

Cloud Orchestration on Peer to Peer Network



By

**CJUO Rooshan Saleem Butt
GC Hamza Shoukat
GC Ahmed Haider Nawaz**

Submitted to Faculty of Computer Software Engineering,
National University of Sciences and Technology, Islamabad in partial fulfillment for
The requirement of a B.E Degree in Computer Software Engineering

JUNE 2014

CERTIFICATE

Certified that the contents and form of project report entitled “Cloud Orchestration on Peer to Peer Network” submitted by Rooshan Saleem Butt, Muhammad Hamza Shaoukat and Ahmed Haider Nawaz have been found satisfactory for the requirement of the degree.

Supervisor: _____

Dr. Sarmad Sadik

ABSTRACT

This is a cloud based application that manages, monitors, coordinates and facilitates the execution of workflows, heterogeneous software services and application processing on independent nodes connected together in a peer to peer network. This application acts as a middleware among the front end applications and maps the relevant processes to individual computing nodes based on its required processing and memory requirements with achieving a necessary level of access and location transparency.

The work focuses on the creation of cloud based orchestration middleware which enables the users to use the processing of computers placed in the lab. Our service maps them dynamically to appropriate nodes. The system will maintain our cloud network on peer to peer nodes and act as computing engine for front end. Our cloud will autonomously manage and coordinate the node resources in a flexible and reusable fashion. This product is the implementation of cloud processing from a network of computers on a small scale inside universities and colleges where students can process their data by using this service being available from the university or college. The main part of the system is the middleware which behaves like a mediator between the users and the computers in peer to peer network.

DECLARATION

We declare that the work presented herewith is the result of sole effort of our group, comprising of Rooshan Saleem Butt, Muhammad Hamza Shoukat and Ahmed Haider Nawazand is free of any kind of plagiarism in part. We also declare that the dissertation has never been submitted previously in part or whole in support of another award or qualification either at this institution or elsewhere.

DEDICATION

In the name of Allah, the Most Merciful, the Most Beneficent

To our parents, without whose unflinching support and cooperation,

a work of this magnitude would not have been possible.

ACKNOWLEDGMENT

We are grateful to Almighty Allah for bestowing us with the strength to undertake and complete the project.

We owe a special debt of gratitude to our supervisor, Dr. Sarmad Sadik for guiding us in understanding the concepts of Cloud Computing and for the continuous supervision, motivation and support provided to us right through the project. Without his supervision we would not have been able to complete this successfully. We would like to thank all our other teachers for guiding us in solving our problems related to our project.

TABLE OF CONTENTS

1. Introduction.....	1
1.1 Background	1
1.2 Problem Domain.....	1
1.3 Goals and Objectives	2
1.3.1 Goals	2
1.3.2 Objectives.....	2
1.4 Deliverables.....	3
2. Literature Review	4
2.1 Google File System	4
2.1.1 Features	4
2.2 Amazon Web Service	5
2.2.1 Features	6
3. Software Requirement Specification.....	8
3.1 Introduction	8
3.2 System Features.....	8
3.2.1 Registration to COPTPN.....	8
3.2.2 Login to COPTPN.....	9
3.2.3 Services Selection	10
3.2.4 Processing the Request.....	11
3.2.5 Terminating the Application	12
3.2.6 Nodes Management.....	13
3.3. Other Nonfunctional Requirements	14
3.3.1. Performance Requirements	14
3.3.2. Safety Requirements	14
3.3.3. Security Requirements	15
3.3.4. Software Quality Attributes	15
3.3.4.1. Usability	15
3.3.4.2. Scalability.....	16
3.3.4.3. Flexible.....	16
3.3.4.4. Maintainability	16
3.3.4.5. Availability.....	16
3.3.4.6. Robustness.....	16
3.3.4.7. Business Rules	16
3.3.5. Other Requirements	16
4. Architecture.....	17
4.1 System Design	17
4.1.1 Cloud Orchestration on Peer to Peer Network Architecture	17
4.1.2 Architectural Design	17
4.1.3 Description	18
4.2 Detailed System Design.....	21
4.2.1 Login	22
4.2.2 Select the Service	23
4.2.3 User sends the input data.....	25
4.2.4 View node Information	26
4.2.5 Approve new Account.....	28
4.2.6 Add new node	29
4.2.7 Delete a node.....	30
4.2.8 Setup a Service.....	31
4.3 Deployment Diagram	33
4.4 Data Structure Design.....	34
4.5 Activity Diagram	35
4.5.1 Activity Diagram for User.....	35

4.5.2	Activity Diagram for Admin	36
4.6	Sequence Diagram	37
4.6.1	Admin Login	37
4.6.2	Admin adding a Node	38
4.6.3	User Signup	38
4.6.4	User Login	39
4.6.5	Selecting a Service	39
5.	Tools and Technologies	41
5.1	Microsoft Visual Studio 2008	41
5.2	Microsoft SQL Server	41
6.	System Implementation	42
6.1	User Login	42
6.2	Select the Service	45
6.3	User sends the input data	45
6.4	View Node Information	46
6.5	Approve New Account	46
6.6	Add a new node	47
6.7	Setup a service	47
6.8	Terminating the Application	47
7.	Testing and Results	49
7.1	Introduction to test Plan	49
7.2	Features to be tested	49
7.3	Features Not to Be Tested	52
7.3.1	Approach	52
7.3.2	Item Pass/Fail Criteria	52
7.3.3	Suspension Criteria and Resumption Requirements	53
7.3.4	Environmental Needs	53
7.3.5	Schedule	53
7.3.6	Risks and Contingencies	53
7.3.7	Functional Testing (Black box)	53
7.4	Test Cases	53
7.4.2.	Test Case no. A02	54
7.4.3.	Test Case no. A03	54
7.4.4.	Test Case no. A04	55
7.4.5.	Test Case no. A05	56
7.4.6.	Test Case no. A06	56
7.4.7.	Test Case no. A07	57
7.4.8.	Test case no. A08	57
7.5	Testing of non-functional features	58
7.5.1	Test Case no. B01	58
7.5.2	Test Case no. B02	59
7.5.3	Test Case no. B03	59
7.6	Integration Testing	60
7.7	System Testing	60
8.	Conclusion	61
9.	USER MANUAL	62
9.1	Reading Instructions	62
9.2	Installation	63
9.2.1	Website	63
9.2.2	Middleware Server	69
9.2.3	Middleware Client for Service	69
9.3	How to use the system	70
10.	Bibliography	71

LIST OF FIGURES

Figure 1: Working of GFS.....	5
Figure 2: Amazon web services.....	7
Figure 3: Amazon web services Dashboard.....	7
Figure 4: System Architectural level Block diagram.....	18

Figure 5: System Use case diagram.....	21
Figure 6: Deployment diagram of the system.....	33
Figure 7: Class diagram of the system.....	35
Figure 8: Activity diagram for the user.....	36
Figure 9: Activity diagram for Administrator.....	37
Figure 10: Sequence diagram of Admin Login.....	38
Figure 11: Sequence diagram of Admin adding a node.....	38
Figure 12: Sequence diagram of User Signup.....	39
Figure 13: Sequence diagram of User Login.....	39
Figure 14: Sequence diagram of Service Selection.....	40

LIST OF TABLES

Table 1: Test items.....	49
--------------------------	----

Table 2: Features to be tested..... 52

CHAPTER 1

INTRODUCTION

1. Introduction:

This document acts as the thesis for the software infrastructure (or Product) that enables the processing of a user data on a network of lab computers through peering, Hence forth termed as **Cloud Orchestration on Peer to Peer Network**, and provides an overall description of it. It describes the problem statement, approach followed to accomplish the project, goals, literature review, functional requirements, non-functional requirements, design and development phase, analysis and testing and the future work which can be done to make this product more enhanced and useful..

1.1 Background:

With increase in the demand of advanced computing, users require machines which can provide them higher processing. Moreover, users also required up-to-date versions of the software to fulfill their processing needs. For this Various cloud orchestration software have been developed by vendors including HP Operation Orchestration, Flexi ant Cloud Orchestrator, Apache ODE etc. which helps in autonomous management and coordination of various software services on heterogeneous devices. But all the previous orchestration involved centralized servers which offer their processing/storage capacities but we are using PCs (Peer-to-Peer network) to offer the processing functionalities to the users and users can access them remotely. This is will make the system cheaper as we are using lab computers so it can be afforded by the smaller.

1.2 Problem Domain:

The problems that are being focused by the project are:

1. PCs in the labs are usually idle or their processing capabilities are not fully utilized specially during night hours.
2. User may have huge amount data to be processed and his/her machine doesn't possess such specifications to process it effectively.
3. User may not have latest version of some software to do his job.

The system shall focus on the following aspects:

- a. Idle PCs in the labs are utilized up to their maximum capabilities.
- b. Good quality services.
- c. Reusability of available resources.
- d. Cheap cost of processing.

1.3 Goals and Objectives:

Following are the academic objectives and goals of the project achieved.

1.3.1 Goals:

The goals which it can achieve are.

1. Development of middleware for creating private cloud on desktop PC.
2. This middleware act as mediator between user, user interface and the processing machines at backend.
3. Development of a web based user-interface.
4. Assist the user in accessing his cloud processing backend.
5. Ensure communication between the local machine and the cloud processing devices/machines over the network.

1.3.2 Objectives:

- a. To learn the concepts of cloud computing.

- b. To understand the concepts and functioning of middleware.
- c. To develop a web application for the web end user.

1.4 Deliverables:

Deliverables of the project are:

- a. Formation of a private cloud on peer to peer network of PCs.
- b. Web Application for end users
- c. Documentation and User Manual.

CHAPTER 2

LITERATURE REVIEW

2. Literature Review:

2.1 Google File System

Google File System, a scalable distributed file system for large distributed data-intensive applications. It provides fault tolerance while running on inexpensive commodity hardware, and it delivers high aggregate performance to a large number of clients. While sharing many of the same goals as previous distributed file systems, the design has been driven by observations of those application workloads and technological environment, both current and anticipated that reflect a marked departure from some earlier file system assumptions. This has led to reexamine traditional choices and explore radically different design points. The file system has successfully met storage needs. It is widely deployed within Google as the storage platform for the generation and processing of data used by our service as well as research and development efforts that require large data sets. The largest cluster to date provides hundreds of terabytes of storage across thousands of disks on over a thousand machines, and it is concurrently accessed by hundreds of clients. Files are organized hierarchically in directories and identified by pathnames. We support the usual operations to *create*, *delete*, *open*, *close*, *read*, and *write* files.

2.1.1 Features:

Some of the features are:

- a. Efficiently handle large data set that don't fit in a single commodity hard disk.
- b. Optimize for the common write pattern.
- c. Minimal state is stored in the master, and state information is gathered in a more real time fashion allowing faster addressing of faults.
- d. Large chunk sizes are used which fits ideally for the design goal of huge files
communication between clients and master is kept minimal.

- e. Chunk replication and chunk check summing improves reliability and fault tolerance.
- f. While operation log is maintained on the master, check pointing of the same is done to improve performance and minimize replay overhead.

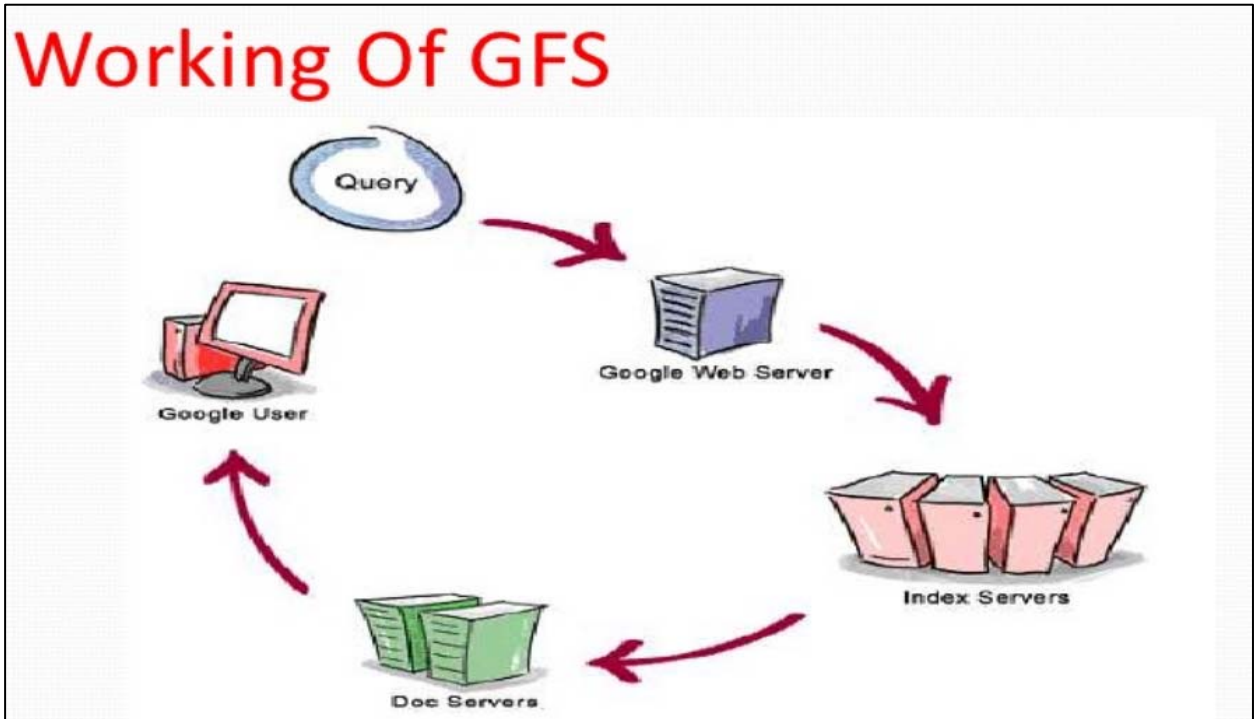


Figure 1: Working of GFS

2.2 Amazon Web Service

Amazon Web Services (abbreviated **AWS**) is a collection of remote computing services (also called web services) that together make up a cloud computing platform, offered over the Internet by Amazon.com. The most central and well-known of these services are Amazon EC2 and Amazon S3. The service is advertised as providing a large computing capacity (potentially many servers) much faster and cheaper than building a physical server farm. AWS is a web service that provides resizable compute capacity in the cloud. It is designed to make web-scale computing easier for developers. AWS is a simple web service interface allows you to obtain and configure capacity with minimal friction. It provides you with complete control of your computing resources and lets you run on Amazon's proven computing environment. AWS reduces the time required to obtain and

boot new server instances to minutes, allowing you to quickly scale capacity, both up and down, as your computing requirements change. AWS changes the economics of cloud computing servers by allowing you to pay only for capacity that you actually use. AWS provides developers the tools to build failure resilient applications and isolate themselves from common failure scenarios.

2.2.1 Features:

Some of the features are:

- a. Elastic Web-Scale computing
- b. Completely Controlled
- c. Flexible Cloud Hosting
- d. Designed for use with other Amazon Web Services
- e. Reliable
- f. Secure
- g. Inexpensive
- h. Easy to Start

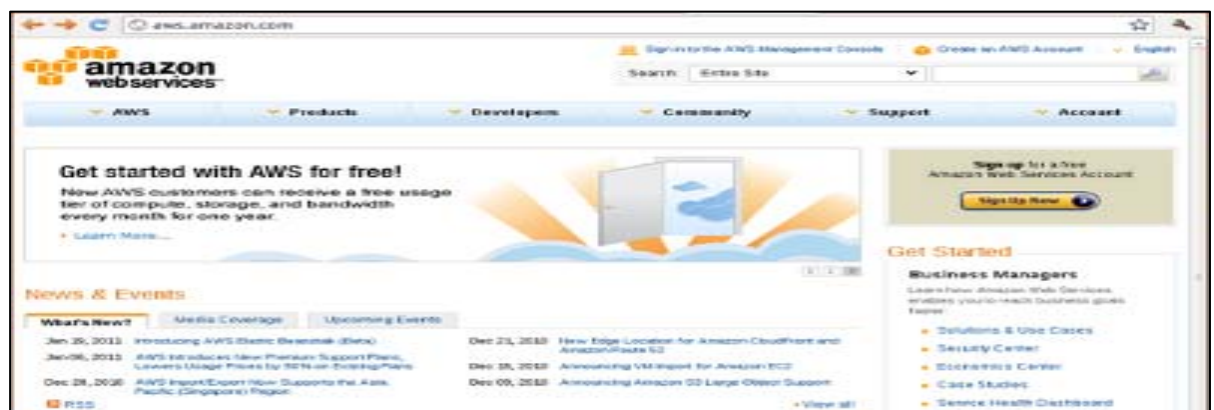


Figure 2: Amazon web services

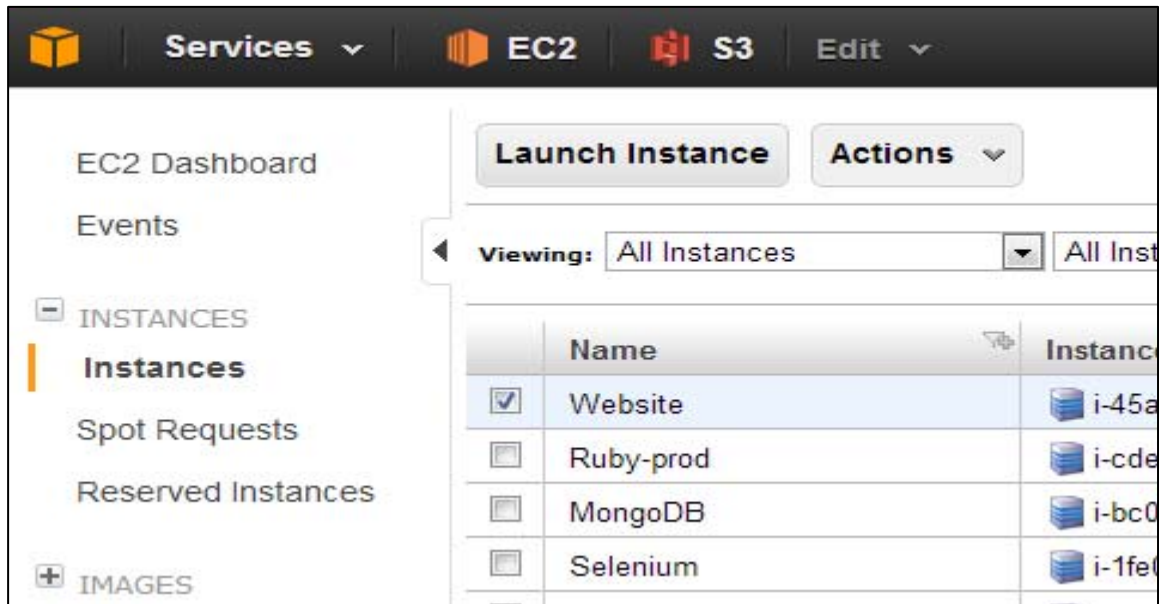


Figure 3: Amazon web services Dashboard

CHAPTER 3

SOFTWARE REQUIREMENT SPECIFICATION

3. Software Requirement Specification:

3.1 Introduction:

The system should be able to perform following functionalities.

1. Users should access it over a web based front end.
2. System should offer the services to the registered users.
3. System should create a user request table for users' request.
4. System should create a user table.
5. System should synchronize the user with the data present inside the cloud for respective user.
6. Check which nodes are available and which are not available.
7. System should calculate the available resources and assign them to user request if they are available.
8. New users can be registered by filling and submitting the registration form.

3.2 System Features

3.2.1 Registration to COTPN:

3.2.1.1 Description and Priority:

Users who want to use the purposed system and are not registered, first they will register themselves.

Priority: High.

3.2.1.2 Stimulus/Response Sequences:

Stimulus: User will click on the signup button.

Response: A form will be provided to the user.

Stimulus: User will fill all the fields of the form and press the submit button.

Response: A message will be displayed, telling the user that he/she has been registered.

Alternate course of action

Stimulus: User will click on the signup button.

Response: A form will be provided to the user.

Stimulus: User doesn't fill all the fields of the form and press the submit button.

Response: An error message will be displayed, telling the user to fill all the fields of the form.

3.2.1.3 Functional Requirements:

REQ-1 System database will have a form to collect the information from the user.

REQ-2 System will store all the data entered by the user in its database.

3.2.2 Login to COTPN

3.2.2.1 Description and Priority:

Administrator and the other members of the university/college who have registered to use this service will first login to the system. After login they can do the further actions accordingly.

Priority: High

3.2.2.2 Stimulus/Response Sequences:

As user accesses our through web browser, login screen will be displayed.

Stimulus: User will enter his/her username and password and presses the login button.

Response: System will authenticate the username and password from the database.

Response: A new window will open containing the services offered by the system.

Alternate course of action

As user accesses our through web browser, login screen will be displayed.

Stimulus: User enters the wrong username or password and presses the login button.

Response: System will authenticate the username and password from the database.

Response: Error message will be displayed.

3.2.2.3 Functional Requirements:

REQ-1 The system will check the validity of the user from database (which have the passwords and usernames already stored).

3.2.3 Services Selection:

3.2.3.1 Description and Priority:

After login into the system, system will show the available services. It includes different applications that are installed on the peers which can be used by the users e.g. WinRAR, Microsoft Word etc. Along with this, processors' utilization will also be shown i.e. how much of the processors' processing capabilities are in use (in percentage).

Priority: High

3.2.3.2 Stimulus/Response Sequences:

Stimulus: User presses the login in button (with correct username and password).

Response: A new window will open containing the list of the services that our system offers and list of the processors will also be displayed along with their usage percentage.

Stimulus: User selects any of the services by clicking upon its icon.

Response: System will add a new user along with his/her job in its database.

Response: Selected application starts running on the node and its interface window appears in web browser.

Alternate course of action

Stimulus: User presses the login in button (with wrong username or password).

Response: Error message will be displayed and user is asked to re-enter the username and password.

3.2.3.3 Functional Requirements:

- REQ-1** The system will check the availability of the required memory, hard disk space and processor usage.
- REQ-2** Node's processor usage is lesser than 95%.
- REQ-3** Node's memory consumption is lesser than 90%.
- REQ-4** Hard disk has more than 4GB free space.
- REQ-5** System is connected to the database.
- REQ-6** Job will be assigned to the selected peer having CPU usage percentage lesser than 95.

3.2.4 Processing the Request:

3.2.4.1 Description and Priority:

If the system has the required resources then the system starts executing the user request.

Priority: High

3.2.4.2 Stimulus/Response Sequences:

Stimulus: User will select a node and service.

Response: Middleware client for service is downloaded on the client side.

Stimulus: User will login the middleware client.

Response: Middleware client asks IP address, port number and file to send.

Stimulus: User enters the values and sends the file.

Response: File is transferred on the cloud computer and processed file is sent back.

Alternate course of action

Stimulus: User will select a node and service.

Response: Middleware client for service is downloaded on the client side.

Stimulus: User will login the middleware client.

Response: Middleware client asks IP address, port number and file to send.

Stimulus: User enters wrong IP address or port number.

Response: An error message is shown.

3.2.4.3 Functional Requirements:

REQ-1 Required application is installed on the node.

REQ-2 Node's processor usage is lesser than 95%.

REQ-3 Node's memory consumption is lesser than 90%.

3.2.5 Terminating the Application:

3.2.5.1 Description and Priority:

After using the required application, user will terminate the Application.

Priority: Medium

3.2.5.2 Stimulus/Response Sequences:

Stimulus: User presses the 'close' button.

Response: Application will be terminated.

Stimulus: System will release all the resources being used by that application.

Response: CPU usage, memory consumption, hard disk space and user tables will be updated.

Alternate course of action

Stimulus: User clicks on 'logout' button.

Response: Application will be terminated and user will be logged out.

Stimulus: System will release all the resources being used by that application.

Response: CPU usage, memory consumption, hard disk space and user tables will be updated.

3.2.5.3 Functional Requirements:

REQ-1 The system will terminate the application and release the resources.

REQ-2 System will show those resources as available for the further user requests.

3.2.6 Nodes Management:

3.2.6.1 Description and Priority:

System notifies about the available nodes at a particular time and discloses any failed nodes.

Priority: High.

3.2.6.2 Stimulus/Response Sequences:

Stimulus: Node 'A' fails

Response: The system immediately notifies the administrator account about the failure of node 'A'.

Stimulus: Administrator drops node 'A' from the peer-to-peer network.

Response: Node 'A' is removed from the website visualization.

Alternate course of action

Stimulus: Node 'A' fails

Response: The system immediately notifies the administrator account about the failure of node 'A'.

Stimulus: Administrator doesn't drop node 'A' from the peer-to-peer network.

Response: Node 'A' is shown on the website but it will not be working as desired.

3.2.6.3 Functional Requirements:

REQ-1 The system checks for the available nodes after every 10 minutes by sending the ping messages and generates a failure report for this kind of failure.

REQ-2 System informs the administrator account about the failure.

3.3. Other Nonfunctional Requirements

3.3.1. Performance Requirements

In order to maintain an acceptable transactional of input and output data maximum number of users should not be more than 20 at any time. Any more users will cause the delay in the response and delay will increase with the increase in the number of users from 20. It also depends upon the type of the application being used by the user. If application requires large amount of input, output and processing than system efficiently can accommodate 15 users at a time. For the applications requiring lesser processing 25 users can be accommodated efficiently by the system at a time.

3.3.2. Safety Requirements

1. The system do not promises the security of the information that is why it is highly recommended that the users should not use this system for their confidential use.

2. There should be some backup for electricity otherwise the whole system shuts down on load shedding.
3. As many computers would be operating at the same time, definitely they will generate heat and the facility should be fully equipped with firefighting and cooling systems.

3.3.3. Security Requirements

The security involved in this system is the authentication of the database administrator to log into the system. The system will have security checks and firewalls to prevent any illegal breaching into the system by unauthorized users. It will keep the log of access in the database and log will be read-once so no one can alter the log.

3.3.4. Software Quality Attributes

3.3.4.1. Usability

The system will provide the user with easy to use and understand GUI interface. User can easily interact with the system with menus and text areas.

3.3.4.2. Scalability

The system shall be handling at least 5 nodes with zero latency.

3.3.4.3. Flexible

Since software is flexible in nature, thus the system can incorporate up to 50%, for the changes are relevant to system, of change in its functionalities according to the changing user demands. For new changes the flexibility rate will be dynamic from 15% to 25 %.

3.3.4.4. Maintainability

After the release of the system, the system will be maintained after every year for unknown bugs or system performance issues. These problems will be corrected at its best by the developer team. The system is able to correct the errors after its release up to 35%.

3.3.4.5. Availability

The failure rate of the system will be 12 hours per month. The system will be available to the user 29 days and 12 hours per month.

3.3.4.6. Robustness

After the failure occurs, the system will be able to recover within 2 hours after restart the system.

3.3.4.7. Business Rules

Administrator would only have the right of authenticating the users. He/she can't be authorized to view the data of students. Students should have their access rights and limitations, they should be allowed to view only there data and content. All other user accounts should not be let disclosed to the normal users of the system.

3.3.5. Other Requirements

System shall communicate with nodes through middleware. The middle ware decides about a node which could be used for processing the data on the basis of equal distribution of load on each node

CHAPTER 4

SOFTWARE DESIGN SPECIFICATIONS

4. Architecture

4.1 System Design:

Following is the architecture design of the system.

4.1.1 Cloud Orchestration on Peer to Peer Network Architecture:

As this is a web based application so we have used client-server architecture which is the best suitable for a web based application. Further, within client and server module we have used loose MVC (Model View Controller) architecture. This was due to the nature of the project, which includes a cloud. Since a cloud is involved, it intrinsically relies on MVC, since for clouds; a Presentation-Platform-Information model is the basis of development. In a cloud environment, Presentation is provided by end user clients like browsers, the Platform is always the middleware atop physical hardware, and the data is processed on the cloud and accessed via the middleware.

4.1.2 Architectural Design:

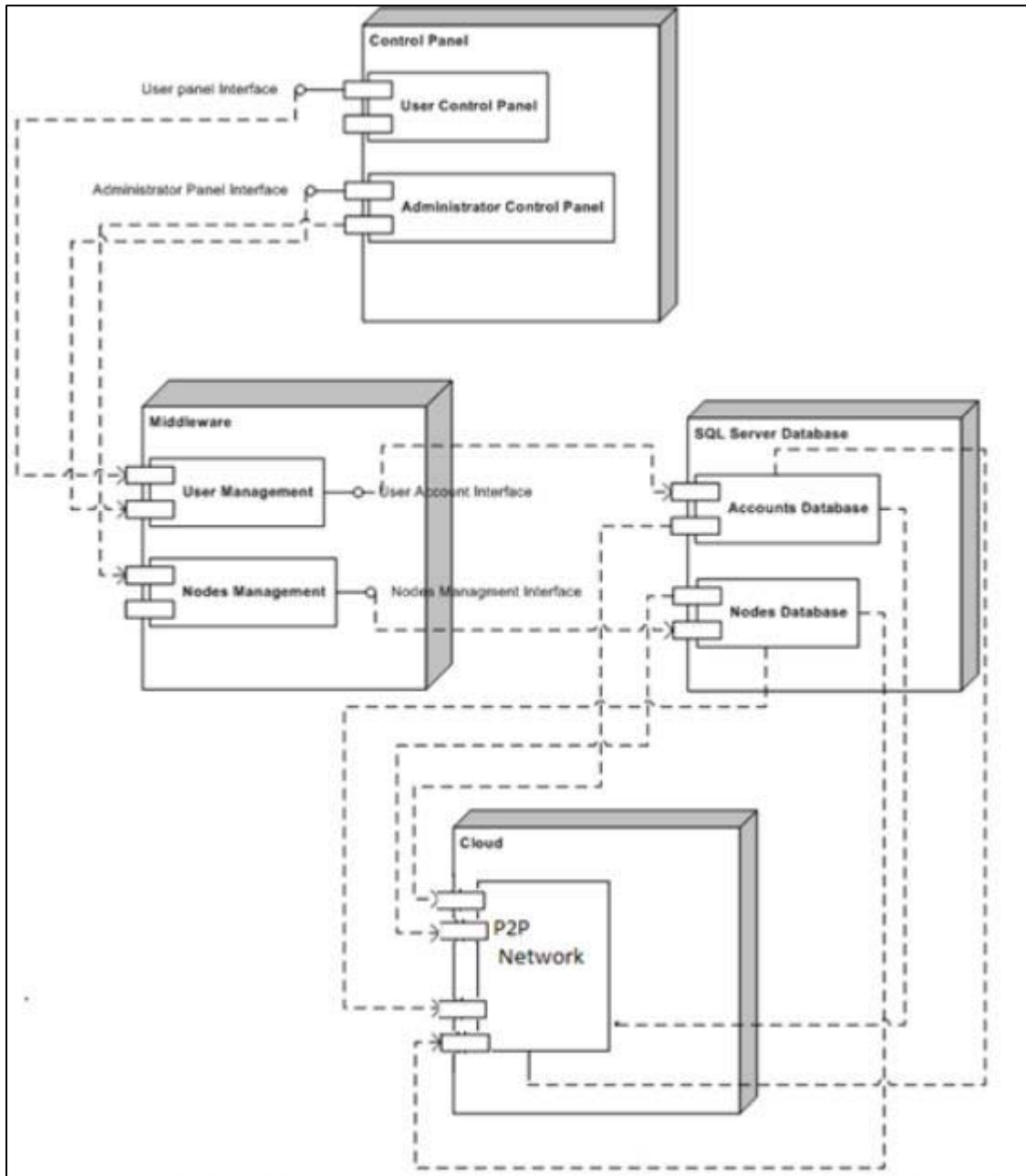


Figure 4: System Architectural level Block diagram

4.1.3 Description:

As shown above, this software system is a set of communicating entities that collaborate to perform a task. This diagram shows these entities, their relationships and the relationship to the actors in the system. This top level is a diagram where each entity has a name, an abstract specification and an interface design.

All entities above are described below:

4.1.3.1 Middleware:

1. **Specification:** A middleware is used in conjunction with server operating systems to glue together multiple machines and to provide clients with a set of available virtual resource pools on which they may upload their files. The benefit of such an approach is derived from grid computing. This middleware shall interact to the incoming http request from web browser to communicate with the database. It depends on the cloud's middleware, that from which exact computer it provides the service to the users. It could be either computer A or computer B, or Computer C.
2. **Interface:** The cloud is configured to let in only users with permitted usernames and passwords. These credentials would be stored on a database hosted on the cloud.

4.1.3.2 Cloud:

1. **Specification:** Cloud will be implemented on a Peer to Peer network of the desktop computers. These peers will appear as single identity to the outside users. On each of the n-odes, middleware is installed which will interact with the user request.
2. **Interface:** The middleware will accept the user request and it will decide on which node the request should be mapped. Similarly, middleware will send the output back to the appropriate user.

4.1.3.3 SQL Server Database:

1. **Specification:** This shall be a relational database handled using SQL Server 2012 RDBMS. It shall be installed on the master node. A database namely COTPN is used for nodes, services and users' management. Main tables of the database are Nodes, NodesStat, Services, Users and UsersLog.

2. **Interface:** The server shall connect the database using TCP protocol. The connection strings and application settings are loaded from a configuration file in the same directory as of website.

4.1.3.4 Administrator Control Panel:

1. **Specification:** The administrator role is assigned the responsibility of maintaining the system. He does it through this portal. In this portal he has the services to monitor the usage of the cloud, check its resource health and also to shut down server if necessary. Lastly, this portal could be used to handle users of the system and to change their rights and privileges.
2. **Interface:** A simple GUI shall be provided to the administrator to view and check the status of cloud. In addition, advanced functions like starting/shut down of the servers. Database administration would be done directly on a tabular interface that shall connect the database to the portal remotely. However, the administrator has to first explicitly login to the portal before making any changes.

4.1.3.5 User Control Panel

1. **Specification:** the user has the privileges to select from the services offered, send input data and system will return the processed output.
2. **Interface:** A simple GUI shall be provided to the user to select the node and perform its tasks.

4.2 Detailed System Design:

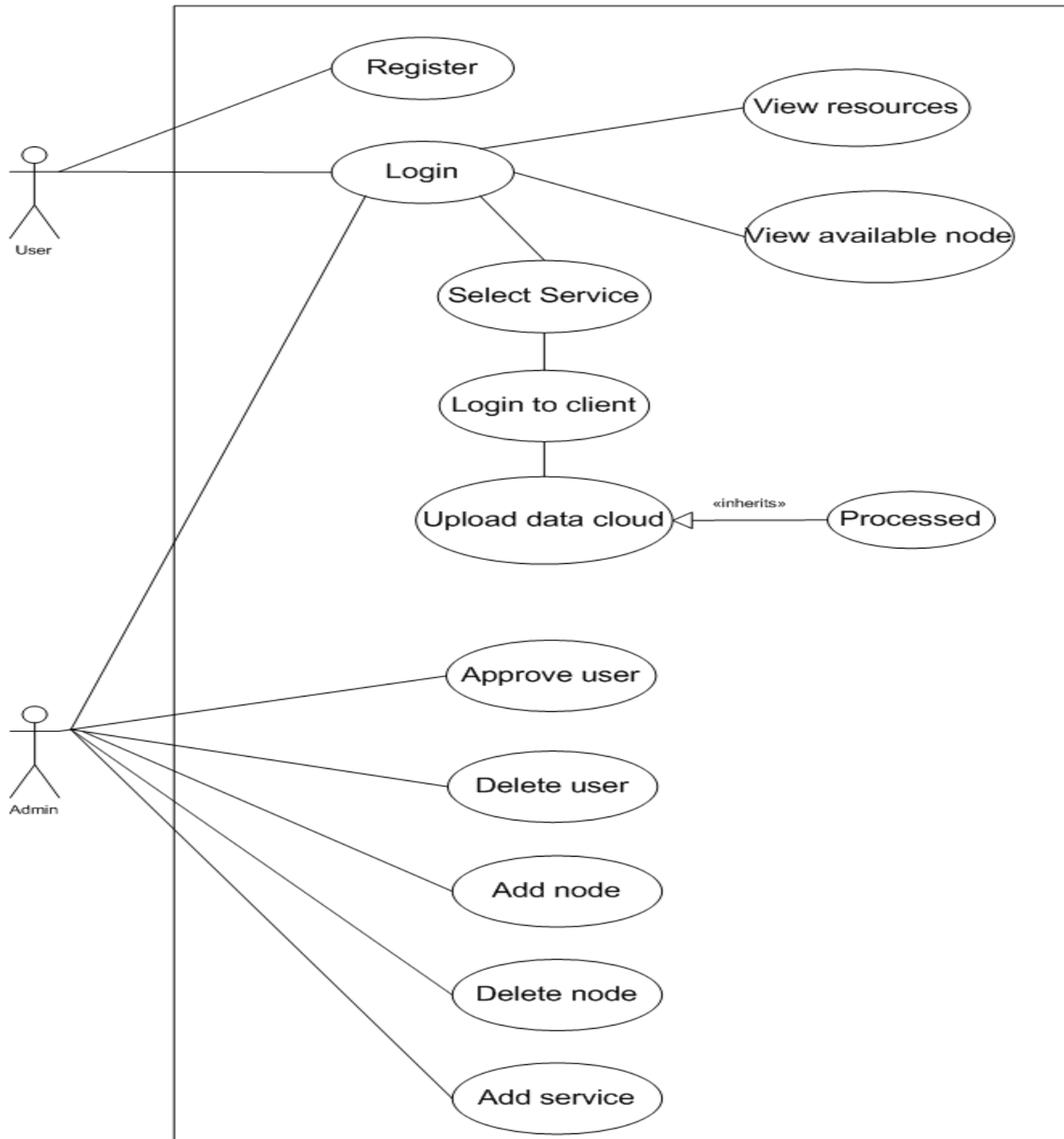


Figure 5: System Use case diagram

4.2.1 Login:

Description and Priority

The user should be allowed to login to the system to use the functionality. The system requires only authenticated users to use it. The proper login mechanism can ensure that no unauthorized person is using the system.

Use Case Paths

Normal:

User logs in successfully.

Exceptional:

User is unable to change his credentials.

Normal Path: User logs in successfully

Externals

User.

Preconditions

The user Portal is up and running, ready to take username and password.

Interactions

- a. The user enters his password and username.
- b. The Portal shall show the user his profile.

Post conditions

Internet is working and portal is running.

Categorization

- a. **Frequency:** High (Daily)
- b. **Criticality:** High
- c. **Probability of Defects:** Low
- d. **Risk:** High

Exceptional path: user is unable to login

Externals

User

Preconditions

The user Portal is up and running, ready to take username and password.

Interactions

- a. The user enters wrong password or username.
- b. Error message shall show saying that password or user name is not correct.
- c. Database errors are displayed.

Post conditions

The User is displayed the log-in screen again with error message “wrong username/password, please enter correct password”.

Categorization

- a. **Frequency:** High (Daily)
- b. **Criticality:** High
- c. **Probability of Defects:** Medium
- d. **Risk:** High

4.2.2 Select the Service:

Description and Priority

System will show the available services. It includes different applications that are installed on the peers (nodes) which can be used by the users. E.g. File Sharing, C++ compiler, image processing etc. It is the core functionality of the system that user must be able to select the offered service.

Use Case Paths

Normal:

Application is selected successfully.

Exceptional:

User is unable to select the application.

Normal Path: User selects the application successfully

Externals

User

Preconditions

- a. The user Portal is up and running.

- b. User log in successfully.

Interactions

1. User will select one of the available nodes
2. User shall click on the service which he/she desires to use.
3. Middleware client for service will be downloaded automatically on the client side.
4. User shall execute middleware client executable.

Post conditions

Middleware client for service will starting executing on the client side.

Categorization

- a. **Frequency:** High (Daily)
- b. **Criticality:** High
- c. **Probability of Defects:** Low
- d. **Risk:** High

Exceptional path: User is unable to select the application

Externals

User

Preconditions

1. The user Portal is up and running.
2. User logs in successfully.

Interactions

- a. User will select one of the available nodes
- b. User shall click on the service which he/she desires to use.
- c. Middleware client does not downloads to the client side.
3. The Portal shall show an error message in case the application is not selected or if application is not available.

Post conditions

An error message with reason and suggestions shall be displayed.

Categorization

- a. **Frequency:** High (Daily)
- b. **Criticality:** High
- c. **Probability of Defects:** Low

d. **Risk:** High

4.2.3 User sends the input data:

Description and Priority

The user should be allowed to enter his data and send it to the cloud .It is the core functionality of the system that user must be able to enter his data. When user logs into the website he/she is shown with the table containing the IP's and ports of nodes. On downloading the client middleware the user enters the contents of the table on the client side.

Use Case Paths

Normal:

Data is sent successfully.

Exceptional:

User is unable to enter or send the data.

Normal Path: User enters and sends the data successfully

Externals

User

Preconditions

- a. The user Portal is up and running.
- b. User log in successfully.
- c. Middleware client for service is downloaded on the client side and running.

Interactions

1. User will use the application by entering his/her code or any other input data.
2. User will press the submit button and data is sent to the cloud.
3. Application will process that data on the allocated node and output is sent back to the client.

Post conditions

System will show the processed data as output to the user.

Categorization

- a. **Frequency:** High (Daily)

- b. **Criticality:** High
- c. **Probability of Defects:** Low
- d. **Risk:** High

Exceptional path: User is unable to enter or send the data

Externals

User

Preconditions

- a. The user Portal is up and running, ready to take username and password.
- b. User log in successfully.

Interactions

1. User will use the application by entering his/her code or any other input data.
2. User will press the submit button and data is sent to the cloud.
3. Application will not return any output.

Post conditions

An error message with reason and suggestions shall be displayed.

Categorization

- a. **Frequency:** Medium (Monthly)
- b. **Criticality:** High
- c. **Probability of Defects:** Low
- d. **Risk:** High

4.2.4 View node Information:

Description and Priority

The user will be able to see the number of node available on the home screen. CPU usage and memory available on different nodes are shown to the users.

Use Case Paths

Normal:

User can see the information of available nodes.

Exceptional:

User is unable to see the available nodes.

Normal Path: User can see the node information

Externals

User

Preconditions

- a. The portal is up and running.
- b. User log in successfully.

Interactions

1. The User login to home screen and can see the following node information:

The number of nodes available.

- a. Different services available on the nodes.
- b. CPU usage of nodes.
- c. Memory available on nodes

Post conditions

System will show the node information.

Categorization

- a. **Frequency:** Medium (Monthly)
- b. **Criticality:** High
- c. **Probability of Defects:** Low
- d. **Risk:** High

Exceptional Path: User can't see the node information

Externals

User

Preconditions

- a. User can't log in successfully.
- b. An error message with reason and suggestions shall be displayed.

Interactions

1. User will use the login menu by entering his/her username and password.
2. User will press the login button and data is sent server.
3. Login error is occurred, user will be redirected to the user page.

Post conditions

- a. The User is displayed the log in screen again with error message “wrong username/password, please enter correct password”
- b. Nodes are not are not shown as user fails to login to his/her home screen.

Categorization

- a. **Frequency:** High (Daily)
- b. **Criticality:** High
- c. **Probability of Defects:** Medium
- d. **Risk:** High

4.2.5 Approve new Account:

Description and Priority

The COPTPN system shall have more than one User and Administrator .The system requires only authenticated users to give access to the system and authorized administrators to access the Administration Portal. System administrator will approve the users using this feature.

Use Case Paths

Normal:

- a. User approval request is pending.
- b. Administrator approves new account successfully.

Normal Path: Administrator approves a new account successfully

Externals

Administrator

Preconditions

- a. The Administration Portal is up and running.
- b. User is logged in Administrator.
- c. The option to approve users is available.

Interactions

1. The Administrator selects the button to approve a new user/admin.
2. A grid is displayed to administrator to see the credentials of the user.
3. A checkbox is available to make the user an administrator.

4. The Administrator approves the information and the Portal displays the message for successful addition.
5. The Administrator can disable an account.

Post conditions

The home screen of the Administration Portal is displayed.

- a. **Categorization**
- b. **Frequency:** Low (Monthly)
- c. **Criticality:** High
- d. **Probability of Defects:** Low
- e. **Risk:** Medium

4.2.6 Add new node

Description and Priority

The COPTPN system shall have multiple nodes .The Administrator can add new nodes if the Middleware Software is installed on the computer then that particular node can be added into the system.

Use Case Paths

Normal:

Administrator adds new node.

Normal Path: Administrator adds a new node successfully

Externals

Administrator

Preconditions

- a. The Administration Portal is up and running.
- b. The Middleware software is installed on the particular computer to be added.

Interactions

1. The Administrator selects the button to add a new node.
2. The Portal displays the message for successful addition.

Post conditions

The home screen of the Administration Portal is displayed.

Categorization

- a. **Frequency:** Low (Monthly)
- b. **Criticality:** High
- c. **Probability of Defects:** Low
- d. **Risk:** Medium

Exceptional Path: Administrator can't add a new node

Externals

Administrator

Preconditions

The Administration Portal is up and running and the Middleware software is not installed on the particular computer to be added.

Interactions

1. The Administrator selects the button to add a new node.
2. The Portal displays the message that the said computer doesn't have the required middleware for successful addition.

Post conditions

The home screen of the Administration Portal is displayed.

Categorization

- a. **Frequency:** Low (Monthly)
- b. **Criticality:** High
- c. **Probability of Defects:** Low
- d. **Risk:** Medium

4.2.7 Delete a node:

Description and Priority

The COPTPN system shall have multiple nodes .The Administrator can delete a node if it is not working properly or to replace it with some other node.

Use Case Paths

Normal:

Administrator deletes a node.

Normal Path: Administrator deletes a node successfully

Externals

Administrator

Preconditions

The Administration Portal is up and running.

Interactions

1. The Administrator selects the button to delete a node.
2. The Portal displays the message for successful deletion.

Post conditions

The home screen of the Administration Portal is displayed.

Categorization

- a. **Frequency:** Low (Monthly)
- b. **Criticality:** High
- c. **Probability of Defects:** Low
- d. **Risk:** Medium

4.2.8 Setup a Service:

Description and Priority

The COPTPN system shall have multiple services .The Administrator can add service through middleware addition. We add the service by the middleware configuration and not through the website. We setup the service at middleware level.

Use Case Paths

Normal:

Administrator adds new service.

Normal Path: Administrator adds a new service successfully

Externals

Administrator

Preconditions

The Administration Portal is up and running and the Middleware is compatible to add new service.

Interactions

1. The Administrator configures the setting at middleware like port setting at middleware server.
2. The Portal displays the message for successful addition of service.

Post conditions

The home screen of the Middleware server is displayed.

Categorization

- a. **Frequency:** Low (Monthly)
- b. **Criticality:** High
- c. **Probability of Defects:** Low
- d. **Risk:** Medium

Exceptional Path: Administrator can't add a new service

Externals

Administrator

Preconditions

The Administration Portal is up and running and the Middleware setting are not configured properly.

Interactions

1. The Administrator selects to add new service.
2. The Portal displays the message that the said service doesn't have the required middleware configuration or the service is not compatible.

Post conditions

The home screen of the Middleware server is displayed.

Categorization

- a. **Frequency:** Low (Monthly)
- b. **Criticality:** High
- c. **Probability of Defects:** Low
- d. **Risk:** Medium

4.3 Deployment Diagram:

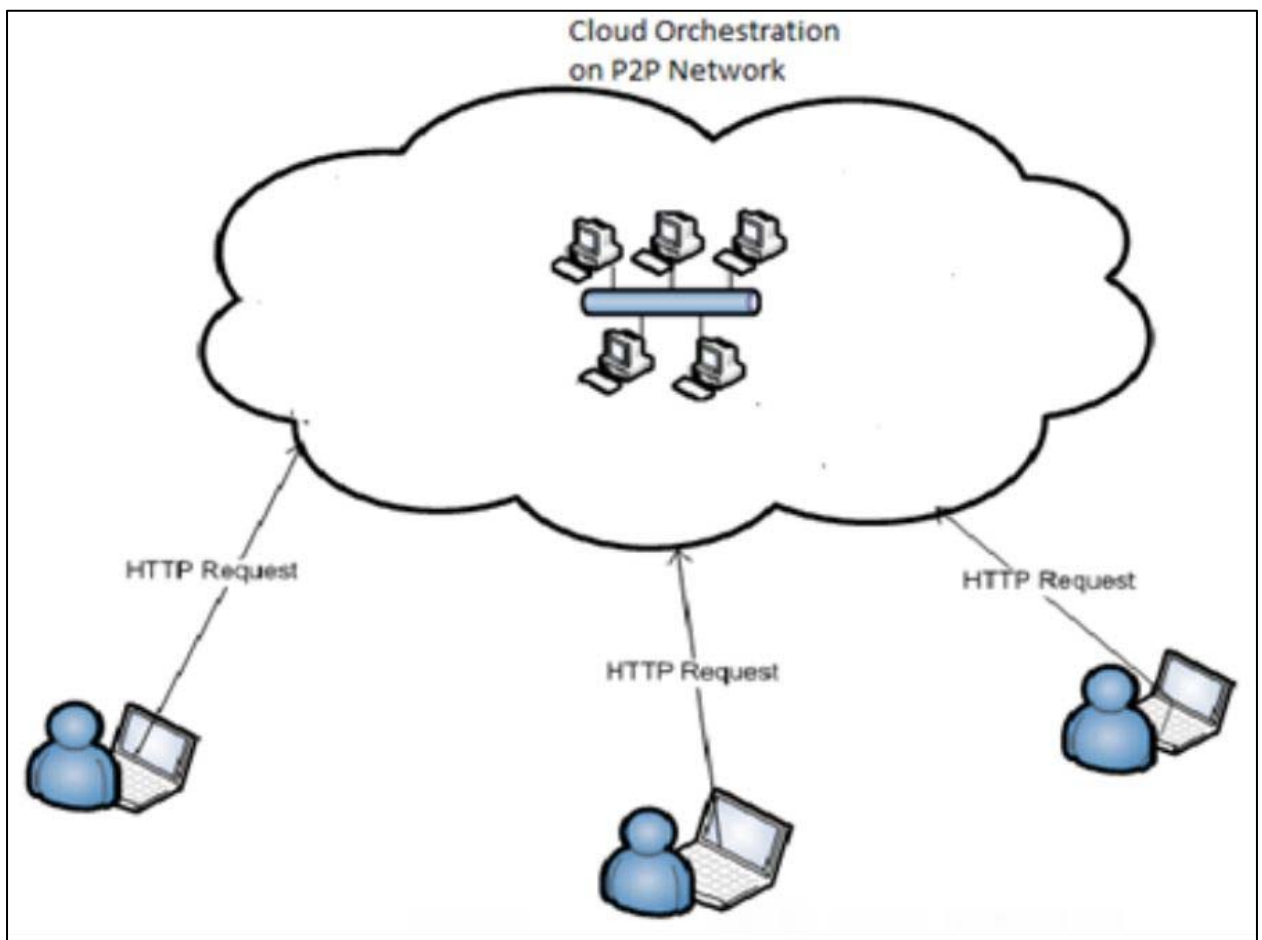


Figure 6: Deployment diagram of the system

The deployment diagram provides a physical look at the system with each processor and device indicated. This provides a background for the rest of the document as no software component can straddle two physical locations. Each physical location will have its own

software unit and units in different physical locations will collaborate to provide the services that logically seem to be straddling the units.

In our system, there is a WLAN on which all the systems are connected and they collaborate with each other to act as a single unit for user.

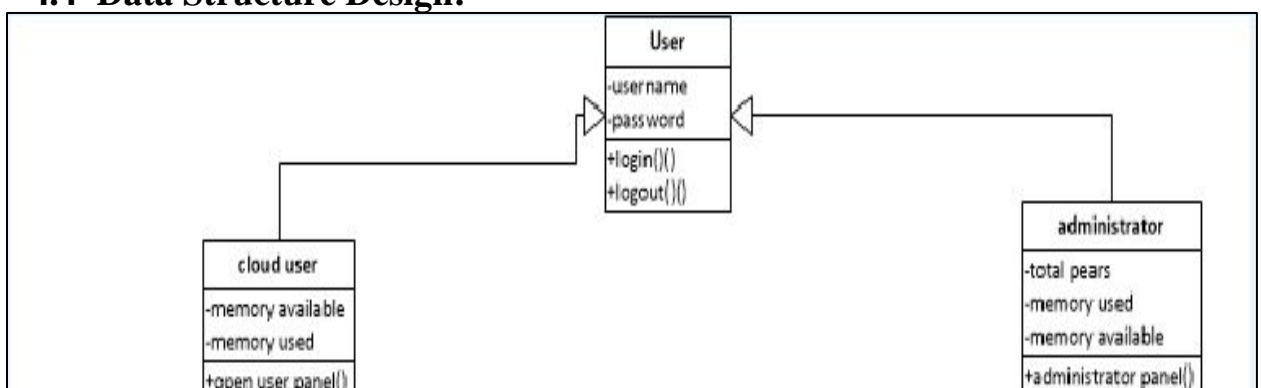
The Private cloud consists of Core i5, 64-bit laptop machines that are configured with a middleware. These machines are connected via wifi. The middleware gels the machines into one single processing unit with combined physical resources and power. These “cloud servers” also host the databases to be used for the project. One of the machines shall be set as the interface machine to which the outside world shall connect. However it shall appear no different than the other laptops, and to the outside world all machines would appear as one.

Lastly, the Administration of the cloud shall be done remotely using a laptop computer that shall also connect to the rest of the system via WLAN.

User and the Administrator are the only two logical users of this system. The User is supposedly an owner of a memory unit and has access to the services of the cloud through web browser. The Administrator on the other hand, is a technical person and has full privileges to change all aspects of the system at will. He is responsible for maintenance and controls the system via his Administration panel on his laptop. He has put in place the protocols for the network and has defined the data in/outflow mechanisms.

Note that the cloud only allows certain people to access its services, using a username and password.

4.4 Data Structure Design:



4.5 Activity Diagram

4.5.1 Activity Diagram for User

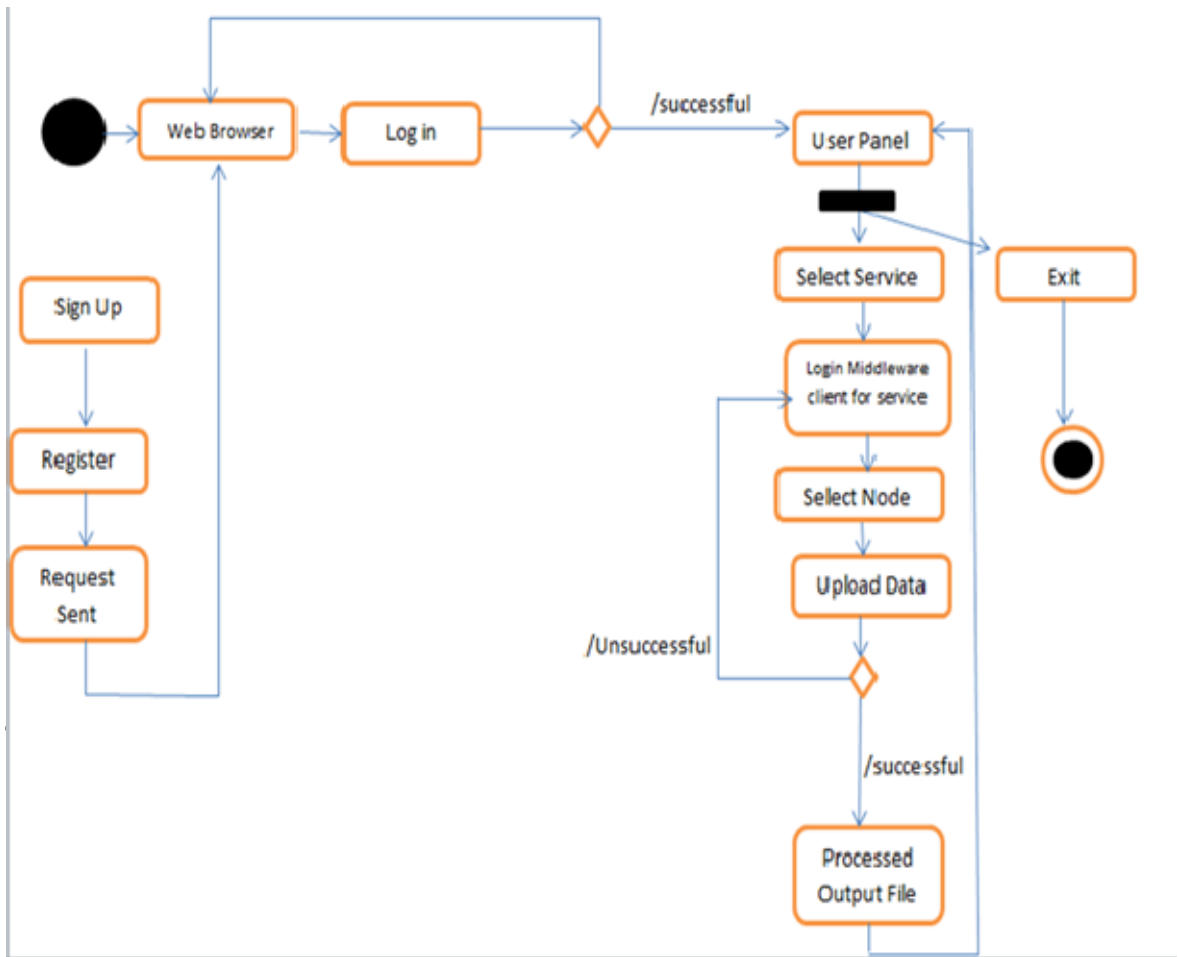
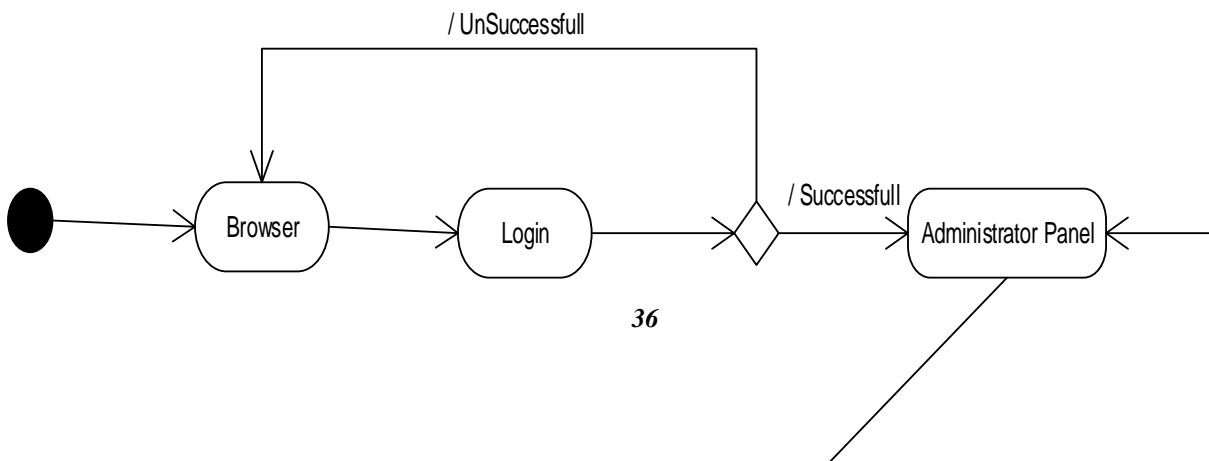
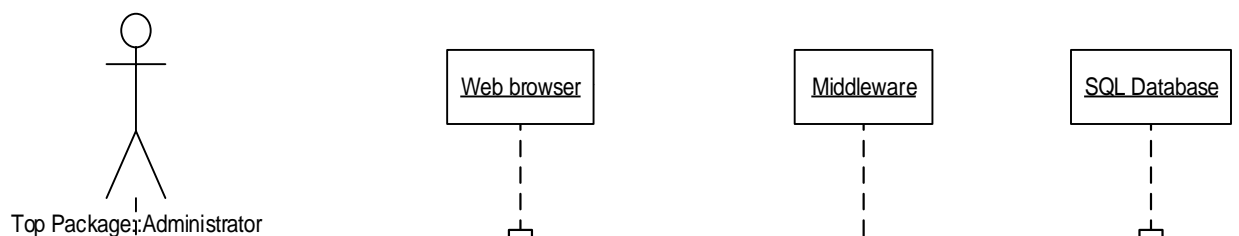


Figure 8: Activity diagram for the user



4.6 Sequence Diagram

4.6.1 Admin Login



4.6.2 Admin adding a Node

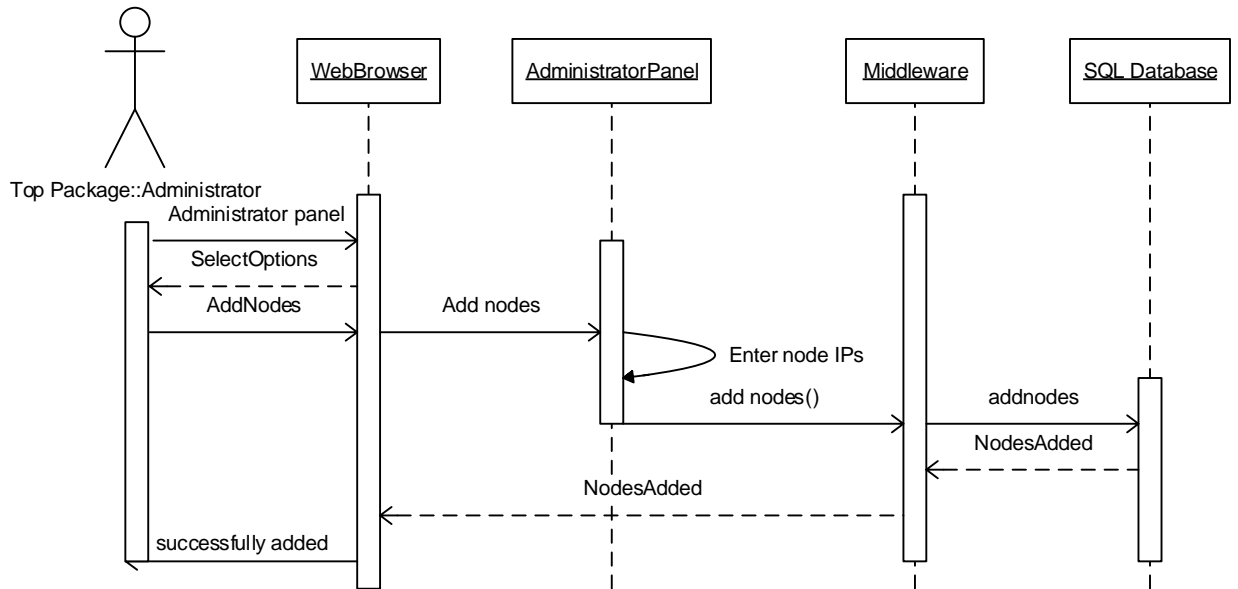
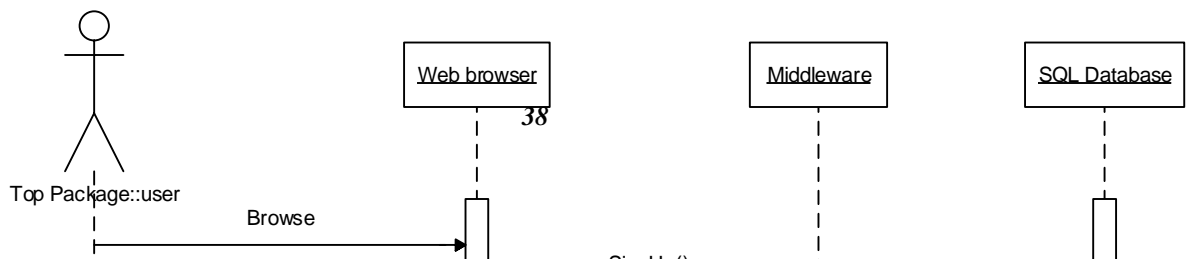


Figure 11: Sequence diagram of Admin adding a node

4.6.3 User Signup



4.6.4 User Login

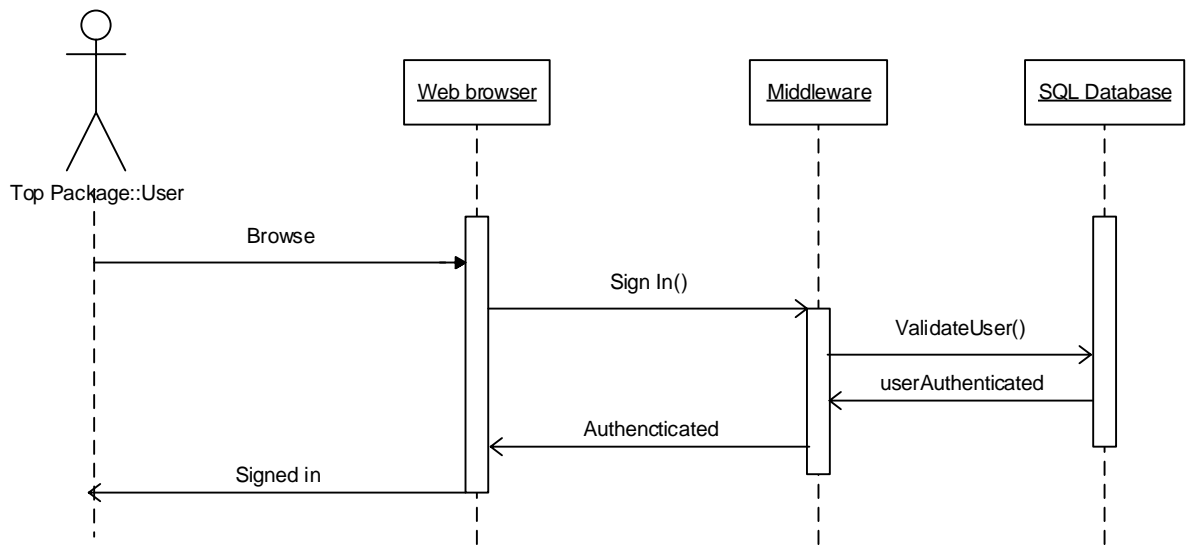
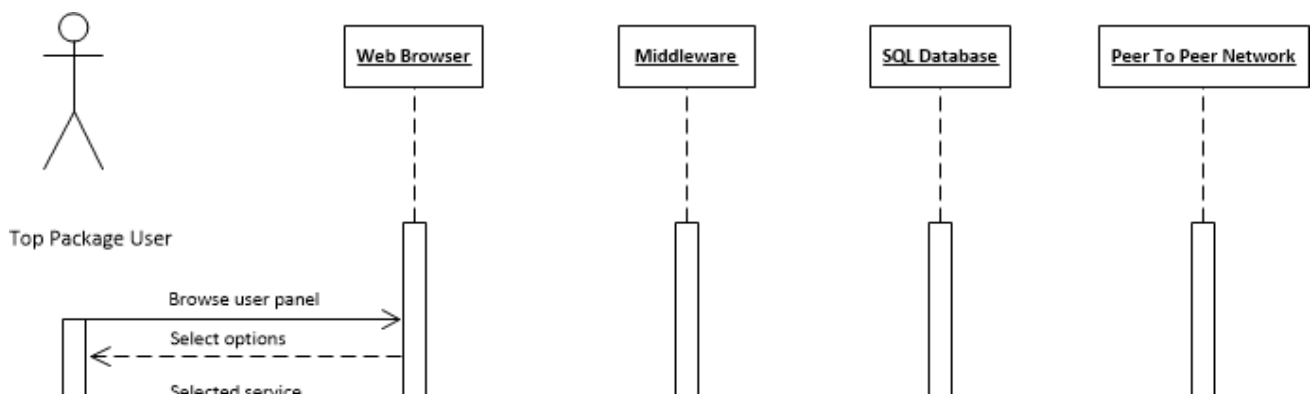


Figure 13: Sequence diagram of User Login

4.6.5 Selecting a Service



CHAPTER 5

CHAPTER 5 IMPLEMENTATION

5. Tools and Technologies

5.1 Microsoft Visual Studio 2008

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop console and graphical user interface applications along With Windows Forms applications, web sites, web applications, and web services in both active code together with managed code for all platforms supported by Microsoft Windows, Windows Mobile, Windows CE, .NET Framework, .NET Compact Framework and Microsoft Silverlight. Visual Studio supports different programming languages by means of language services, which allow the code editor and debugger to support (to varying degrees) nearly any Programming language, provided a language-specific service exists. Built-in languages Include C/C++ (via Visual C++), VB.NET (via Visual Basic .NET), and C # (via Visual C #) And F#. We have implemented our system in C# and ASP.NET.

5.2 Microsoft SQL Server

Microsoft SQL Server is a relational database management system developed by Microsoft. As a database, it is a software product whose primary function is to store and retrieve data as requested by other software applications, be it those on the same computer or those running on another computer across a network (including the Internet).we have used Microsoft SQL Server to design our database.

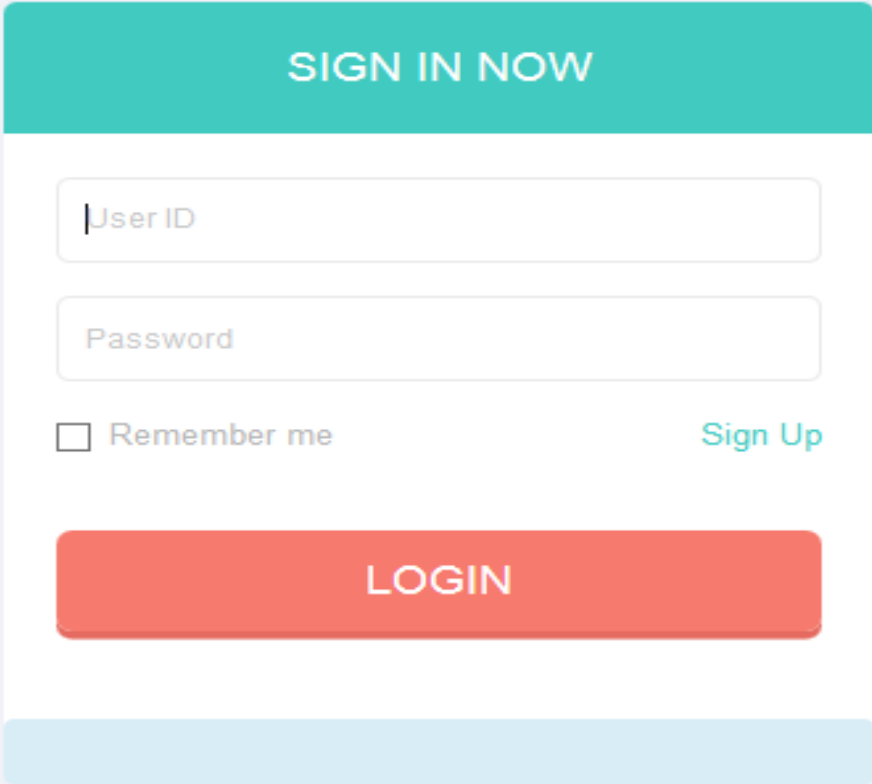
CHAPTER 6

SYSTEM IMPLEMENTATION

6. System Implementation

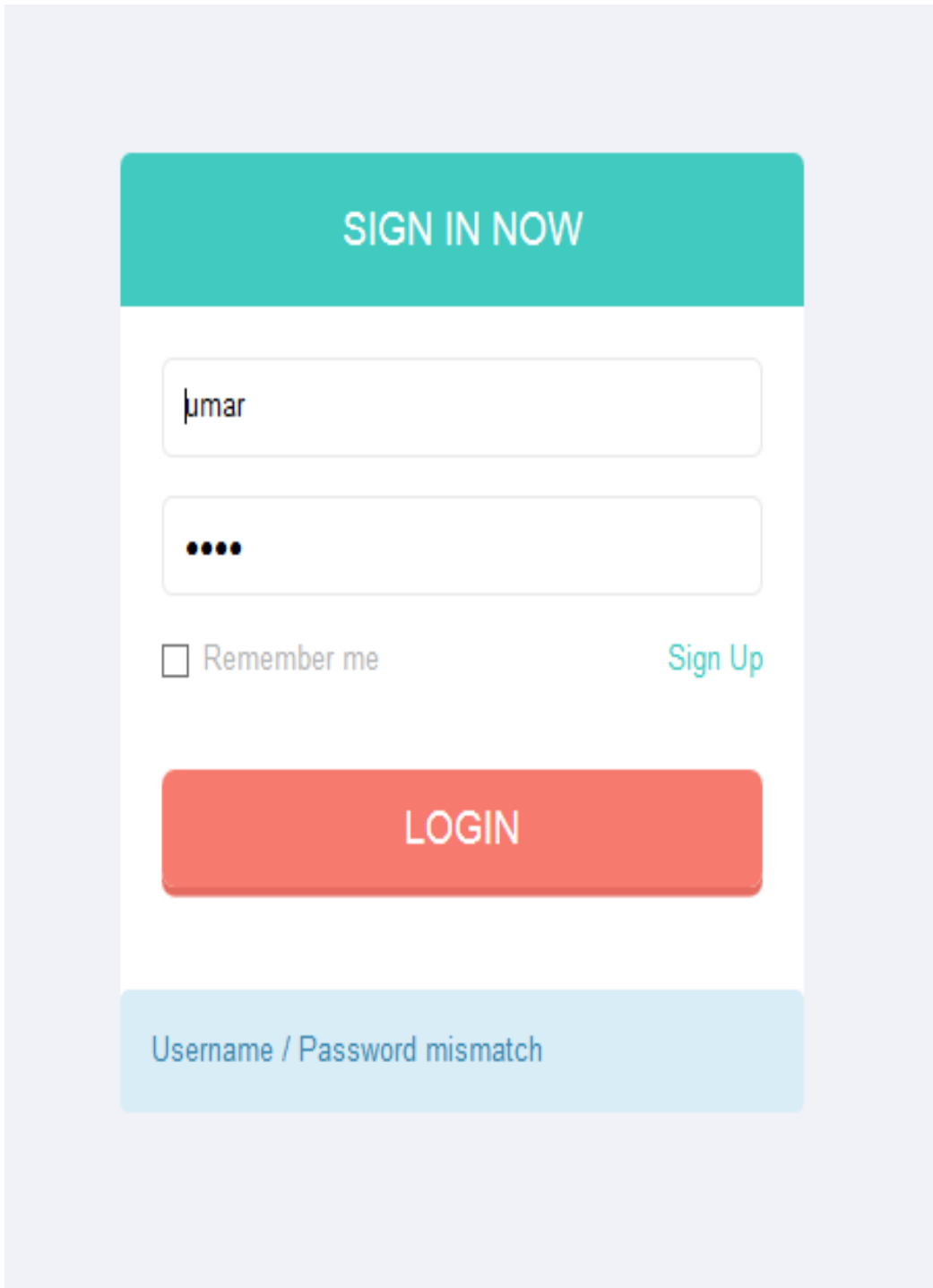
6.1 User Login

Our system has a login page where registered users can login. The users have to enter their user name and password. The system requires only authenticated users to use it. The proper login mechanism can ensure that no unauthorized person is using the system. Figure Below shows the login Page.



The screenshot displays a login interface with a teal header containing the text "SIGN IN NOW". Below the header, there are two input fields: "User ID" and "Password". To the left of the "Password" field is a checkbox labeled "Remember me". To the right of the "Remember me" checkbox is a teal link labeled "Sign Up". At the bottom of the form is a large red button with the text "LOGIN".

Screenshot 1: Showing system Login Screen



Screenshot 2: Showing system accepts the username and password

```

if (loginfield.Text.ToString() != "" &&password.Text.ToString() != "")
    {
DataSet result = model.LoginCheck(loginfield.Text, password.Text);

intUserid = int.Parse(result.Tables[0].Rows[0]["ID"].ToString());
if (Userid> -1)
    {
error.InnerText = "Login Successful";
Session["usersid"] = Userid;
Session["userstype"] = result.Tables[0].Rows[0]["type"].ToString();
string Approved = result.Tables[0].Rows[0]["approved"].ToString();
if (Approved.Contains("not"))
    {
error.InnerText = "Your account has not been approved, please try later or contact admin";
return;
    }
string Active = result.Tables[0].Rows[0]["active"].ToString();
if (Active.Contains("not"))
    {
error.InnerText = "Your account has not been disabled, please try later or contact admin";
return;
    }
Session["users"] = result.Tables[0].Rows[0];
Page_Load(sender, e);
System.Threading.Thread.Sleep(1000);
if(Session["userstype"].ToString()!="admin")
Response.Redirect("homepage.aspx");
else
Response.Redirect("adminhome.aspx");
    }
else
    {
error.InnerText = "Username / Password mismatch"
    }
}

```

Screenshot 3: Showing code for login screen

6.2 Select the Service

System will show the available services. It includes different applications that are installed on the peers (nodes) which can be used by the users. E.g. File Sharing, C++ compiler etc. It is the core functionality of the system that user must be able to select the offered service.



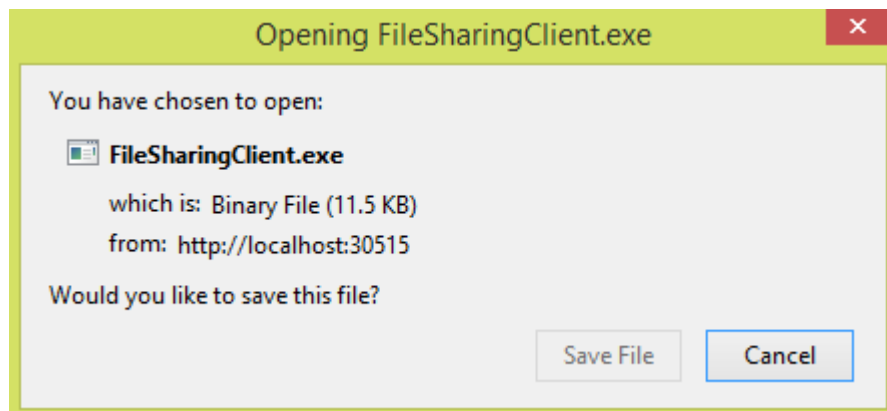
Request Service	ID	Name	Path	NodeID
<input checked="" type="checkbox"/>	1	Matlab	C:\Matlab	1
<input checked="" type="checkbox"/>	2	Winrar Compression	C:\program file(x86)\winrar\rar.exe a	-1

Screenshot 4: Showing services offered by the system

6.3 User sends the input data

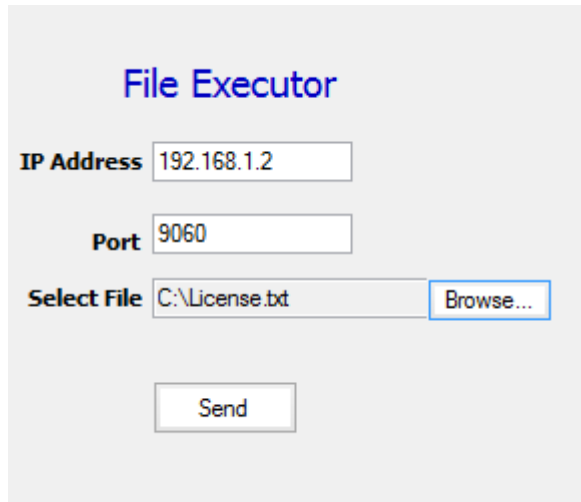
The user should be allowed to enter his data and send it to the cloud. It is the core functionality of the system that user must be able to enter his data *and send it to the cloud*. User will upload his file onto the client application and send it to cloud for processing. User will press the submit button and data is sent to the cloud.

The client exe file which is downloaded onto the user computer is shown below.



Screenshot 5: Showing Middleware client for service executable being download to the client side

The user selects the file containing the input data.



Screenshot 6: Showing user interface of the Middleware client for service

6.4 View Node Information

The user will be able to see the number of node available on the home screen. Different CPUs available for new operations, current CPU usage and the services available on different nodes.

All Nodes									
	ID	NodeName	NodeIP	Memory	Processor	Harddisk	Network	Active	datetime
<input checked="" type="checkbox"/>	1	home	192.168.20.30	100	100	100	100	active	1/1/2014 12:00:00 AM
<input checked="" type="checkbox"/>	2	office	192.168.20.31	1	1	1	1	active	1/1/2014 12:00:00 AM
<input checked="" type="checkbox"/>	3	university	192.168.20.50	2	3.5	500	100	active	4/9/2014 6:43:40 AM

Screenshot 7: Showing Nodes available to the user

6.5 Approve New Account

The COPTPN systemization shall have more than one User and Administrator .The system requires only authenticated Users to give access to access the system and authorized administrators to access the Administration Portal. The User and Administrator can be under check and security loopholes are minimized. System administrator will approve the user using this feature.

Figure shows the list of pending user waiting for approval by the administrator.

All Pending Users												
Approve User	ID	name	email	nustid	password	username	course	approved	active	servicerequested	type	Node
<input checked="" type="checkbox"/>	2002	hamza	hamza@mcs.edu.pk	2010-NUST-BECSE-250	hamzi	hamzi	BESE 16	not approved	active	none	user	

Screenshot 8: Showing all pending users

6.6 Add a new node

The COPTPN systemization have multiple 2-3 nodes .The Administrator can add new nodes if the Middleware Software is installed on the computer then that particular node can be added into the system.

6.7 Setup a service

The COPTPN systemization can have multiple services .The Administrator can add service through middleware addition. We add the service by the middleware configuration. For this administrator definite the name of the service and add the path of the executable.

```
Public partial class AddService :System.Web.UI.UserControl
{
protected void Page_Load(object sender, EventArgs e)
{
}
}
ModelLayer model = new ModelLayer();
protected void addSrcv_Click(object sender, EventArgs e)
{
model.sp_Services_Insert(Name.Text, path.Text);
error.InnerText = "Service added successfully";
Page_Load(sender, e);
}
}
```

Screenshot 9: Showing code to add new service

6.8 Terminating the Application

After using the required application, user will terminate the Application.

CPU usage, memory consumption and user tables will be updated. Application will be terminated when the user will log out. System will release all the resources being used by that application. CPU usage, memory consumption and user tables will be updated.

CHAPTER 7

TESTING AND RESULTS

7. Testing and Results

7.1 Introduction to test Plan

This software test plan is to provide the description of test cases for COPTPN (Cloud Orchestration on Peer to Peer Network) describing the scope and approach of intended test activities. This document will describe the test cases for different features of the system such as user login, users' management, resource management, assigning user request to the selected node etc.

Software Requirement and Specification document of COPTPN supports this Software Test Plan.

Test Items

Following are test items and their version:

Test Item Name	Test Item Version Number	Test Type
Overall System	Ver 1	Blackbox
User Registration	Ver 1	Blackbox
User Login	Ver 1	Blackbox
Users' Management	Ver 1	Blackbox
Nodes Management	Ver 1	Blackbox
Services Management	Ver 1	Blackbox
Resources Calculation	Ver 1	Blackbox
Executing Users' Request	Ver 1	Blackbox
Overall System	Ver 1	Blackbox

Table 1: Test items

7.2 Features to be tested

Feature	Parent Component/System	Overview
Overall System Working	COPTPN	a. Run the Server. b. Open http website on

		<p>client.</p> <ul style="list-style-type: none"> c. Login the system. d. Select the node and service. e. Perform task. f. Logout the system.
User Registration	Web Application	<ul style="list-style-type: none"> a. Run the Server b. Open http website on client. c. Open Signup form. d. Fill-in the credentials. e. Submit the form. f. Check the status.
User Login	Web Application	<ul style="list-style-type: none"> a. Open the website. b. Enter username and password. c. Check the status.
Users' Management	Databases	<ul style="list-style-type: none"> a. Open the website. b. Login as administrator. c. Check all the registered users are shown along with their status. d. Edit the rights of the users.
Nodes Management	Database, Middleware	<ul style="list-style-type: none"> a. Open the website. b. Login as administrator.

		<ul style="list-style-type: none"> c. Add the details of a node. d. Check the status. e. Remove a node. f. Check the status.
Services Management	Database, Middleware	<ul style="list-style-type: none"> a. Open the website. b. Login as administrator. c. Add/delete service to/from a node. d. Check the status.
Resources Calculation	Database, Middleware	<ul style="list-style-type: none"> a. Open the website. b. Login as administrator. c. Checks the resources d. Select a service and give input to the system. e. Check the resources again. f. Check the resources once again when job is done.
Executing Users' Request	Database, Middleware	<ul style="list-style-type: none"> a. Open the website. b. Login as a client. c. Select the available node and service required. d. Download the

		<p>middleware client for service.</p> <p>e. Execute the middleware client.</p> <p>f. Login to middleware client.</p> <p>g. Give inputs and submit the request.</p> <p>h. Check the results.</p>
--	--	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Table 2: Features to be tested

7.3 Features Not to Be Tested

All the features of COPTPN need to be tested.

7.3.1 Approach

The overall approach of this test plan is Gray Box testing i.e. hybrid of Black Box and White Box testing. Testing approach will be same for all features of the system. System will be tested in different perspectives such as GUI testing etc.

No separate tool will be used to test the system. Code will be reviewed on Microsoft Visual Studio 2012. Testing will be comprehensive so that to ensure high quality of the system. Minimum degree of comprehensive required is that all features should be tested at least once. To judge the comprehensiveness a record will be maintained of the features which have been tested completely.

7.3.2 Item Pass/Fail Criteria

The entrance criteria's for each phase of testing must be met before the next phase can commence. Any test item will be declared pass if it conforms to the requirements specified in the SRS and fail if it does not.

7.3.3 Suspension Criteria and Resumption Requirements

The only case in which test activity needs to be suspended is failure of a feature. In that case the development team will be reported to fix the error of that feature after which the testing of the whole system will start over again.

7.3.4 Environmental Needs

Windows based system is required for testing of this system. Microsoft Visual Studio 2013 and .NET Framework 4.5 is required to run the software.

7.3.5 Schedule

Testing process has taken 1 month and 3 days

7.3.6 Risks and Contingencies

Risk lies in incomplete execution of test cases and ignoring little bugs of the systems which may result in big faults and failures. Also the testing process needs to be done in the specified time. If not, the whole schedule of development team may get disturbed resulting in lack of customer's trust on development team.

7.3.7 Functional Testing (Black box)

The software program or system under test is viewed as a "black box". The selection of test cases for functional testing is based on the requirement or design specification of the software entity under test. Functional testing emphasizes on the external behavior of the software entity.

7.4 Test Cases

7.4.1 Test Case no. A01

Name: User Registration.

Description: It will test registration of a user with distinct username.

Precondition: System is correctly installed, bug free and working properly.

Input Values: End user have to open the website and signup form.

Steps:

1. Open the website.
2. Press the signup button
3. Now fill up the registration form.
 - a. User name must be identical.
 - b. All the fields must be filled.
 - c. Submit the form.

Expected Output: If username is identical and all the fields are filled correctly then the form should be submitted and user must be registered. And if username is not identical or any field is unfilled or filled wrongly then error message should appear asking user to fill the form correctly.

Output: Output was same as expected.

Result: Pass

7.4.2. Test Case no. A02

Name: User Login

Description: It will test registered user logs in successfully into the system.

Precondition: System is correctly installed, bug free, working properly and user is registered.

Input Values: End user has to enter a valid username and correct passwords.

Steps:

1. Open the website.
2. Enter username and password.

Expected Output: If the username and password are correct then user should login the system. Otherwise error message should be shown.

Output: Output was same as expected.

Result: Pass

7.4.3. Test Case no. A03

Name: Users' Management

Description: It will test users' management by the administrator.

Precondition: System is correctly installed, bug free, working properly, user in logged in from the administrator account.

Input Values: Administrator must click on the users' management button.

Steps:

1. Open the website.
2. Login as administrator.
3. Check all the registered users are shown along with their status.
4. Approve the users' requests (if present).
5. Delete a user.

Expected Output: All the registered users should be shown along with those whose approval is required. Any user can be given the status of administrator. If administrator selects the option "Delete user" then that specific user should be deleted.

Output: System responded as expected.

Result: Pass

7.4.4. Test Case no. A04

Name: Nodes Management

Description: It will test nodes management by the system.

Precondition: System is correctly installed, bug free, working properly and user is logged in as administrator.

Input Values: administrator selects the nodes management button.

Steps:

1. Open the website.
2. Login as administrator.
3. Add the details of a node.
4. Check the status.
5. Remove a node.
6. Check the status.

Expected Output: Node should be added into the system and shown in the database when administrator adds it and it should be deleted when administrator selects the option “delete node”.

Output: System behaved as expected

Result: Pass

7.4.5. Test Case no. A05

Name: Services Management.

Description: It will test services management by the system.

Precondition: System is correctly installed, bug free, website is running properly and user is logged in as administrator.

Input Values: End user has to select option “Services”.

Steps:

1. Open the website.
2. Login as administrator.
3. Add service to a node.
4. Check the status.

Expected Output: Service should be added/deleted to/from a node as selected by the administrator.

Output: For step 3 the output was the same as expected.

Result: Pass

7.4.6. Test Case no. A06

Name: Resources Calculation.

Description: It will test calculation of available resources.

Precondition: System is correctly installed, bug free, website is running properly and user is logged in.

Input Values: End user has to login the system.

Steps:

1. Open the website.
2. Login the system.
3. Checks the resources timeline.

Expected Output: System should show all the available resources.

Output: For step 3 the output was the same as expected.

Result: Pass

7.4.7. Test Case no. A07

Name: Executing User's Request

Description: It will test the execution of user's request.

Precondition: System is correctly installed, bug free, website is running properly.

Input Values: End user has to select a node from the available nodes and service offered on that node.

Steps:

1. Open the website.
2. Login as a client.
3. Select the available node and service required.
4. Download the middleware client for service.
5. Execute the middleware client.
6. Login to middleware client.
7. Give inputs and submit the request.
8. Check the results.

Expected Output: System should accept the input from the user and process that on the selected node.

Output: Output was same as expected.

Result: Pass.

7.4.8. Test case no. A08

Name: System overall function.

Description: It will test the overall functionality of the system and the working of the System.

Precondition: System is correctly installed, bug free and working properly.

Input Values: Enter the port number in the server.

Steps:

1. Run the COPTPN i.e.
 - a. Enter the proxy IP address.
 - b. Enter the port number
2. Now check the text boxes for invalid inputs i.e.
 - a. In the textbox dealing with numbers in regex of IP address write alphabets and special characters in it.
 - b. Port number is integer and within the range of 4000-65000
 - c. Use SQL injection attacks for exploit the data.

Expected Output: System should work properly as a proxy server.

Output: For step 1 output is same as expected. For step 2 Output is same as expected as the text box won't allow entering invalid input.

Result: Pass

7.5 Testing of non-functional features:

7.5.1 Test Case no. B01

Name: Web Interface

Description: It will test if all the hyperlinks provided in the website are responding.

Precondition: The user requests for the website.

Input Values: Navigation to all the shown hyperlinks by user.

Steps:

1. Open the website.

2. Navigate through all the shown hyperlinks.
3. For all the hyperlinks, a new page or tab must open.

Expected Output: All the hyperlinks open a meaningful page and none of the hyperlinks is dead.

Output: For step 3 the output was the same as expected.

Result: Pass.

7.5.2 Test Case no. B02

Name: Web Interface

Description: It will test if the web interface is user friendly.

Precondition: The user requests for the website.

Input Values: Navigation to all the shown hyperlinks by user.

Steps:

1. Open the website.
2. Navigate through all the shown hyperlinks.
3. For all the hyperlinks, a new page or tab must open.
4. The website must show the sitemap.
5. The user must not find it difficult to find any item or page and the website should be self-explanatory.
6. Color scheme of the website is soft and not disturbing for eyes of the user.

Expected Output: The user find should find it easy to navigate through and navigate back to different pages and the interface should be user friendly.

Output: For step 4, 5, 6 the output was the same as expected.

Result: Pass

7.5.3 Test Case no. B03

Name: Remote access to website

Description: It will test if the administrator can access the website from any system on the network.

Precondition: The administrator deploys COPTPN on the network.

Input Values: User requests for the website from a remote computer (other than server) on the network.

Steps:

1. The user makes a request for COPTPN website from a remote computer.
2. The website opens and administrator can login.
3. The administrator can change the user rights, add nodes, delete nodes, add services, delete services, view all registered users.

Expected Output: The administrator should be able to access the website and he should be able to change the user rights, add nodes, delete nodes, add services, delete services, view all registered users and available resources.

Output: For step 3, 4 the output was the same as expected.

Result: Pass.

7.6 Integration Testing

1. Initially hardware modules and Application were tested for integration.
2. Then Web service module was integrated and tested to see if all the data was flowing properly.

7.7 System Testing

System testing was performed at the end of development. Complete system was tested in different inputs with different conditions to verify that those conditions do not disrupt the performance of the system and then testing the whole system for performance and other attributes (failures, response delays etc.).

CHAPTER 8

CONCLUSION AND FUTURE WORK

8. Conclusion

COPTPN is a small scale cloud of desktop computers which is being used to provide its processing abilities for the users of the college. This would be a private cloud where students and faculty members can register themselves and can use its processing. The desktop computers would be using a peer to peer protocol and a middleware should be deployed on each of the computer which manages the resources inside the cloud. The

middleware should be responsible for allocating user request on cloud on the basis of CPU consumption of the Peers. The administrator of this private cloud should be responsible for authenticating a user and removing him/her when necessary.

The main driving force behind the creation of a cloud from desktop computers instead of a traditional server was that this cloud provides reusability of resources and this is very cheap and can be implemented easily in working environments like, schools, colleges, universities and offices.

COPTPN can be deployed in the CSE-lab. By doing so peer to peer network will have more peers and number of users can be increased and via LAN it can be made accessible to users residing in the hostels i.e. Iqbal Coy and Jinnah Coy

CHAPTER 9

USER MANUAL

9. USER MANUAL

9.1 Reading Instructions

This Manual is a guide to the system “Cloud Orchestration on Peer to Peer Network (COPTPN)”

It contains essential instructions for setup and operations.

The system provides a user friendly interface which allows you to directly interact and monitor the system.

This Manual should be read in the order given.

9.2 Installation

.NET framework must be installed in the system before installing COPTPN. It can be downloaded from the following link:

<http://www.microsoft.com/en-pk/download/details.aspx?id=17718>

COPTPN has three main components:

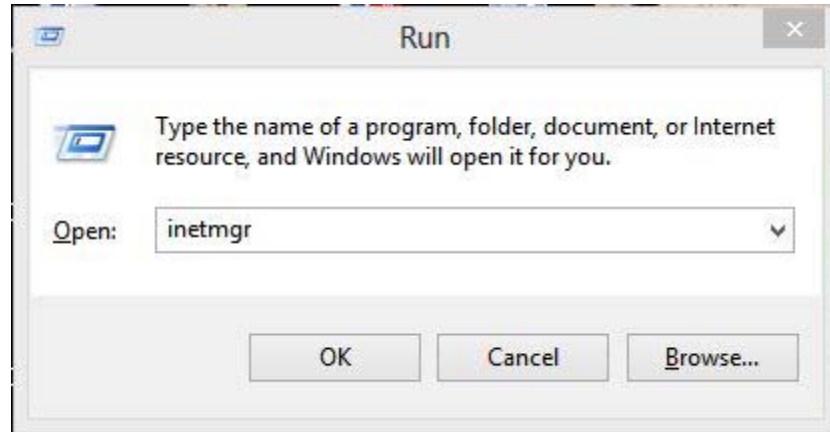
- i. Website
- ii. Middleware client
- iii. Middleware server

Installing instructions of these components are as follow:

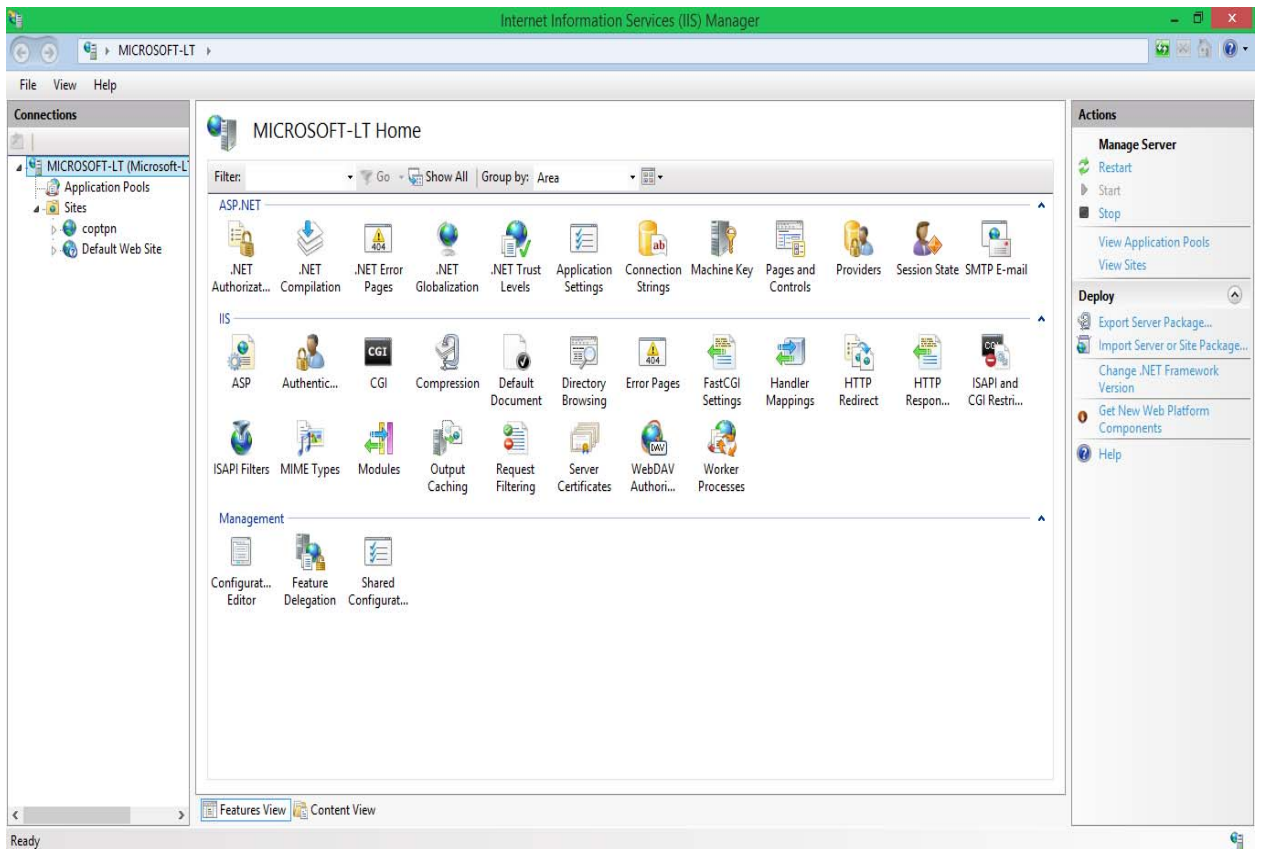
9.2.1 Website:

For website Microsoft IIS (Internet Information Services) server 7.5 or above must be installed on the system to server the website.

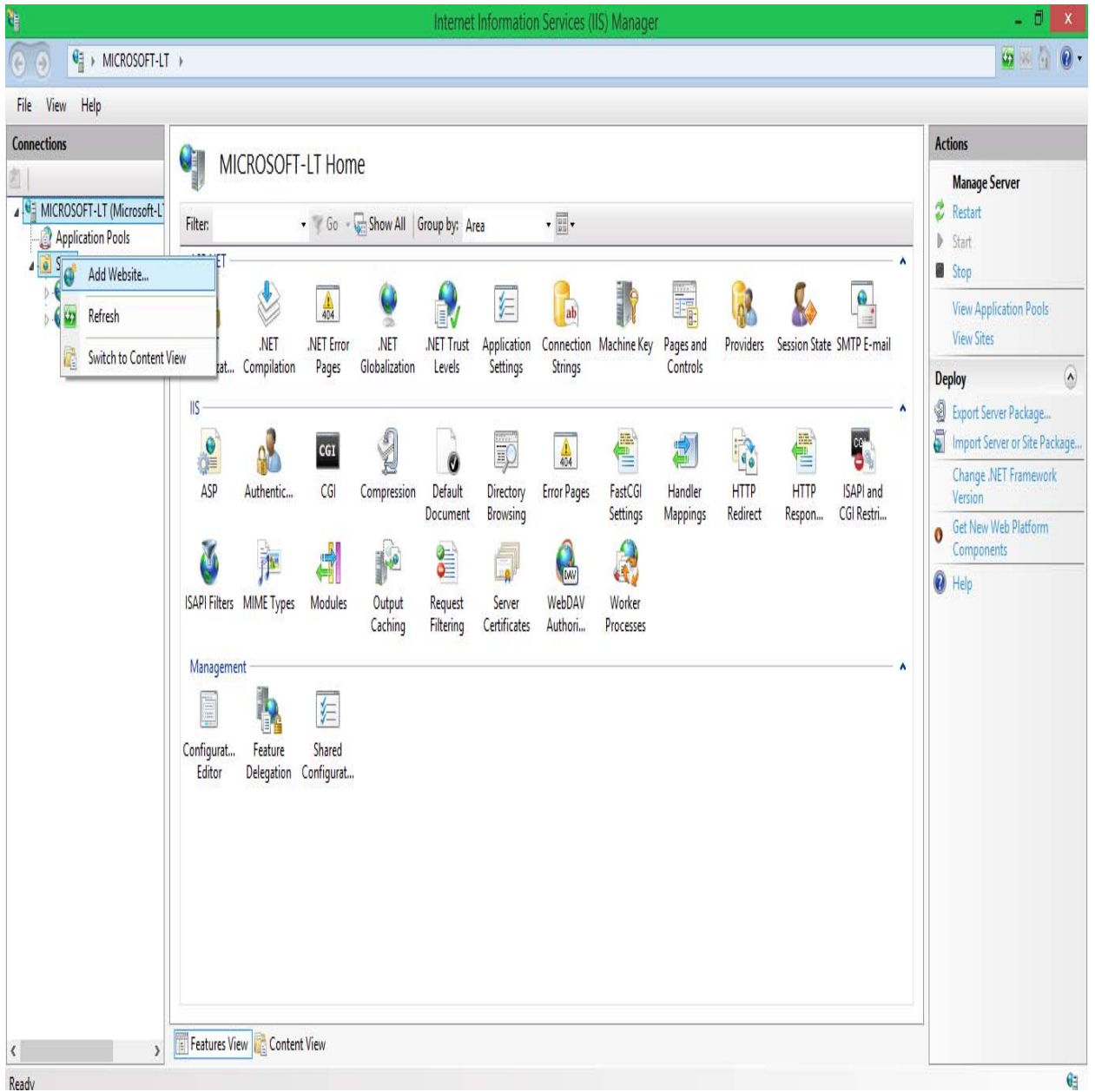
To publish the website, track the subsequent screenshots:



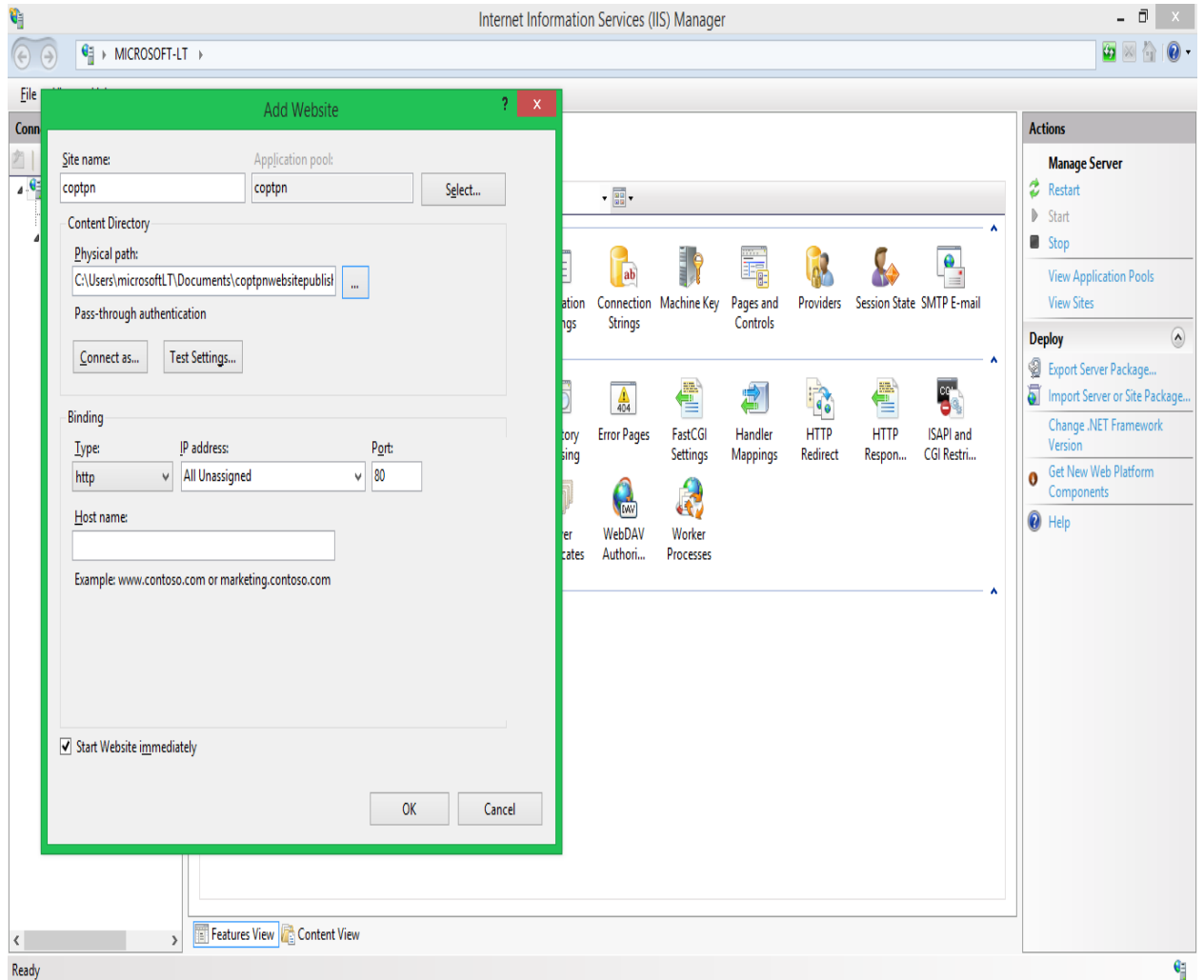
Screenshot 10: Showing how to publish the website



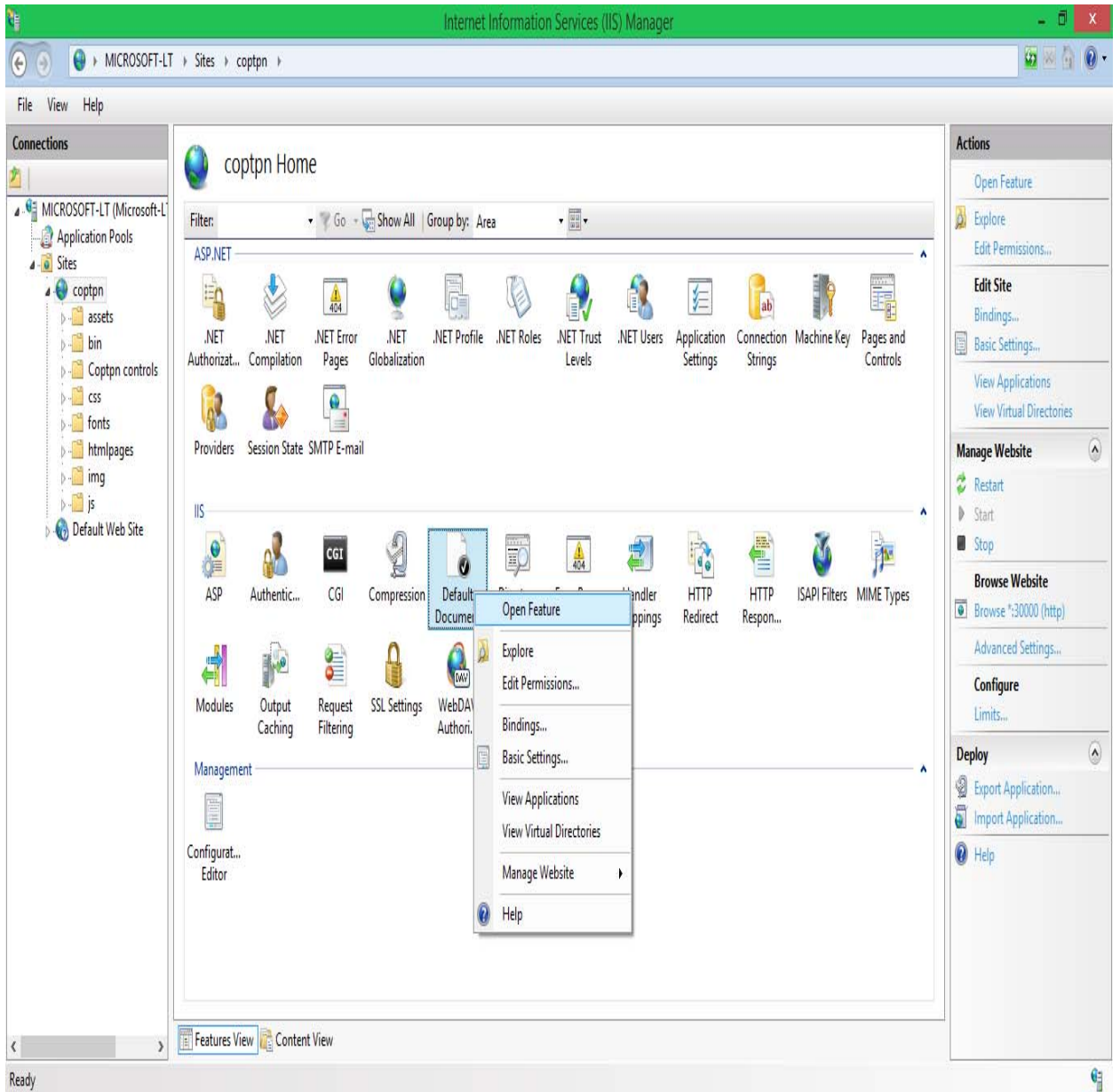
Screenshot 11: Showing how to publish the website



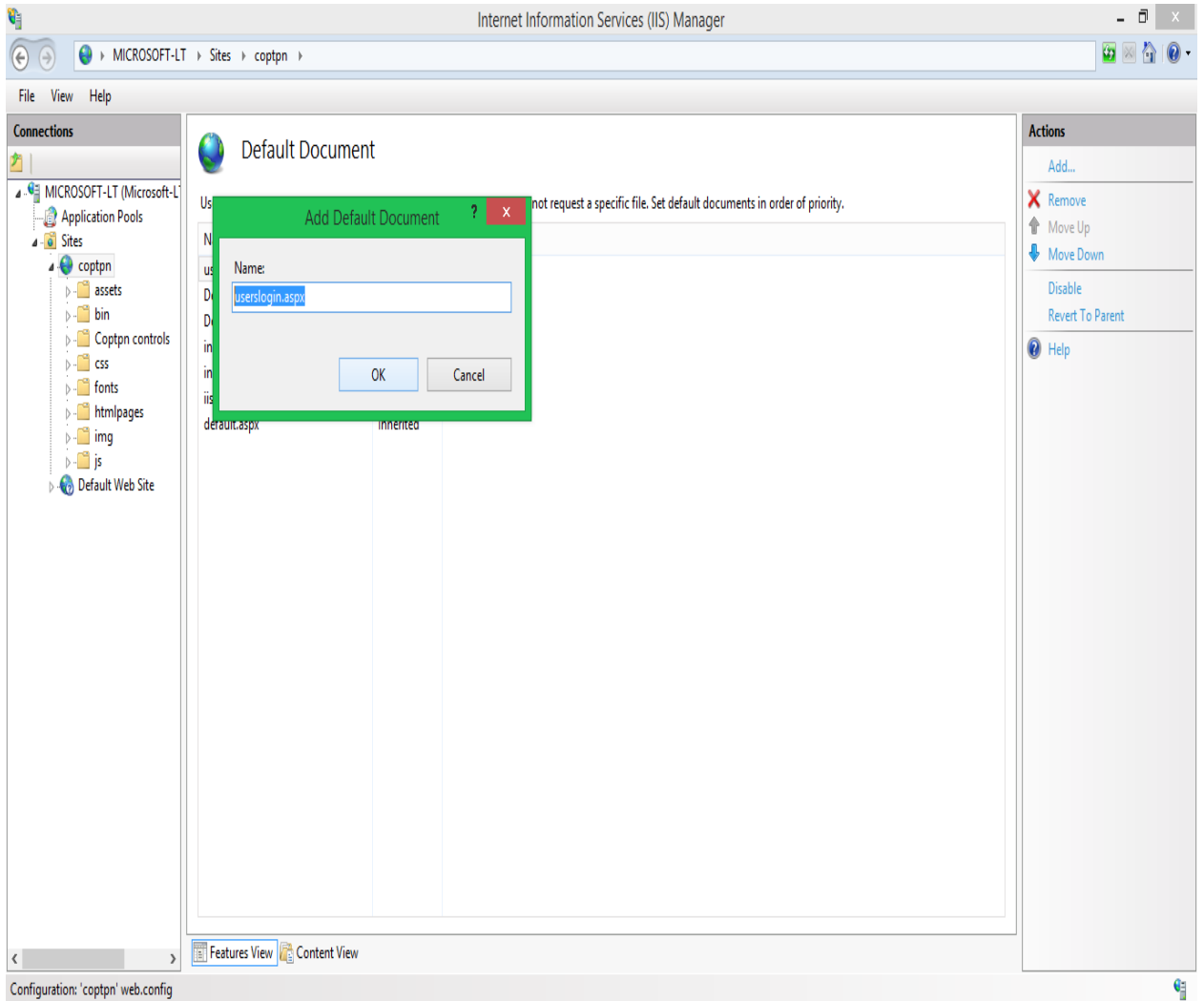
Screenshot 12: Showing how to publish the website



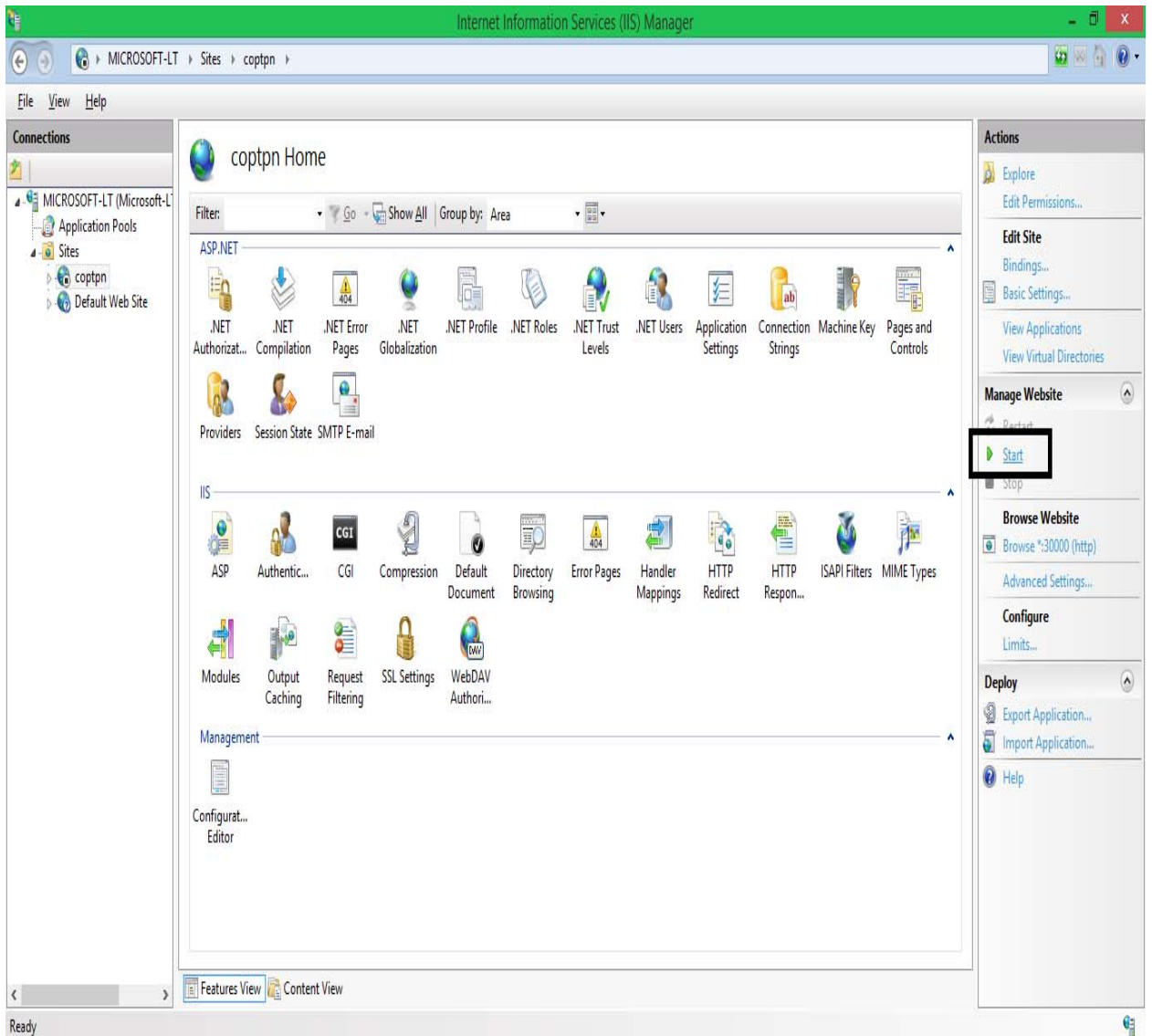
Screenshot 13: Showing how to publish the website
Press the 'OK' button.



Screenshot 14: Showing how to publish the website



Screenshot 15: Showing how to publish the website



Screenshot 16: Showing how to publish the website

Press the 'Start' button.

9.2.2 Middleware Server:

Install middleware server by simply double clicking on the .exe file.

9.2.3 Middleware Client for Service:

Run the middleware client for service simple by double clicking.

9.3 How to use the system

Operation of COTPN comprises of following steps:

1. Access the website by entering IP of the interface computer.
2. Register in COTPN.
3. Login your Account.
4. Administrator can view all the registered users, view resources, approve user request, add node/service, delete node, delete user by selecting it from the side menu.
5. Other users can view the nodes, services and resource available.
6. Select the service.
7. Middleware client for service will be automatically downloaded on the client side.
8. Run the Middleware client.
9. Login the Middleware client.
10. Fill in the IP of the desirable node as shown on the website.
11. Fill in the port number.
12. Upload the file.
13. File will be processed on the cloud and output file will be sent to the client.
14. Use some other service or Logout your account.

BIBLIOGRAPHY

10. Bibliography

- i. <http://www.statsoft.com/textbook/time-series-analysis/>
- ii. <http://www.tutor2u.net/economics/revision-notes/as-markets-Cloud.html>
- iii. <https://www.imf.org/external/np/speeches/2009/031809.htm>
- iv. <http://www.ask.com/question/factors-affecting-Cloud-prices>
- v. <http://www.economicshelp.org/blog/3809/Cloud/factors-affecting-Cloud-prices-in-short-term-and-long-term/>
- vi. <http://www.investментu.com/2012/January/the-five-factors-moving-Cloud-prices-this-year.html>
- vii. <http://c1wsolutions.wordpress.com/2012/04/30/factors-affect-price-of-Cloud/>
- viii. <http://cordis.europa.eu/fp7/ict/ssai/docs/cloud-report-final.pdf>
- ix. <https://www.us-cert.gov/sites/default/files/publications/CloudComputingHuthCebula.pdf>
- x. http://www.enterprisearchitect.org.uk/so-what-is-cloud/?gclid=CL7svf2_j7sCFQdY3godJAwAhQ
- xi. <http://www.ijcaonline.org/nccse/number2/SPE163T.pdf>
- xii. http://en.wikipedia.org/wiki/Cloud_computing
- xiii. <http://www.ijodls.in/uploads/3/6/0/3/3603729/vol. 2 july - sept. 2012 part-2.pdf>
- xiv. <http://www.microsoft.com/en-pk/download/details.aspx?id=17718>