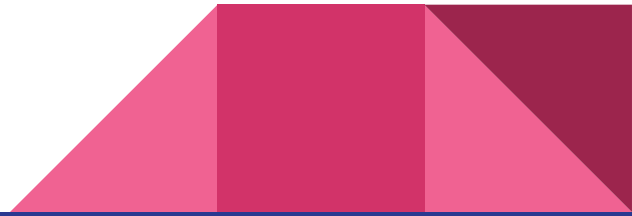
Access Review

Sponsor: Coupa Software
Point of Contact: Philip Cox
Advisor: Patrick Tague
Team Member: Kaiyu Liu, Rundong Liu, Haoran Liu

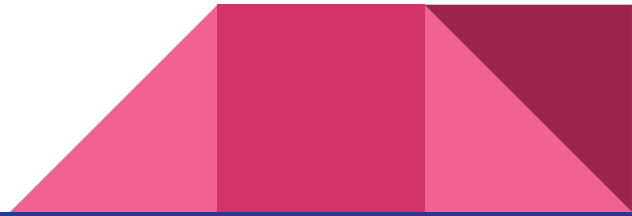
Roadmap

- Introduction
- Motivation
- Related work
- System design
- System implementation
- Demo
- Experiments/analysis
- Conclusions and future work



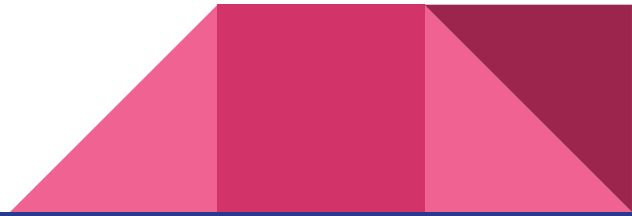
Introduction

- Least privilege is the basic security principle used in resource management
- Access Review helps to enforce this security principle
- It plays an important role when dealing with restricted resources



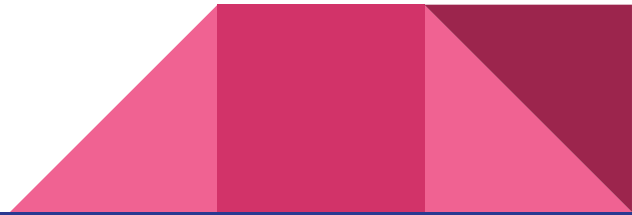
Motivation

- Lots of applications are used in Coupa’s business
- Employees have different access permissions for these applications.
- The access review for these permissions used to be done manually and waste a lot of time.
- We need to come out of a solution to make access review process easier to manage and realize better visualization.



Related Work

- Did research of the access control at the very beginning
  - Access control systems perform authorization, identification, authentication etc. of entities
  - We mainly use role-based access control (RBAC) in our project
  - In RBAC, we can easily created, changed, or deleted roles. These can be done without updating the privileges for every user.



Related Work

- Did some research of the Play and Django and Ruby on Rails framework.

	Play	Django	Rails
Language	Java/Scala	Python	Ruby
design pattern	MVC	MVC	MVC
Main feature	It aims to optimize developer productivity by using convention over configuration, hot code reloading and display of errors in the browser.	Django's primary goal is to ease the creation of complex, database-driven websites	It encourages and facilitates the use of web standards such as JSON or XML for data transfer, and HTML, CSS and JavaScript for display and user interfacing.

Related Work

- Did some research of the PostgreSQL and MySQL and SQLite database

MySQL	PostgreSQL	SQLite
Easy to work with	Strong community	Good for developing and testing
secure and speedy to use	Strong third-party support	file-based
Scalable and powerful	Extensible	no user management
has some known issues	Performance is not good as MySQL	Lack of possibility to tinker with for additional performance

Related Work

- Discuss with the client
  - Visit Coupa Software HQ to discuss about the project in detail
- Design the project and the division of work is clear-cut
  - everyone being charged with specific responsibilities
- Worked out the schedule that can meet the need
  - Use Agile development method

System Design

- There are 4 kinds of users in the company
  - Administrator
  - Manager/reviewer
  - Auditor
  - Normal user
- Different users should have different content displayed when logged into the system
- Different users should have different read/write permission of the data
- All the modification of the data would be stored into database

System Design

Model:

- Admin -The one that controls the whole app
- User -The normal user in the app
- Auditor -The one that audit all the applications
- Manager -The one that reviews applications assigned to him
- Application -Services involved in the company

System Design

Relation

- App-User-Permission

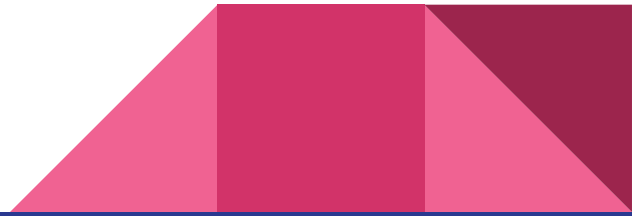
  Access control list shows user permission

- App-Manager

  Show which manager is assigned to an app

- App-Auditor

  Show which auditor is assigned to an app



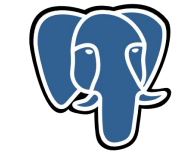
System Design

- Framework-Django

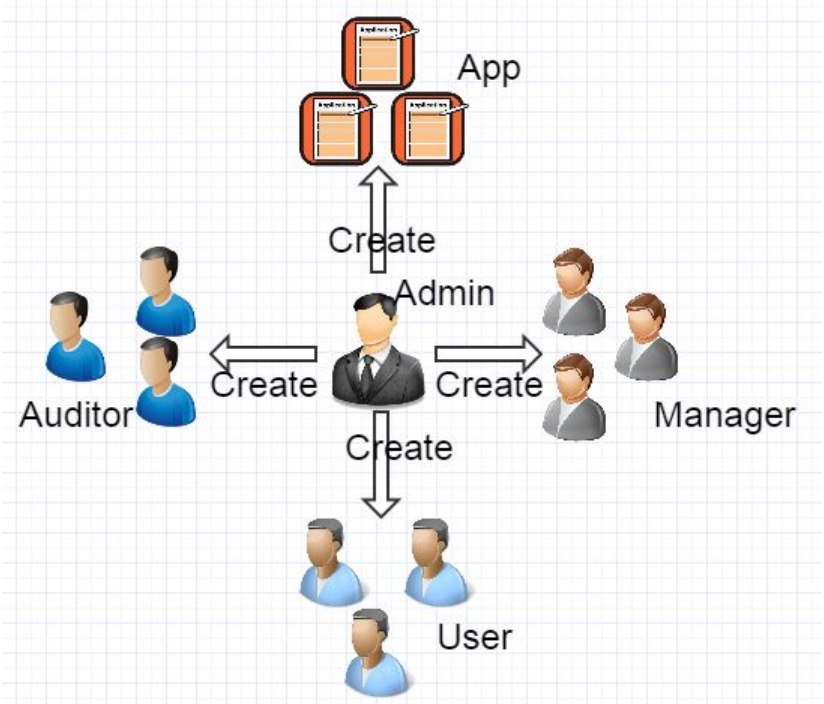
  Django is a high-level MVC Python Web framework that make web development quick and clean. Django is free and open source. It helps developers to create complicated, database-driven websites.

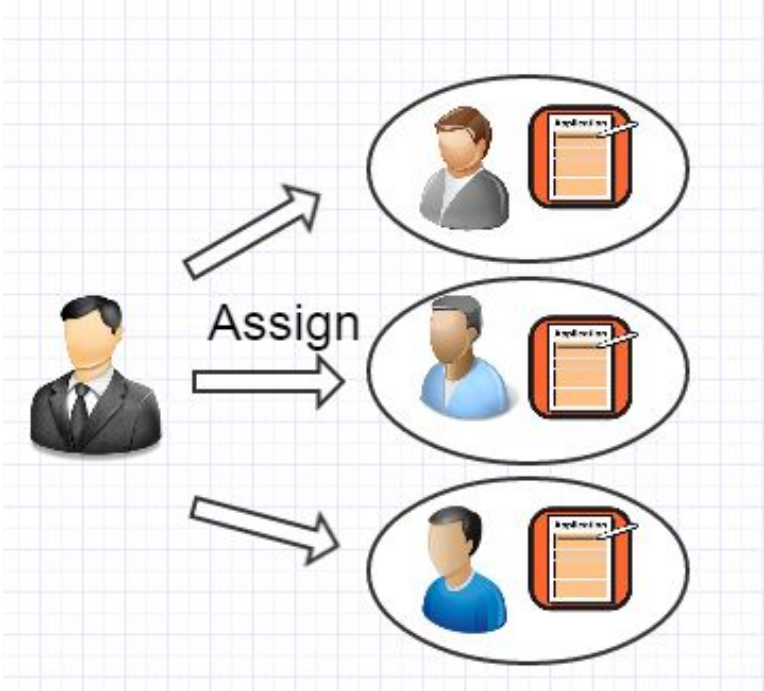
- Database-Sqlite, PostgreSQL

  Sqlite in development process, PostgreSQL in testing process.

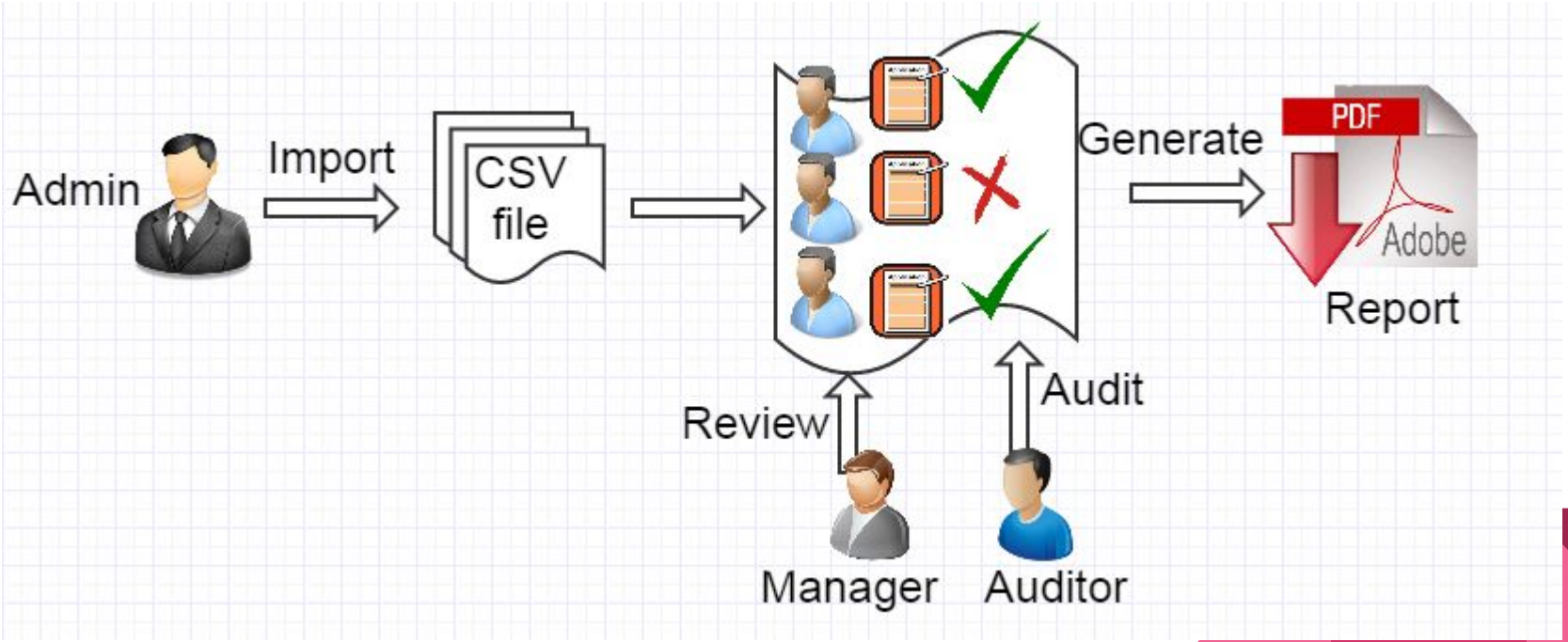


System Implementation-Workflow



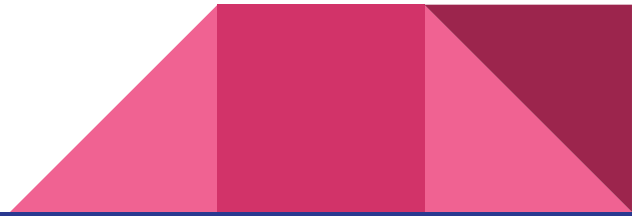


System Implementation-Workflow



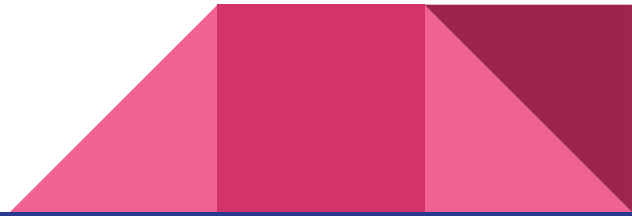
System Implementation

- Implement registration and authentication of different users
- Design web interface for interaction and operations
- Design workflow for the system
- Extract data supplied by customer
- Deal with multithread database update
- Support ACID(Atomicity, Consistency, Isolation, Durability) of the database



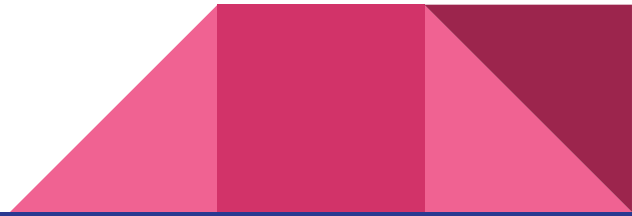
System Implementation

- One-click import access control list
  - Create method to convert csv file to abstract access control list and
  - Insert record to backend database
- Generate PDF report - ReportLab python package
  - Reportlab is a python package used to generate customized pdf file
  - We use it to generate access review report in a pdf format



System Implementation

- Multi-threading
  - We make our web application multithreading.
  - We implement mechanism to ensure database transaction atomicity



System Implementation-Security

For security reason

We add anti-csrf token in forms and requests made by browser.

```
form id="view_form" class="form-horizontal" name="update-form" method="post" action="/permission/1">
 <input type="hidden" value="WxsPAxdvKWu2VqPvpOUUeEBW4Ds6EUKf" name="csrfmiddlewaretoken"></input>
<div class="table-responsive"></div>
<div class="form-actions"></div>
```

We add session based access control in the system to only allow legitimate user to access to private data

```
# check whether user in customer group
def in_manager(function=None):
    actual_decorator = user_passes_test(
        lambda u: u.is_authenticated() and u.groups.filter(name='Manager').exists()
    )
    return actual_decorator(function)
```

How to Run

git clone https://github.com/KearneyLiu/Access_Review.git

python manage.py migrate

python manage.py runserver 0.0.0.0:8000

Demo!

Experiments/analysis

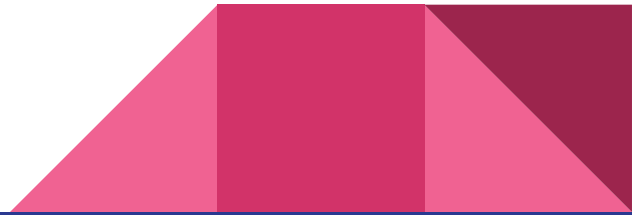
We test our system while developing it.

It is mainly focus on the correctness and performance of the system.

- At first, we used simple data to test our application and make sure everything works good.
- We log in the system using different entities and the contents and control panels are showed as expected.
    - The admin can manage the whole system. He/she changed some users permission to see the result. And also the admin can One-click import access control list.

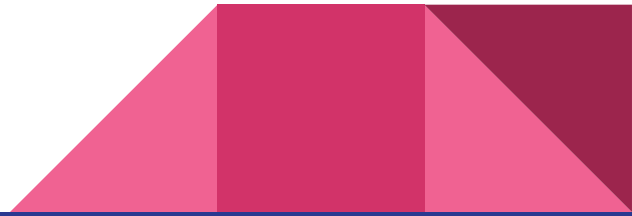
Experiments/analysis

- Cont’d
  - The manager can modify all the applications’ user list that he is responsible for. He/she can change the “read”/”read & write”.
  - The normal users can only review the userlist that he is related to.
  - The auditors can see all the access review lists, they can verify the list.
- Do experiments to download the access review result as a PDF format file.
- The system has good concurrency control mechanism, works well in the multi threaded environment



Conclusions

- We designed schema based on the provided data, and designed the project based on the real need.
- We did the agile development and revise the code during the develop process and finally implemented all the functions we planned to deliver.
- The project facilitates the access review process and realizes better visualization.
- The project is a good solution for role based access control and can work with real world resource management



Future work

- Refine schemas to deal with more complicated situations beyond this project
- Refine the front-end part to make it more beautiful.
- Combine system with real world authentication for dynamic access control
- If the users forget the password, we will provide a method for them to reset their password.

Reference

http://searchsecurity.techtarget.com/definition/access-control

http://csrc.nist.gov/publications/nistbul/csl95-12.txt

http://www.diogonunes.com/blog/rails-vs-django-vs-play-frameworks/

http://codemonkeyism.com/playing-play-framework-java/

https://en.wikipedia.org/wiki/Ruby_on_Rails

https://en.wikipedia.org/wiki/Django_(web_framework)

https://www.digitalocean.com/community/tutorials/sqlite-vs-mysql-vs-postgresql-a-comparison-of-relational-database-management-systems

https://pypi.python.org/pypi/reportlab

Questions