Cloud computing project- "PragatiDrive"

Anmol Anubhai (121004)
anmol.anubhai@iet.ahduni.edu.in

Kunjan Patel (121027)
kunjan.patel@iet.ahduni.edu.in

Rahul Patel (121040)
rahul.patel@iet.ahduni.edu.in

Shashwat Sanghavi (121049)
shashwat.sanghavi@iet.ahduni.edu.in

Guided by:
Dr. Sanjay Chaudhary
sanjay.chaudhary@ahduni.edu.in

Institute of Engineering & Technology, Ahmedabad University

November 11, 2015

1 Project Brief

PragatiDrive is a SAAS (Software as a service) based storage over a cloud application. We have ideated PragatiDrive for universities and schools. The stakeholders of universities and schools can upload and maintain their documents in a fast and secured way using this cloud storage service. Also the regulations about what and how much to store are customizable according to the needs of the organization. SAAS help provide key functionalities like Multi-Tenancy, Customizable Business Logic, Customizable GUI, Billing Information, Privacy and security and the fact that each tenant only needs to pay for the features and resources used by him.

Django, a python based framework using RESTful programming has been used. REST is the underlying architectural principle of the web. One of the key advantages of RESTful programming is that the client and server can interact in a number of ways without the client actually knowing about the server or the resources it is hosting. The key constraint is that the server and client must both agree on the media used, which in the case of the web is HTML. The server needs to provide whatever information the client needs to interact with the service. The client stays unaware of the API used.

Next, we used SQLite to design the database. Since SQLite database is integrated with the framework by default, we made use of the same. SQLite offers key features like Zero Configuration, Sever less, Single Database File,Stable Cross Platform Database File,Compact, Manifest Typing, Readable Source code,SQL statements compile into virtual machine codes, Public Domain and SQL language extensions. We started by installing the Citric Xenserver on bare metal. The key advantages of the Xenserver are as follows:

- Strongest and the fastest virtualization software
- A complete, managed virtualization software
- It can handle any kind of server workload. It is specially designed for Windows and Linux virtual servers.
- It has a virtual machine that runs in ten minutes.
- The XenServer is cloud-proven virtualization platform that's packed with all the things needed in order to create and manage any kind of virtual infrastructure.
- XenMotion Live Migration-This allows the VMs to be transferred between physical servers with no interruption at all.

The Xenserver has many advantages and key functionalities which we shall discuss ahead. Thus, for PragatiDrive we made use of Django framework using SQLite database and Xenserver.

2 Functionalities

PragatiDrive was born out of a growing need for a storage based cloud application especially designed to meet the requirements of universities and schools. In today's age when education is becoming more and more technology friendly as well as extremely easy to access, it is mandatory for universities and schools to have a cloud based application for data storage. This will enable students as well as professors to get efficient and easy access to their files as and when needed. Though this also poses challenges like:

- Security and Privacy

- Regulating the size of files
- Regulating the types of files

To be able to solve these challenges efficiently, we made use of SAAS (Software as a Service).

SAAS has many key functionalities and advantages as stated in the previous section. We shall now discuss each of these in detail.

1. **Multitenancy** : A single instance of a software application serves multiple users. Software development and maintenance costs are shared by these multiple users also known as tenants. Thus, this model proves to be more economical. One update by the provider serves all which is not true in the case of single tenancy. The multiple tenants can customize certain aspects like color and GUI but they cannot change the core application's code. Though one tenant cannot access another tenant's data and thus security as well as privacy is ensured.

   In the case of PragatiDrive as well we have ensured that tenants i.e. in our case different universities/colleges can access as well as store their own data and customize certain aspects like the storage capacity given to a user or the type of file allowed. But one university cannot view another's files.

2. **Customizable Business Logic** : SAAS helps to provide customizable business logic by allowing tenants to choose certain features as well as set their own terms. In the case of PragatiDrive we do not allow complete customization of business logic as in most cases of universities and colleges certain key fields and requirements are common. Thus, there is no ardent need for complete customization of business logic. Though we do allow the admin to decide as to how much space should be allocated to every user. He also decides the type of file that should be allowed to be uploaded. He can create users as well.

3. **Billing Information** : In the case of PragatiDrive the total space used by a certain admin i.e. the admin provides certain storage space to every user, is summed up and the admin only has to pay for the storage space used by him. This enables a transparent system in which the admin can view the space used by him and pay accordingly. One of the key advantages of SAAS cloud applications is the fact that a user has to pay only for the functionalities or resources used. Thus, it out competes the old model when a user had to pay and buy the entire software irrespective of the functionalities/resources he will be using of the same.

4. **Privacy and Security** : In the case of SAAS cloud applications, security and privacy are of utmost importance. This is because no tenant will want another tenant to be able to access his data. This is ensured by giving every tenant a separate authorization ID and password. In the case of SAAS cloud applications related Privacy and Security issues the following questions are key.

- Who owns the data?
- How is the data segregated?
- What are the measures taken in case of a security breach?

Many data analysts believe that tenants have a more secure public environment cloud than with an internal data center. Using a cloud provider may in fact give you a valid disaster backup plan, because they have tested the fact that the data replicates in different locations and can be accessed. This may be something that your internal data center never did. In the case of PragatiDrive, we have ensured that every admin is given a unique authorization ID as well as password. This ensures that one tenant cannot view another's files. The superuser can view all tenants and their storage spaces used.

3 Hardware & Software used

3.1 Django framework

We have used Django, a Python based framework for RESTful programming, to build PragatiDrive. SQLite database is integrated with the framework by default so we have used the same. The biggest advantage in using Django is, it provides a very nice platform to develop a SAAS application and supports structured programming.

Different components of the application like database, business logic, user interaction, data handling, metadata, etc. are written in different files and all these components are managed by Django. Every application has a file called views.py which handles the business logic, fires queries on the database, manipulates the data according to the need and then passes the data to UI module for presenting it to the user. We shall look into the key features as well as advantages of using Django.

- **Python** : Python has an easy to understand syntax. It has multiple libraries of standard as well as contributed modules that cover almost all possible functionalities. it has an object oriented model. Also, Python's runtime environment is fast and stable. It supports a variety of platforms: UNIX, LINUX, Windows and MAC. Python supports a variety of database servers as well.

- **MTV Architecture** : M stands for Model, the data access layer, which contains everything about the data how to access it, how to validate it, its characteristics, and the relationships between data. T stands for Template, the presentation layer, which contains presentation-related decisions how something should be displayed on a Web page. V stands for View, which basically acts as controller which renders an HTML page using templates and models.

- **Robust ORM System** : Django supports a large number of database engines. If a decision is made to change from one database engine to another then all it takes is to change the configuration file. Thus, it ensures greater flexibility.

- **Readable URLs** : With Django the URLS created are more readable than any ASP or PHP website. They also ensure easy of creation.
- **Templates** : It speeds up development while making it easy. It allows creation of templates with injectable program logic. Thus, this ensures modularity and simplifies complex codes.
- **Automatic Admin Interface** : Django comes with an admin interface by default. This is a web portal which allows the admin to control various web applications.
- **Integrated development environment** : Django provides a complete development environment. It comes with a lightweight Web server for development and testing. When the debugging mode is enabled, Django provides very thorough and detailed error messages, with a lot of debugging information which, in turn, makes debugging easier, and allows one to quickly isolate bugs.
- **Caching** : Django comes with a robust cache system that lets you save dynamic pages so they don't have to be calculated for each request. For convenience, Django offers different levels of cache granularity: You can cache the output of specific views, you can cache only the pieces that are difficult to produce, or you can cache your entire site.

3.2 File structure

Each Django Project folder contains the following key files:

- settings.py: This contains project settings, such as database connection information, email server configuration, installed apps, media folder location and static folder location with their URLs, template folder location and reference to some Django modules.
- urls.py: Can be considered the DNS of our Django app. Each request to the application goes through urls.py, and entries in the urlpatterns list in this decide which request is to be forwarded to which view of a particular app.
- manage.py: Contains commands that, as the name suggests, are used for managing our Django project for example, startapp, syncdb, etc. Each Django Web application consists of smaller modules called apps. Each app inside our project contains three files:
  - models.py: This is the place for all your application data (that you want to be persistent). These are basically classes extending models.Model, which is provided by Django ORM library. It is recommended that business logic should be written in models so that it remains intact with models and can be used by any other app of your Django project.

- views.py: This is the place where you render HTML pages for the HTTP requests that are redirected to particular view function via urls.py. Basically, views contains functions, each of which get the HTTP request as first parameter. They get data from models and render HTML pages using Templates, or redirect to another view function.
- tests.py: This is the place for your test cases. These are functions performing unit test cases. This file should be effectively used for regression testing (to check the side effects of your code changes).

3.3 Database

Next, we shall discuss in detail the key features of the SQLite database as stated in the previous section. The key functionalities and advantages are as under:

- **Zero Configuration** : SQLite does not need to be "installed" before it is used. There is no need for an administrator to create a new database instance or assign access permissions to users. SQLite uses no configuration files.
- **Sever-less** : Most Database engines communicate with the server through some interprocess(TCP/IP) when they want to read or write from the database. In the case of SQLite, there is no need of any kind of interprocess. The process that wants to read or write from the database directly does so without communicating with any server. No separate server needed to install and maintain.
- **Single Database File** : The entire database gets stored in a single disk file located in the directory. SQLite can read this disk file and the database. Also if they are writable then SQLite can also make any changes needed in the database. In the case of other database engines, the database is stored in multiple files which though improves security also makes access difficult.
- **Cross Platform compatibility** : SQLite database files can be copied and made to run on other machines with different architectures just as easily.
- **Compact** : When optimized for size, the whole SQLite library with everything enabled is less than 500KiB in size
- **Manifest Typing** : In most SQL database engines a datatype is associated with each column in a table and only values of that particular datatype are allowed to be stored in that column. In manifest typing, the datatype is a property of the value itself, not of the column in which the value is stored. SQLite thus allows the user to store any value of any datatype into any column regardless of the declared type of that column.
- **Variable Length Records**: SQLite, uses only the amount of disk space actually needed to store the information in a row. If you store a single character in a VARCHAR(100) column, then only a single byte of disk space is consumed.

- **Readable Source code**: All procedures, data structures and variables are carefully commented to give useful information about what they do.

3.4 RESTful programming in Django

Next we shall look into what exactly is RESTful programming as the Django framework makes use of Restful programing. A "REST API" is two things: Its a web service and it's RESTful.

REST API has some loose coupling. The client need not be aware of internal implementation details. One of the key advantages of RESTful programming is that the client and server can interact in a number of ways without the client actually knowing about the server or the resources it is hosting. It is stateless in nature.

A RESTful API breaks down a transaction to create a series of small modules, each of which addresses a particular underlying part of the transaction. This modularity provides developers with a lot of flexibility but can also be challenging for developers to design from scratch. RESTful API also provides with very large scalability. They simply use "PUT" to change the state of or update a resource, which can be an object, file or block; "GET" to retrieve a resource; POST" to create that resource; and "DELETE" to remove it.

3.5 XenServer

We installed XenServer on bare metal. We shall discuss XenServer in detail. The key features as well as functionalities are as below:

- **Easy configuration** The Citrix XenServer has a virtual machine that runs in as ten minutes. Furthermore, there's no need to configure a complicated management infrastructure nor come up with a dedicated storage network.
- **Powerful Server Management** : It is equipped with all tools needed to create and maintain any kind of virtual infrastructure.
- **XenCentre Management Control** : This allows one to monitor and control several VMs coming from the same centralized management console. One can start, stop, create, migrate or create a backup of VMs in just a few clicks.
- **XenMotion Live Migration** : This allows the VMs to be transferred between physical servers without any interruption at all.
- **VM Protection and Recovery** : One can set policies that will regularly gather the VM snapshots and archive them. This also enables us to take the VM to a previous stage and help recovery.
- **Dynamic Memory** : This feature is used to share the unused server memory between VMs, resulting in an improvement in the application performance and a greater number of VMs.

- **Distributed Virtual Switching** : This can be used to create a network that can handle multi-tenancy. The basis for building both public and private cloud.

  Using these hardware and software resources we have devised PragatiDrive which starts with a login page asking the user to enter his authorization ID and password. Next, if he is a superuser then he gets a complete access to all tenants' information. He can view their names as well as the storage space utilized. Next, if he is a admin then he can create a user and decide the space that will get allocated to that user along with the type of file that he/she can upload. If he is a normal user then on logging in he can upload files as well as view the space being utilized by him.

4 Software architecture

The digram shown in Figure 1 contains the software architecture of PragatiDrive. The client first needs to login in to the system with his credentials. Once the credentials are verified the client is redirected to the respective page. Whenever a new client request comes it is trapped by the server. This request is further scrutinized to find the requested URL.

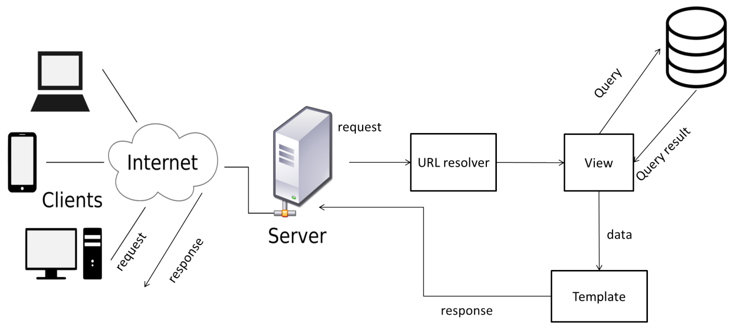


Figure 1: Software architecture of PragatiDrive

The list of all valid URLs is contained in a file against the request URL is compared. If a valid match is not found the system throws error. Each URL is mapped to a function which is called upon when a valid URL is found. These functioned are defined in the views.py file. They primarily contain the business logic and are responsible for the interaction with database.
After the data relevant to the given request is fetched it is injected into the respective template. Template is a static file (HTML, CSS, etc) responsible for the look and feel of the web page. Thus Django helps in efficiently managing the front and back end by separating the two layers. Also, the templates allow inheritance which helps in easily deriving the structure of the page from the parent template. Individual structures can be over-rided in the child template as per the requirement.

5 Database design

The database mainly contains 5 models. The database schema is represented in the form of class diagram and hence it is independent of the back end database.

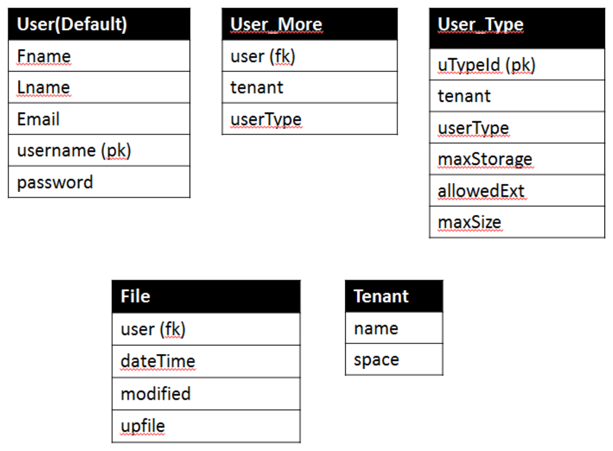


Figure 2: Database design in terms of the class diagram for the proposed SAAS application

Figure 2 contains the class representations of all tables used in building the application. The *User* class is the default table provided by Django. It contains a set of bare minimum attributes. If more attributes are required for the entity user we need to extend that class. *User_More* contains the other details which are required by the our SAAS application. *User_Type* class is used to store details about the tenant to which it is linked to, allowed file extensions which can be uploaded, maximum storage space allocated and maximum size of file to be uploaded. *File* class contains the details about the uploaded file like where it is stored, at what time it was uploaded and when was it last modified.

6 Screenshots

We shall look at the different screens of PragatiDrive that the user will interact with in this section. We have the following screens and their respective images as shown below:

1. Login
2. Super User Page
3. Privileged User Page
4. Standard User Page
5. Add Tenant

6. Edit Tenant
7. Tenant Admin Pages (A and B)
8. Error Message Page

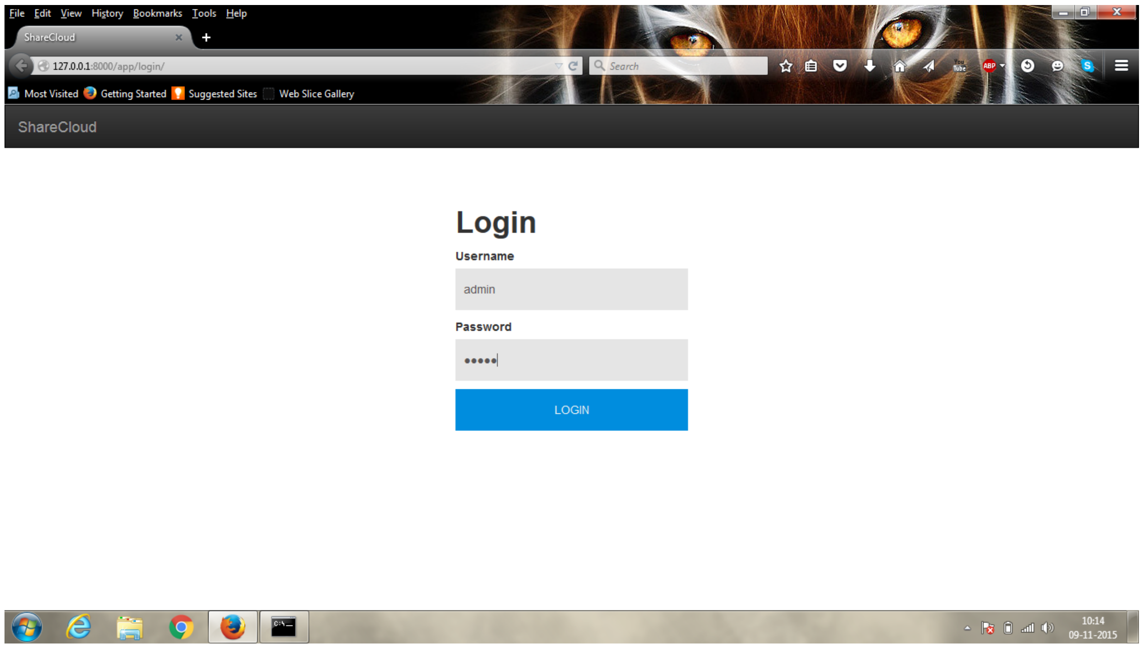
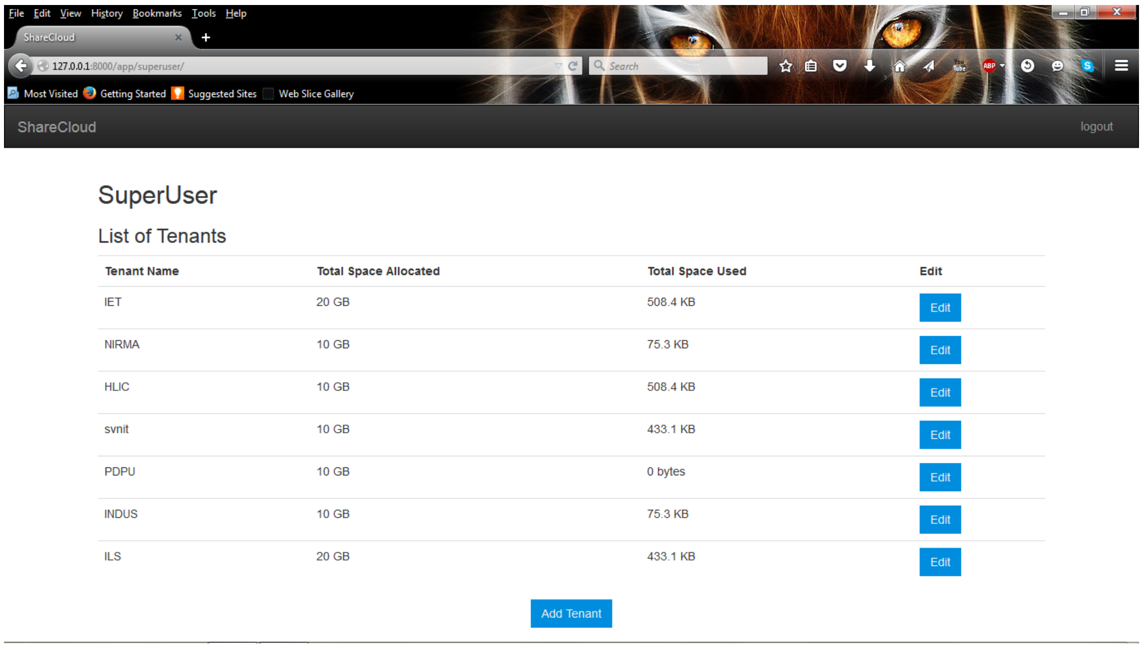


Figure 3: 1. The login page takes in the authorization ID and password from the user

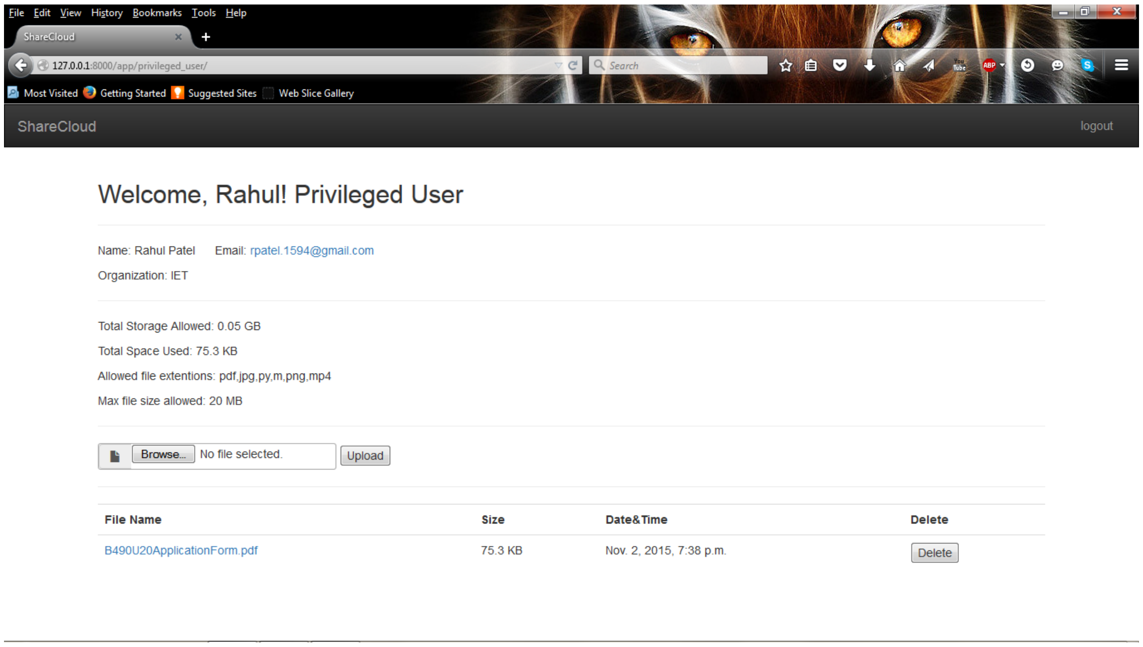

File Edit View History Bookmarks Tools Help
ShareCloud
127.0.0.1:8000/app/superuser/
Search
Most Visited Getting Started Suggested Sites Web Slice Gallery
ShareCloud
logout

SuperUser

List of Tenants

Tenant Name	Total Space Allocated	Total Space Used	Edit
IET	20 GB	508.4 KB	Edit
NIRMA	10 GB	75.3 KB	Edit
HLIC	10 GB	508.4 KB	Edit
svnit	10 GB	433.1 KB	Edit
PDPU	10 GB	0 bytes	Edit
INDUS	10 GB	75.3 KB	Edit
ILS	20 GB	433.1 KB	Edit

Figure 4: 2. The SuperUser can view all the tenants and the space they are utilizing


File Edit View History Bookmarks Tools Help
ShareCloud
127.0.0.1:8000/app/privileged_user/
Search
Most Visited Getting Started Suggested Sites Web Slice Gallery
ShareCloud
logout

Welcome, Rahul! Privileged User

Name: Rahul Patel Email: rpatel.1594@gmail.com

Organization: IET

Total Storage Allowed: 0.05 GB

Total Space Used: 75.3 KB

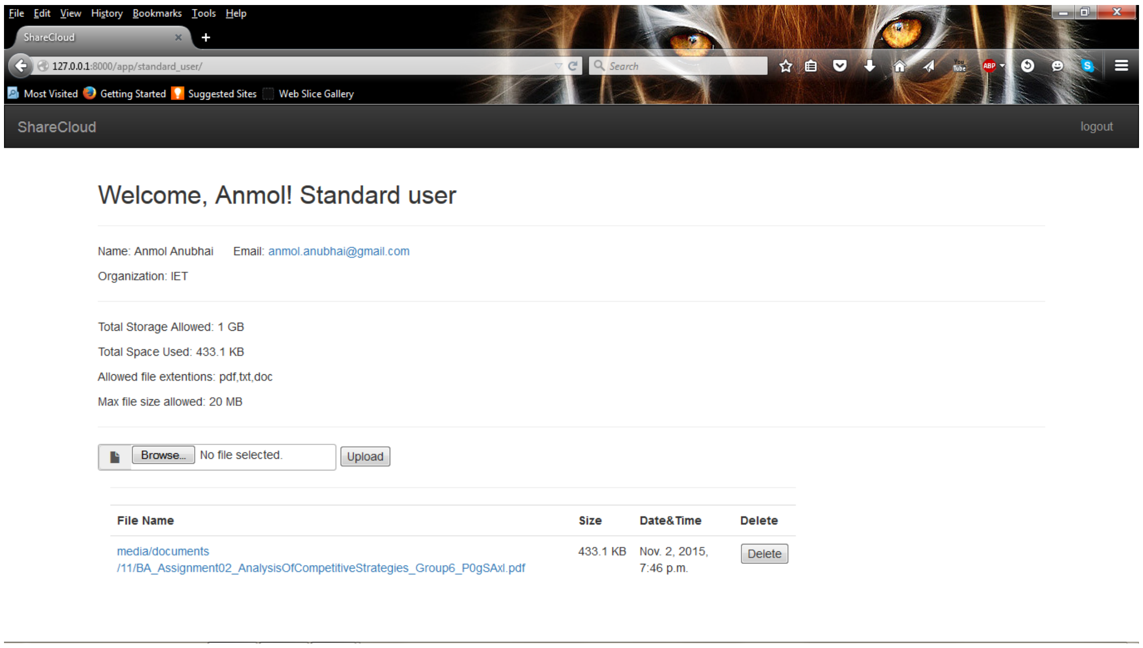
Allowed file extentions: pdf,jpg,py,m,png,mp4

Max file size allowed: 20 MB

Browse... No file selected.
Upload

File Name	Size	Date&Time	Delete
B490U20ApplicationForm.pdf	75.3 KB	Nov. 2, 2015, 7:38 p.m.	Delete

Figure 5: 3. The Privileged User can upload his files which fall in the allowed category and also view the space being utilized by him


File Edit View History Bookmarks Tools Help
ShareCloud
127.0.0.1:8000/app/standard_user/
Most Visited Getting Started Suggested Sites Web Slice Gallery
ShareCloud
logout

Welcome, Anmol! Standard user

Name: Anmol Anubhai Email: anmol.anubhai@gmail.com

Organization: IET

Total Storage Allowed: 1 GB

Total Space Used: 433.1 KB

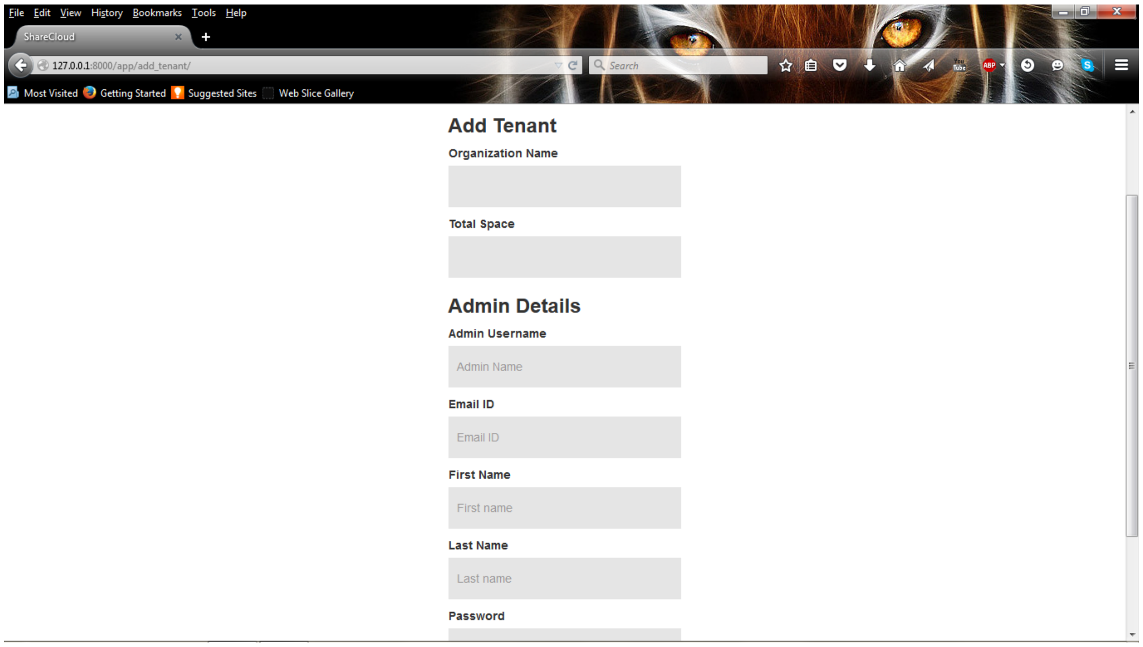
Allowed file extentions: pdf,txt,doc

Max file size allowed: 20 MB

Browse... No file selected. Upload

File Name	Size	Date&Time	Delete
media/documents /11/BA_Assignment02_AnalysisOfCompetitiveStrategies_Group6_P0gSAxl.pdf	433.1 KB	Nov. 2, 2015, 7:46 p.m.	Delete

Figure 6: 4. The standard user can upload files which fall in his allowed category and also view the space being utilized


File Edit View History Bookmarks Tools Help
ShareCloud
127.0.0.1:8000/app/add_tenant/
Most Visited Getting Started Suggested Sites Web Slice Gallery

Add Tenant

Organization Name

Total Space

Admin Details

Admin Username

Admin Name

Email ID

Email ID

First Name

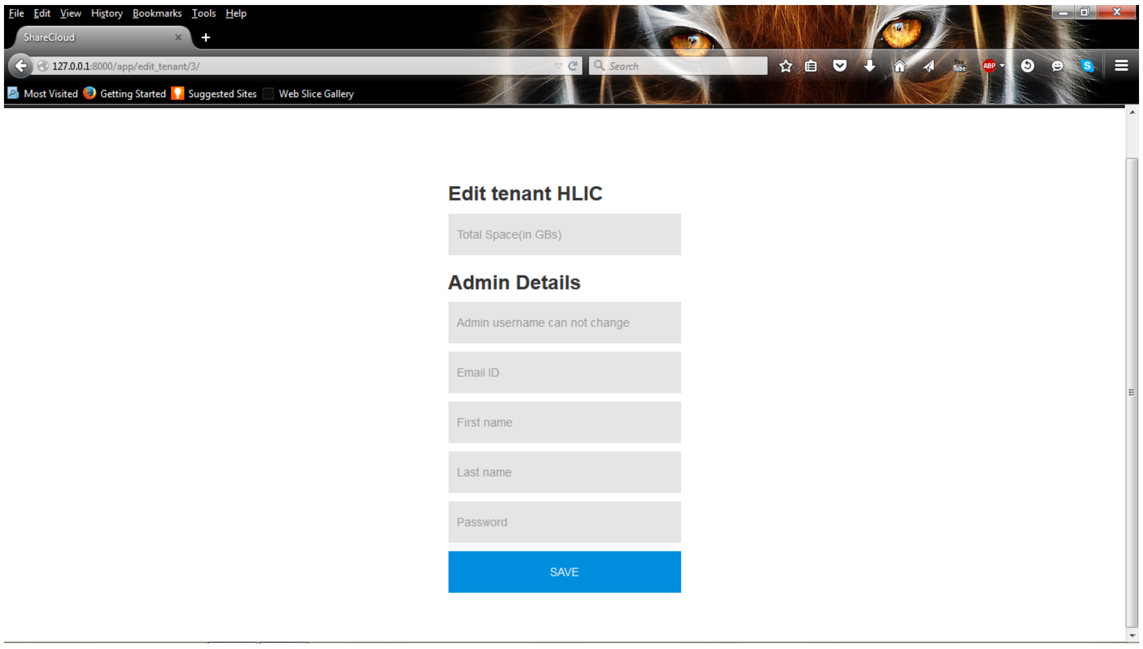
First name

Last Name

Last name

Password

Figure 7: 5. The Super User can add tenants by filling in their information


File Edit View History Bookmarks Tools Help
ShareCloud
127.0.0.1:8000/app/edit_tenant/3/
Most Visited Getting Started Suggested Sites Web Slice Gallery

Edit tenant HLIC

Total Space(in GBs)

Admin Details

Admin username can not change

Email ID

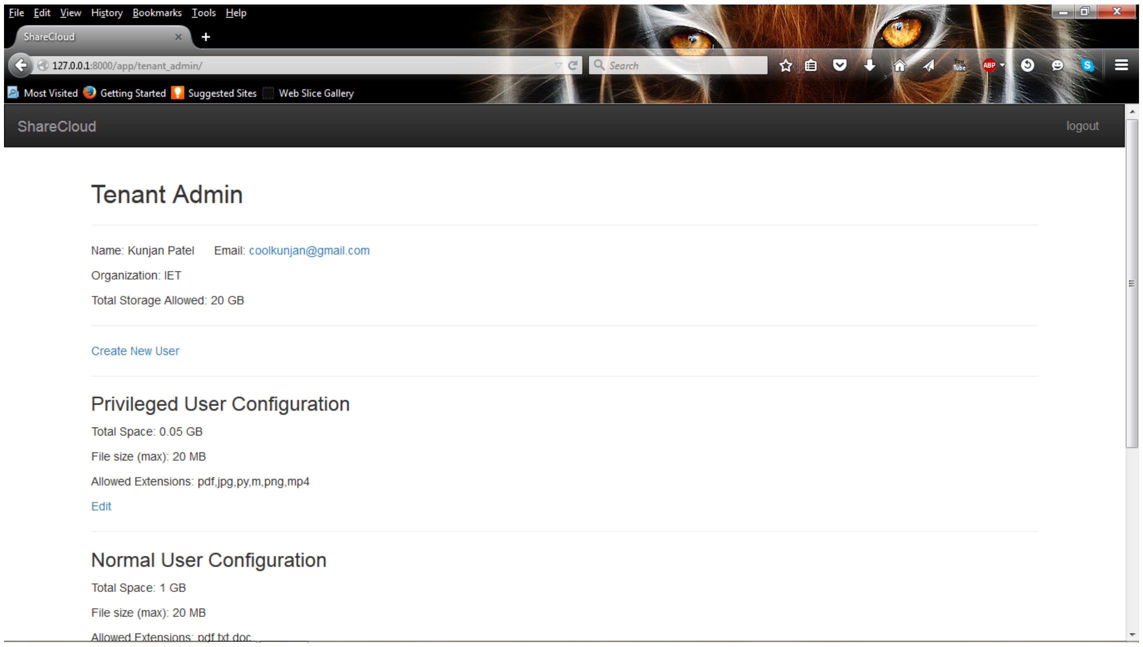
First name

Last name

Password

SAVE

Figure 8: 6. The Super User can edit a tenant's information as and when required.


File Edit View History Bookmarks Tools Help
ShareCloud
127.0.0.1:8000/app/tenant_admin/
Most Visited Getting Started Suggested Sites Web Slice Gallery
ShareCloud
logout

Tenant Admin

Name: Kunjan Patel Email: coolkunjan@gmail.com

Organization: IET

Total Storage Allowed: 20 GB

Create New User

Privileged User Configuration

Total Space: 0.05 GB

File size (max): 20 MB

Allowed Extensions: pdf,jpg,py,m,png,mp4

Edit

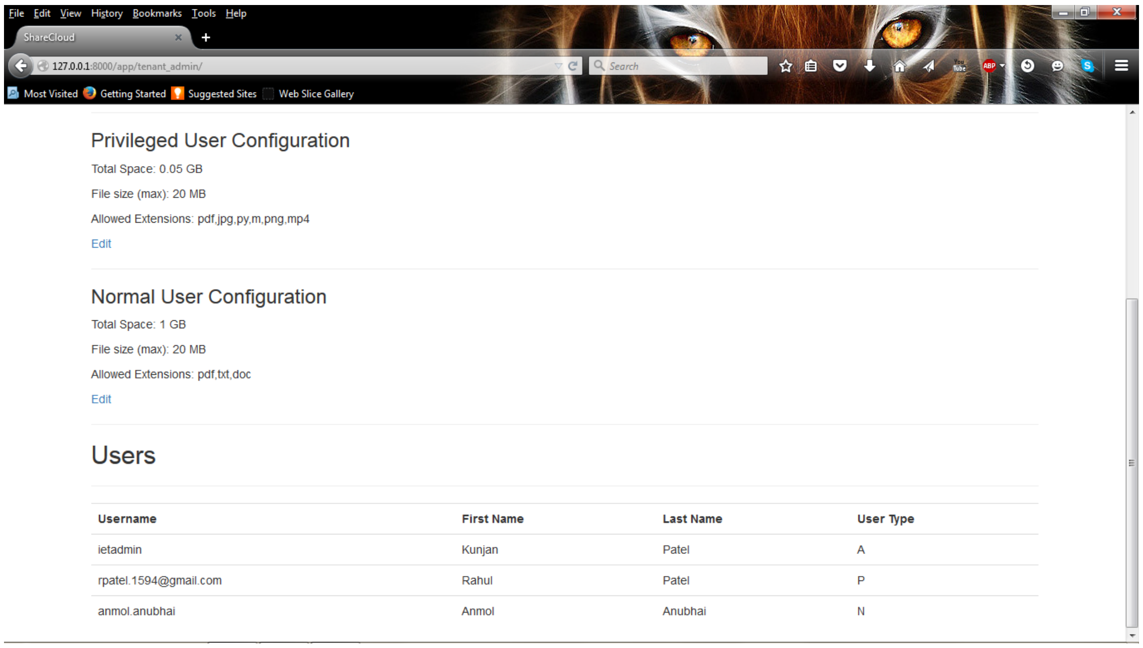
Normal User Configuration

Total Space: 1 GB

File size (max): 20 MB

Allowed Extensions: pdf,txt,doc

Figure 9: 7A. The tenant admin can view all the rights and regulations decided by him for the privileged and standard users as shown above.


File Edit View History Bookmarks Tools Help
ShareCloud
127.0.0.1:8000/app/tenant_admin/
Search
Most Visited Getting Started Suggested Sites Web Slice Gallery

Privileged User Configuration

Total Space: 0.05 GB

File size (max): 20 MB

Allowed Extensions: pdf,jpg,py,m,png,mp4

Edit

Normal User Configuration

Total Space: 1 GB

File size (max): 20 MB

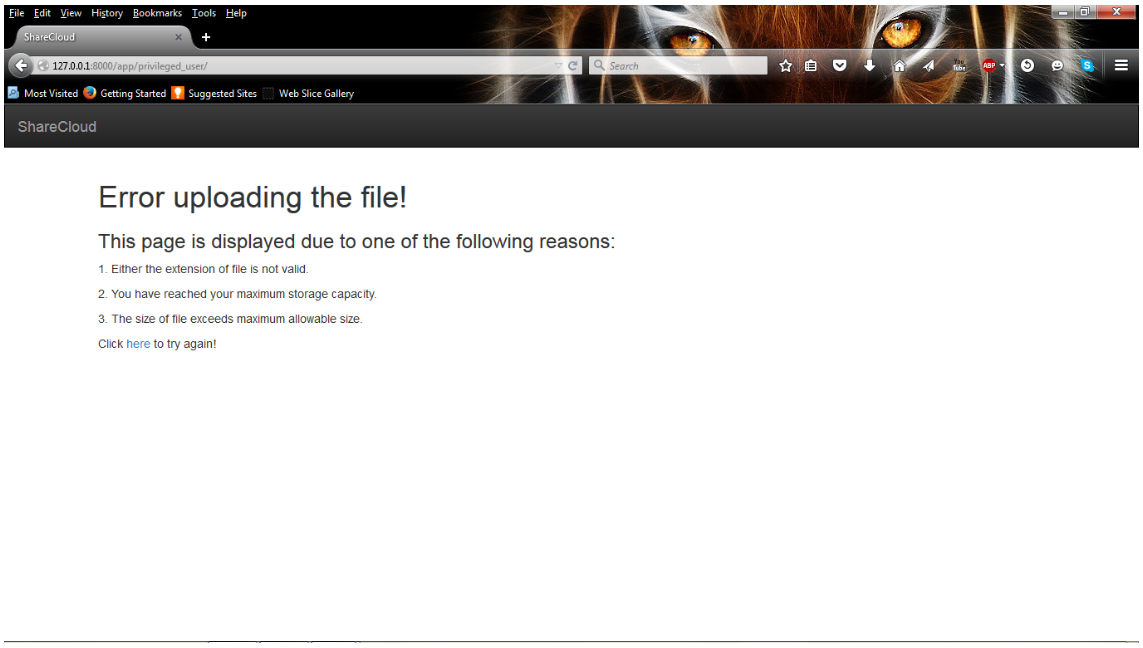
Allowed Extensions: pdf,txt,doc

Edit

Users

Username	First Name	Last Name	User Type
ietadmin	Kunjan	Patel	A
rpatel.1594@gmail.com	Rahul	Patel	P
anmol.anubhai	Anmol	Anubhai	N

Figure 10: 7B. The tenant admin can view all the rights and regulations decided by him for the privileged and standard users as shown above.


File Edit View History Bookmarks Tools Help
ShareCloud
127.0.0.1:8000/app/privileged_user/
Search
Most Visited Getting Started Suggested Sites Web Slice Gallery
ShareCloud

Error uploading the file!

This page is displayed due to one of the following reasons:

1. Either the extension of file is not valid.
2. You have reached your maximum storage capacity.
3. The size of file exceeds maximum allowable size.

Click here to try again!

Figure 11: 8. In case of a login error an error message is displayed.

7 References

1. www.quora.com
2. www.blog.pythonlibrary.org
3. www.djangoproject.com
4. www.sqlite.org/different.html
5. www.searchcloudstorage.techtarget.com
6. www.djangobook.com