Cloud Application Portability

A Project Report

Submitted by

Suryanarayan Sudersanam (B001),

Shon Bangale (B010),

G.V.A. Sashank (B024),

Srija Ganguly (B026)

Under the Guidance of

Prof. Swarnalata Bollavarapu

In partial fulfillment for the award of the degree of

B.Tech

IN Computer Engineering

At

SVKM'S NMIMS Deemed-to-be UNIVERSITY

Mukesh Patel School of Technology Management and Engineering, NMIMS,Mumbai

April, 2017

DECLARATION

I,Suryanarayan S., Shon B, G.V.A. Sashank,Srija G., Roll No. B001, B010, B024, B026 B.Tech (Computer Engineering), VIII semester understand that plagiarism is defined as anyone or combination of the following:

1. Un-credited verbatim copying of individual sentences, paragraphs or illustration (such as graphs, diagrams, etc.) from any source, published or unpublished, including the internet.
2. Un-credited improper paraphrasing of pages paragraphs (changing a few words phrases, or rearranging the original sentence order)
3. Credited verbatim copying of a major portion of a paper (or thesis chapter) without clear delineation of who did wrote what. (Source:IEEE, The institute, Dec. 2004)
4. I have made sure that all the ideas, expressions, graphs, diagrams, etc., that are not a result of my work, are properly credited. Long phrases or sentences that had to be used verbatim from published literature have been clearly identified using quotation marks.
5. I affirm that no portion of my work can be considered as plagiarism and I take full responsibility if such a complaint occurs. I understand fully well that the guide of the seminar/ project report may not be in a position to check for the possibility of such incidences of plagiarism in this body of work.

Signature of the Student:

Name:	Suryanarayan Sudersanam	Shon Bangale	G.V.A Sashank,	Srija Ganguly
Roll No.	B001	B010	B024	B026

Place: Mumbai

Date: 20-April-2017

CERTIFICATE

This is to certify that the project entitled "Cloud Application Portability" is the bonafide work carried out by Suryanarayan S. Shon B. G.V.A Sashank Srija G. of B.Tech (Computer Engineering), MPSTME (NMIMS), Mumbai, during the VIII semester of the academic year 2016-2017, in partial fulfillment of the requirements for the award of the Degree of Bachelors of Engineering as per the norms prescribed by NMIMS. The project work has been assessed and found to be satisfactory.

Prof. Swarnalata Bollavarapu

Internal Mentor

Examiner 1

Examiner 2

Dean

Dr. S.Y. Mhaiskar

Table of contents

List of Figures

List of Tables

Abbreviations

Abbreviation	Description
CLOUD	Communities and Libraries Online Union Database
NA	Not Available

Abstract

Cloud computing is a growing field. People have been relying upon this technology a lot more nowadays, yet some issues still remain which become hindrances to the development of this sphere. One such issue is the difficulty of portability of applications in clouds. Due to several reasons, inter-portability of applications remains an unachieved climb for researchers. With this progress, clouds will become a very successful and affordable venture. There are various needs of portability of applications including prevention of vendor lock-in, flexibility in services desired, more reliability on clouds, prevention of data loss and increasing the scope of the application in terms of usage and environment. Moreover this particular step helps in promoting dependency on technology more than on hardware. Portability is the capacity of an application to run effectively on different systems without having to change its configurations. Our motive is to achieve the same in case of applications supported by Cloud Computing.

To proceed with application portability we will be initially working on developing an application which we shall be publishing on two or more clouds. The application-"Universal converter" will have the objective to convert a file into other possible file formats. For example converting a pdf fine to a "doc" (Word document) file and vice versa. The application will online and the user will upload the document, choose possible conversions and download the file on completion of conversion. The application will be accessible to users who have signed up, and can login later with their password. The document or file could be also saved directly on other clouds like Google drive, Pivotal. Hence completely removing requirement of a of local device software.

In the action plan, we intend to develop a simple document type convertor and deploy it on a cloud platform which may be built from scratch or may be a third party provider. Our major objectives would be-

- To study operation of one single application in different types of cloud environment- ensuring that the same application built for a cloud should be compatible with other clouds too.
- To establish and study a virtual connection between Cloud and a premise(device or any other non-cloud platform)
- To study remote processor invocation (request and reply). The objective is to ensure that the data transfer is taking place correctly between the connections mentioned.

- Managing the application running in the target cloud.
- Testing the problems of hosting application and how it can be made platform- independent.

It can be assumed that we will be facing a number of challenges like-

- Application uses features that are specific to the platform, or if the platform interface is non-standard.
- Vendor agreement problems and difficulty in standardization.
- Supporting the application performance directly.
- Portability between development and operational environments.

Coding with a standard language and maintaining efficiency of

Hence by the end of project we would not only achieve cloud application portability but also develop a universal converter which will come in handy not only for in-house collage purposes but also to any student or computer user, as the type of local device or software of local device of user will not affect the application. Making every computer up to date for any format of document.

1. INTRODUCTION

What is Cloud?

The word cloud (Communities and Libraries Online Union Database) substitutes for the statement the Internet, so the phrase Cloud Computing means a type of Internet-based computing, having different aspects such as server authentication, security, storage, data-transfer and applications are delivered to an organization's computers and devices through the Internet [1].

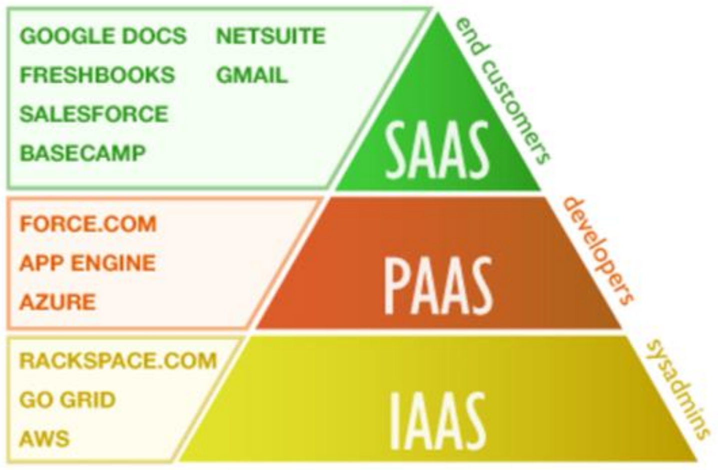


Fig. 1.1: Layers of Cloud Structure [2]

What is portability? [2]

Ability of a software to run (with little or no modification) on different hardware and/or software platforms, or work with different versions of the same hardware or program. In general, software written in Java has this ability.

What is interoperability? [3]

Interoperability is the ability of a system or a product to work with other systems or products without special effort on the part of the customer. Interoperability becomes a quality of increasing importance for information technology products as the concept that "The network is the computer" becomes a reality. For this reason, the term is widely used in product marketing descriptions.

Products achieve interoperability with other products using either or both of two approaches:

- By adhering to published interface standards
- By making use of a "broker" of services that can convert one product's interface into another product's interface "on the fly"

1.1 **Project Overview**

The project is about creating a simple application and test it with respect to implementing portability in clouds.

The application is to implement a type convertor. The application converts a particular file in one type (ex- pdf) into any other type (ex- pptx) according to the user’s choice before downloading. The conversion is supposed to happen in the application itself. The main procedure that the application have are:

- The user chooses a file to be converted.
- The application converts the file.
- The converted file is made available to the user.

The major advantage of such kind of application would be that the user wouldn’t need to download the file into his computer in order to convert its type. This would not only save storage space in the hardware but also would make the conversion procedure a single step achievement rather than a series of steps to follow. Also the technology is being prompted more in this case.

The major steps we intend to follow in the action plan are:

- Cloud creation or hiring third party services.
- Coding the application.
- Testing the application for portability depending upon suitable features.

1.2 **Hardware Requirements**

The hardware requirement is almost negligible. In order to access Clouds, the user requires:

- A fully working personal computer or mobile phone or other computing devices.
- Wi-Fi or LAN adapters/routers or broadband connections providing reliable Internet connection.

1.3 **Software Requirements**

For accessing the application, the following software should be available to the user:

- Operating System (ex: Windows, IOS, and Linux etc.).
- Client side browser (ex: Google Chrome, Mozilla Firefox etc.).

1.4 **Motivation**

The project is mainly focusing on achieving portability in case of applications supported by Cloud Computing. Portability in this sphere is a problem because of various reasons like –

- Different policies of different cloud vendors which leads to difference in the environments in which the application will run.
- Lack of compatibility due to differences in cloud infrastructure and environment. Hence differences in the source and target clouds in multiple ways.
- Geographical location of the data stores or cloud vendor which leads to differences in vendor policies again.
- Lack of a standard language or protocol that can be followed by all the clouds in order to support the same application to run.

As an example, Cog head, an online application development platform supporting the development and hosting of data-driven applications. The platform had managed to attract hundreds of developers before it suddenly announced that it would stop operating, calling all customers to export the data that was stored in their applications, but not giving them the option to port the actual applications to some other platform.

Hence we were motivated to choose this topic by the following main reasons:

- Problems of insufficient services/ features in one platform.
- Lack of fast and hassle free resource sharing.
- Innovation in a trending field and it is a lesser touched topic.
- Study of a virtual connection and efforts to know the technology in more depth.
- Social work.

The application that we intend to develop in order to test the portability of it in Cloud Computing, was thought of when we saw the cumbersome job of faculty members of having to download a type of file(say pdf) and carry out a hectic conversion of the file type to another(docx,pptx) in order to access its contents. Thus the document convertor that we intend to make will be useful to many faculty members as claimed by them.

This kind of application is already available but only for one single conversion type and the user needs to definitely have a copy of the file on his/her device while using the software to convert. The application is intended towards being a convertor for more than one major type of files into another and it won't be necessary for the user to have a downloaded copy of the file. Also the major focus would be on trying to make this application available to more than one Cloud platform with no intention of re-establishing its configuration everywhere.

The organizations with multi-cloud strategies surely make use of this idea to migrate their data resources and to collaborate or share them, but very few examples exist which have agreed to port their resources to another Cloud host because of lack of Vendor agreements. In 2014, Amazon claimed that AWS workloads can be managed by VMware users from VCenter but VMware cautioned users about the restriction of moving workloads back to one of the user's data centers or to another cloud provider highlighting the issue of portability even more[4] . Cloud Foundry is a development, scaling and deploying portal for Cloud applications that need to be ported [5].

Possible future scope or improvements would be ensuring effective data communication within the application itself and entailing the user to a direct conversion of a file found at a link. Also a major step would be to make the application platform-independent in such a way that a majority of Cloud platforms can host it under the similar standardization criteria.

1.5 **Scope of project**

Application areas:

- The application will be widely used for any official purpose where changing Document format is necessary, for eg. Doc to pdf.
- It can be widely used by faculty and students of colleges too. It will be very useful when one downloads a pdf document from blackboard (Lab manual, Syllabus, etc.) To convert to desired format for editing or saving purposes.
- Later the concept of achieved portability can be extended to the IT industry as well.

Keeping in mind the application of this project the different modules and their functionalities or features are:-

- Firstly the application will have a user account consisting of user details. For this to happen user will need login mentioned in the next point.
- Login: Validates User id/Password
- Application: Takes in the Input file, asks for Output file type. Fetches the right algorithm from ‘converters’ class.
- Input file Text: to take in file types like Pdf, doc, docx, mp3, mp4 and avi. Validate the input and download it to the application for conversion.
- Converter: Check the input file. Validate if the input file can be converted to required output file. E.g Invalid: mp4 to Pdf, Valid: Mp4 to mp3.The return the right logic and algorithm to ‘application’ class.
- Output: The required file, can be downloaded by user.

2. LITERATURE REVIEW

When it comes to literature review every group member individually has referred to a minimum of 4 research/ review papers to gain knowledge about Cloud Computing as a domain. Apart from these 16 papers another set of papers related to the project's problem statement were studied. Finally from this vast set of research papers referenced, these three review papers stood as strong pillars to not only help solve the problem statement but also gauge which web development framework and coding language to use for the project.

2.1 Towards Application Portability in Platform as a Service [6]

In this paper the writers have discussed various issues which occur while porting an application to different cloud. They have studies over 68 different cloud vendors, and derived a standardized profile with a common set of capabilities that can be found among PaaS providers and matched with one another to check application portability based on ecosystem capabilities.

A general PaaS cloud can be divided into three layers. Each layer has some parameters and functions. Which play an important role in efficient working of cloud. Understanding and replicating these properties in the target cloud is necessary in order to achieve application portability.

2.1.1 Infrastructure layer –

- This layer deals with the hardware part of cloud like CPU, RAM and Storage units. To achieve portability of application it is necessary to ensure that the hardware specifications of target cloud match to that of the source cloud.
- Another important factor is the geographical region the application will be deployed in. This is particularly interesting because of legal and performance reasons. As bandwidth capacities keep increasing on the customer's end, latency is one of the main constraining factors for publicly hosted applications

2.1.2 Platform layer –

- The platform is the main deliverable of a PaaS offering and includes the application hosting environment delivered as a service. It consists of two stacks, both stacks can be combined by the customers via bindings. Those bindings are generally environment variables that include important properties of the services like endpoint URLs, credentials, and other configuration information.
  - *The runtime stack* includes the basic runtimes offered by the PaaS, i.e. the programming languages that applications can be written in. Furthermore, we see the vast popularity of language-specific frameworks like Ruby on Rails which are leveraged to develop today's applications.
  - *The service stack* includes mostly latency and performance critical core services like data stores. Add-on services are supplied by third-party service providers that integrate with the PaaS.

2.1.3 Management layer –

- It allows control over the deployed applications and the configuration settings of the platform. The management layer includes the abilities to deploy and manage the lifecycle of the applications. This encompasses pushing, starting, and stopping of applications. Moreover, the provisioning of all native services and add-ons is initiated from the management tier.

From the above we can say that managing several layers and the possible manifestations of the inherited components and capabilities is complex, thus portability between PaaS is a difficult task. Nearly every vendor has its particular management API, platform configuration and restrictions, resulting in a strong dependency to a certain provider and significant costs when migrating from one vendor to another

In order to make PaaS offerings comparable and match able the writer of this paper grouped a set of core properties of their business and ecosystem perspectives into a standardized machine-readable PaaS profile.

A PaaS profile contains the following information:

- *Meta Information* are stored in order to keep track of the profile itself, whether it is updated or not, is it vendor verified, when was the profile last verified.
- *Business Properties* include the official name of the PaaS offering, URL to the PaaS webpage, stage of lifecycle like ‘Beta’ or ‘Alpha’, pricing model and other related information (refer to paper)
- *Ecosystem Properties* contains information regarding horizontal & vertical scaling, hosting properties, information regarding infrastructure, runtime, middleware, frameworks and services, languages supported, version of the system etc.

They created profiles for over 68 PaaS offerings by various vendors with their core properties and functionalities. In order to access and evaluate these profiles they developed a Web Application through which one can check various characteristics of a PaaS offering and compare them with other PaaS offerings. This will help in identifying how the source and target cloud are similar / different, which framework / language the source and target cloud support, it will also help the user identify the target cloud which best suits the needs.

But the main issue with this paper is that it doesn’t focus / consider low level portability issues like standardization of the management interface, management of API differences etc... The researches of this paper stated that they are conducting studies on the above issue and are trying to come up with a solution soon.

2.2 Automatic cloud vendor API analysis for supporting cloud application portability [7]

Cloud is an emerging field in today's world and it is becoming important by the day as new providers keep arising every day. It is very difficult to find a firm that can provide all the services under one roof. The issue that arises are invoking and describing services to the specified requirements and to communicate.

Cloud application portability a concept as explained earlier is one which refers to the ability to move applications between cloud vendors with minimum level of integration issues.

This paper offers an API mapping approach commonly known as API alignment technique. This technique shines light on the concept that to provide support to cloud application portability, is to understand functionalities offered by APIs and map them with other provider APIs. Alignment and porting of the application is done through the extraction of knowledge from APIs (i.e. their functionalities).

To put all of this under one roof one must have good knowledge on two things, they are:-

- Reverse engineering- It is taking apart an object to see how it works so that duplication or enhancement of the object is possible.
- Program comprehension and its stages/directions- It is a domain of computer science concerned with the ways software engineers maintain existing source code. The cognitive and other processes involved are identified and studied. The results are used to develop tools and training.

API alignment technique

Points that need to be looked into for this technique are:-

- Mapping of APIs with APIs of other providers.
- Combination of API, UML design, manuals and documentation are used to extract information on the functionalities, which are then ported along with semantic annotations.
- Analysis of the workflow, the architecture and prototypical implementation.

This technique has three ways in which it can be implemented, they are:-

1. API alignment with neutralization

   Terms to be known in this method-

   - Neutral API- It also called as the leader API which contains all the functionalities without the architectural details.
   - Graph- Assume that an API for a given domain is available which can be semantically described using a graph.

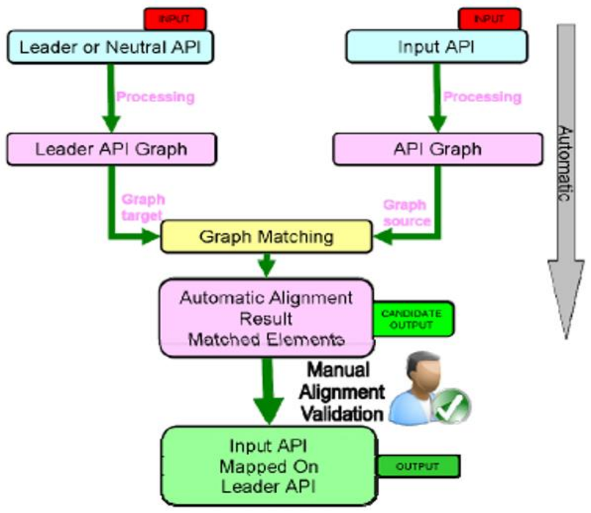


Fig. 2.2.1: API Alignment

Process of alignment-

- The input and the leader API are converted to their respective graphs which need to be matched.
- Then automatic matching and element result alignment is carried out which is also called as the candidate output.
- Now that the API alignment is done, a manual alignment validation is carried to verify the automatic process and then the output is achieved.

It is to be noted that here mapping of functionalities of the services provided by cloud vendors' takes place which in turn makes inter portability easy.

2. API alignment using semantic approach

Terms to be known in this method are the ones learnt in the previous method as well as the following-

- Ontology- It is an explicit specification of conceptualization in which every node is a concept in itself.
- OWL- Web Ontology Language is a semantic mark-up language for sharing and publishing ontologies on the World Wide Web. OWL was developed as a vocabulary extension of RDF.
- Annotated API-APIs that are self-explanatory and are more descriptive because of their add-on functionalities.

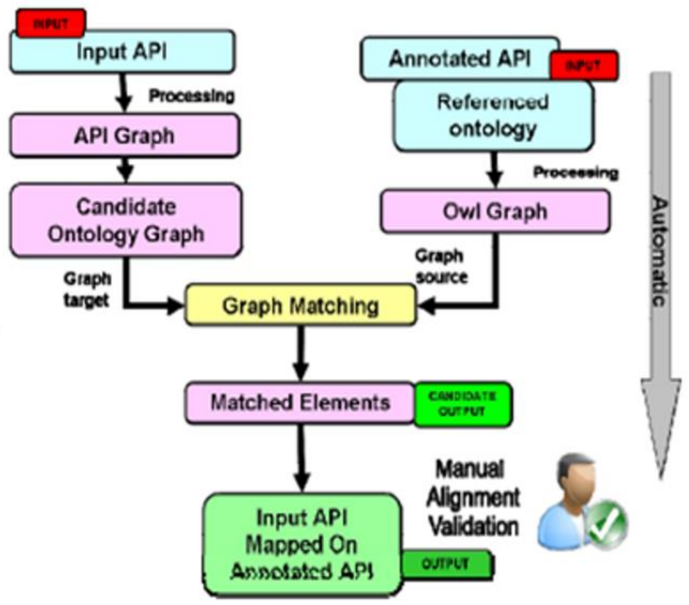


Fig. 2.2.2: API Automatic Alignment.

Process of alignment-

- The input and the annotated API are converted to their respective graphs which need to be matched. The input API is converted to the API graph which then is converted to the candidate ontological graph as it has to be mapped with the annotated API’s graph and the annotated API with the help of a referenced ontology is converted into an OWL graph.
- Then automatic graph matching and element result alignment is carried out which gives the candidate output.

- Now that the API alignment is done, a manual alignment validation is carried to verify the automatic process and then the output is achieved.

There are general things to be noted here that this method is better than the 1st method as here all the semantics of both the functionalities are mapped which increases the mapping efficiency.

3. API alignment using semantic annotations

   Terms to be known in this method are the ones learnt in the previous method as well as the following-

   - Base Ontology- This ontology is declared in this document both in human-readable form (what you see in front of you now) and machine-readable simple HTML ontology extension i.e. SHOE form (which you can see from viewing the html source of this document).

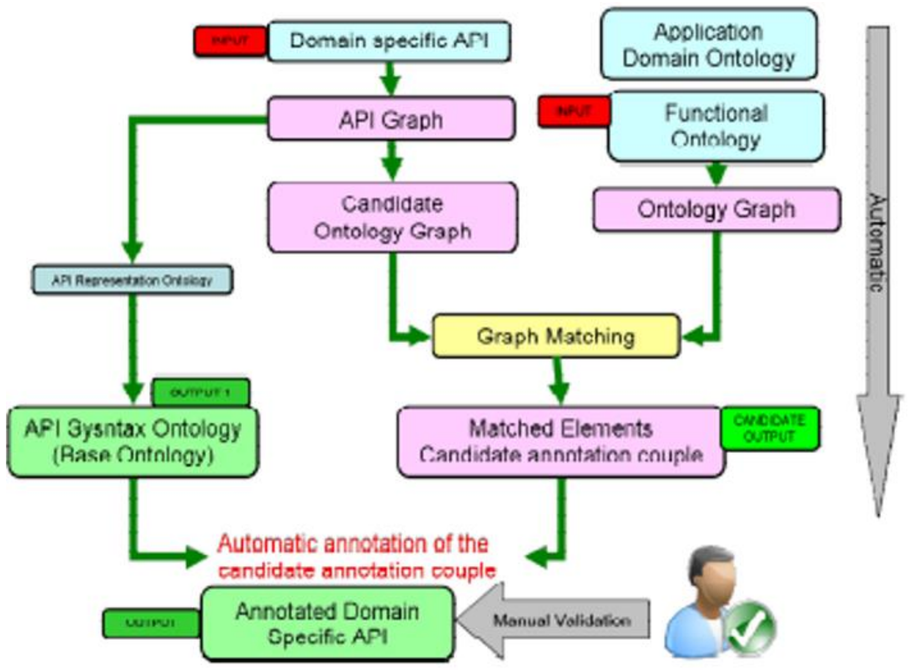

INPUT
Domain specific API
Application Domain Ontology
API Graph
INPUT
Functional Ontology
Candidate Ontology Graph
Ontology Graph
Automatic
API Representation Ontology
Graph Matching
OUTPUT 1
API Sysntax Ontology (Base Ontology)
Matched Elements Candidate annotation couple
CANDIDATE OUTPUT
Automatic annotation of the candidate annotation couple
OUTPUT
Annotated Domain Specific API
Manual Validation

Fig. 2.2.3: API automatic semantic annotation.

Process of alignment-

- The domain specific API and the application domain ontology are converted to their respective graphs which need to be matched. The inputted domain specific API are converted to API graphs and the application domain ontology along with an external input (functional ontology) is converted to the ontology graph.
- The API graph is then converted to a candidate graph as well as into an API syntax ontology (base ontology) with the help of an ontological API representation. The candidate and the ontology graphs are matched and then automatic element result alignment is carried out which gives the candidate output.
- The matched elements or the candidate output undergoes automatic annotation along with the API base ontology to form annotated domain specific API
- Now that the domain specific API alignment is done, a manual alignment validation is carried to verify the automatic process and then the output is achieved.

In this approach it is imperative to note that the matched and aligned API that we receive is domain specific and is more accurate than both the previous methods.

2.3 Cloud Application Portability: An Initial View [8]

One of challenge in Cloud Computing is application portability between two or more cloud environments. The application developed in this paper was ported on four different platform Open Shift, Google App Engine, Heroku, and Amazon Elastic Beanstalk.

PaaS, a key benefit is that users can develop and deploy applications without the burden of setting up and maintaining the necessary programming environment and infrastructure that the application is executed on.

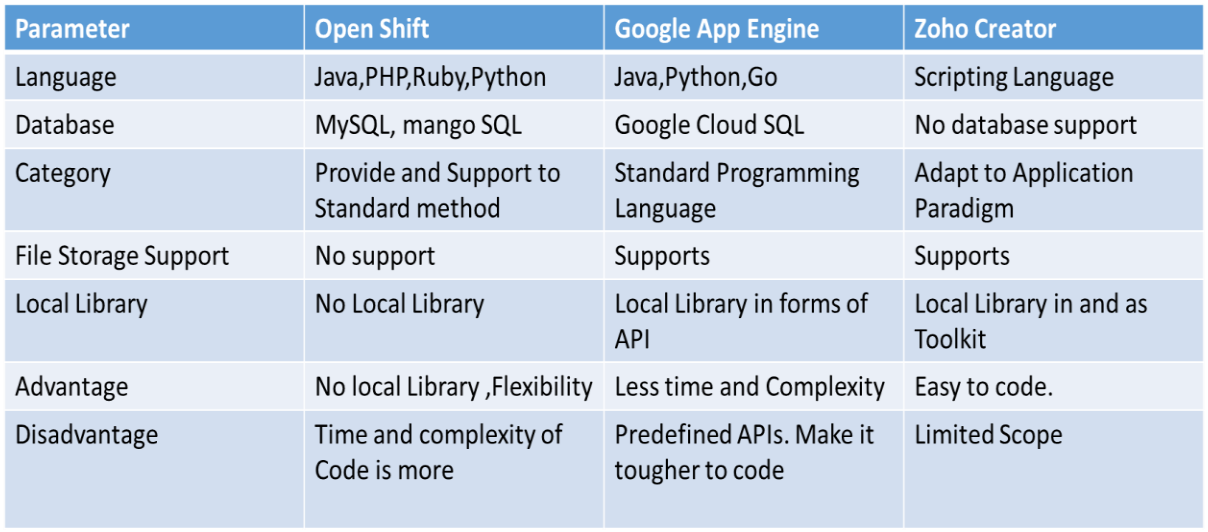
Different cloud application platform offerings are characterized by considerable heterogeneity. Because of incompatibilities, users that develop applications on a specific platform may encounter significant problems when trying to deploy their application in a different environment. This gives rise to the familiar problem of vendor lock-in. Consumers need to be able to easily change between cloud providers and should be free to choose the one that better serves their needs in terms of quality and/or cost

A real example to illustrate this argument is the case of Cog-head, an online application development platform supporting the development and hosting of data-driven applications. The platform had managed to attract hundreds of developers before it suddenly announced that it would stop operating, calling all customers to export the data that was stored in their applications, but not giving them the option to port the actual applications to some other platform.

2.3.1 Cloud platforms:

This section talks about different cloud developing platforms, this will also help to understand what type or what cloud platform one should use for portability.

Table 2.3.1: Comparison in different application development platform



Parameter	Open Shift	Google App Engine	Zoho Creator
Language	Java,PHP,Ruby,Python	Java,Python,Go	Scripting Language
Database	MySQL, mango SQL	Google Cloud SQL	No database support
Category	Provide and Support to Standard method	Standard Programming Language	Adapt to Application Paradigm
File Storage Support	No support	Supports	Supports
Local Library	No Local Library	Local Library in forms of API	Local Library in and as Toolkit
Advantage	No local Library ,Flexibility	Less time and Complexity	Easy to code.
Disadvantage	Time and complexity of Code is more	Predefined APIs. Make it tougher to code	Limited Scope

Portability issues in cloud applications

1. Programming languages and/or frameworks the specific programming languages and frameworks that an application has been built with is obviously a major determinant for cross-platform deployment. Each cloud platform supports certain languages, frameworks, and versions thereof.
2. Platform specific services, an important characteristic of several cloud platforms is that they provide certain services via specific APIs. A service can be considered as high-level functionality that the provider can use without the need to implement it from scratch.

A portability issue arises when the application needs to be ported to a different cloud platform. There are two cases:

a. The target platform doesn't provide the full set of services that the application uses. For example, SMS services and monitoring services are not supported. In this case the developer would need to recreate the missing functionality from scratch on the new target platform.
b. The target platform supports the services that the application uses but provides different APIs in order to use them. In this case the developer would need to modify the application code and align it with the APIs of the new target platform.

3. Data Storage: Data storage is an essential part of an application. There are two types of data storage:

Database stores: used for storing structured data while the second one could be perceived as an analogy to a hard disc drive on the cloud. Almost every modern cloud application needs to access data from a database Different platforms often support different types of database. Therefore the following conflicts may arise when trying to port an application across various platforms:

1) Incompatible data structures: As it became clear there is a wide range of available databases where each one of them adopts a different data structure. Portability issues are bound to arise when trying to move data from a SQL to a NoSQL database, but also between different types of NoSQL databases, e.g. when moving from a key-value store to a document store.

2) Different query languages: Apart from the incompatibility due to different data structures, conflicts may occur at the way of querying data. Databases use
Their own APIs or query languages. Therefore even when data is moved across databases of the same category (e.g. a document store), portability issues may arise concerning the way the database is accessed.

3) Data migration (export/import formats): Another issue that should be considered is data migration. It may happen that an exported database cannot directly be imported to another database engine due to incompatible data formats.

- Use of graphical interface. Human users of the cloud application can manually perform operations on the file storage space.

d) Platform specific configuration files Similar to the configuration files that traditional software applications require in order to instruct the hosting environment on how to execute the applications, cloud platforms may require analogous configuration files. For example Google App Engine uses the “appengine-web.xml” file. The process of adapting the configuration files to each target cloud platform adds to the overall overhead of cross-platform deployment of a cloud application.

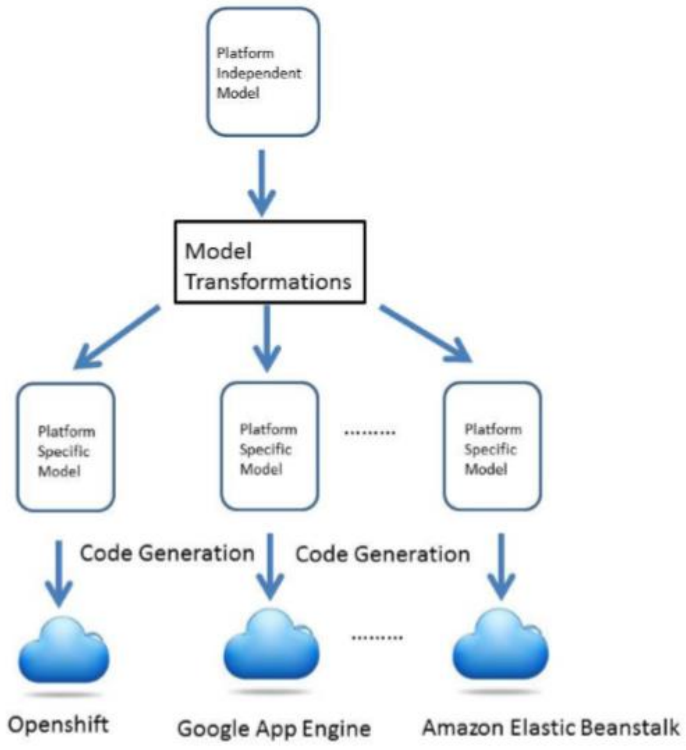


Fig 2.3.1: Intermediation.

2.3.2 Conclusion of the paper

High degree of heterogeneity between cloud application platforms, any approach to tackle application portability needs to target a specific set/class of platforms that present similar characteristics.

Problems: such as programming languages and frameworks, database offerings, file storage service, platform specific services and configuration files.

We presented related work aimed at addressing the challenge of cloud application portability and distinguished between two generic approaches to tackle the issue of application portability: standardization and intermediation. Standardization addresses cross-platform portability through the adoption of common standards by cloud providers. Alternatively intermediation enables developers to create applications independently of a specific platform and then bind them to particular target platforms through some form of automatic translation.

2.4 Review of Java frameworks

A lot of java based web development frameworks were considered for the project. Some of them being the following:-

- Spring MVC framework
- Apache Struts 1
- Grails
- Aura
- React

The following table has is the measure of which framework suits the project best, leaving aside the cost factor.

Table: 2.4.1: Comparison of Frameworks

Project	Language	Ajax	MVC-framework	MVC push-ull	Testing framework(s)	Security framework(s)
Spring	Java	Yes	Yes	Push	Mock objects, unit tests	Spring Security(formerly Acegi)
Aura	JavaScript	No	No only interface to model	-	Unit tests	-

From this list of frameworks, two promising frameworks were chosen for detailed examination. They were:-

- Spring MVC framework
- Aura framework

Based on this matrix it is observed that Spring MVC supports java, AJAX (which is an ad-on) and has a set testing framework of its own. It is even secure and has multiple MVC modules. Aura falls short in all these aspects so Spring MVC framework becomes the evident porting development framework.

2.5 Review of Coding Languages

Based on which languages seemed learnable and executable in the limited time frame two programing languages were considered for detailed analysis. They were:-

- Python
- Java

2.5.1 Python Features[9]

- Easy-to-learn: Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- Easy-to-read: Python code is more clearly defined and visible to the eyes.
- Easy-to-maintain: Python's source code is fairly easy-to-maintain.
- A broad standard library: Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- Interactive Mode: Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- Portable: Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- Extendable: You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- Databases: Python provides interfaces to all major commercial databases.
- GUI Programming: Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of UNIX.
- Scalable: Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below:

- IT supports functional and structured programming methods as well as OOP.

- It can be used as a scripting language or can be compiled to byte-code for building large applications.

2.5.2 Java Features[10]

- Object Oriented: In Java, everything is an Object. Java can be easily extended since it is based on the Object model.
- Platform independent: Unlike many other programming languages including C and C++, when Java is compiled, it is not compiled into platform specific machine, rather into platform independent byte code. This byte code is distributed over the web and interpreted by virtual Machine (JVM) on whichever platform it is being run.
- Simple: Java is designed to be easy to learn. If you understand the basic concept of OOP Java would be easy to master.
- Secure: With Java's secure feature it enables to develop virus-free, tamper-free systems. Authentication techniques are based on public-key encryption.
- Architectural-neutral: Java compiler generates an architecture-neutral object file format which makes the compiled code to be executable on many processors, with the presence of Java runtime system.
- Portable: Being architectural-neutral and having no implementation dependent aspects of the specification makes Java portable. Compiler in Java is written in ANSI C with a clean portability boundary which is a POSIX subset.
- Robust: Java makes an effort to eliminate error prone situations by emphasizing mainly on compile time error checking and runtime checking.
- Multithreaded: With Java's multithreaded feature it is possible to write programs that can do many tasks simultaneously. This design feature allows developers to construct smoothly running interactive applications.
- Interpreted: Java byte code is translated on the fly to native machine instructions and is not stored anywhere. The development process is more rapid and analytical since the linking is an incremental and light weight process.
- High Performance: With the use of Just-In-Time compilers, Java enables high performance.
- Distributed: Java is designed for the distributed environment of the internet.

2.5.3 Conclusion

On careful examination and analysis of the different features offered by the two programming languages, Java was selected to continue with the project for the following reasons:-

- It is easily compatible with Spring MVC.
- We had better exposure of working in Java rather than Python.

3. ANALYSIS AND DESIGN

This aspect consists of the different phases involved in understanding what the project might require and how to get what is required for the project. Basically one can compare this aspect with the requirements gathering and planning phase in the software development life cycle.

3.1. Application Creation

3.1.1 File Converter

The application of File Convertor will have some basic features and functions that have been together explained through a Control Flow Diagram.

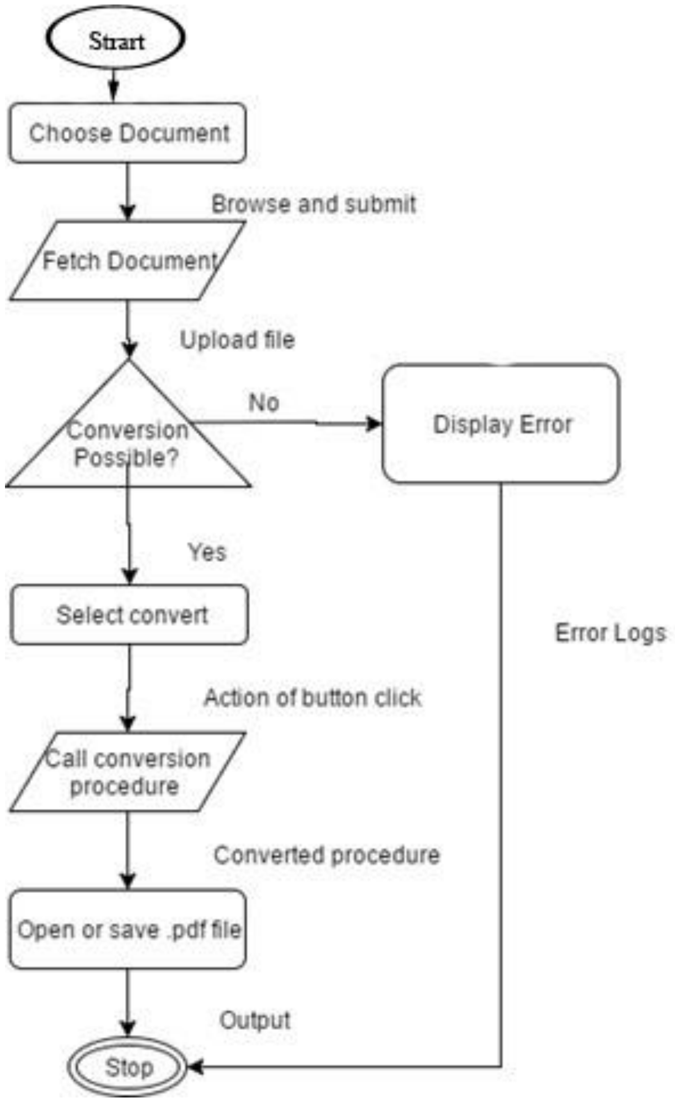


Fig 3.1.1: Application CFD File Converter.

Description of the Figure:

It suggests a simple control logic of the application:

- Step 1- User enters document URL.
- Step 2- The document is fetched. If there is an error then it is displayed and the current process stops.
- Step 3- After fetch action, the user is prompted to choose a conversion choice.
- Step 4- If the conversion is possible for the current document, the correct conversion procedure is called and if the conversion is not possible then the procedure stops.
- Step 5- The converted file is displayed if conversion takes place.
- Step 6- The file can be saved or be used in any other way the user desires.

3.1.2 Metric Converter

The application of Metric Convertor will have some basic features and functions that have been together explained through a Control Flow Diagram.

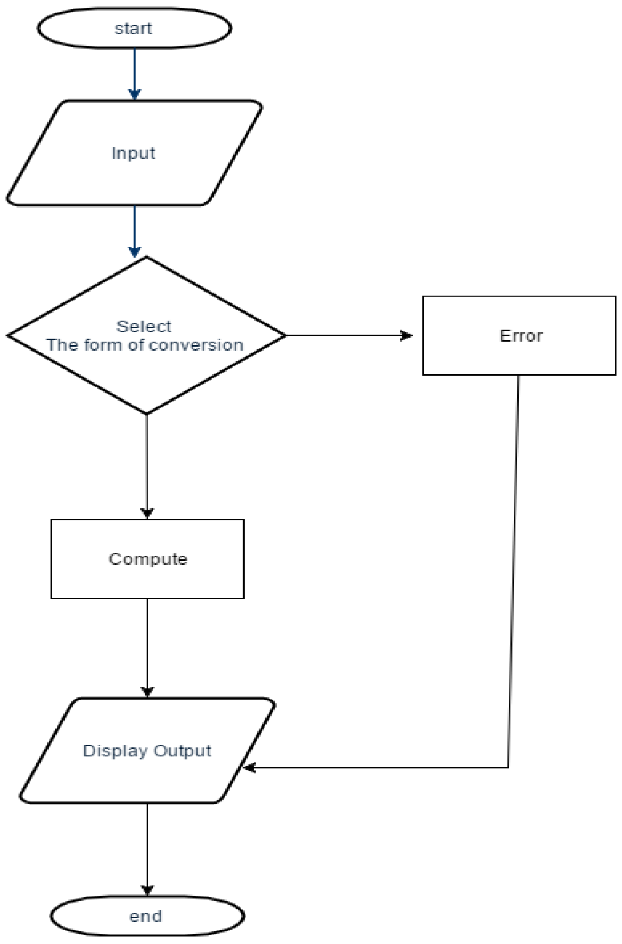


Fig 3.1.2: Application CFD Metric Converter.

Description of the Figure:

It suggests a simple control logic of the application:

- Step 1 – The user Enters the input Value
- Step 2- User Selects the required conversion
- Step 3- Application checks if conversion exists.
- Step 4- Compute the conversion.
- Step 5- Display output.

3.2 Application Class Diagram

The Class diagram or the object oriented diagram of an application provides a small look at how the modules are connected to each other. This basically shows the components and their connectors leading to an architecture view of the application.

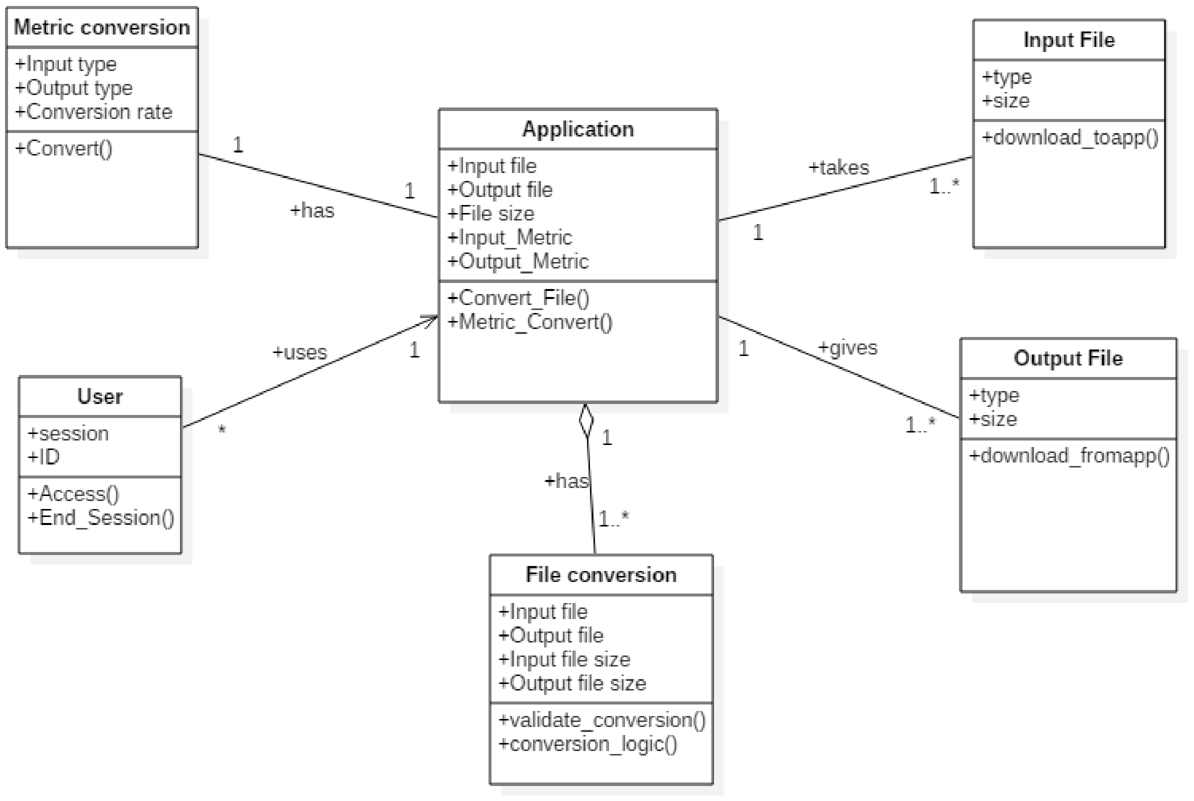


Fig 3.2.3: Class Diagram of Application.

Description of Figure – The class diagram can be explained according to the main modules selected.

The application module- It is there to act as the main class that takes in a file input and eventually provides a file output. It also checks the file size.

Metric Converter module- This deals with the Metric conversions. If user wants to execute metric conversion. He shall input a value, the application verifies if conversion is possible, compute and display.

User module- It is the one that takes care of the user ID and is the sub module of the login module.

Input file module- This is used to upload a file to the application by the user. This also checks the type and size of file.

Input file audio module- This is used to upload an audio as an input. Its type and size is also checked.

Output file module- This module is used to display a file that is processed by the convertor.

Convertors module- It is the main logic behind the application. The convertor checks the input file type, size and calls the conversion logic according to the choice of the user after it is validated.

3.3. Project Action Plan

The project as a whole is one step more than the creation of the application that is the application needs to be deployed in more than one Cloud platform to see if it is accessed from both the Clouds in a standardized way.

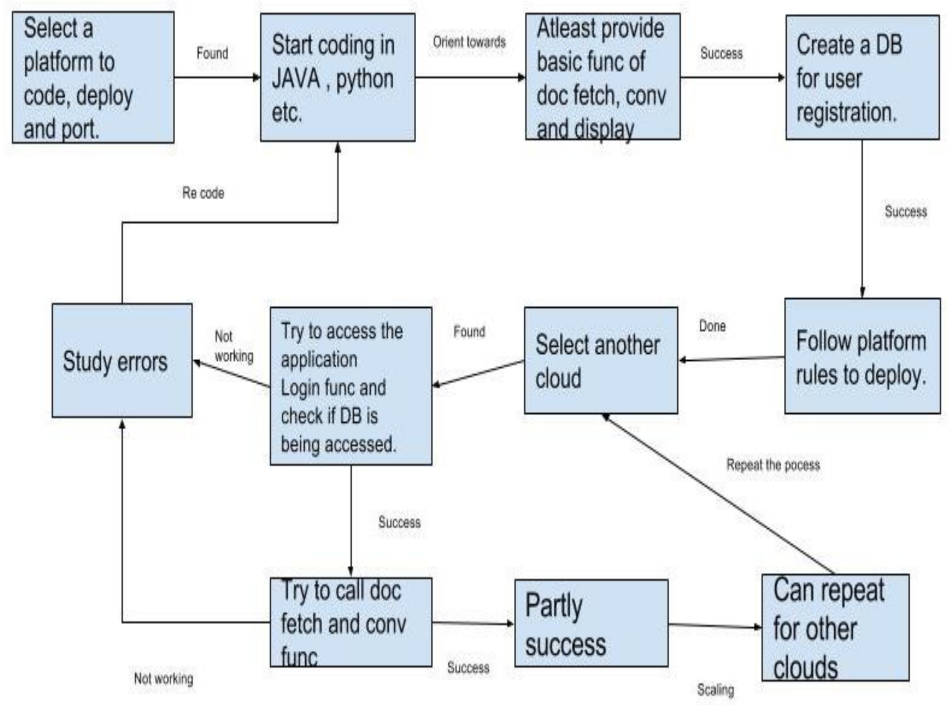

Select a platform to code, deploy and port.
Found
Start coding in JAVA , python etc.
Orient towards
Atleast provide basic func of doc fetch, conv and display
Success
Create a DB for user registration.
Re code
Success
Study errors
Not working
Try to access the application Login func and check if DB is being accessed.
Found
Select another cloud
Done
Follow platform rules to deploy.
Repeat the pocess
Success
Try to call doc fetch and conv func
Partly success
Can repeat for other clouds
Not working
Success
Scaling

Fig 3.3.1: Action plan for project.

Description of Figure- The project plan can be highlighted through this image and the corresponding blocks. Each block represents an action to be taken for completing the project in a whole.

- Selecting a Platform to code the project (i.e. Java) that makes it easy for deploying to Cloud. The framework (i.e. Spring MVC) has already been chosen by extensive study and trials to port the application.
- Start coding in Java and PHP.
- Provide basic functions of fetching a document, conversion and display that will be refer to as a working application.

- Creating a Database and its validation techniques to register users and keep track of them.
- Follow the specific rules of the platform and framework that are used to deploy the working application to the Cloud.
- The next block is assumed when all the previous ones are a success. Another Cloud platform is selected to check portability of application.
- To check if the basic functions like accessing the database and login are being able to be accessed from the other Cloud platform.
- If not working, then errors are to be studied. The study can lead us to change certain code in the application that achieves more standardization across several Cloud platforms.
- If the basic functions of user login are working from the next Cloud platform selected, then the application checks if document fetching, conversion and displaying are working.
- If that’s not working, then again errors are studied. Changes are made according to results gathered.
- If the basic functions are working, then the project is a partly successful project that enables an application to be ported from one Cloud platform to another Cloud platform.
- The process is repeated for another Cloud platform too.

4. METHODS IMPLEMENTED

To implement the idea presented above, an extensive search for the appropriate framework was conducted. It was observed that Spring MVC frame was the most suitable for the application that we are developing.

The Spring web MVC framework provides model-view-controller architecture and ready components that can be used to develop flexible and loosely coupled web applications. The MVC pattern results in separating the different aspects of the application (input logic, business logic, and UI logic)[11]

Using spring MVC following has been implemented:

4.1 User interface in spring

Spring MVC provides support of HTML, CSS, Bootstrap and various other interface extensions, with the help of which an interactive but universally portable view can be made, since these are not dependent on any system specifications.

4.2 Uploading and downloading Files from web-server

One of the most important features of the application is Conversion of file types, for that to happen we have developed a system through which the user can upload the files he wishes to convert on to the web server and download the file from web server.

The file is uploaded to the web server, and can be downloaded back by clicking the download button.

- In this application we have implemented file validation methods, which checks if the file has been successfully uploaded or not.
- The special java model control the flow of files through the bucket list, and is responsible for giving access to upload and download options.

4.3 File Conversion

Conversion of the file takes place using Java language and iText API.

iText API[12]:

iText 7 Core is the bedrock of iText 7. It lets you assemble, expand, extract, split and interact with any PDF file. It is compliant with most ISO PDF standards as well as standards for digital signatures (PAdES) it allows developer to spend user time more productively by automating routine documentation and archiving tasks. User documents can be read by any machine, at any time, no matter the original input formats, and will stay legible for a very long time. iText 7 Core further serves as the platform that can be expanded with add-ons for specific value-added PDF tasks. Look below for the add-ons that are available - more will be added as iText 7 continues to grow and evolve.

iText 7 Community is the same as iText 7 Core in terms of functionalities, but can be used only with the AGPL license, meaning users must disclose their own source code in iText projects and must disclose any modifications they make. In addition, the iText 7 add-ons are not available for the Community version, and the AGPL license does not come with support from the development teams.

4.4 Save/Download

The file is stored in file buffer where it will open up in system viewer. The system viewer then has the functionality to store the document.

4.5 Metric Conversion

The user inputted value will be converted to expected metric value, currently the application can perform the following conversions:

Centimeter to Inch
Inch to Centimeter
Feet to Centimeter
Meter to Centimeter
Meter to Feet
Kilo-meter to Miles

Application

Diagram as per Methods:

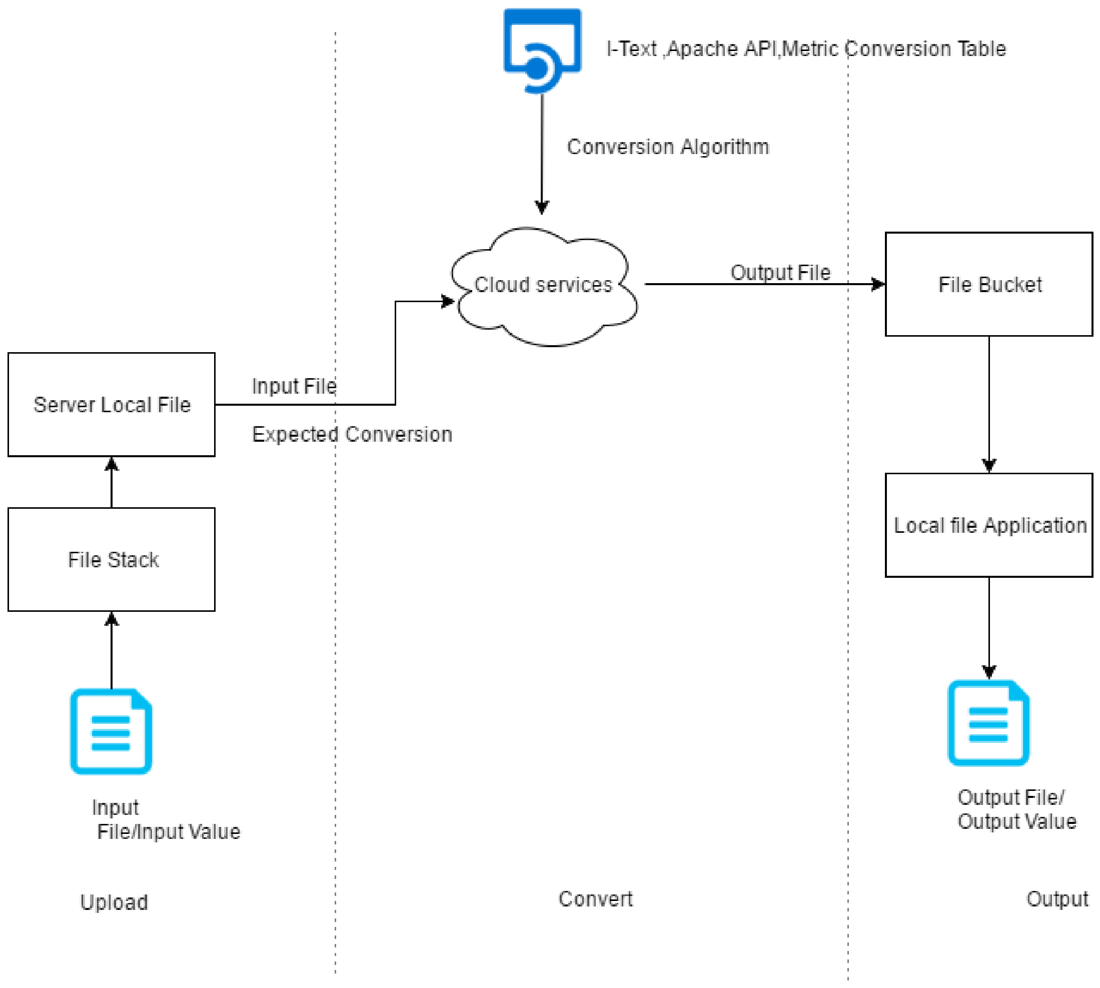


Fig 4.1: Application as per Methods

Figure 4.1 is a pictorial representation that the application incorporates. Including Upload, Input, Download, Display, File Conversion API, Metric conversion Table, and the Cloud service where the application shall be deployed.

5. Testing

5.1 INTRODUCTION

Application Portability Converter is an application to convert file of one form to other, for example JPEG to PDF.

Main functionality:

Upload: To validate file from local file or web URL and store in server file.

Convert: Convert file from one format to another.

Save/Open: Save or open file in expected format.

5.2 OBJECTIVES AND TASKS

The main objective of testing modules are:

- Comparison of performance
- Debug application

5.3 SCOPE

- General
  - Testing of Different components
  - Usability of Software
  - Testing Algorithm Complexity

- Tactics
  - Testing of Different components
    - Check Upload, Convert, Download components
  - Usability of Software
    - Interface testing, scalabilty , availblity
  - Testing Algorithm Complexity
    - Based on Function point,LOC.

- TESTING STRATEGY
    - Upload, Download :
- Bulk Testing
- Beta Testing
- Interface usability
    - Conversion:
- Unit Testing
- Functionality testing
- Integration testing

5.4 DEPENDENCIES

Cloud Server: Availability depends on Cloud and internet provider in availability of any may cause fail in testing and system failure,

5.5 RISKS/ASSUMPTIONS

- Cloud server in-availability
- Denial of Service
- Lack of time for testing
- Undetermined number of user.

5.6 APPROVALS

- Application working on two or more cloud with same performance.

5.7 Testing Results

5.7.1 Local Server

Table: 5.7.1.1: DESIGNING TEST CASES – MODULE 1

Test Case Template	
Test Case ID: PR_1	Test Designed By: Surya S.
Test Priority : High	Test Designed Date: 31/03/2017
Test Title: Verification of file Upload	Test Executed By: GVA. Sashank
Description: The main objective of the test the effect on speed to upload by different file size or quality(RESOLUTION)	Test Execution Date: 01/04/2017
Preconditions: File Upload verification and type validation.	

Steps	Test Steps	Test Data	Expected Result	Actual Result	Status	Notes
1	Entering File Path	The File path or link should be explicit Without any general flaws	Successful acceptance of File address and fetch exact link	Link fetched	True	Check for web URL too
2	Upload file	File link fetched previously is now used to access file into file bucket and upload in temporary Location	File successfully uploaded	File successfully uploaded	True	File uploaded successfully, But name of file changes to default value.
3	Clicking on Submit button	File and conversion algorithms	Show all possible conversion for particular file.	Fail to validate file type. For desired function	False	File should validate file type of file for particular conversion.

Table 5.7.1.2: DESIGNING TEST CASES – MODULE 2

Test Case Template	
Test Case ID: PR_2	Test Designed By: Shon B.
Test Priority : High	Test Designed Date: 31/03/2017
Test Title: Validation of Conversion	Test Executed By: Srija G.
Description: The main objective of the test is quality of PDF to conversion for different resolution values	Test Execution Date: 01/04/2017
Preconditions: The user-ID and password of the user’s should be same and match .Also it is Case sensitive.	

Steps	Test Steps	Test Data	Expected Result	Actual Result	Status	Notes
1	Input file from File bucket and fetch from server location.	Input File for conversion	Successful File input into algorithm buffer.	Successful inputted	True	-
2	Implement conversion algorithm	File (INPUT FORMAT)	Successful conversion to expected file format	Successful conversion to expected file format	True	File converted to Expected output. (JPEG to PDF)
3	Clicking on Save or download file	File(OUTPUT FORMAT)	File open or download	File downloaded	True	Default location to download.

Table 5.7.1.3: Metric Conversion

Test Case Template						
Test Case ID: PR_3			Test Designed By: Surya S.			
Test Priority : High			Test Designed Date: 31/03/2017			
Test Title: Validation of Metric Conversion			Test Executed By: Shon B.			
Description: The main objective of the test is cheek correctness of metric conversion			Test Execution Date: 01/04/2017			
Preconditions: The user-ID and password of the user's should be same and match .Also it is Case sensitive.						
Steps	Test Steps	Test Data	Expected Result	Actual Result	Status	Notes
1	Input Value	Input Value for conversion	Successful Loading and initialization of variable.	Successful inputted	True	-
2	Implement conversion algorithm	Value (INPUT FORMAT)	Successful conversion to expected Metrics, Display error if conversion is not possible	Successful conversion to expected output value	True	-
3	Display	Value(OUTPUT Metrics)	Display correct value	Value displayed	True	-

Execution of the applications and Comparison of quality of the software in terms of:

A. memory management and time taken

Table 5.7.1.4: Test for Time to Upload

File Size(UPLOAD)(MB)	Time to upload(seconds)
0.5	0.1
2.5	1.2
5.0	2.5

Table 5.7.1.5: Test for Time to Convert

File Size(UPLOADED)	Time to Convert(Seconds)
0.5	1.3
2.5	1.6
5.0	2.2

Table 5.7.1.6: Test for Time to Download

File Size(UPLOAD)	Time to download
0.5	0.2
2.5	0.8
5.0	1.9

B. Accuracy

Table 5.7.1.7: Test for Quality of Uploaded file

File Size(UPLOAD)	Quality of input file(UPLOADED)
0.5	Same
2.5	Same
5.0	Same

Table 5.7.1.8: Test for Quality of Converted file

File Size(UPLOADED)	Post Conversion File(Distorted)
0.5	Same
2.5	Medium
5.0	Low

5.7.2 Pivotal Cloud Foundry/Heroku Platform

Table: 5.7.2.1: DESIGNING TEST CASES – MODULE 1

Test Case Template						
Test Case ID: PCF_1			Test Designed By: Srija G.			
Test Priority : High			Test Designed Date: 02/04/2017			
Test Title: Verification of file Upload			Test Executed By: GVA. Sashank			
Description: The main objective of the test the effect on speed to upload by different file size or quality(RESOLUTION)			Test Execution Date: 02/04/2017			
Preconditions: File Upload verification and type validation.						
Steps	Test Steps	Test Data	Expected Result	Actual Result	Status	Notes
1	Entering File Path	The File path or link should be explicit Without any general flaws	Successful acceptance of File address and fetch exact link	Link fetched	True	Check for web URL too
2	Upload file	File link fetched previously is now used to access file into file bucket and upload in temporary Location	File successfully uploaded	File upload unsuccessful	False	File upload didn’t work on local file system.

Table 5.7.2.2: DESIGNING TEST CASES – MODULE 2

Test Case Template	
Test Case ID: PCF_2	Test Designed By: Shon B.
Test Priority : High	Test Designed Date: 02/04/2017
Test Title: Validation of Conversion	Test Executed By: Surya S.
Description: The main objective of the test is quality of PDF to conversion for different resolution values	Test Execution Date: 02/04/2017
Preconditions: The user-ID and password of the user's should be same and match .Also it is Case sensitive.	

Steps	Test Steps	Test Data	Expected Result	Actual Result	Status	Notes
1	Input file from File bucket and fetch from server location.	Input File for conversion	Successful File input into algorithm buffer.	Input unsuccessful	False	File system path is not supported.
2	Implement conversion algorithm	File (INPUT FORMAT)	Successful conversion to expected file format	Unsuccessful conversion to expected file format	False	File didn't convert as it couldn't be fetched.
3	Clicking on Save or download file	File(OUTPUT FORMAT)	File open or download	File didn't download	False	Default location to download is not accepted by Cloud.

Table 5.7.2.3: Metric Conversion

Test Case Template	
Test Case ID: PCF_3	Test Designed By: GVA. Sashank
Test Priority : High	Test Designed Date: 06/04/2017
Test Title: Validation of Metric Conversion	Test Executed By: Surya S.
Description: The main objective of the test is cheek correctness of metric conversion	Test Execution Date: 06/04/2017
Preconditions: The user-ID and password of the user's should be same and match .Also it is Case sensitive.	

Steps	Test Steps	Test Data	Expected Result	Actual Result	Status	Notes
1	Input Value	Input Value for conversion	Successful Loading and initialization of variable.	Successful inputted	True	-
2	Implement conversion algorithm	Value (INPUT FORMAT)	Successful conversion to expected Metrics, Display error if conversion is not possible	Successful conversion to expected output value	True	No need for files or database.
3	Display	Value(OUTPUT Metrics)	Display correct value	Value displayed	True	Flawless run.

Execution of the applications and Comparison of quality of the software in terms of:

A. Memory management and time taken

Table 5.7.2.4: Test for Time to Upload

File Size(UPLOAD)(MB)	Time to upload(seconds)
0.5	Fail
2.5	Fail
5.0	Fail

Table 5.7.2.5: Test for Time to Convert

File Size(UPLOADED)	Time to Convert(Seconds)
0.5	NA
2.5	NA
5.0	NA

Table 5.7.2.6: Test for Time to Download

File Size(UPLOAD)	Time to download
0.5	NA
2.5	NA
5.0	NA

(NA: Not Available due to inability in execution)

B. Accuracy

Table 5.7.2.7: Test for Quality of Uploaded file

File Size(UPLOAD)	Quality of input file(UPLOADED)
0.5	NA
2.5	NA
5.0	NA

Table 5.7.2.8: Test for Quality of Converted file

File Size(UPLOADED)	Post Conversion File(Distorted)
0.5	NA
2.5	NA
5.0	NA

(NA: Not Available due to inability in execution)

6. RESULT AND DISCUSSIONS

The work progress that has been recorded is result of the weekly discussions with our mentor. Various scenario or concept of Cloud Application Portability as discussed as a problem. A research paper has been published in INPRESSCO IJCET discussing the different problems in cloud like, Bottleneck in cloud, API configuration and Data security in Cloud.

Frameworks like Aura and Spring on which the application would run were considered. Interface was strictly going to be JavaScript based. Upon discussions and detailed study, Spring framework was selected to proceed with.

On running the application on Tomcat (version 7) server, it successfully uses the local file system to upload a file and later convert it to PDF format. The metric convertor too works successfully and displays correct results upon conversion.

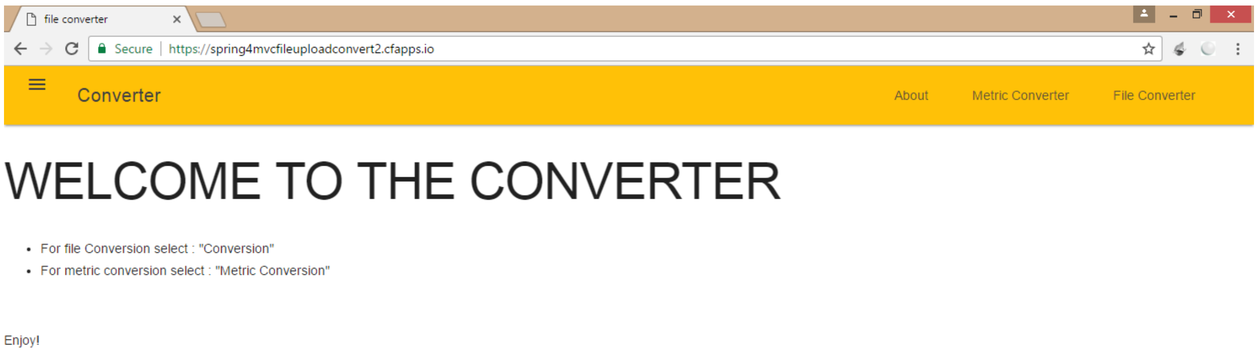
Two cloud platforms were chosen to host our application, namely, Pivotal Cloud Foundry and Heroku. The application upon being deployed to Pivotal Cloud Foundry, didn't work successfully for uploading a chosen file to the local file system. Moreover, the cloud platform doesn't support local file systems which is a hurdle in the way of centralized applications. However, the metric conversion module worked flawlessly while invoking the correct conversion logic and displaying the correct result, since it does not require Local File and Database System.

Screenshots of application

<u>User Interface</u>

The User Interface is one of the parts of the application though which users can interact with it. It needs a simple yet attractive design to be able to make it friendly to use. The software characteristics for a user interface should be followed that makes it simple, easy to understand, users don't have to remember, and also there should be abstraction.

The user interface that is designed for the first page involves a welcome message for the users along with a top bar for navigation.



WELCOME TO THE CONVERTER

- For file Conversion select : "Conversion"
- For metric conversion select : "Metric Conversion"

Enjoy!

Fig 6.1.1: First page design of user interface.

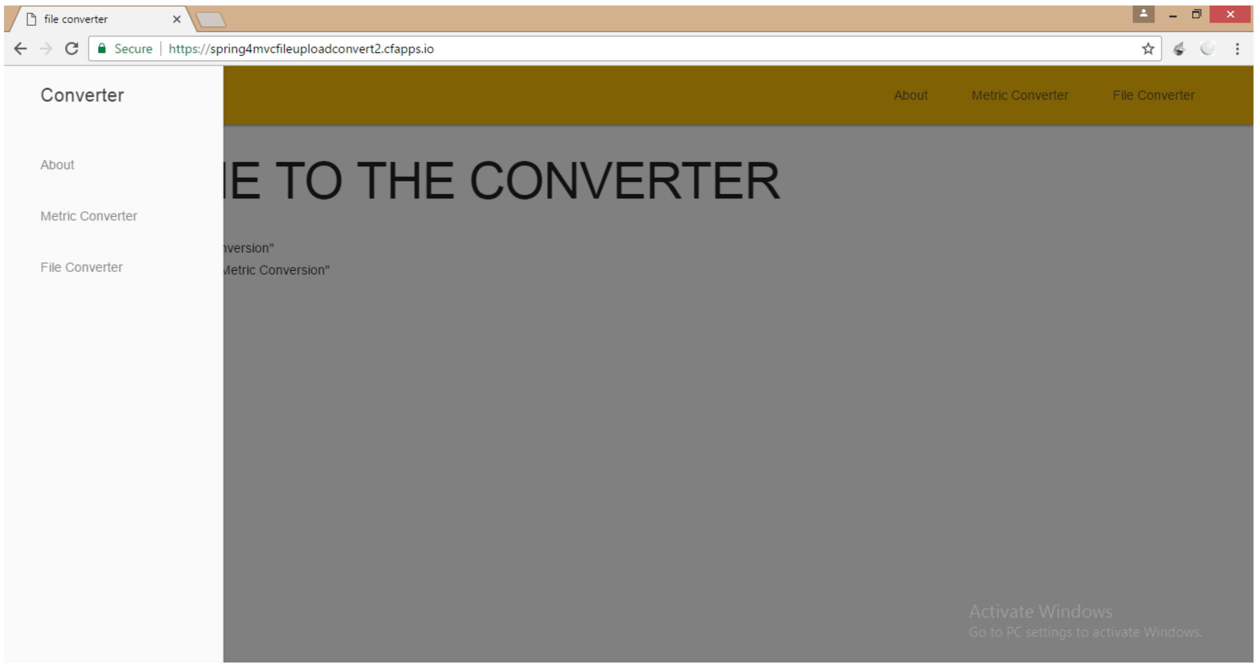
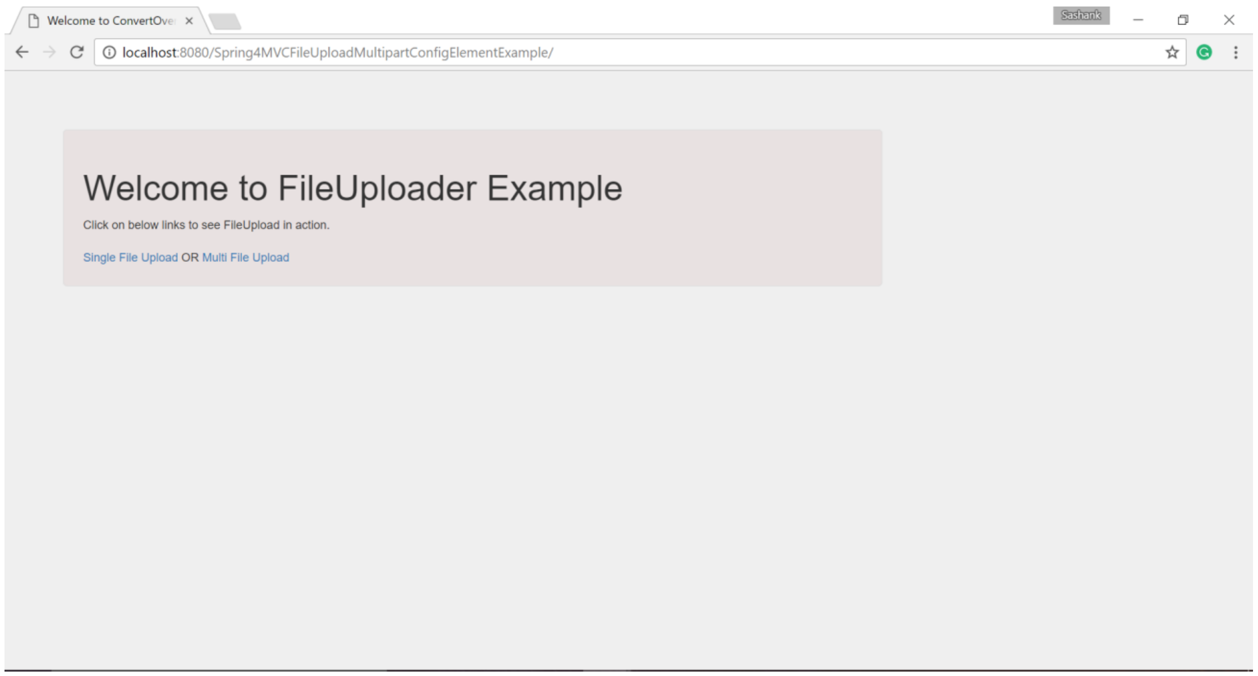


Fig 6.1.2: First page drawer of menu options.

<u>File Conversion</u>

The conversion page will have options to convert the document from one form to another. There is a short description of each form of file in the conversion card. The example provides forms of video conversion that may be incorporated and how the cards and buttons are designed to appear. The upper bar of menu can help to navigate. Like this other conversion pages will be made according to the type of file.



Welcome to ConvertOve
Sashank
localhost:8080/Spring4MVCFileUploadMultipartConfigElementExample/

Welcome to FileUploader Example

Click on below links to see FileUpload in action.

Single File Upload OR Multi File Upload

Fig 6.1.3: User interface in spring.

We have implemented modules which lets user upload single or multiple files to the web server at a time.

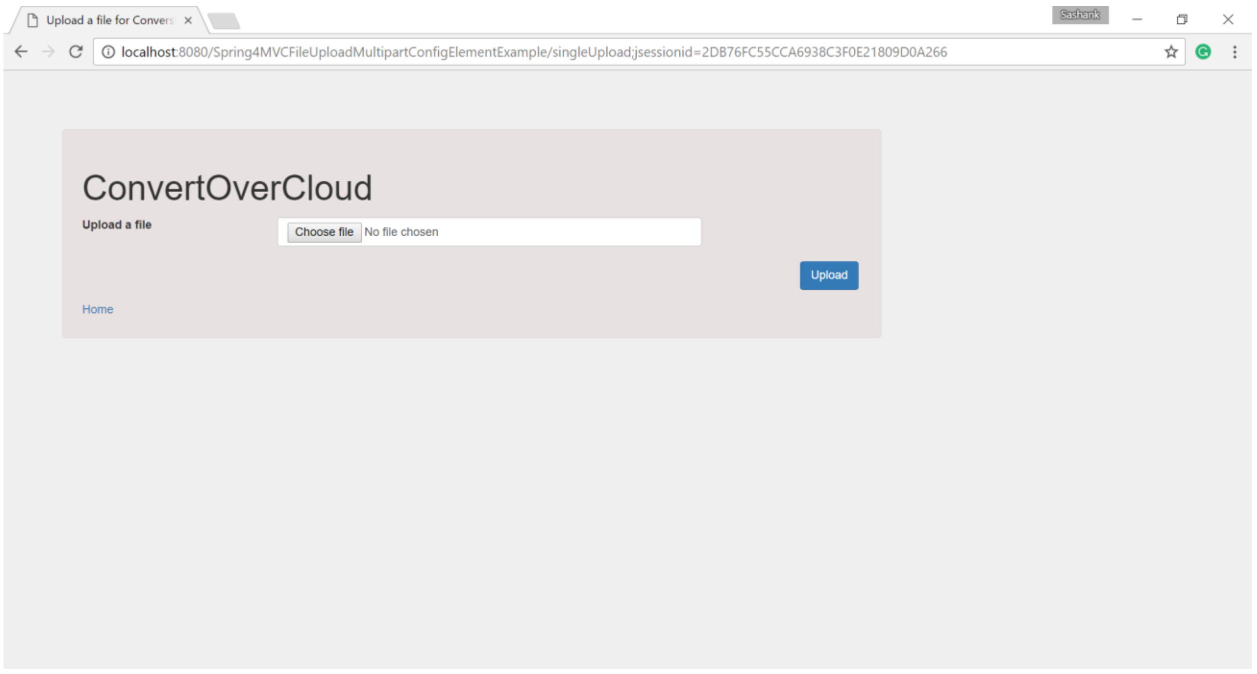

Upload a file for Convers
Sashank
localhost:8080/Spring4MVCFileUploadMultipartConfigElementExample/singleUpload;jsessionid=2DB76FC55CCA6938C3F0E21809D0A266
ConvertOverCloud
Upload a file
Choose file No file chosen
Upload
Home

Fig 6.1.4 uploading a file in spring.

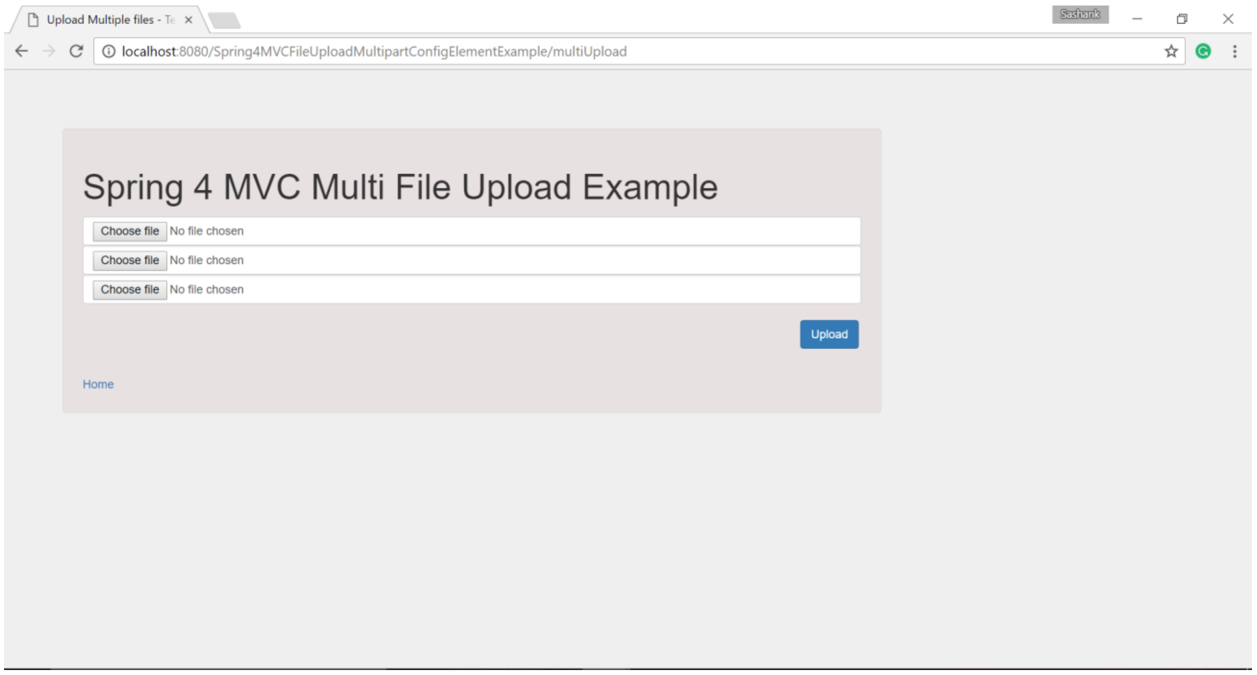
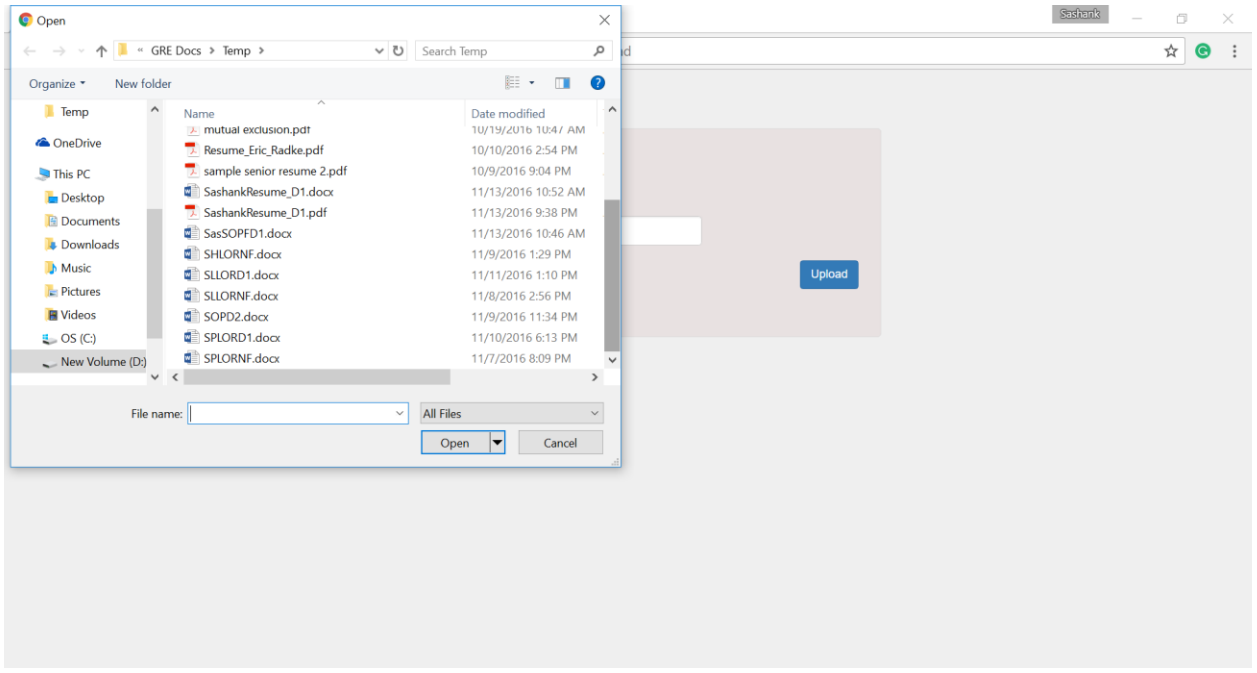
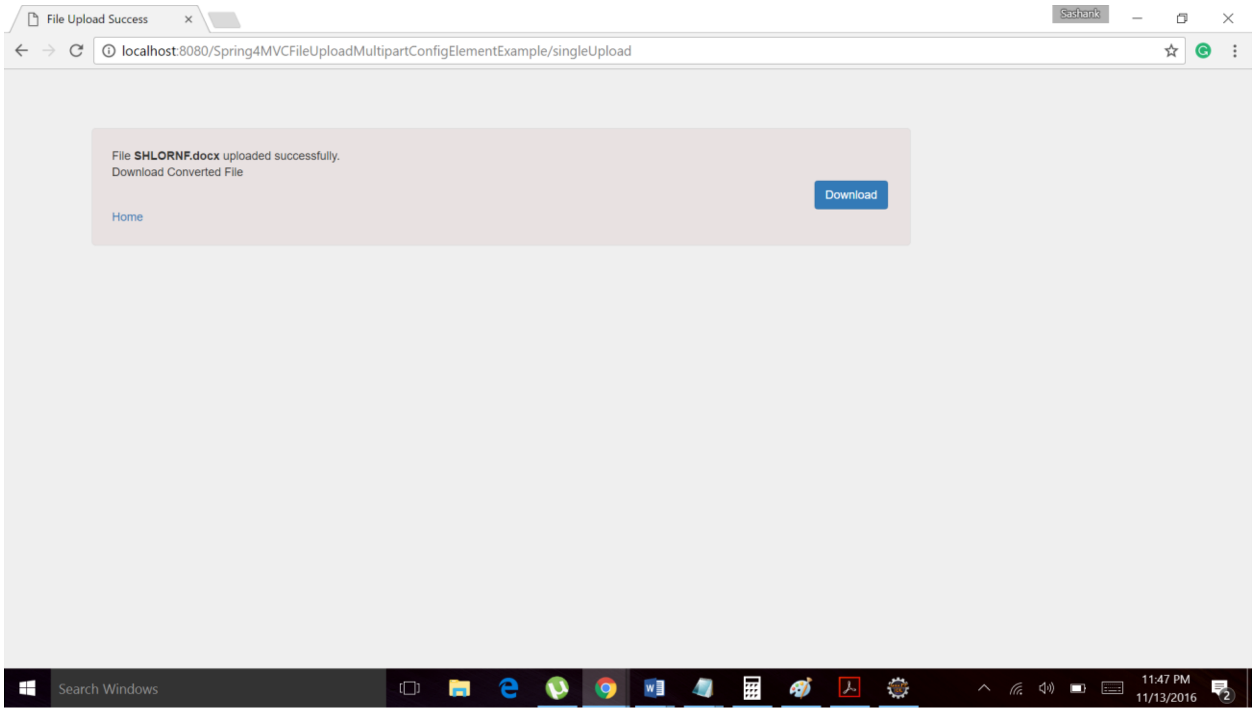

Upload Multiple files - Te
Sashank
localhost:8080/Spring4MVCFileUploadMultipartConfigElementExample/multiUpload
Spring 4 MVC Multi File Upload Example
Choose file No file chosen
Choose file No file chosen
Choose file No file chosen
Upload
Home

Fig 6.1.5: User can select the file he wants to convert.



Name	Date modified
mutual exclusion.pdf	10/19/2016 10:47 AM
Resume_Eric_Radke.pdf	10/10/2016 2:54 PM
sample senior resume 2.pdf	10/9/2016 9:04 PM
SashankResume_D1.docx	11/13/2016 10:52 AM
SashankResume_D1.pdf	11/13/2016 9:38 PM
SasSOPFD1.docx	11/13/2016 10:46 AM
SHLORNF.docx	11/9/2016 1:29 PM
SLLORD1.docx	11/11/2016 1:10 PM
SLLORNF.docx	11/8/2016 2:56 PM
SOPD2.docx	11/9/2016 11:34 PM
SPLORD1.docx	11/10/2016 6:13 PM
SPLORNF.docx	11/7/2016 8:09 PM

Fig 6.1.6: Choosing file pop up menu.



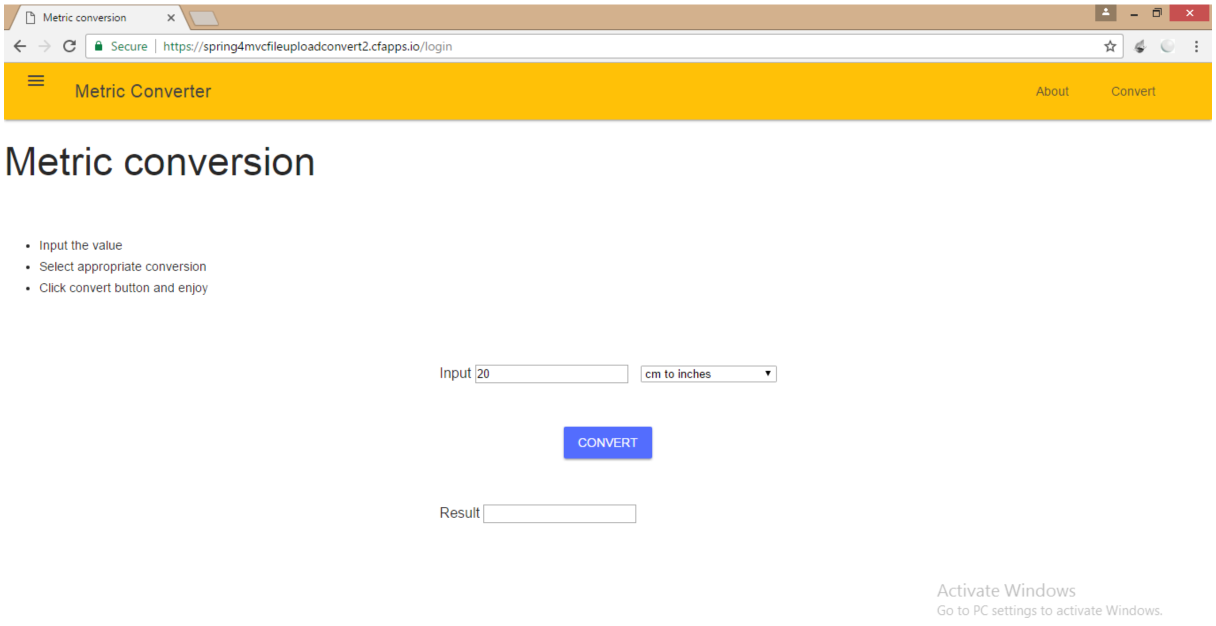
localhost:8080/Spring4MVCFileUploadMultipartConfigElementExample/singleUpload

File **SHLORNF.docx** uploaded successfully.
Download Converted File

Fig 6.1.7: File uploaded message.

The file is uploaded to the web server, and can be downloaded back by clicking the download button.

Metric Conversion



Metric conversion
Secure | https://spring4mvcfileuploadconvert2.cfapps.io/login
Metric Converter
About
Convert

Metric conversion

- Input the value
- Select appropriate conversion
- Click convert button and enjoy

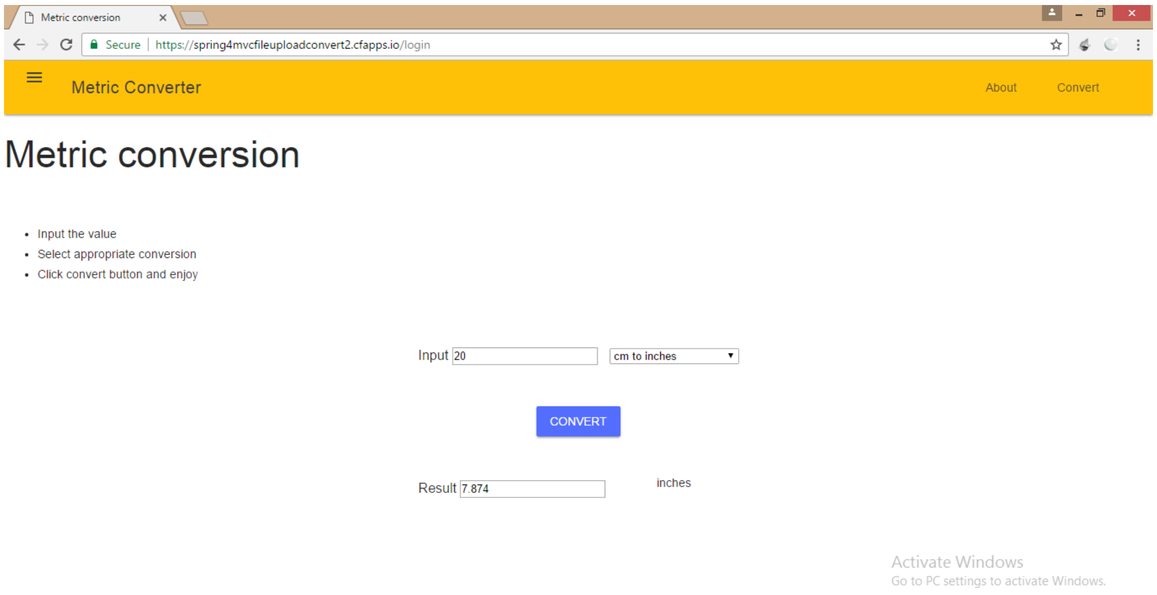
Input 20 cm to inches

CONVERT

Result

Fig 6.1.8: Metric Conversion Page

This page will be visible when Metric Conversion is selected.



Metric conversion
Secure | https://spring4mvcfileuploadconvert2.cfapps.io/login
Metric Converter
About
Convert

Metric conversion

- Input the value
- Select appropriate conversion
- Click convert button and enjoy

Input 20 cm to inches

CONVERT

Result 7.874 inches

Fig 6.1.9: Metric Conversion Page (Inches to Centi-Meter)

This is how the output will look for inches to centimeter conversion there are other five possible conversions.

7. CONCLUSION AND FUTURE SCOPE

The increase in demand of Cloud Computing in industry and the necessity to solve the problem of Portability. Though being complex, it’s not impossible. It is necessary to ensure that there are at least reasonable ways of providing the same feature/function on multiple platforms, even if the same APIs won't work across them all.

We developed a Cloud application to test the possibility of portability by testing the application across two Cloud platforms.

This Application will do the following:

- Type Conversion, e.g. Doc to PDF.
- Download PDF file.
- Metric conversion of various types.

However the vendor lock in problem still persist hence even on deploying application on clouds the Cloud could not support Application portability for Local File System and Database. The conclusion was when an application uses File systems, it is more difficult to port it than the one which doesn’t use any centralized database or local file system. We learnt that, if possible and supported by the platform, Cloud storage can be used for maintaining system database.

The metric converter did not require file system or database service, this module of the application was successfully ported and we computed its performance in our testing modules, to verify and validate Cloud Application Portability.

Future scope:

Our main aim for suture scope would be try to solve the vendor lock in file system and database in various clouds and try porting our original application with full functionalities. This application will be useful for those member of society who have older versions of software with simple net connection and a computing device one can access and download the Document in the required format.

We will also try to incorporate Signup and Login Modules to maintain security aspects of Cloud.

8. REFERENCES

[1] Vangie Beal, “Cloud Computing: The cloud.”, on Webopedia[Online], Available:”http://www.webopedia.com/quick_ref/cloud_computing_terms.asp.”

[2] Business dictionary,”portability”, [*Online*], Available: “http://www.businessdictionary.com/definition/portability.html”

[3] Techtarget ,“interoperability”, [Online], Available: “http://searchmicroservices.techtarget.com/definition/interoperability”

[4] Bailey Caldwell, “Cloud Migration and Portability: What VMware and AWS aren’t telling you” (June 04, 2014), CLOUD MANAGEMENT BLOG *[online],* available: http://www.rightscale.com/blog/enterprise-cloud-strategies/cloud-migration-and-portability-what-vmware-and-aws-arent-telling-you

[5] Chloe Jackson, “Designing Portable Applications With Cloud Foundry” (December 13, 2012), *[online],* available: https://blog.pivotal.io/pivotal-cloud-foundry/products/designing-portable-applications-with-cloud-foundry

[6] Stefan Kolb and Guido Wirtz, “Towards Application Portability in Platform as a Service”, in University of Bamberg Bamberg, Germany.

[7] Giuseppina cretella and Beniamino Di Martino (2012), “Towards Automatic Analysis of Cloud Vendors APIs for Supporting Cloud Application Portability”, IEEE © 2012 IEEE, pp. 61-67

[8] Fotis Gonidis Anthony J. H. Simons Iraklis Paraskakis Dimitrios Kourtesis,” Cloud Application Portability: An Initial View”, BCI’13, September 19-21, 2013, Thessaloniki, Greece. Copyright 2013 ACM 978-1-4503-1851-8/13/09

[9] “Python Overview”, [*Online*], Available : “https://www.tutorialspoint.com/python/python_overview.htm”

[10] “Features of Java”,[*Online*], Available: “http://www.javatpoint.com/features-of-java”

[11] “Spring MVC Documentation”,[*Online*], Available: “https://docs.spring.io/spring/docs/current/spring-framework-reference/html/mvc.html.”

[12] “API Itext Developer”, [*Online*], Available: “http://developers.itextpdf.com/apis”

[13] Sudersasnam, Suryanarayan, Shon Bangale, G. V. A. Sashank, and Srija Ganguly. "Application Portability: A Necessity." (2016).