**Error! Unknown document property name.**

**Error! Unknown document property name.**

High Level Design & Low Level Design

The purpose of this document is to provide with a template for documenting both HLD & LLD.

**Document Control :**

| **Project Revision History** | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | |  |  |  |  |  |
| **Date** | **Version** | **Author** | **Brief Description of Changes** | | | | **Approver Signature** | |
|  |  |  |  | | | |  | |
|  |  |  |  | | | |  | |
|  |  |  |  | | | |  | |

[**1. Introduction 3**](#_3znysh7)

[1.1. Intended Audience 3](#_2et92p0)

[1.2. Acronyms/Abbreviations 3](#_tyjcwt)

[1.3. Project Purpose 3](#_3dy6vkm)

[1.4. Key Project Objectives 3](#_1t3h5sf)

[1.5. Project Scope and Limitation 3](#_4d34og8)

[*1.5.1. In Scope 3*](#_2s8eyo1)

[*1.5.2. Out of scope 3*](#_17dp8vu)

[1.6. Functional Overview 3](#_3rdcrjn)

[1.7. Assumptions, Dependencies & Constraints 3](#_26in1rg)

[1.8. Risks 3](#_lnxbz9)

[**2. Design Overview 3**](#_35nkun2)

[2.1. Design Objectives 3](#_1ksv4uv)

[*2.1.1. Recommended Architecture 3*](#_44sinio)

[2.2. Architectural Strategies 3](#_2jxsxqh)

[*2.2.1. Design Alternative 3*](#_z337ya)

[*2.2.2. Reuse of Existing Common Services/Utilities 3*](#_3j2qqm3)

[*2.2.3. Creation of New Common Services/Utilities 3*](#_1y810tw)

[*2.2.4. User Interface Paradigms 3*](#_4i7ojhp)

[*2.2.5. System Interface Paradigms 3*](#_2xcytpi)

[*2.2.6. Error Detection / Exceptional Handling 3*](#_1ci93xb)

[*2.2.7. Memory Management 3*](#_3whwml4)

[*2.2.8. Performance 3*](#_2bn6wsx)

[*2.2.9. Security 3*](#_qsh70q)

[*2.2.10. Concurrency and Synchronization 3*](#_3as4poj)

[*2.2.11. Housekeeping and Maintenance 3*](#_1pxezwc)

[**3. System Architecture 3**](#_49x2ik5)

[3.1. System Architecture Diagram. (Not Necessary) 3](#_2p2csry)

[3.2. System Use-Cases 3](#_147n2zr)

[3.3. Subsystem Architecture 3](#_3o7alnk)

[3.4. System Interfaces 3](#_23ckvvd)

[*3.4.1. Internal Interfaces 3*](#_ihv636)

[*3.4.2. External Interfaces 3*](#_32hioqz)

[**4. Detailed System Design 3**](#_1hmsyys)

[4.1. Key Entities 3](#_41mghml)

[4.2. Detailed-Level Database Design 3](#_2grqrue)

[*4.2.1. Data Mapping Information 3*](#_vx1227)

[*4.2.2. Data Conversion 3*](#_3fwokq0)

[4.3. Archival and retention requirements 3](#_1v1yuxt)

[4.4. Disaster and Failure Recovery 3](#_4f1mdlm)

[4.5. Business Process workflow 3](#_2u6wntf)

[4.6. Business Process Modeling and Management (as applicable) 3](#_19c6y18)

[4.7. Business Logic 3](#_3tbugp1)

[4.8. Variables 3](#_28h4qwu)

[4.9. Activity / Class Diagrams (as applicable) 3](#_nmf14n)

[4.10. Data Migration 3](#_37m2jsg)

[*4.10.1. Architectural Representation 3*](#_1mrcu09)

[*4.10.2. Architectural Goals and Constraints 3*](#_46r0co2)

[*4.10.3. Logical View 3*](#_2lwamvv)

[*4.10.4. Architecturally Significant Design Packages 3*](#_111kx3o)

[*4.10.5. Data model 3*](#_3l18frh)

[*4.10.6. Deployment View 3*](#_1egqt2p)

[**5. Environment Description 3**](#_3ygebqi)

[5.1. Time Zone Support 3](#_2dlolyb)

[5.2. Language Support 3](#_sqyw64)

[5.3. User Desktop Requirements 3](#_3cqmetx)

[5.4. Server-Side Requirements 3](#_1rvwp1q)

[*5.4.1. Deployment Considerations 3*](#_4bvk7pj)

[*5.4.2. Application Server Disk Space 3*](#_2r0uhxc)

[*5.4.3. Database Server Disk Space 3*](#_1664s55)

[*5.4.4. Integration Requirements 3*](#_3q5sasy)

[*5.4.5. Jobs 3*](#_25b2l0r)

[*5.4.6. Network 3*](#_kgcv8k)

[*5.4.7. Others 3*](#_34g0dwd)

[5.5. Configuration 3](#_1jlao46)

[*5.5.1. Operating System 3*](#_43ky6rz)

[*5.5.2. Database 3*](#_2iq8gzs)

[*5.5.3. Network 3*](#_xvir7l)

[*5.5.4. Desktop 3*](#_3hv69ve)

[**6. References 3**](#_1x0gk37)

[**7. Appendix 3**](#_4h042r0)

# 

# Introduction

Call forwarding, or call diversion, is a telephony feature of all telephone switching systems which redirects a telephone call to another destination, which may be, for example, a mobile or another telephone number where the desired called party is available.

When a call is forwarded to another number, the caller is redirected to the cellular or landline phone connected to that number. This is highly useful in case of a network error and various other conditions. That said, the caller may be notified that they're being forwarded to another number.

## 1.1. Intended Audience

| All Mobile User |  |
| --- | --- |
|  |  |
|  |  |

## 1.2. Acronyms/Abbreviations

| CFSS | Call Forwarding System Simulator |
| --- | --- |
| User\_Reg | User Registration |
| Dis | Display |
|  |  |

**1.3. Project Purpose**

The main aim of this project Call forwarding typically can redirect incoming calls to any other domestic telephone number, but the owner of the forwarded line must pay any toll charges for forwarded calls.

Call Forwarding is a feature that allows you to send all the calls that come to your landline to another phone of your choosing. Selective Call Forwarding allows you to send selected incoming calls to a different number.

Call forwarding improves productivity by getting more calls answered, especially in the current mobile work environment. If someone is out of the office for vacation or work travel, incoming calls can be sent to an alternate number, ensuring that callers get questions answered and issues resolved.

## 1.4. Key Project Objectives

1. Objective of the project to provide Better Customer-Agent Interactions.

2. Always Availability and Better Business Presence.

3. Multiple Extensions on a Single Number

## 1.5. Project Scope and Limitation

The call forwarding feature can be a lifesaver for businesses. It allows you to route calls to the appropriate person or department, even when unavailable. The call forwarding service is a phone feature that allows users to forward incoming calls to another phone number, usually when the user is not able to answer the phone. Using a call forwarding system includes being able to answer calls even when you are not near your phone and being able to forward calls to another phone number in case of an emergency.

Call forwarding limits your ability to control the process. For example, if a call is forwarded to someone that is unavailable, important customer calls may not be attended to promptly.

### 1.5.1. In Scope

**Increase in Agent Availability** – Apart from taking several calls, the agents also need to work on some other official work which is impossible to take care of with constant calls coming through the line. Call forwarding redirects the calls to another agent or department as per the requirement in case the concerned agent is working on something else. Call forwarding solution is also useful for the agent to take a small breather in between calls and compose themselves for the next client in a more prepared way.

**Scope for Multitasking** – The call forwarding solutions in India are very useful in allowing agents to multitask. Agents can take time off calling and work on research or update the database while the calls are rerouted to other available lines without the customer waiting for hours or the call being dropped or redirected to an automated machine.

**The convenience of Preference** – We all know that customers love to talk to a human representative rather than talking to an automated voice. With call forwarding, it’s easier to reach an actual person for troubleshooting or query resolution rather than punching numbers with a robot for solutions.

### 1.5.2. Out of scope

1. Increased costs due to diverted calls

2. Higher team mobile costs due to remote working

3. Cost of additional time needed due to reduced internal communication and messaging between departments.

4.Losing internal phone transferring means your team are regularly having to take and forward messages adding extra work for them and increasing customer response times.

## 1.6. Functional Overview

Call Forwarding is a feature that allows you to send all the calls that come to your landline to another phone of your choosing. Selective Call Forwarding allows you to send selected incoming calls to a different number.

## 1.7 Assumptions, Dependencies & Constraints

**Hardware**- RAM,-4 .processor -i5

**Software**- Linux

**Operating System**- Ubuntu

**Constrains**

1.If the person receiving forwarded calls is unavailable calls can be missed.

2.Customers may be asked to recall numbers if they can't be transferred or unable to speak to team members relying on messages to be passed and calls returned, adding extra steps and delays to their requests.

## 1.8 Risks

1. Call forwarding a new way of cyber crime.

2.Online fraudsters are now using the call forwarding facility, which enables call transfer and forwarding, if the phone is busy, switched off or out of the network coverage area, to dupe people.

3.Call forwarding can risk into lots of frauds

4.Software and Hardware failure

# Design Overview

The project is designed by dividing into 7 files.i.e main file, menu file, log file,call file,admin file library file,RegAut file

These files contain different functions and declarations for different cases.these functions are called based on the user inputs.

The main file is the first from where the program will start.It has the welcome message .

The second file is a log file in which there are three logins.One login is for admin, one is for new\_user and one is for existing\_user.

The third file is the menu file.There are two menus one for admin and one for user.

Admin menu has three functions called unregister\_user, display\_database, update\_database.

In unregister\_user admin can delete registered user. Display function is used for displaying the file (database which we used in this project ). The update function is used for showing that the database is updated.Next file is RegAut file. In that file user can register his\her self. And the data will be taken from user and that will be printed as output and data will be stored in file as well.User will get access to call forwarding if he will be a registered user.

*.*

## 2.1. Design Objectives

The objective of the call forwarding system is , a user is calling another user and another user is using call forwarding activation on his phone number.So when the user calls another user his call will be forwarded to another number.

In the project we used three types of call forwarding activation mode.One mode is unconditional mode in which the call will be forwarded to another number.Second mode is no\_reply mode in this mode when the user will make a call he will get no\_reply message.Third mode is busy mode ,in this mode user will get that the receiver is busy.

### 2.1.1. Recommended Architecture

In this design a user can register himself before calling .Without being registered and authenticated (that his information is in the database). He cannot activate the call forwarding process.

## 2.2. Architectural Strategies

In this design the user is authenticated,this is the most important part of this project .

In the authentication process we are fetching data from the file of the registered user ,and match them by taking input from the user .If the authentication is successful ,he will be forwarded to call forwarding.

### 2.2.1. Design Alternative

Not Applicable

### 2.2.2. Reuse of Existing Common Services/Utilities

user\_reg();

This is used for the admin and user both*.*

### 2.2.3. Creation of New Common Services/Utilities

In this project we are using two menus ,in that user and admin can access according to their requirements.

### 2.2.4 User Interface Paradigms

This system will allow admin to unregister ,display and update database of a user by giving one numeric input

And allows users to use call forwarding system design ,by assurance of authentication.

### 

### 2.2.5. System Interface Paradigms

Good design creates good projects.If the system has a good interface and satisfies user requirements , then the software can reach new heights.Call forwarding system offers user to use unconditional ,no\_reply and busy types of call forwarding services.

### 2.2.6. Error Detection / Exceptional Handling

The C programming language does not support exception handling nor error handling. It is an additional feature offered by C. In spite of the absence of this feature, there are certain ways to implement error handling in C. Generally, in case of an error, most of the functions either return a null value or -1.

### 2.2.7. Memory Management

**Memory management** or memory allocation, is the process by which computer programs are assigned to physical or virtual memory space. Computer memory is a finite resource that must be efficiently managed.

For Memory management we will use Valgrind.It analyzes the program as it runs and shows errors and issues when it finds , such as memory leaks etc.

**Syntax–**

**valgrind–leak-check=full ./FileName**

.

### 2.2.8. Performance

When the user is registered ,then he can activate call forwarding processes.

Call forwarding will give authentication to the user.

Admin has the rights of unregistering a user only.

Display function is only applicable to the user.

### 2.2.9 Security

The Call ForwardingSystem is based on user input. That is secured by using the login method which gives assurance to user and admin that the system is authorized.

And the authentication function makes it more authorized.

### 2.2.10. Concurrency and Synchronization

Not Applicable

### 2.2.11. Housekeeping and Maintenance

Not Applicable

# System Architecture

To Create the CFSS program we are Functions given below

->Login()

->Admin\_Login()

->Existed\_User\_Login()

->New\_User\_Login()

->User\_Menu()

->Register\_User()

->Authentication()

->Type\_Of\_Call\_Forwording()

->Unconditional()

->No\_Reply()

->Busy()

->Dial()

->Admin\_Menu()

->Unregister\_User()

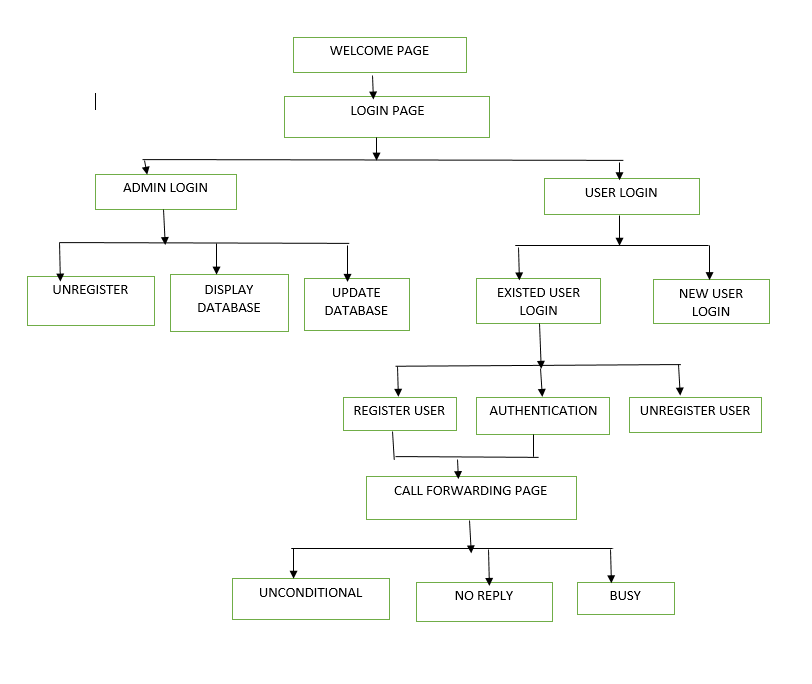
->Display\_Database()

->Update\_Database()

## 

## 

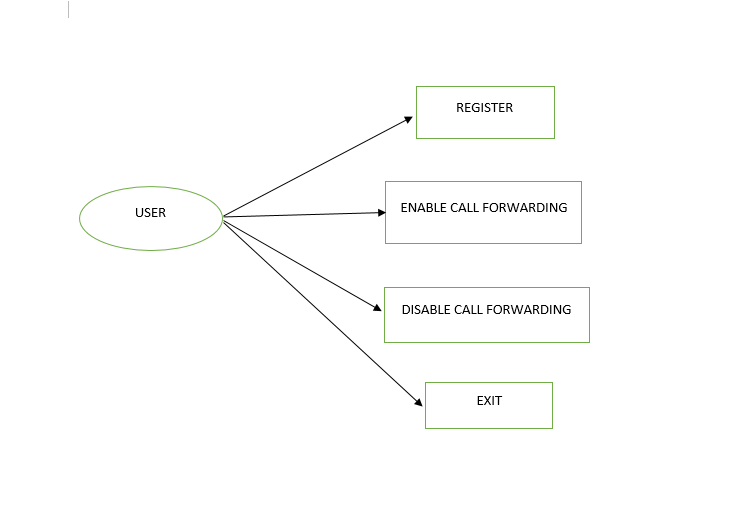
## 3.1. System Architecture Diagram.



## 

## 

## 3.2. System Use-Cases

**

## 3.3. Subsystem Architecture

To create this Call Forwarding System Design we create 7 files including one library.h files which have all the decorations or interfaces of this project, and other 6 files have definitions of those functions.And library.h file is included in all the files.

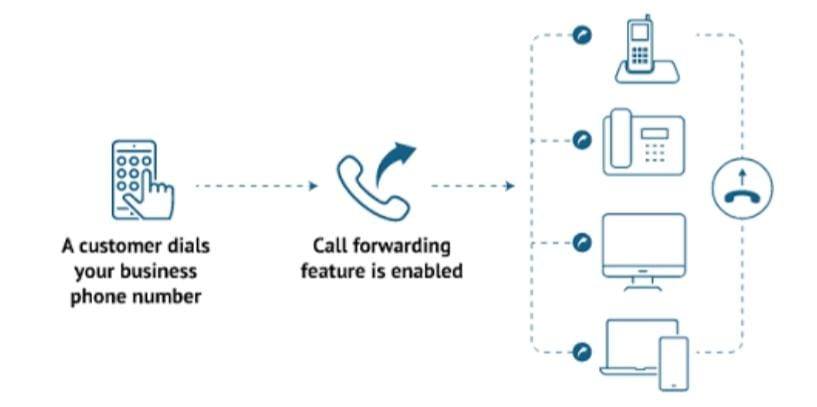
All the files have their own functionalities.

i.e:

1.RegAut file have Register\_user() and Authentication() function in a file.Which is used by user only.If he gave right input like he gave at the time of registering process he can use call forwarding service.

2.Admin files have the access to unregister() ,display(),and update\_database() of registered users.

## 3.4. System Interfaces

**

### 3.4.1 Internal Interfaces

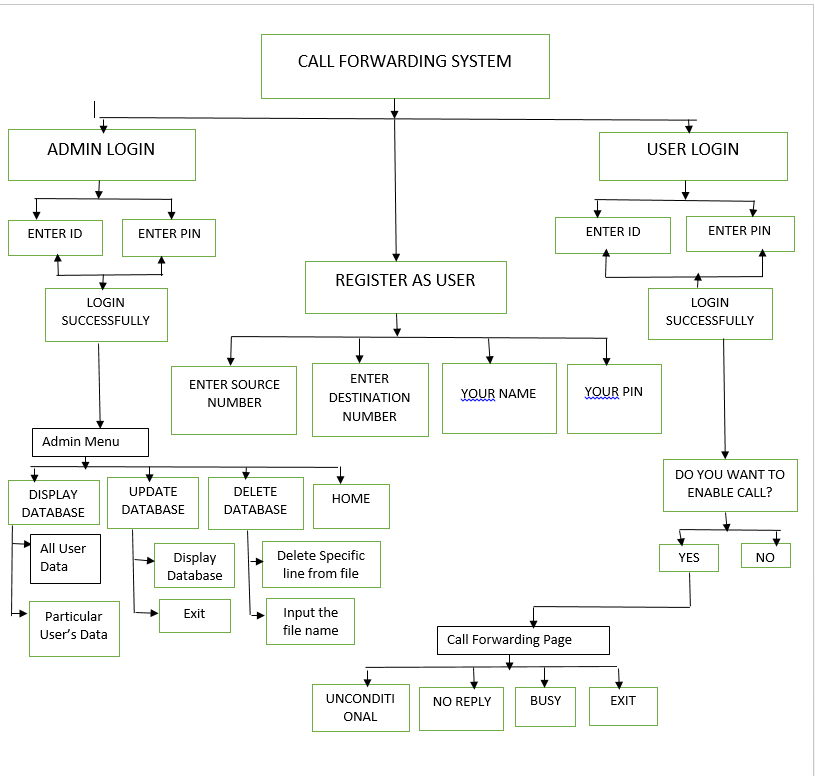
As an Internal interface we are using Ubuntu Linux Distribution. It is an Operating System that is made up of collection of software based on Linux Kernel or you can say that distribution contain the Linux Kernel and Supporting libraries and softwares

### 3.4.2 External Interfaces

1.Desktop or Linux system

2. Internet

# Detailed System Design

**

## 4.1. Key Entities

1. ADMIN\_LOGIN

-> Admin Id

->pin

2.User\_Reg

->name

-> pin

->source

->destination

3. User\_Login

-> user id

-> pin

## 4.2. Detailed-Level Database Design

* Not Applicable

### 4.2.1. Data Mapping Information

* + - * Not Applicable

### 4.2.2. Data Conversion

* Not Applicable

## 4.3. Archival and retention requirements

* Not Applicable

## 4.4. Disaster and Failure Recovery

* Not Applicable

## 4.5. Business Process workflow

* + Not Applicable

## 4.6. Business Process Modeling and Management (as applicable)

Not Applicable

## 4.7. Business Logic

Not Applicable

## 4.8. Variables

## Not Applicable

## 4.9. Activity / Class Diagrams (as applicable)

Not Applicable

## 4.10. Data Migration

Not Applicable

### 4.10.1. Architectural Representation

Not Applicable

### 4.10.2. Architectural Goals and Constraints

Not Applicable

### 4.10.3. Logical View

Not Applicable

### 4.10.4. Architecturally Significant Design Packages

Not Applicable

### 4.10.5. Data model

Legacy system data model

Proposed system data model

Interface data model

### 4.10.6. Deployment View

Not Applicable

# 5. Environment Description

The Environment we are using Linux Operating System

## 5.1. Time Zone Support

It’s Support all Time Zone

## 

## 5.2. Language Support

C language is used in this project.It was created in the 1970s by Dennis Ritchie,and remains very widely used.

## 5.3. User Desktop Requirements

Windows : 7 or above

Processor:Minimum 1GHz and more

Hard Drive: 32GB and more

Memory(RAM):Minimum 1GB and more

## 5.4. Server-Side Requirements

Not Applicable

### 5.4.1. Deployment Considerations

Not Applicable

### 5.4.2. Application Server Disk Space

Not Applicable

### 5.4.3. Database Server Disk Space

Not Applicable

### 5.4.4. Integration Requirements

Not Applicable

### 5.4.5. Jobs

Not Applicable

### 5.4.6. Network

Not Applicable

### 5.4.7. Others

Not Applicable

## 5.5. Configuration

For making CFSS we need the Windows version above 7 and Ubuntu Linux Distribution in our system.

### 5.5.1. Operating System

Windows : 7 or above

Processor:Minimum 1GHz and more

Hard Drive: 32GB and more

Memory(RAM):Minimum 1GB and mor*e*

### 5.5.2. Database

Not Applicable

### 5.5.3. Network

Not Applicable

### 5.5.4. Desktop

We are using linux Ubuntu 3.2

# 6. References

1.<https://stackoverflow.com/questions/5128664/how-to-split-a-c-program-into-multiple-files>

2.[https://www.tutorialspoint.com/write-a-c-program-to-read-a-data-from-file-an d-display](https://www.tutorialspoint.com/write-a-c-program-to-read-a-data-from-file-and-display)

3. https://youtu.be/NRA46yNC\_P0

# 7. Appendix

C Function

Pointers

File Handling

**Change Log**

| **QMS Template Version Control (Maintained by QA)** | | | | | |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
| **Date** | **Version** | **Author** | | **Description** | |
| 28-May-2015 | 1.0 | QA Team | | Initial Version | |
|  |  |  | |  | |
|  |  |  | |  | |
|  |  |  | |  | |