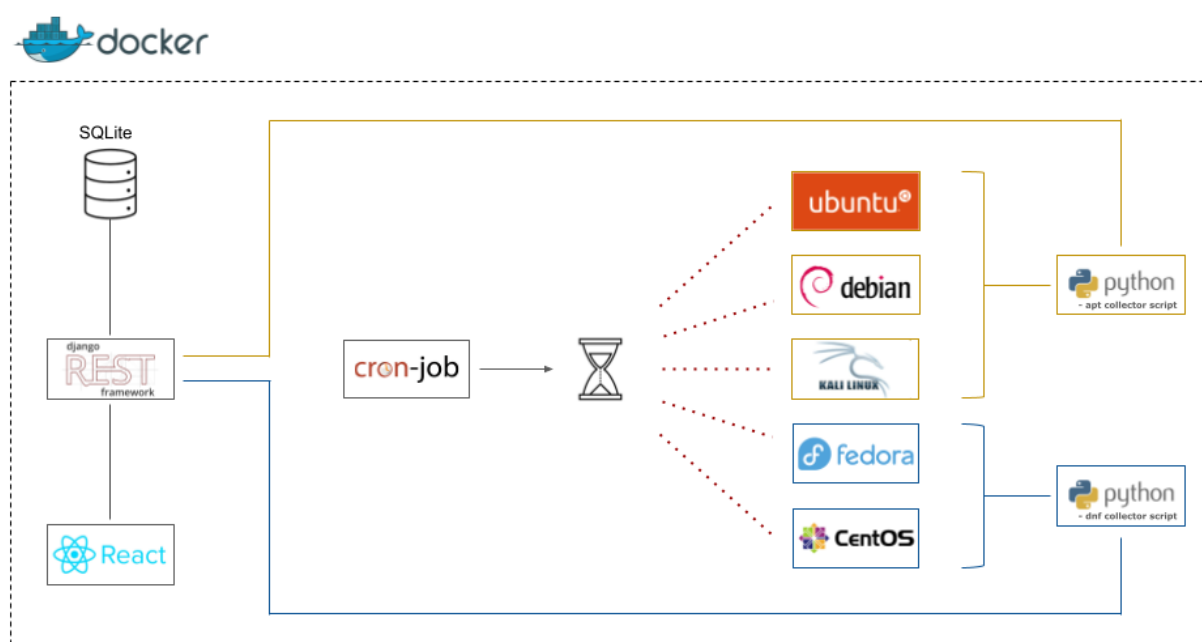
- **name**
- **architecture**
- **size**
- **binary file URL**

*Note: each package can have multiple versions*

The whole application is containerized and split into services which are orchestrated via a Docker-Compose file. Because of this, it can be easily run by just one command:

```
$ docker-compose up - --build
```

The diagram below depicts a representative visualization of the app's architecture and technology stack.



III.1: whole app's architecture

As we can see on the diagram there are three main services:

- **the collection mechanism:** a Cron-Job that builds the Linux distributions Docker images (Ubuntu, Fedora etc.) and spawns the corresponding containers inside of which the python scripts, responsible for the package info collection, are being executed.
- **the back-end:** developed using the Django REST framework. It is responsible for populating the database with the extracted attributes, querying it based on specific search criteria and feeding the front-end with the corresponding results. It utilizes a SQLite database.
- **the front-end:** developed using the React.js library. Its role is to consume the queried results provided by the back-end and present them in a well structured and intuitive way.

## 4.1 Package information Collection

The package information collection mechanism is triggered once a month, through a Cron-Job service which runs endlessly on a Docker container. Once it is triggered, it executes the *run\_all\_collector\_scripts.sh* script, which is responsible for building the image for every supported Linux distribution and executing the right collector python script, within the corresponding container. The *run\_all\_collector\_scripts.sh* script spawns the new Docker containers (for every supported Linux distribution) on the host machine. It also contains some environment variables that are passed on the newly spawned containers.

```
1 0 0 1 * * /bin/sh /collector/run_all_collector_scripts.sh > /proc/1/fd/1 2> /proc/1/fd/2
2 # the script will be executing at 00:00 on the 1st of every month
```

Fig.9: cron job file for executing the *run\_all\_collector\_scripts.sh* script

```
# vars
DOCKER_VOL_MOUNT="-v /var/run/docker.sock:/var/run/docker.sock"
WORK_DIR="/collector" # must be identical to collector Dockerfile's 'WORKDIR'
SHM_SIZE="5.07gb" # considering that your machine has 16 GB of RAM - change it accordingly otherwise
BACKEND_API="http://localhost:8000/api/v1" # considering that the backend runs on the same machine - change it accordingly otherwise
GITHUB_TOKEN="" # your github account token - fill it out
USERNAME="" # your credentials to login to backend API - fill it out
PASSWORD="" # your credentials to login to backend API - fill it out

# Ubuntu:
docker build -t ubuntu_distro -f $WORK_DIR/ubuntu.Dockerfile $WORK_DIR
docker run --rm $DOCKER_VOL_MOUNT --net host --shm-size=$SHM_SIZE \
  --env GITHUB-TOKEN=$GITHUB_TOKEN --env USERNAME=$USERNAME --env PASSWORD=$PASSWORD \
  ubuntu_distro python3 apt_collector.py -d Ubuntu -u $BACKEND_API -a http://archive.ubuntu.com/ubuntu

# Debian:
docker build -t debian_distro -f $WORK_DIR/debian.Dockerfile $WORK_DIR
docker run --rm $DOCKER_VOL_MOUNT --net host --shm-size=$SHM_SIZE \
  --env GITHUB-TOKEN=$GITHUB_TOKEN --env USERNAME=$USERNAME --env PASSWORD=$PASSWORD \
  debian_distro python3 apt_collector.py -d Debian -u $BACKEND_API -a http://ftp.debian.org/debian

# ...
```

Fig.10: *run\_all\_collector\_scripts.sh*

There are two python scripts for extracting and collecting the package attributes:

- **apt\_collector.py**: used for the Ubuntu, Debian and Kali distributions
- **dnf\_collector.py**: used for the Fedora and CentOS distributions

They communicate with the back-end using HTTP requests via a RESTful API.

Both of them share the following command line arguments:

- **--max-concurrency, -c <int>** [default: 50]: Number of maximum packages that will be processed concurrently

- **--distro, -d <str>**: The linux distribution name
- **--API-URL, -u <url>**: Back-end API base URL
- **--archives-url, -a <url>**: Linux distribution's archives base URL

Apart from those, the *dnf\_collector.py* also accepts the:

- **--repos-url, -r <url>**: Linux distribution's packages code repository base URL

Both scripts expect the following environment variables to be passed on:

- **GITHUB-TOKEN**: an access token for Github, since its API will be utilized to extract some package attributes that the package managers may not always provide.
- **USERNAME / PASSWORD**: user's credentials to login to the back-end API, since not everyone is allowed to add package information to the database.

Both scripts utilize the Ray framework. Ray is an open source project that makes it simple to scale any compute-intensive python workload. With a rich set of libraries and integrations built on a flexible distributed execution framework, Ray makes distributed computing easy and accessible. Parallelization is crucial on the particular application, due to the big amount of packages each Linux distribution provides (e.g. Ubuntu has almost 60,000 packages).

The scripts' logic can be depicted in the following pseudo code:

```

1 for pkg in list_distro_pkgs(): # parallel processing
2     exist_pkg_info = fetch_pkg_info(pkg) # HTTP request to backend
3     if exist_pkg_info is None: # new package
4         pkg_info = extract_pkg_info(pkg)
5         extra_pkg_info = get_pkg_info_from_github(pkg) or \
6             get_pkg_info_from_distro_code_repo(pkg)
7
8         save_pkg_info(pkg_info, extra_pkg_info) # HTTP request to backend
9     else: # package already exists check for new versions
10        new_versions = []
11        for version in extract_pkg_versions(pkg):
12            if version not in exist_pkg_info["versions"]: # new version
13                new_versions.append(extract_version_info(pkg, version))
14
15        save_new_pkg_version_info(pkg, new_versions) # HTTP request to backend
16

```

Fig.11: apt/dnf collector scripts pseudo code

One thing to note is that the apt and dnf package managers do not provide exactly the same information for packages. For example the apt one does not provide the license attribute as the dnf does. Furthermore, none of them normally provide URLs for the packages source code

repositories. For this reason, these attributes are attempted to be extracted from their github or their distribution's equivalent online repositories (e.g. salsa.debian.org for Debian) API. However this is not always possible.

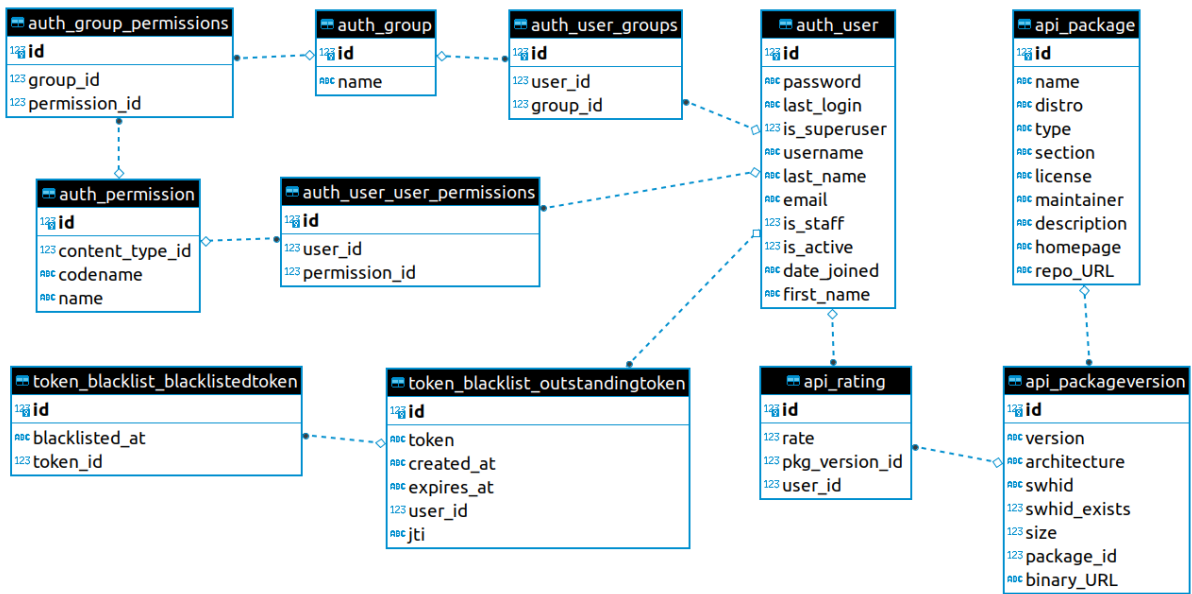
```
id: 7029
description: "389-ds-base packaging"
name: "389-ds-base"
name_with_namespace: "FreeIPA packaging / 389-ds-base"
path: "389-ds-base"
path_with_namespace: "freeipa-team/389-ds-base"
created_at: "2018-01-24T14:15:55.010Z"
default_branch: "master"
tag_list: []
ssh_url_to_repo: "git@salsa.debian.org:freeipa-team/389-ds-base.git"
▶ http_url_to_repo: "https://salsa.debian.org/freeipa-team/389-ds-base.git"
web_url: "https://salsa.debian.org/freeipa-team/389-ds-base"
▶ readme_url: "https://salsa.debian.org/-/blob/master/README.md"
▶ license_url: "https://salsa.debian.org/se/-/blob/master/LICENSE"
▼ license:
  key: "other"
  name: "Other"
  nickname: null
  html_url: "http://choosealicense.com/licenses/other/"
  source_url: null
avatar_url: null
forks_count: 2
star_count: 0
last_activity_at: "2022-02-10T17:39:23.329Z"
```

Fig.12: salsa.debian.org API's JSON response

## 4.2 Back-end

The back-end was developed using the Django REST framework. It is responsible for populating the database with the extracted attributes (feeded by the collector scripts), querying it based on specific search criteria (feeded by the front-end) and feeding the front-end with the corresponding results. It utilizes a SQLite database.

On the diagram below, the database's schema is depicted. The tables for storing the packages' information are the *api\_package* and *api\_packageversion*. The *api\_rating* table is used for storing users' ratings on package versions. The *auth\_user* table is for the default Django user data. The *token\_blacklist\_outstandingtoken*, *token\_blacklist\_blacklistedtoken* tables are for authorization purposes. All the other tables are for the Django's integrated permission system.



III.2: database schema

The application supports four user categories:

- **visitors:** unauthenticated users that have read rights to every resource and no write permission to anything.
- **regular users:** authenticated users that can additionally rate package versions.
- **package collector:** authorized users that can additionally add package information to the database using the API.
- **admins:** authorized users that have access to every resource with read and write permissions. Furthermore, they have access to Django's admin panel and can upgrade a regular user to a package collector.

As for the authorization the OAuth 2.0 protocol was followed utilizing JWTs. The JSON Web Token (JWT) is an open standard (RFC 7519) that defines a compact and self-contained way for securely transmitting information between parties as a JSON object. This information can be verified and trusted because it is digitally signed. JWTs can be signed using a secret (with the HMAC algorithm) or a public/private key pair using RSA or ECDSA.

JWT is widely used nowadays, because of its small overhead and its ability to be easily used across different domains. (*JSON Web Token Introduction - Jwt.io*, n.d.)

ALGORITHM HS256

Encoded

Decoded

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiaWF0Ij0iOnRyZWV9.TJVA95OrM7E2cBab30RMhRHDcEfxjYZgeFONFh7HgQ
```

HEADER:

```
{  "alg": "HS256",  "typ": "JWT"}
```

PAYLOAD:

```
{  "sub": "1234567890",  "name": "John Doe",  "admin": true}
```

VERIFY SIGNATURE

```
HMACSHA256(  
  base64UrlEncode(header) + "." +  
  base64UrlEncode(payload),  
  secret  
)  
secret base64 encoded
```

Fig.13: JWT format

OAuth 2.0 uses Access Tokens and Refresh Tokens.

The Access token contains all the information the server needs to know if the user can access the resource he is requesting or not. They are expired tokens with a short validity period.

The refresh token is used to generate a new access token. Typically, if the access token has an expiration date, once it expires, the user would have to authenticate again to obtain an access token. With refresh token, this step can be skipped and with a request to the API get a new access token that allows the user to continue accessing the application resources. That is the case for the particular application. (*Refresh Token With JWT Authentication in Node.js*, n.d.)



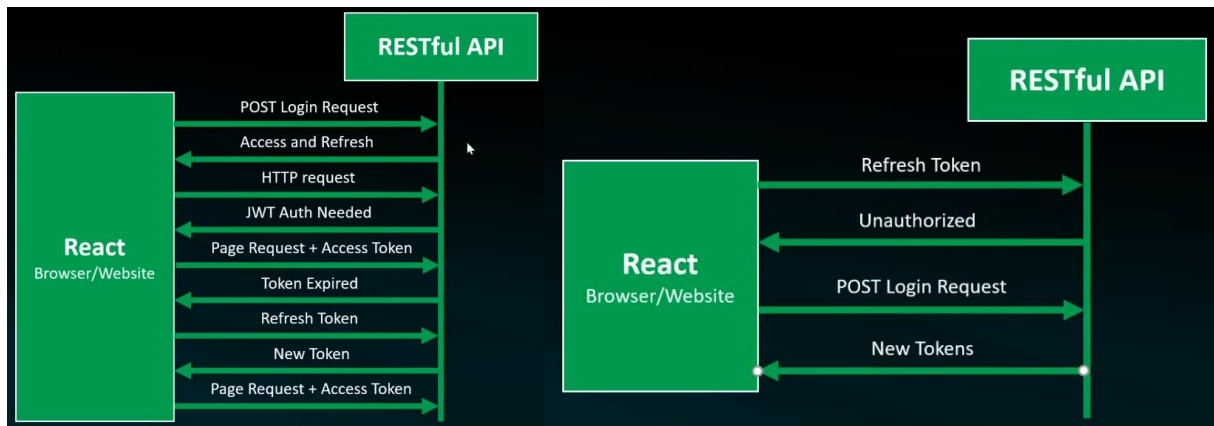


Fig.14: OAuth 2.0 Access / Refresh tokens flow

The API endpoints were implemented using Django REST's ModelViewSet, whereas for data serialization the Django REST's ModelSerializer was used.

```
class PackageSerializer(serializers.ModelSerializer):
    versions = PackageVersionSerializer(many=True)
    rating = serializers.SerializerMethodField()
    def get_rating(self, obj):
        if obj.rating is None:
            return 0
        return obj.rating

    def validate_versions(self, versions):
        if len(versions) == 0:
            raise serializers.ValidationError("This field must not be empty.")
        return versions

    class Meta:
        model = Package
        fields = "__all__"

    def create(self, validated_data):
        versions = validated_data.pop('versions')
        pkg = Package.objects.create(**validated_data)
        for version in versions:
            PackageVersion.objects.create(package=pkg, **version)
        return pkg
```

Fig.15: Django REST serializer for Package information

For the database creation the Django models were utilized.

```
class Package(models.Model):
    name = models.CharField(max_length=100)
    distro = models.CharField(max_length=50)
    type = models.CharField(max_length=20, default='')
    section = models.CharField(max_length=50, blank=True, default='')
    license = models.CharField(max_length=300, blank=True, default='')
    maintainer = models.CharField(max_length=200, blank=True, default='')
    description = models.CharField(max_length=300, blank=True, default='')
    homepage = models.URLField(max_length=500, blank=True, default='')
    repo_URL = models.URLField(max_length=500, blank=True, default='')
```

Fig.16: Django model for Package information

Due to the huge amount of stored packages Django REST's pagination functionality has been enabled, returning multiple pages of 10 results per page.

## API

A full API documentation can be found on [<BACK\\_END\\_BASE\\_URL>/api/v1/swagger/](http://<BACK_END_BASE_URL>/api/v1/swagger/).

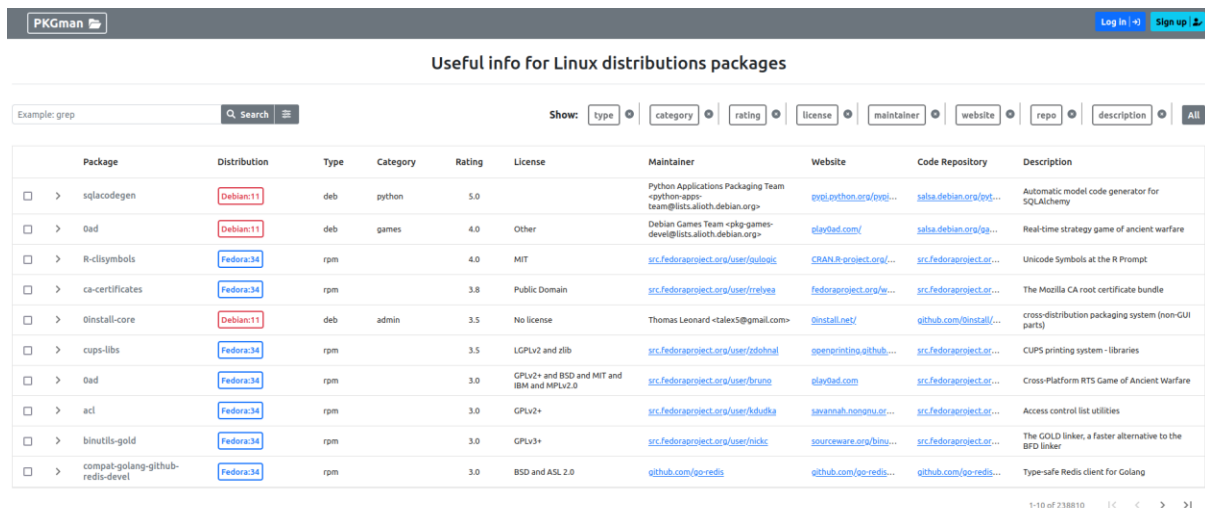
packages			▼
GET	/packages/	packages_list	🔒
POST	/packages/	packages_create	🔒
POST	/packages/dockerfile/	packages_dockerfile	🔒
GET	/packages/versions/	packages_versions_list	🔒
POST	/packages/versions/	packages_versions_create	🔒
GET	/packages/versions/ratings/	packages_versions_ratings_list	🔒
POST	/packages/versions/ratings/	packages_versions_ratings_create	🔒
GET	/packages/versions/ratings/{id}/	packages_versions_ratings_read	🔒
PUT	/packages/versions/ratings/{id}/	packages_versions_ratings_update	🔒
PATCH	/packages/versions/ratings/{id}/	packages_versions_ratings_partial_update	🔒
DELETE	/packages/versions/ratings/{id}/	packages_versions_ratings_delete	🔒
GET	/packages/versions/{id}/	packages_versions_read	🔒
PUT	/packages/versions/{id}/	packages_versions_update	🔒
PATCH	/packages/versions/{id}/	packages_versions_partial_update	🔒
DELETE	/packages/versions/{id}/	packages_versions_delete	🔒
GET	/packages/{id}/	packages_read	🔒
PUT	/packages/{id}/	packages_update	🔒
PATCH	/packages/{id}/	packages_partial_update	🔒
DELETE	/packages/{id}/	packages_delete	🔒
POST	/packages/{id}/versions/	packages_versions	🔒
users			▼
POST	/users/login/	users_login_create	🔒
POST	/users/login/token/refresh/	users_login_token_refresh_create	🔒
POST	/users/logout/	users_logout_create	🔒
POST	/users/register/	users_register_create	🔒

Fig.17: Back-end API endpoints

## 4.3 Front-end

The front-end was developed using the React.js library. Its role is to consume the queried results provided by the back-end and present them in a well structured and intuitive way. It communicates with the back-end with HTTP requests via a RESTful API.

For the package presentation the react-data-table-component npm package was used. The default results consist of the highest rated packages, in a descending order. The results are paginated, meaning that when the next page icon is clicked a new batch of results will be fetched from the back-end. The pagination is essential due to the enormous amount of the results.



The screenshot shows the PKGman landing page. At the top, there's a header with the PKGman logo and links for 'Log in' and 'Sign up'. Below the header, the title 'Useful info for Linux distributions packages' is centered. A search bar with the placeholder 'Example: grep' and a 'Search' button is on the left. To the right of the search bar, there's a 'Show:' dropdown menu followed by several filter buttons: 'type', 'category', 'rating', 'license', 'maintainer', 'website', 'repo', 'description', and 'All'. Below these filters is a table with 10 columns: Package, Distribution, Type, Category, Rating, License, Maintainer, Website, Code Repository, and Description. The table lists 10 packages, each with a checkbox and a right-pointing arrow in the 'Package' column. The 'Distribution' column shows 'Debian:11' for the first two packages and 'Fedora:34' for the others. The 'Rating' column shows values like 5.0, 4.0, 4.0, 3.8, 3.5, 3.5, 3.0, 3.0, 3.0, and 3.0. The 'License' column shows various licenses like MIT, Public Domain, No license, LGPLv2 and zlib, GPLv2+ and BSD and MIT and IBM and MPLv2.0, GPLv2+, GPLv3+, and BSD and ASL 2.0. The 'Maintainer' column shows email addresses or team names. The 'Website' column shows URLs. The 'Code Repository' column shows GitHub repository links. The 'Description' column provides a brief description of each package. At the bottom right of the table, there's a pagination link '1-10 of 238810' and navigation icons for previous, next, and other pages.

Package	Distribution	Type	Category	Rating	License	Maintainer	Website	Code Repository	Description
<input type="checkbox"/> > sqlacodegen	Debian:11	deb	python	5.0		Python Applications Packaging Team <python-apps-team@lists.alioth.debian.org>	<a href="https://pypi.org/project/sqlacodegen/">pypi.org/project/sqlacodegen/</a>	<a href="https://salsa.debian.org/python-apps-team/sqlacodegen">salsa.debian.org/python-apps-team/sqlacodegen</a>	Automatic model code generator for SQLAlchemy
<input type="checkbox"/> > 0ad	Debian:11	deb	games	4.0	Other	Debian Games Team <pkg-games-devel@lists.alioth.debian.org>	<a href="https://0ad.com/">0ad.com/</a>	<a href="https://salsa.debian.org/games-team/0ad">salsa.debian.org/games-team/0ad</a>	Real-time strategy game of ancient warfare
<input type="checkbox"/> > R-clisymbols	Fedora:34	rpm		4.0	MIT	<a href="https://src.fedoraproject.org/user/oulois">src.fedoraproject.org/user/oulois</a>	<a href="https://cran.r-project.org/web/packages/R-clisymbols/index.html">CRAN.R-project.org/...</a>	<a href="https://src.fedoraproject.org/packages/R-clisymbols">src.fedoraproject.org/...</a>	Unicode Symbols at the R Prompt
<input type="checkbox"/> > ca-certificates	Fedora:34	rpm		3.8	Public Domain	<a href="https://src.fedoraproject.org/user/trelves">src.fedoraproject.org/user/trelves</a>	<a href="https://fedoraproject.org/wiki/CA_Certificates">fedoraproject.org/wiki/...</a>	<a href="https://src.fedoraproject.org/packages/ca-certificates">src.fedoraproject.org/...</a>	The Mozilla CA root certificate bundle
<input type="checkbox"/> > 0install-core	Debian:11	deb	admin	3.5	No license	Thomas Leonard <talex5@gmail.com>	<a href="https://0install.net/">0install.net/</a>	<a href="https://github.com/0install/0install-core">github.com/0install/...</a>	cross-distribution packaging system (non-GUI parts)
<input type="checkbox"/> > cups-libs	Fedora:34	rpm		3.5	LGPLv2 and zlib	<a href="https://src.fedoraproject.org/user/rdohal">src.fedoraproject.org/user/rdohal</a>	<a href="https://source.redhat.com/projects/cups">source.redhat.com/projects/cups</a>	<a href="https://src.fedoraproject.org/packages/cups-libs">src.fedoraproject.org/...</a>	CUPS printing system - libraries
<input type="checkbox"/> > 0ad	Fedora:34	rpm		3.0	GPLv2+ and BSD and MIT and IBM and MPLv2.0	<a href="https://src.fedoraproject.org/user/bruno">src.fedoraproject.org/user/bruno</a>	<a href="https://0ad.com/">0ad.com</a>	<a href="https://src.fedoraproject.org/packages/0ad">src.fedoraproject.org/...</a>	Cross-Platform RTS Game of Ancient Warfare
<input type="checkbox"/> > acl	Fedora:34	rpm		3.0	GPLv2+	<a href="https://src.fedoraproject.org/user/dufka">src.fedoraproject.org/user/dufka</a>	<a href="https://savannah.nongnu.org/projects/acl">savannah.nongnu.org/...</a>	<a href="https://src.fedoraproject.org/packages/acl">src.fedoraproject.org/...</a>	Access control list utilities
<input type="checkbox"/> > binutils-gold	Fedora:34	rpm		3.0	GPLv3+	<a href="https://src.fedoraproject.org/user/nickc">src.fedoraproject.org/user/nickc</a>	<a href="https://sourceware.org/binutils/">sourceware.org/binu...</a>	<a href="https://src.fedoraproject.org/packages/binutils-gold">src.fedoraproject.org/...</a>	The GOLD linker, a faster alternative to the BFD linker
<input type="checkbox"/> > compat-golang-github-redis-devel	Fedora:34	rpm		3.0	BSD and ASL 2.0	<a href="https://github.com/gomods/redis">github.com/gomods/redis</a>	<a href="https://github.com/gomods/redis">github.com/gomods/redis</a>	<a href="https://github.com/gomods/redis">github.com/gomods/redis...</a>	Type-safe Redis client for Golang

Fig.18: PKGman landing page with the highest rated packages in descending order

Each row of the data-table is collapsible and when expanded each package's corresponding versions are presented.

Useful info for Linux distributions packages									
grep		Search		Show: type category rating license maintainer website repo description All					
Package	Distribution	Type	Category	Rating	License	Maintainer	Website	Code Repository	Description
<input type="checkbox"/> > sqlacodegen	Debian:11	deb	python	5.0		Python Applications Packaging Team <python-apps-team@lists.alioth.debian.org>	<a href="#">pyvel.python.org/pyv...</a>	<a href="#">salisa.debian.org/pyv...</a>	Automatic model code generator for SQLAlchemy
<input type="checkbox"/> > Oad	Debian:11	deb	games	4.0	Other	Debian Games Team <pkg-games-devel@lists.alioth.debian.org>	<a href="#">qlayad.com/</a>	<a href="#">salisa.debian.org/qa...</a>	Real-time strategy game of ancient warfare
<input type="checkbox"/> > R-clisymbols	Fedora:34	rpm		4.0	MIT	<a href="#">src.fedoraproject.org/user/duoloi</a>	<a href="#">CRAN.R-project.org/...</a>	<a href="#">src.fedoraproject.org/...</a>	Unicode Symbols at the R Prompt
<input type="checkbox"/> > ca-certificates	Fedora:34	rpm		3.8	Public Domain	<a href="#">src.fedoraproject.org/user/treveya</a>	<a href="#">fedoraobject.org/w...</a>	<a href="#">src.fedoraproject.org/...</a>	The Mozilla CA root certificate bundle
<div> <div>Version</div> <div>Architecture</div> <div>Size</div> <div>Average Rating</div> <div>Binary Download Link</div> </div> <div> <input type="checkbox"/> 2021.2.52-1.0.fc34           <div>noarch</div> <div>362 Bytes</div> <div>3.5</div> <div><a href="#">koilekas.fedoraproject.org/packages/ca-certificates/2021.2.52-1.0.fc34/src.rpm</a></div> </div> <div> <input type="checkbox"/> 2020.2.41-7.fc34           <div>noarch</div> <div>936 Bytes</div> <div>4.0</div> <div><a href="#">koilekas.fedoraproject.org/packages/ca-certificates/2020.2.41-7.fc34/src.rpm</a></div> </div>									
<input type="checkbox"/> > Oinstall-core	Debian:11	deb	admin	3.5	No license	Thomas Leonard <tales5@gmail.com>	<a href="#">Oinstall.net/</a>	<a href="#">github.com/Oinstall/...</a>	cross-distribution packaging system (non-GUI parts)
<input type="checkbox"/> > cups-libs	Fedora:34	rpm		3.5	LGPLv2 and zlib	<a href="#">src.fedoraproject.org/user/rdohal</a>	<a href="#">openerprinting.github...</a>	<a href="#">src.fedoraproject.org/...</a>	CUPS printing system - libraries
<div> <div>Version</div> <div>Architecture</div> <div>Size</div> <div>Average Rating</div> <div>Binary Download Link</div> </div> <div> <input type="checkbox"/> 2.3.3op2-13.fc34           <div>x86_64</div> <div>1.3 kB</div> <div>2.0</div> <div><a href="#">koilekas.fedoraproject.org/packages/cups-libs/2.3.3op2-13.fc34/src.rpm</a></div> </div> <div> <input type="checkbox"/> 2.3.3op2-11.fc34           <div>x86_64 i686</div> <div>666 Bytes</div> <div>5.0</div> <div><a href="#">koilekas.fedoraproject.org/packages/cups/2.3.3op2-11.fc34/src.rpm</a></div> </div>									
<input type="checkbox"/> > Oad	Fedora:34	rpm		3.0	GPLv2+ and BSD and MIT and IBM and MPLv2.0	<a href="#">src.fedoraproject.org/user/bruno</a>	<a href="#">qlayad.com</a>	<a href="#">src.fedoraproject.org/...</a>	Cross-Platform RTS Game of Ancient Warfare
<input type="checkbox"/> > acl	Fedora:34	rpm		3.0	GPLv2+	<a href="#">src.fedoraproject.org/user/tdubka</a>	<a href="#">savannah.nongnu.or...</a>	<a href="#">src.fedoraproject.org/...</a>	Access control list utilities

Fig.19: PKGman collapsible package results with corresponding versions

It also contains a number of search filters so that the user can conveniently search for packages.

Example: grep

Search

Type

deb X

Distribution

Debian:11 X Ubuntu:20.04 X

Architecture

x86\_64 X

Category

database X editors X

☐ Show only exact search matches

Search

Package	Distribution	Type	Category	Rating	License	Maintainer	Website	Code Repository	Description
<input type="checkbox"/> > sqlacodegen	Debian:11	deb	python	5.0		Python Applications Packaging Team <python-apps-team@lists.alioth.debian.org>	<a href="#">pyvel.python.org/pyv...</a>	<a href="#">salisa.debian.org/pyv...</a>	Automatic model code generator for SQLAlchemy
<input type="checkbox"/> > Oad	Debian:11	deb	games	4.0	Other	Debian Games Team <pkg-games-devel@lists.alioth.debian.org>	<a href="#">qlayad.com/</a>	<a href="#">salisa.debian.org/qa...</a>	Real-time strategy game of ancient warfare
<input type="checkbox"/> > R-clisymbols	Fedora:34	rpm		4.0	MIT	<a href="#">src.fedoraproject.org/user/duoloi</a>	<a href="#">CRAN.R-project.org/...</a>	<a href="#">src.fedoraproject.org/...</a>	Unicode Symbols at the R Prompt
<input type="checkbox"/> > ca-certificates	Fedora:34	rpm		3.8	Public Domain	<a href="#">src.fedoraproject.org/user/treveya</a>	<a href="#">fedoraobject.org/w...</a>	<a href="#">src.fedoraproject.org/...</a>	The Mozilla CA root certificate bundle
<input type="checkbox"/> > Oinstall-core	Debian:11	deb	admin	3.5	No license	Thomas Leonard <tales5@gmail.com>	<a href="#">Oinstall.net/</a>	<a href="#">github.com/Oinstall/...</a>	cross-distribution packaging system (non-GUI parts)
<input type="checkbox"/> > cups-libs	Fedora:34	rpm		3.5	LGPLv2 and zlib	<a href="#">src.fedoraproject.org/user/rdohal</a>	<a href="#">openerprinting.github...</a>	<a href="#">src.fedoraproject.org/...</a>	CUPS printing system - libraries
<input type="checkbox"/> > Oad	Fedora:34	rpm		3.0	GPLv2+ and BSD and MIT and IBM and MPLv2.0	<a href="#">src.fedoraproject.org/user/bruno</a>	<a href="#">qlayad.com</a>	<a href="#">src.fedoraproject.org/...</a>	Cross-Platform RTS Game of Ancient Warfare
<input type="checkbox"/> > acl	Fedora:34	rpm		3.0	GPLv2+	<a href="#">src.fedoraproject.org/user/tdubka</a>	<a href="#">savannah.nongnu.or...</a>	<a href="#">src.fedoraproject.org/...</a>	Access control list utilities
<input type="checkbox"/> > binutils-gold	Fedora:34	rpm		3.0	GPLv3+	<a href="#">src.fedoraproject.org/user/bock</a>	<a href="#">sourceware.org/binu...</a>	<a href="#">src.fedoraproject.org/...</a>	The GOLD linker, a faster alternative to the BFD linker
<input type="checkbox"/> > compat-golang-github-redis-devel	Fedora:34	rpm		3.0	BSD and ASL 2.0	<a href="#">github.com/gomred</a>	<a href="#">github.com/gomred</a>	<a href="#">github.com/gomred</a>	Type-safe Redis client for Golang

Fig.20: PKGman search filters

Additionally the user has the option to hide/show specific columns (package attributes) as he pleases. Apart from that, multiple columns can be sorted in ascending or descending order. The sorting is performed on the back-end side for performance purposes.

PKGman

Log in | Sign up

Useful info for Linux distributions packages

Example: grep

Search

Show: type rating description All

Package	Distribution	Type	Rating	Description
<input type="checkbox"/> > sqlacodegen	Debian:11	deb	5.0	Automatic model code generator for SQLAlchemy
<input type="checkbox"/> > oad	Debian:11	deb	4.0	Real-time strategy game of ancient warfare
<input type="checkbox"/> > R-clisymbols	Fedora:34	rpm	4.0	Unicode Symbols at the R Prompt
<input type="checkbox"/> > ca-certificates	Fedora:34	rpm	3.8	The Mozilla CA root certificate bundle
<input type="checkbox"/> > oinstall-core	Debian:11	deb	3.5	cross-distribution packaging system (non-GUI parts)
<input type="checkbox"/> > cups-libs	Fedora:34	rpm	3.5	CUPS printing system - libraries
<input type="checkbox"/> > oad	Fedora:34	rpm	3.0	Cross-Platform RTS Game of Ancient Warfare
<input type="checkbox"/> > acl	Fedora:34	rpm	3.0	Access control list utilities
<input type="checkbox"/> > binutils-gold	Fedora:34	rpm	3.0	The GOLD linker, a faster alternative to the BFD linker
<input type="checkbox"/> > compat-golang-github-redis-devel	Fedora:34	rpm	3.0	Type-safe Redis client for Golang

1-10 of 238810 < > >>

Fig.21: PKGman selected table columns

Furthermore, a user can select one or multiple packages. He can even pick specific package versions. The selected packages (versions) are presented on top of the data-table in a collapsed list. From there he can easily handle his selection by removing one or all of them. After he is done with his selection he can extract their information in a CSV / JSON format or even create a working Dockerfile that contains the appropriate commands for installing them. If the latter is the case all selected packages must be of the same Linux distribution.

PKGman

Log out

Useful info for Linux distributions packages

Example: grep

Q Search

Show:

type

category

rating

license

maintainer

website

repo

description

All

2 packages have been selected

• oinstall-core, 2.16-1, Debian:11

• sqlacodegen, Debian:11

Deselect All

Create Dockerfile

Export to CSV

Export to JSON

Package	Distribution	Type	Category	Rating	License	Maintainer	Website	Code Repository	Description
<input checked="" type="checkbox"/> > sqlacodegen	Debian:11	deb	python	5.0		Python Applications Packaging Team <python-apps-team@lists.alioth.debian.org>	pyth-onthon.org/ovp...	salsa.debian.org/ovt...	Automatic model code generator for SQLAlchemy
<input type="checkbox"/> > oad	Debian:11	deb	games	4.0	Other	Debian Games Team <pkg-games-devel@lists.alioth.debian.org>	elavted.com/	salsa.debian.org/og...	Real-time strategy game of ancient warfare
<input type="checkbox"/> > R-clisymbols	Fedora:34	rpm		4.0	MIT	src.fedoraproject.org/user/fojouis	CRAN.R-project.org/...	src.fedoraproject.or...	Unicode Symbols at the R Prompt
<input type="checkbox"/> > ca-certificates	Fedora:34	rpm		3.8	Public Domain	src.fedoraproject.org/user/freelva	fedoraproject.org/te...	src.fedoraproject.or...	The Mozilla CA root certificate bundle
<input checked="" type="checkbox"/> > oinstall-core	Debian:11	deb	admin	3.5	No license	Thomas Leonard <talox5@gmail.com>	0install.net/	github.com/0install/...	cross-distribution packaging system (non-GUI parts)
<input checked="" type="checkbox"/>	Version	Architecture		Size	Average Rating	Personal Rating	Binary Download Link		
<input checked="" type="checkbox"/>	2.16-1	x86_64		37.8 kB	3.5	4	ftp.debian.org/debian/pool/main/o/zeroinstall-injector/0install-core_2.16-1_amd64.deb		
<input type="checkbox"/> > cups-libs	Fedora:34	rpm		3.5	LGPLv2 and zlib	src.fedoraproject.org/user/fdothnal	openprinting.org/aihub...	src.fedoraproject.or...	CUPS printing system - libraries
<input type="checkbox"/> > oad	Fedora:34	rpm		3.0	GPLv2+ and BSD and MIT and IBM and MPLv2.0	src.fedoraproject.org/user/fbcune	elavted.com	src.fedoraproject.or...	Cross-Platform RTS Game of Ancient Warfare
<input type="checkbox"/> > acl	Fedora:34	rpm		3.0	GPLv2+	src.fedoraproject.org/user/fdudka	savannah.nongnu.or...	src.fedoraproject.or...	Access control list utilities
<input type="checkbox"/> > binutils-gold	Fedora:34	rpm		3.0	GPLv3+	src.fedoraproject.org/user/nickc	sourceware.org/bin...	src.fedoraproject.or...	The GOLD linker, a faster alternative to the

Fig.22: PKGman selected packages (versions)

```
{
  "id": 6,
  "versions": [
    {
      "id": 32,
      "user_rating": {
        "rating_id": 1,
        "rate": 4
      },
      "rating": 3.5,
      "version": "2.16-1",
      "architecture": "x86_64",
      "swbid": "",
      "swbid_exists": null,
      "size": 38719,
      "binary_URL": "http://ftp.debian.org/debian/pool/main/z/zeroinstall-injector/0install-core_2.16-1_amd64.deb",
      "package": 6
    }
  ],
  "rating": 3.5,
  "name": "0install-core",
  "distro": "Debian:11",
  "type": "deb",
  "section": "admin",
  "license": "No license",
  "maintainer": "Thomas Leonard <talex5@gmail.com>",
  "description": "cross-distribution packaging system (non-GUI parts)",
  "homepage": "http://0install.net/",
  "repo_URL": "https://github.com/0install/0install-debian",
  "selectedVersion": "2.16-1"
},
{
  "id": 160815,
  "versions": [
    {
      "id": 170999,
      "user_rating": null,
      "rating": 5,
      "version": "1.1.6-3",
      "architecture": "all",
      "swbid": "",
      "swbid_exists": null,
      "size": 58,
      "binary_URL": "http://ftp.debian.org/debian/pool/main/s/sqlacodegen/sqlacodegen_1.1.6-3_all.deb",
      "package": 160815
    }
  ],
  "rating": 5,
  "name": "sqlacodegen",
  "distro": "Debian:11",
  "type": "deb",
  "section": "python",
  "license": "",
  "maintainer": "Python Applications Packaging Team <python-apps-team@lists.aliases.debian.org>",
  "description": "Automatic model code generator for SQLAlchemy",
  "homepage": "https://pypi.python.org/pypi/sqlacodegen",
  "repo_URL": "https://salsa.debian.org/python-team/applications/sqlacodegen"
}
]
```

Fig.23: PKGman selected packages information extracted in JSON format

```
# Generated by PKGman at 2022-02-25 22:19:00
FROM debian:11

RUN apt-get update

RUN apt-get install 0install-core=2.16-1 -y && \
  apt-get install sqlacodegen -y
```

Fig.24: PKGman generated Dockerfile with user's selected packages (versions)

PKGman also provides log in/sign up functionality. A user must be authenticated in order to be able to rate a package version.

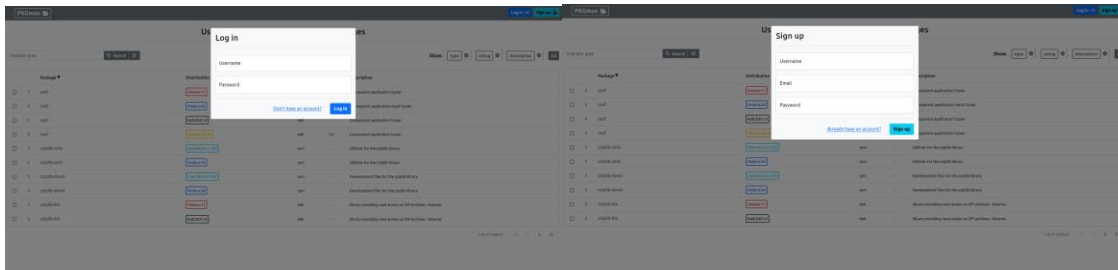


Fig.25: PKGman log in/sign up forms

## 5. Conclusions and future work

### 5.1 Conclusions

The end result is a web application that can benefit IT professionals and regular linux users alike. More than 240,000 software packages have been collected, in total. The project currently supports Ubuntu 20.04, Debian 11, Kali 2021.4, Fedora 34 and CentOS 8.4.2105, but it can easily be extended to include even more distributions that utilize the apt or dnf/yum package manager. The whole application is containerized and thus easily deployable via a Docker-Compose file.

### 5.2 Future Work

Despite the fact that the end result is a fully functional and deployable application, some additions and improvements could be made:

1. Support more Linux distributions.
2. Calculate, store and present the Software Heritage Identifier (SWHID) for each package version.
3. Add export options to more formats.
4. Add full text search functionality.



## References

- (n.d.). Ray - Scaling Python made simple, for any workload. Retrieved February 26, 2022, from <https://www.ray.io/>
- About SQLite*. (n.d.). SQLite. Retrieved February 26, 2022, from <https://www.sqlite.org/about.html>
- Bearnes, B. (2016, January 4). *Package Management Basics: apt, yum, dnf, pkg*. DigitalOcean. Retrieved February 26, 2022, from <https://www.digitalocean.com/community/tutorials/package-management-basics-apt-yum-dnf-pkg>
- Cron Job: a Comprehensive Guide for Beginners 2022*. (2022, February 9). Hostinger. Retrieved February 26, 2022, from <https://www.hostinger.com/tutorials/cron-job>
- Django introduction - Learn web development | MDN*. (2022, February 18). MDN Web Docs. Retrieved February 26, 2022, from <https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Introduction>
- Haas, J. (2020, September 11). *A Basic Guide to Linux Packages - Software & Apps*. Lifewire. Retrieved February 26, 2022, from <https://www.lifewire.com/guide-to-linux-packages-2202801>
- JSON Web Token Introduction - jwt.io*. (n.d.). JWT.io. Retrieved February 26, 2022, from <https://jwt.io/introduction>
- Refresh token with JWT authentication in Node.js*. (n.d.). Izertis. Retrieved February 26, 2022, from <https://www.izertis.com/en/-/refresh-token-with-jwt-authentication-in-node-js>
- Subramanian, P. (2018, July 10). *How To View Detailed Information About A Package In Linux*. 2DayGeek. Retrieved February 26, 2022, from <https://www.2daygeek.com/how-to-view-detailed-information-about-a-package-in-linux>