

# **1. INTRODUCTION**

Internet plays a vital role in today's world of communication. Today the world is running on the basis of internet. No work can be done without use of internet. Electronic mail i.e. email is the most important part in day to day life. But some of the people in today's world don't know how to make use of internet, some are blind or some are illiterate. So it goes very difficult to them when to live in this world of internet. Nowadays there are various technologies available in this world like screen readers, ASR, TTS, STT, etc. but these are not that much efficient for them. Around 39 million people are blind and 246 people have low vision and also 82 of people living with blindness are 50 aged and above. We have to make some internet facilities to them so they can use internet. Therefore we came up with our project as voice based email system for blinds which will help a lot to visually impaired peoples and also illiterate peoples for sending their mails. The users of this system don't need to remember any basic information about keyboard shortcuts as well as location of the keys. Simple mouse click operations are needed for functions making system easy to use for user of any age group. Our system provides location of where user is prompting through voice so that user doesn't have to worry about remembering which mouse click operation he/she wants to achieve.

Internet is considered as a major storehouse of information in today's world. No single work can be done without the help of it. It has even become one of the de facto methods used in communication. And out of all methods available email is one of the most common forms of communication especially in the business world. However not all people can use the internet. This is because in order to access the internet you would need to know what is written on the screen. If that is not visible it is of no use. This makes internet a completely useless technology for the visually impaired and illiterate people. Even the systems that are available currently like the screen readers TTS and ASR do not provide full efficiency to the blind people so as to use the internet. As nearly 285 million people worldwide are estimated visually impaired it become necessary to make internet facilities for communication usable for them also.

Therefore we have come up with this project in which we will be developing a voice based email system which will aid the visually impaired people who are naive to computer systems to use email facilities in a hassle free manner. The users of this system would not need to have any basic information regarding keyboard shortcuts or where the keys are located. All functions are based on simple mouse click operations making it very easy for any type of user to use this system.

Also the user need not worry about remembering which mouse click operation he/she needs to perform in order to avail a given service as the system itself will be prompting them as to which click will provide them with what operations.

The most common mail services that we use in our day to day life cannot be used by visually challenged people. This is because they do not provide any facility so that the person in front can hear out the content of the screen. As they cannot visualize what is already present on screen they cannot make out where to click in order to perform the required operations

For a visually challenged person using a computer for the first time is not that convenient as it is for a normal user even though it is user friendly. Although there are many screen readers available then also these people face some minor difficulties. Screen readers read out whatever content is there on the screen and to perform those actions the person will have to use keyboard shortcuts as mouse location cannot be traced by the screen readers. This means two things; one that the user cannot make use of mouse pointer as it is completely inconvenient if the pointer location cannot be traced and second that user should be well versed with the keyboard as to where each and every key is located. A user is new to computer can therefore not use this service as they are not aware of the key locations.

Another drawback that sets in is that screen readers read out the content in sequential manner and therefore user can make out the contents of the screen only if they are in basic HTML format.

## **2. SYSTEM ANALYSIS**

### **2.1 EXISTING SYSTEM**

The mail services that are available today are of no use to the people who are visually impaired. This is because these systems are not helpful to them in anyway as it cannot provide any audio feedback to read out the contents for them. As they are unable to visualize things that are present on the screen, they find it difficult to perform operations such as performing mouse click specifically.

Although, there are screen readers available but, they impose some or the other kind of difficulty to them. Screen readers basically read out the content on the screen for them and in order to respond to it, they need to provide input through a keyboard. So, in order to accomplish this, the user needs to be aware of the positions of the keys on the keyboard. Hence, a person who has never made use of a computer will never be able to use such kind of a system.

Also, the screen-readers that are available, read the contents sequentially and hence, only if the content is in the basic HTML format, then only the user is able to make out clearly what actually the content is. Also, the advance WebPages of email system which proves to be user friendly to a person with normal eyesight turns out to be complicated to them. Hence, in order to avoid the drawbacks of the current available systems, we are developing an email system that will help these blind people in many ways.

### **DRAWBACKS OF EXISTING SYSTEM**

- ✓ Screen readers read out the content in sequential manner and therefore user can make out the contents of the screen only if they are in basic HTML format.
- ✓ Thus the new advanced web pages which do not follow this paradigm in order to make the website more user-friendly only create extra hassles for these people

## **2.2 PROPOSED SYSTEM**

The proposed system is based on a completely novel idea and is nowhere like the existing mail systems. The most important aspect that has been kept in mind while developing the proposed system is accessibility. A web system is said to be perfectly accessible only if it can be used efficiently by all types of people whether able or disable. The current systems do not provide this accessibility. Thus the system we are developing is completely different from the current system. Unlike current system which emphasizes more on user friendliness of normal users, our system focuses more on user friendliness of all types of people including normal people visually impaired people as well as illiterate people.

The complete system is based on IVR- interactive voice response. When using this system the computer will be prompting the user to perform specific operations to avail respective services and if the user needs to access the respective services then he/she needs to perform that operation. One of the major advantages of this system is that user won't require to use the keyboard. All operations will be based on mouse click events. Now the question that arises is that how will the blind users find location of the mouse pointer. As particular location cannot be tracked by the blind user the system has given the user a free will to click blandly anywhere on the screen. Which type of click will perform which function will be specified by the IVR.

Thus user need not worry about location of the mouse at all. This system will be perfectly accessible to all types of users as it is just based on simple mouse clicks and speech inputs and there is no need to remember keyboard shortcuts. Also because of IVR facility those who cannot read need not worry as they can listen to the prompting done by the system and perform respective actions.

### **ADVANTAGES OF PROPOSED SYSTEM**

- ✓ User won't require to use the keyboard.
- ✓ All operations will be based on mouse click events.
- ✓ User need not worry about location of the mouse at all.

### **3. SYSTEM CONFIGURATION**

#### **3.1 HARDWARE REQUIREMENTS**

<b>SYSTEM</b>	:	Pentium i3 processor 2.4 GHz.
<b>HARD DISK</b>	:	40 GB.
<b>FLOPPY DRIVE</b>	:	1.44 Mb.
<b>MONITOR</b>	:	15 VGA Color.
<b>MOUSE</b>	:	Logitech.
<b>RAM</b>	:	512 MB.

#### **3.2 SOFTWARE REQUIREMENTS**

<b>OPERATING SYSTEM</b>	:	Windows XP Professional.
<b>CODING LANGUAGE</b>	:	PHP
<b>DATABASE</b>	:	MySql

## **4. SOFTWARE DESCRIPTION**

### **4.1 ABOUT THE FRONTEND**

PHP is a server-side scripting language created in 1995 and designed for web development but also used as a general-purpose programming language. As of January 2013, PHP was installed on more than 240 million websites (39% of those sampled) and 2.1 million web servers. Originally created by Rasmus Lerdorf in 1994, the reference implementation of PHP (powered by the Zend Engine) is now produced by The PHP Group. While PHP originally stood for Personal Home Page, it now stands for PHP: Hypertext Preprocessor, which is a recursive backronym.

PHP code can be simply mixed with HTML code, or it can be used in combination with various templating engines and web frameworks. PHP code is usually processed by a PHP interpreter, which is usually implemented as a web server's native module or a Common Gateway Interface (CGI) executable. After the PHP code is interpreted and executed, the web server sends resulting output to its client, usually in form of a part of the generated web page; for example, PHP code can generate a web page's HTML code, an image, or some other data. PHP has also evolved to include a command-line interface (CLI) capability and can be used instandalone graphical applications.

The canonical PHP interpreter, powered by the Zend Engine, is free software released under the PHP License. PHP has been widely ported and can be deployed on most web servers on almost every operating system and platform, free of charge.

Despite its popularity, no written specification or standard existed for the PHP language until 2014, leaving the canonical PHP interpreter as a de facto standard. Since 2014, there is ongoing work on creating a formal PHP specification.

#### **What can PHP do?**

Anything. PHP is mainly focused on server-side scripting, so you can do anything any other CGI program can do, such as collect form data, generate dynamic page content, or send and receive cookies. But PHP can do much more.

There are three main areas where PHP scripts are used.

- Server-side scripting. This is the most traditional and main target field for PHP. You need three things to make this work. The PHP parser (CGI or server module), a web server and a web browser. You need to run the web server, with a connected PHP installation. You can access the PHP program output with a web browser, viewing the PHP page through the server. All these can run on your home machine if you are just experimenting with PHP programming. See the installation instructions section for more information.
- Command line scripting. You can make a PHP script to run it without any server or browser. You only need the PHP parser to use it this way. This type of usage is ideal for scripts regularly executed using cron (on \*nix or Linux) or Task Scheduler (on Windows). These scripts can also be used for simple text processing tasks. See the section about Command line usage of PHP for more information.
- Writing desktop applications. PHP is probably not the very best language to create a desktop application with a graphical user interface, but if you know PHP very well, and would like to use some advanced PHP features in your client-side applications you can also use PHP-GTK to write such programs. You also have the ability to write cross-platform applications this way. PHP-GTK is an extension to PHP, not available in the main distribution.

PHP can be used on all major operating systems, including Linux, many Unix variants (including HP-UX, Solaris and Open BSD), Microsoft Windows, Mac OS X, RISC OS, and probably others. PHP has also support for most of the web servers today. This includes Apache, IIS, and many others. And this includes any web server that can utilize the Fast CGI PHP binary, like lighted and nginx. PHP works as either a module, or as a CGI processor.

So with PHP, you have the freedom of choosing an operating system and a web server. Furthermore, you also have the choice of using procedural programming or object oriented programming (OOP), or a mixture of them both.

With PHP you are not limited to output HTML. PHP's abilities includes outputting images, PDF files and even Flash movies (using libswf and Ming) generated on the fly. You can also output easily any text, such as XHTML and any other XML file. PHP can auto generate these files, and save them in the file system, instead of printing it out, forming a server-side cache for your dynamic content.

One of the strongest and most significant features in PHP is its support for a wide range of databases. Writing a database-enabled web page is incredibly simple using one of the database specific extensions (e.g., for mysql), or using an abstraction layer like PDO, or connect to any database supporting the Open Database Connection standard via the ODBC extension. Other databases may utilize cURL or sockets, like CouchDB.

PHP also has support for talking to other services using protocols such as LDAP, IMAP, SNMP, NNTP, POP3, HTTP, COM (on Windows) and countless others. You can also open raw network sockets and interact using any other protocol. PHP has support for the WDDX complex data exchange between virtually all Web programming languages. Talking about interconnection, PHP has support for instantiation of Java objects and using them transparently as PHP objects.

PHP has useful text processing features, which includes the Perl compatible regular expressions (PCRE), and many extensions and tools to parse and access XML documents. PHP standardizes all of the XML extensions on the solid base of libxml2, and extends the feature set adding Simple XML, XML Reader and XML Writer support.

And many other interesting extensions exist, which are categorized both alphabetically and by category. And there are additional PECL extensions that may or may not be documented within the PHP manual itself, like» XDebug.

As you can see this page is not enough to list all the features and benefits PHP can offer. Read on in the sections about installing PHP, and see the function reference part for explanation of the extensions mentioned here.

## **Advantages of PHP**



PHP has some advantages that have made it so popular, and it's been the go-to language for web servers for more than 15 years now. Here are some of PHP's benefits:

- **Cross-Platform:** PHP is platform-independent. You don't have to have a particular OS to use it because it runs on every platform, whether it's Mac, Windows, or Linux.
- **Open Source:** PHP is open source. The original code is made available to everyone who wants to build upon it. This is one of the reasons why one of its frameworks, Laravel, is so popular.
- **Easy to learn:** PHP is not hard to learn for absolute beginners. You can pick it up pretty if you already have programming knowledge.
- **PHP syncs with all Databases:** You can easily connect PHP to all Databases, relational and non-relational. So it can connect in no time to MySQL, Postgress, MongoDB, or any other database.
- **Supportive Community:** PHP has a very supportive online community. The official documentation provides guides on how to use the features and you can easily get your problem fixed while stuck.

## **Who Uses PHP**

A number of established companies and tech giants use PHP to run their servers and make a lot of incredible things.

- **Facebook:** Facebook uses PHP to power its site. In turn, the company contributed to the community by creating an implementation known as Hip Hop for PHP.
- **Wikipedia:** one of the world's largest sources of information on any topic, Wikipedia is built in PHP.
- **Content Management Systems (CMSs):** the world's most popular content management system, WordPress, is built in PHP. Other content management

systems such as Drupal, Joomla, and Magento are also built in PHP. Shopify runs on PHP too.

- Web Hosting Platforms: a lot of Web Hosting Platforms such as BlueHost, Site ground, and Whogohost run their hosting servers using PHP.

### **Common uses of PHP**

- PHP performs system functions, i.e. from files on a system it can create, open, read, write, and close them.
- PHP can handle forms, i.e. gather data from files, save data to a file, through email you can send data, return data to the user.
- You add, delete, modify elements within your database through PHP.
- Access cookies variables and set cookies.
- Using PHP, you can restrict users to access some pages of your website.
- It can encrypt data.

### **Characteristics of PHP**

Five important characteristics make PHP's practical nature possible –

- Simplicity
- Efficiency
- Security
- Flexibility
- Familiarity

### **Performance:**

PHP script is executed much faster than those scripts which are written in other languages such as JSP and ASP. PHP uses its own memory, so the server workload and loading time is automatically reduced, which results in faster processing speed and better performance.

**Open Source:**

PHP source code and software are freely available on the web. You can develop all the versions of PHP according to your requirement without paying any cost. All its components are free to download and use.

**Familiarity with syntax:**

PHP has easily understandable syntax. Programmers are comfortable coding with it.

**Embedded:**

PHP code can be easily embedded within HTML tags and script.

**Platform Independent:**

PHP is available for WINDOWS, MAC, LINUX & UNIX operating system. A PHP application developed in one OS can be easily executed in other OS also.

**Database Support:**

PHP supports all the leading databases such as MySQL, SQLite, ODBC, etc.

**Error Reporting -**

PHP has predefined error reporting constants to generate an error notice or warning at runtime. E.g., E\_ERROR, E\_WARNING, E\_STRICT, E\_PARSE.

**Loosely Typed Language:**

PHP allows us to use a variable without declaring its datatype. It will be taken automatically at the time of execution based on the type of data it contains on its value.

**Web servers Support:**

PHP is compatible with almost all local servers used today like Apache, Netscape, Microsoft IIS, etc.

**Security:**

PHP is a secure language to develop the website. It consists of multiple layers of security to prevent threats and malicious attacks.

**Control:**

Different programming languages require long script or code, whereas PHP can do the same work in a few lines of code. It has maximum control over the websites like you can make changes easily whenever you want.

**A Helpful PHP Community:**

It has a large community of developers who regularly updates documentation, tutorials, online help, and FAQs. Learning PHP from the communities is one of the significant benefits.

**Web Development**

PHP is widely used in web development nowadays. PHP can develop dynamic websites easily. But you must have the basic the knowledge of following technologies for web development as well.

- HTML
- CSS
- JavaScript
- Ajax
- XML and JSON
- jQuery

**Prerequisite**

Before learning PHP, you must have the basic knowledge of HTML, CSS, and JavaScript. So, learn these technologies for better implementation of PHP.

- HTML - HTML is used to design static webpage.
- CSS - CSS helps to make the webpage content more effective and attractive.
- JavaScript - JavaScript is used to design an interactive website.

## **Data types**

PHP stores whole numbers in a platform-dependent range, either a 64-bit or 32-bit signed integer equivalent to the C-language long type. Unsigned integers are converted to signed values in certain situations; this behavior is different from other programming languages. Integer variables can be assigned using decimal (positive and negative), octal, hexadecimal, and binary notations.

Floating point numbers are also stored in a platform-specific range. They can be specified using floating point notation, or two forms of scientific notation. PHP has a native Boolean type that is similar to the native Boolean types in Java and C++. Using the Boolean type conversion rules, non-zero values are interpreted as true and zero as false, as in Perl and C++.

The null data type represents a variable that has no value; Null is the only allowed value for this data type.

Variables of the "resource" type represent references to resources from external sources. These are typically created by functions from a particular extension, and can only be processed by functions from the same extension; examples include file, image, and database resources.

Arrays can contain elements of any type that PHP can handle, including resources, objects, and even other arrays. Order is preserved in lists of values and in hashes with both keys and values, and the two can be intermingled. PHP also supports strings, which can be used with single quotes, double quotes, now doc or here doc syntax.

The Standard PHP Library (SPL) attempts to solve standard problems and implements efficient data access interfaces and classes.

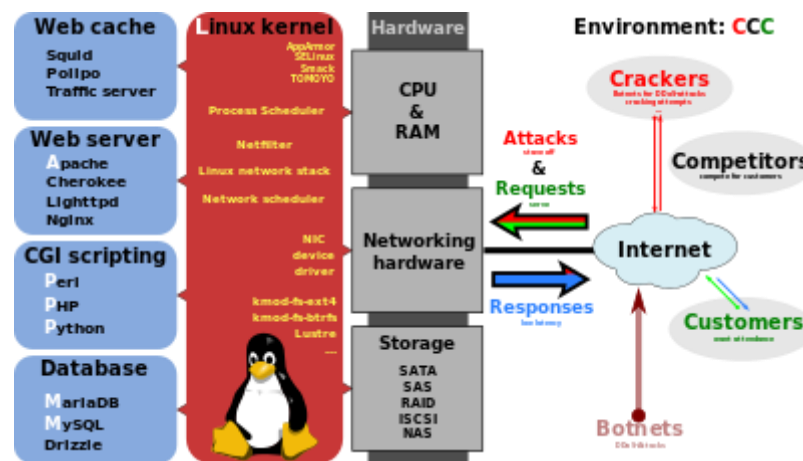
## **Objects**

Basic object-oriented programming functionality was added in PHP 3 and improved in PHP 4. Object handling was completely rewritten for PHP 5, expanding the feature set and enhancing performance. In previous versions of PHP, objects were handled like value types. The drawback of this method was that the whole object was

copied when a variable was assigned or passed as a parameter to a method. In the new approach, objects are referenced by handle, and not by value.

PHP 5 introduced private and protected member variables and methods, along with abstract classes, final classes, abstract methods, and final methods. It also introduced a standard way of declaring constructors and destructors, similar to that of other object-oriented languages such as C++, and a standard exception handling model. Furthermore, PHP 5 added interfaces and allowed for multiple interfaces to be implemented. There are special interfaces that allow objects to interact with the runtime system.

## Use

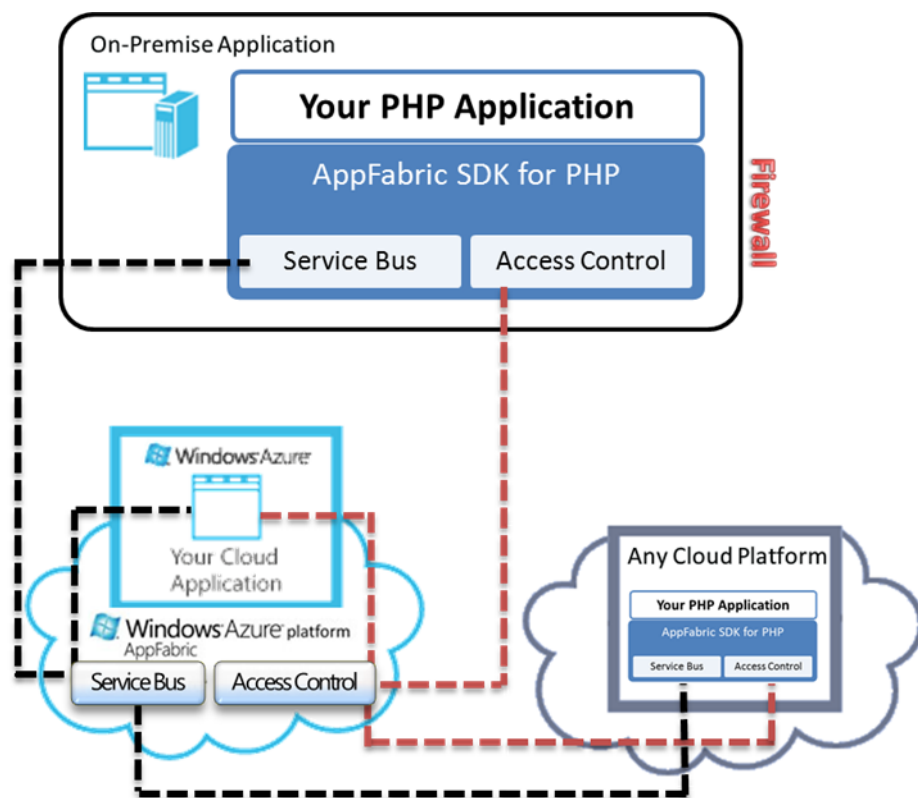


A broad overview of the LAMP software bundle, displayed here together with Squid.

PHP is a general-purpose scripting language that is especially suited to server-side web development, in which case PHP generally runs on a web server. Any PHP code in a requested file is executed by the PHP runtime, usually to create dynamic web page content or dynamic images used on websites or elsewhere. It can also be used for command-line scripting and client-side graphical user interface (GUI) applications. PHP can be deployed on most web servers, many operating systems and platforms, and can be used with many relational database management systems (RDBMS). Most web hosting providers support PHP for use by their clients. It is available free of charge, and the PHP Group provides the complete source code for users to build, customize and extend for their own use.

## Architecture and Concepts

The query cache plug-in is implemented as a PHP extension. It is written in C and operates under the hood of PHP. During the startup of the PHP interpreter, it gets registered as a `mysqlnd` plug-in to replace selected `mysqlnd` C methods. Hereby, it can change the behavior of any PHP MySQL extension (`mysqli`, `PDO_MYSQL`, `mysql`) compiled to use the `mysqlnd` library without changing the extensions API. This makes the plugin compatible with each and every PHP MySQL application. Because existing APIs are not changed, it is almost transparent to use. Please, see the `mysqlnd` plug-in API description for a discussion of the advantages of the plug-in architecture and a comparison with proxy based solutions.



### *Transparent to use*

At PHP run time PECL/`mysqlnd_qc` can proxy queries send from PHP (`mysqlnd`) to the MySQL server. It then inspects the statement string to find whether it shall cache its results. If so, result set is cached using a storage handler and further executions of the statement are served from the cache for a user-defined period. The

Time to Live (TTL) of the cache entry can either be set globally or on a per statement basis.

A statement is either cached if the plugin is instructed to cache all statements globally using a or, if the query string starts with the SQL hint (*/\*qc=on\**). The plugin is capable of caching any query issued by calling appropriate API calls of any of the existing PHP MySQL extensions.

### ***Flexible storage: various storage handler***

Various storage handler are supported to offer different scopes for cache entries. Different scopes allow for different degrees in sharing cache entries among clients.

- *default* (built-in): process memory, scope: process, one or more web requests depending on PHP deployment model used
- *APC*: shared memory, scope: single server, multiple web requests
- *SQLite*: memory or file, scope: single server, multiple web requests
- *MEMCACHE*: main memory, scope: single or multiple server, multiple web requests
- *user* (built-in): user-defined - any, scope: user-defined - any

Support for the *APC*, *SQLite* and *MEMCACHE* storage handler has to be enabled at compile time. The *default* and *user* handler are built-in. It is possible to switch between compiled-in storage handlers on a per query basis at run time. However, it is recommended to pick one storage handler and use it for all cache entries.

### ***Built-in slam defense to avoid overloading***

To avoid overload situations the cache plugin has a built-in slam defense mechanism. If popular cache entries expires many clients using the cache entries will try to refresh the cache entry. For the duration of the refresh many clients may access the database server concurrently. In the worst case, the database server becomes overloaded and it takes more and more time to refresh the cache entry, which in turn



lets more and more clients try to refresh the cache entry. To prevent this from happening the plug in has a slam defense mechanism. If slam defense is enabled and the plug in detects an expired cache entry it extends the life time of the cache entry before it refreshes the cache entry.

This way other concurrent accesses to the expired cache entry are still served from the cache for a certain time. The other concurrent accesses do not trigger a concurrent refresh. Ideally, the cache entry gets refreshed by the client which extended the cache entries lifespan before other clients try to refresh the cache and potentially cause an overload situation.

### ***Unique approach to caching***

PECL/mysqldb has a unique approach to caching result sets that is superior to application based cache solutions. Application based solutions first fetch a result set into PHP variables. Then, the PHP variables are serialized for storage in a persistent cache, and then unserialized when fetching. The mysqldb query cache stores the raw wire protocol data sent from MySQL to PHP in its cache and replays it, if still valid, on a cache hit. This way, it saves an extra serialization step for a cache put that all application based solutions have to do. It can store the raw wire protocol data in the cache without having to serialize into a PHP variable first and de-serializing the PHP variable for storing in the cache again.

### **Object Oriented Programming Principles**

The three major principles of OOP are;

**Encapsulation** – this is concerned with hiding the implementation details and only exposing the methods. The main purpose of encapsulation is to;

**Reduce software development complexity** – by hiding the implementation details and only exposing the operations, using a class becomes easy.

**Protect the internal state of an object** – access to the class variables is via methods such as get and set, this makes the class flexible and easy to maintain.

The internal implementation of the class can be changed without worrying about breaking the code that uses the class.

**Inheritance** – this is concerned with the relationship between classes. The relationship takes the form of a parent and child. The child uses the methods defined in the parent class. The main purpose of inheritance is;

**Re-usability**– a number of children, can inherit from the same parent. This is very useful when we have to provide common functionality such as adding, updating and deleting data from the database.

**Polymorphism** – this is concerned with having a single form but many different implementation ways. The main purpose of polymorphism is;

Simplify maintaining applications and making them more extendable.

In OOP (i.e. object-oriented programming), you will combine codes and data to create an object. A computer application created using this style consists of different objects that can communicate with each other. Often, these objects are self-contained and possess different methods and properties. Properties serve as an object's data. Thus, these are variables owned by the object they point to. The methods, on the other hand, are functions an object supports. Classes serve as templates for a programming object.

They describe the properties and methods that an object will possess. In the example given below, the class describes a vehicle. For each vehicle inside your computer program, you may create an instance of the class to represent that vehicle's data. For instance, if two vehicles in your application are named "Car" and "Motorcycle," you will create two instances of your class and initialize the appropriate variable for each vehicle.

To initialize the variables, you need to invoke the method named setName() for the two vehicles. The members and methods that interacting objects can utilize are known as the "contract" of the class. For the example below, the car's contracts are the "get" and "set" methods, getType() and setType().

**private** – To access an object's private members, you need to be inside one of the methods of that object. You can't access these members while you are inside a method of derived objects. Since you can't "see" private properties while you're inside an inheriting class, two different classes can declare identical private properties.

**protected** – A protected member is similar to a private one in that you can only access it from inside the method of an object. The only difference between these members is that a protected member is visible from an inheriting class. In this kind of situation, you need to use the `$this` variable to access the protected members.

**public** – You can access a public member both from inside the object (i.e. using the `$this` variable) and outside the object (i.e. through `$object→Member`). These rules will apply if a different class inherits public members. Here, you can access the members both from inside the class's methods and outside its objects.

## **How to Handle Exceptions**

Programmers consider exception handling as the most difficult part of software development. In general, errors (e.g. network failure, database failure, program bug, etc.) pose serious problems to program developers. For instance, developers need to make decisions regarding the errors that occurred, insert checks to prevent failure, and invoke the right function to manage it.

Additionally, programmers need to make sure that their program will work as normal after handling the error. Currently, most computer languages offer their own version of "try/catch/throw" (i.e. a popular paradigm for handling exceptions). The construct named "try/catch" protects the code it belongs to and informs the computer language about its security tasks. Here, the program will "throw" errors and exceptions as soon as they are detected. Then, the language (e.g. PHP) will scan its execution stack to know whether there's a "try/catch" construct that can handle the problem.

This method offers a lot of advantages. For example, it allows you to write robust programs without having to write "if" statements in each of your code blocks. That means you can minimize the codes that you need to write. With the

“try/catch/throw” paradigm, you can just enclose code blocks with “try/catch” constructs and manage errors once they occur. Moreover, upon detecting an exception via the “throw” construct, you may go back to a part of the code that can handle and continue the program’s execution.

## **Common Errors**

Programmers make mistakes sometimes. If you will subscribe to security-related email lists, you will discover new weaknesses of PHP programs each week. Let’s discuss some of the most popular errors related to PHP:

**Global Variables** – In some cases, program developers fail to initialize global variables correctly. You can prevent this mistake by setting the ‘register\_globals’ directive to “off.” However, this problem can still occur so you need to be careful. Users whose register\_globals are on might abuse your website application. For example, customers might gain admin-level access to your system just by running arbitrary codes on your site.

**Cross-Site Scripting** – A hacker might use “cross-site scripting” to run client-side languages (e.g. JavaScript) to steal cookies and confidential information. This scripting technique is easy and simple. The hacker just needs to enter raw information into the website’s HTML.

**SQL Injection** – This method requires the hacker to insert malicious codes into his/her database queries.

## **4.2 ABOUT THE BACKEND**

### **MYSQL**

The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation.

MySQL is a popular choice of database for use in web applications, and is a central component of the widely used LAMP open source web application software stack (and other 'AMP' stacks). LAMP is an acronym for "Linux, Apache, MySQL, Perl/PHP/Python." Free-software-open source projects that require a full-featured database management system often use MySQL.

### **SQL**

SQL (Structured Query Language) is a special-purpose programming language designed for managing data held in a relational database management system (RDBMS), or for stream processing in a relational data stream management system (RDSMS). SQL consists of a data definition language and a data manipulation language. The scope of SQL includes data insert, query, update and delete, schema creation and modification, and data access control. Although SQL is often described as, and to a great extent is, a declarative language (4GL), it also includes procedural elements.

### **Security**

As most products do, MySQL comes "ready-to-work" out of the box. Usually, security is not a major consideration when installing this kind of product. Often, the most important issue is to get it up and running as quickly as possible so that the organization can benefit. This document is intended as a quick security manual to help you bring an installed MySQL database server into conformity with best security practices

## **Memory Management**

The MySQL Native Driver manages memory different than the MySQL Client Library. The libraries differ in the way memory is allocated and released, how memory is allocated in chunks while reading results from MySQL, which debug and development options exist, and how results read from MySQL are linked to PHP user variables.

The following notes are intended as an introduction and summary to users interested at understanding the MySQL Native Driver at the C code level

## **Interfaces**

MySQL is a relational database management system (RDBMS), and ships with no GUI tools to administer MySQL databases or manage data contained within the databases. Users may use the included command line tools, or use MySQL "front-ends", desktop software and web applications that create and manage MySQL databases, build database structures, back up data, inspect status, and work with data records. The official set of MySQL front-end tools, MySQL Workbench is actively developed by Oracle, and is freely available for use.

## **Features**

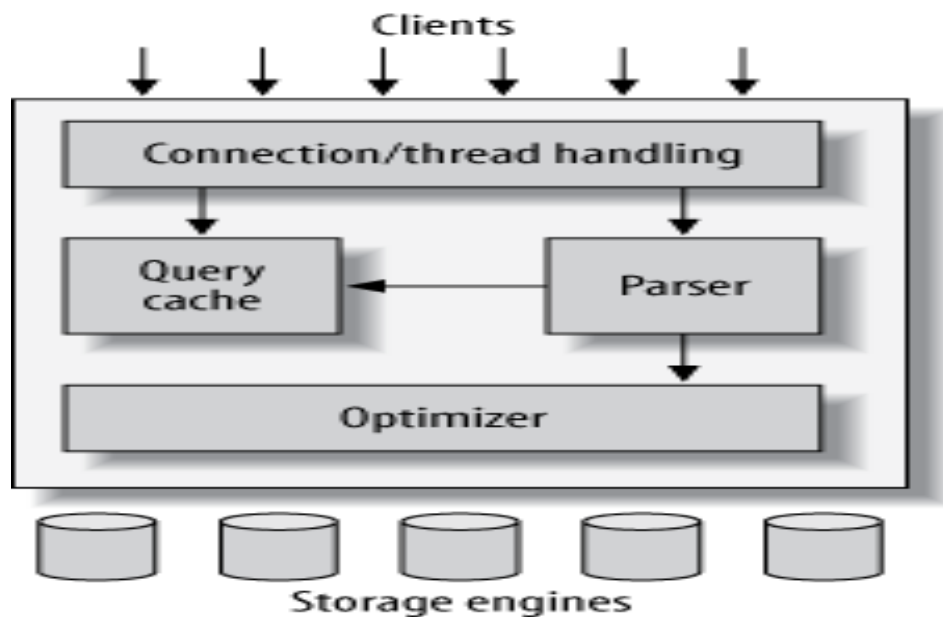
MySQL is offered under two different editions: the open source MySQL Community Server and the proprietary Enterprise Server. MySQL Enterprise Server is differentiated by a series of proprietary extensions which install as server plug in, but otherwise shares the version numbering system and is built from the same code base.

- A broad subset of ANSI SQL 99, as well as extensions
- Cross-platform support
- Stored procedures, using a procedural language that closely adheres to SQL/PSM<sup>[33]</sup>
- Triggers

- Cursors
- Updatable views
- Online DDL when using the InnoDB Storage Engine.
- Information schema
- Performance Schema
- A set of SQL Mode options to control runtime behavior, including a strict mode to better adhere to SQL standards.

### MySQL's Logical Architecture

The topmost layer contains the services that aren't unique to MySQL. They're services most network-based client/server tools or servers need: connection handling, authentication, security, and so forth.



### High availability

Ensuring high availability requires a certain amount of redundancy in the system. For database systems, the redundancy traditionally takes the form of having a primary server acting as a master, and using replication to keep secondary's available to take over in case the primary fails. This means that the "server" that the application

connects to is in reality a collection of servers, not a single server. In a similar manner, if the application is using a sharded database, it is in reality working with a collection of servers, not a single server. In this case, a collection of servers is usually referred to as a farm.

One of the projects aiming to provide high availability for MySQL is MySQL Fabric, an integrated system for managing a collection of MySQL servers, and a framework on top of which high availability and database sharding is built. MySQL Fabric is open-source and is intended to be extensible, easy to use, and to support procedure execution even in the presence of failure, providing an execution model usually called resilient execution.

## **Deployment**

MySQL can be built and installed manually from source code, but this can be tedious so it is more commonly installed from a binary package unless special customizations are required. On most Linux distributions the package management system can download and install MySQL with minimal effort, though further configuration is often required to adjust security and optimization settings.

Though MySQL began as a low-end alternative to more powerful proprietary databases, it has gradually evolved to support higher-scale needs as well. It is still most commonly used in small to medium scale single-server deployments, either as a component in a LAMP-based web application or as a standalone database server. Much of MySQL's appeal originates in its relative simplicity and ease of use, which is enabled by an ecosystem of open source tools such as phpMyAdmin. In the medium range, MySQL can be scaled by deploying it on more powerful hardware, such as a multi-processor server with gigabytes of memory.

## **The Pros**

It is popular – MySQL possesses the largest market share in the database management industry. Almost all hosting service providers offer access to MySQL. Additionally, you can easily find reading materials about MySQL. That means you can get and learn this database management system quickly and easily.



It is intuitive – Once you have established your MySQL database, managing your data becomes a straightforward process. An administrator should configure the initial access to a database. Thus, if you will share your database with other people, you need to give them access to MySQL first.

It is free – This database management system is open-source. That means you can use it without shelling out any money.

MySQL supports two kinds of queries:

- (1) buffered queries and
- (2) unbuffered queries.

Let's discuss these queries in detail:

### **Buffered Queries**

A buffered query retrieves the result and stores it in the client-side machine. If the user will run subsequent queries, his/her requests will go through the memory of his local machine first. The main advantage of a buffered query is that you can search inside it. That means you can control the row pointer inside the result set according to your needs (considering that the information is stored in the local machine). Its primary disadvantage, on the other hand, is that the results require extra memory.

**Unbuffered Queries** An unbuffered query doesn't allow you to move its row pointer. Thus, you have to follow the exact sequence of the search results. The advantage offered by this query is that it doesn't require excessive storage space. You may collect and process data rows once your MySQL server begins to return them. While using the result of an unbuffered query, you should get all of the rows using "mysqli\_fetch\_row" or free them using "mysqli\_free\_result". Your server won't respond to your requests until you issue one of these commands.

## **5. SYSTEM DESIGN**

### **5.1 MODULES**

1. Registration
2. Login
3. BMAIL APP Page
4. Compose Mail
5. Inbox
6. Sent mail

#### **Modules Description**

##### **Registration**

This is the first module of system. Any of the users who want to use the system should first register himself to obtain his/her own username and password. Registration module will obtain all the details about user by voice commands given by the system that where to fill which information. The user should speak the details as the system requires. If the information is incorrect then the system will be telling about re-enter the information again.

##### **Login**

This is the second module of system. Once the registration is done the user can login to the system. Login module will ask user to provide username and password. Here the process goes in speech to text conversation of user. User is told to validate whether he/she entered details are correct or not. If the details are correct then the user is authorized and will enter to the main page.

##### **BMAIL APP Page**

The user is directed to this page once login is done successfully. From this page now the user can perform operations that the user wishes to perform. The options available are: • Inbox • Compose User will be guided with the help of IVR in which direction he has to move. Also there is an icon for logout, which would read as

“logout” when mouse goes or rolls over it. So, when the user wants can logout from the system.

### **Compose Mail**

These are not only the most used mail function but also a very important feature of mailing systems. Without compose, one cannot mail. Since the system is for visually challenged people and keyboard operations are completely avoided composing mail is totally done on voice input and mouse operations. No typed input will be required, as the system totally focuses on simple mouse click operations.. User can record the messages by clicking on the small mic option present in front of every box. Here, the STT technology gets used, that means speech gets converted to text.

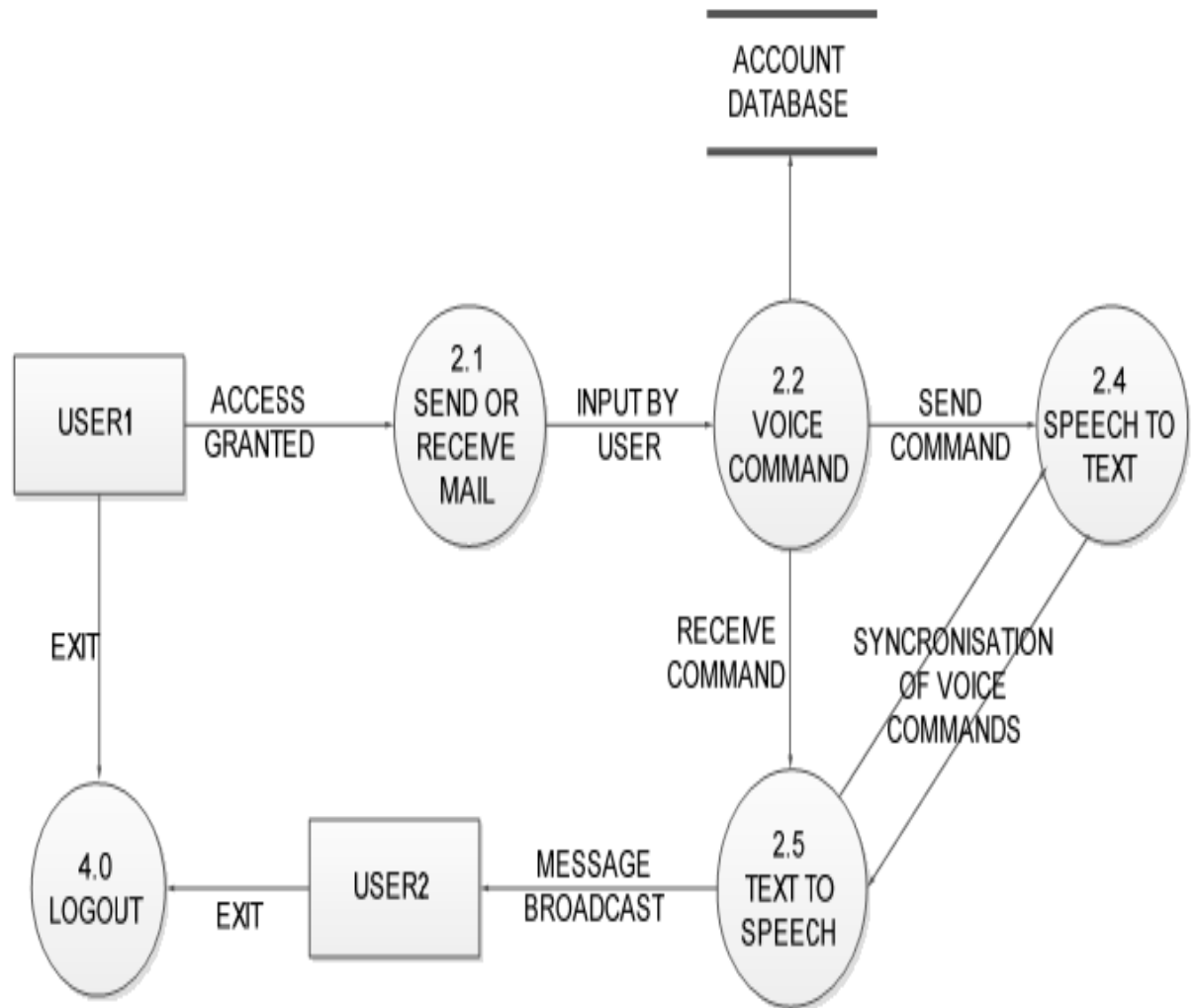
### **Inbox**

The user can view all the mails received to the account through this option. Once in this option the system will be continuously prompting the user about which click operation needs to be performed to navigate through the page and perform operations on the received mail. The user also has the option to delete the mails received. The deleted mails will be stored in the trash section.

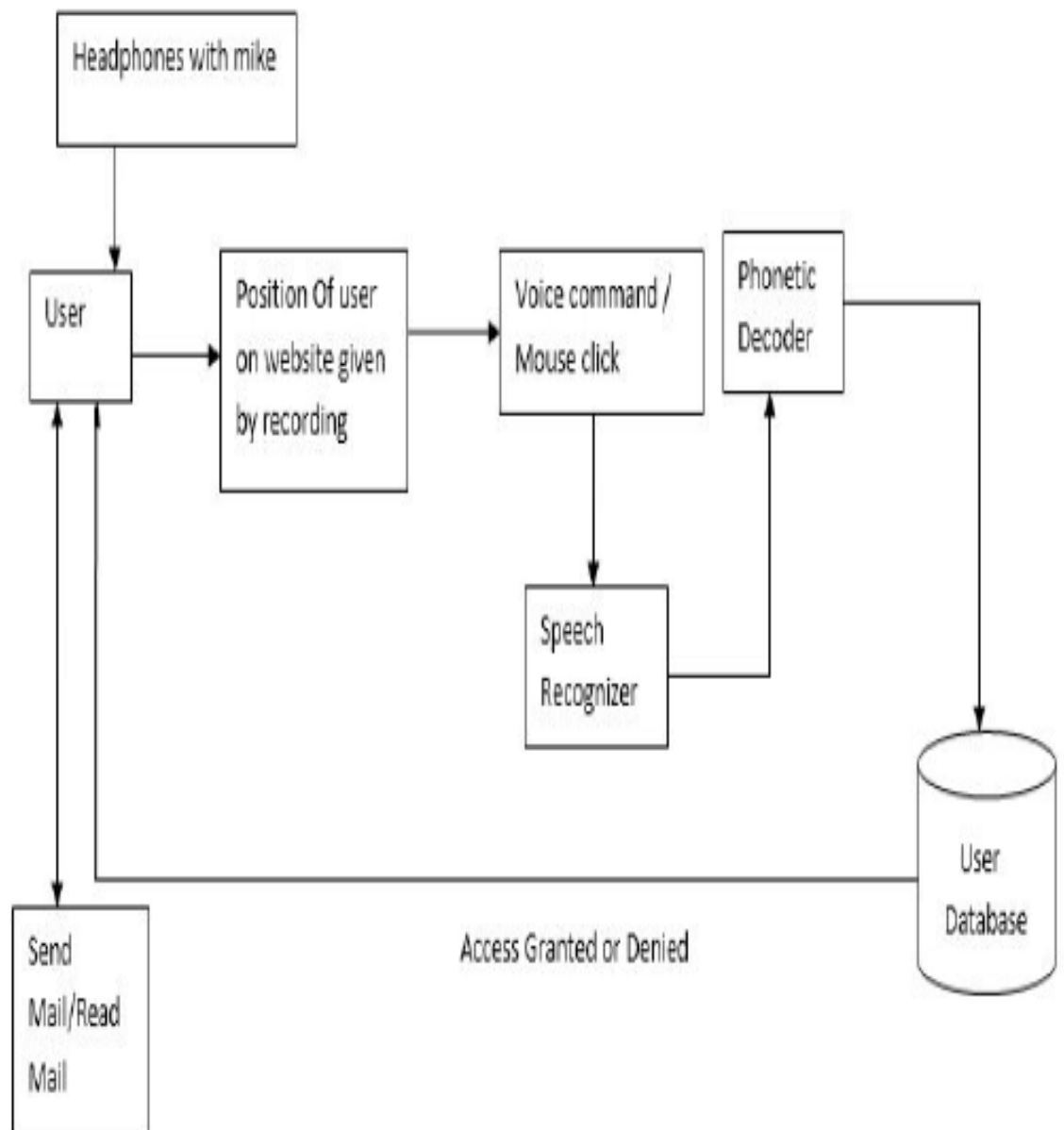
### **Sent mail**

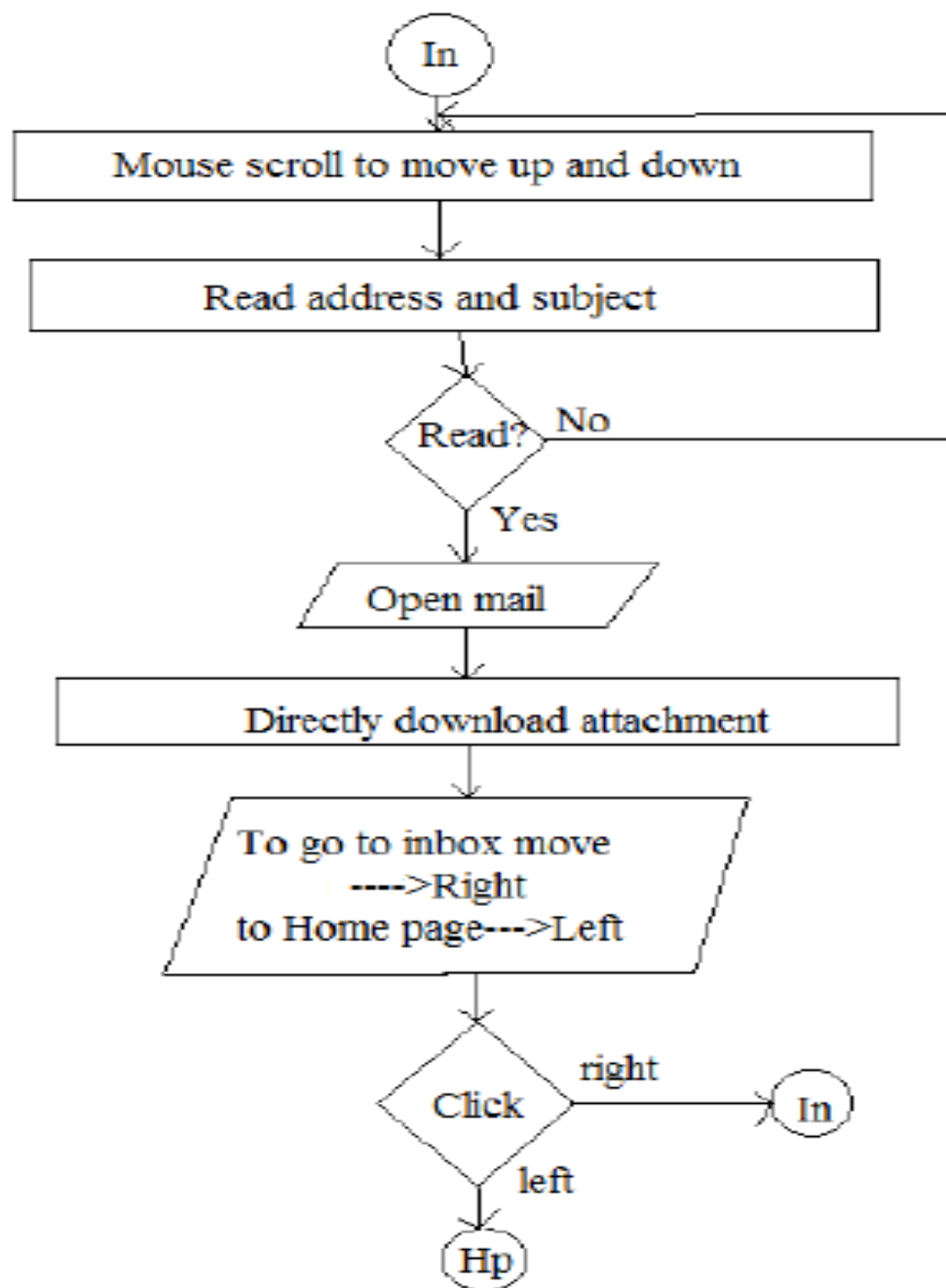
This option will keep a track of all the mails sent by the user. If the user wants to access these mails, this option will provide them with their needs. In order to access the sent mails user will need to perform the actions provided by the prompt to navigate between mails. When the control lands on particular mail user will be prompted as who the receiver was and what is the subject of the mail. This will help the user in efficiently understanding and extracting the required mail.

## 5.2 BLOCK DIAGRAM

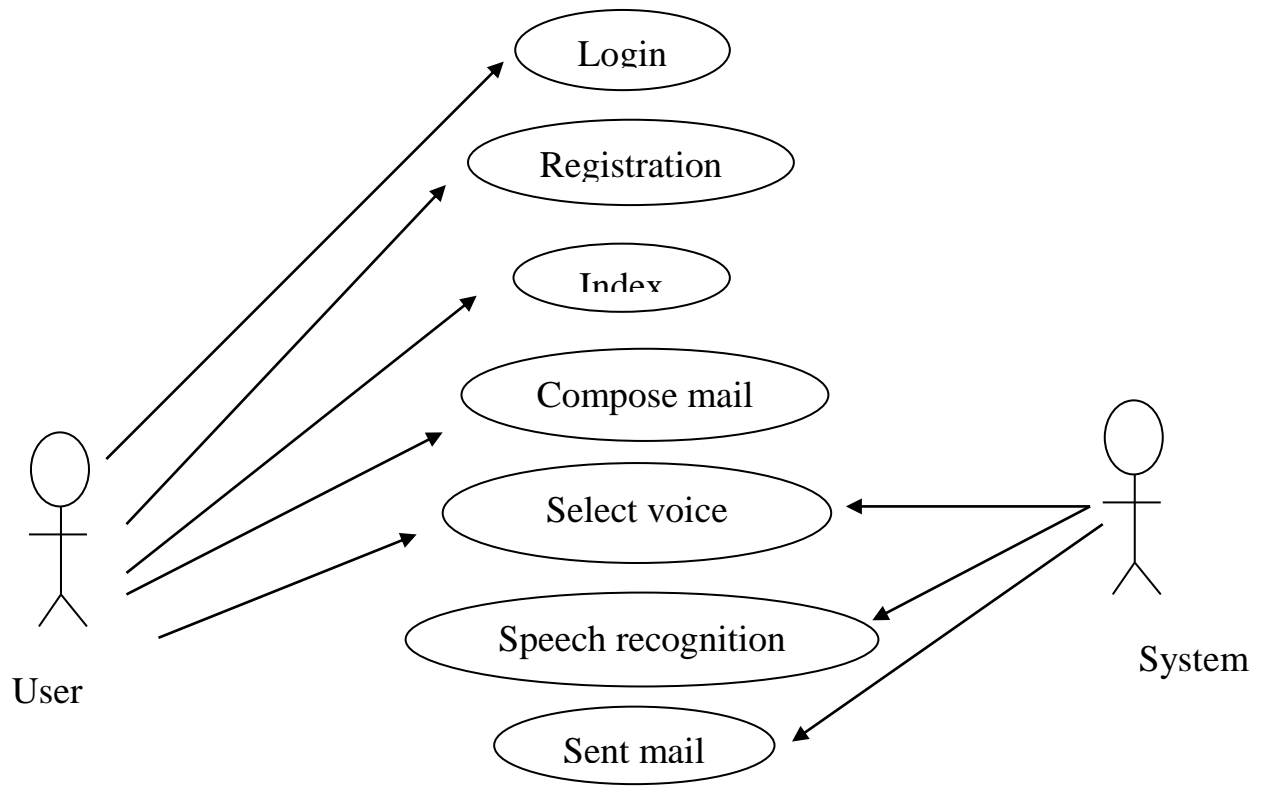


### 5.3 DATA FLOW DIAGRAM





## 5.4 USE CASE DIAGRAM



## **6. SYSTEM STUDY**

### **6.1 FEASIBILITY STUDY**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ◆ ECONOMICAL FEASIBILITY
- ◆ TECHNICAL FEASIBILITY
- ◆ SOCIAL FEASIBILITY

#### **ECONOMICAL FEASIBILITY**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

#### **TECHNICAL FEASIBILITY**

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.



## **SOCIAL FEASIBILITY**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

## **7. TESTING AND IMPLEMENTATION**

### **7.1 TESTING**

Testing is a critical stage in software development life cycle (SDLC) and is vital to provide quality assurance and for ensuring reliability of software. Testing forms the first step in determining the error in the program testing is the process of executing a program with intent of finding error product.

#### **Unit Testing**

Unit testing focuses verification effort on the smallest unit of software design, the module. The unit testing we have is white box oriented and some modules the steps are conducted in parallel.

#### **White Box Testing**

This type of testing ensures that

- All independent paths have been exercised at least once
- All logical decisions have been exercised on their true and false sides
- All loops are executed at their boundaries and within their operational bounds
- All internal data structures have been exercised to assure their validity.

To follow the concept of white box testing we have tested each form .we have created independently to verify that Data flow is correct, All conditions are exercised to check their validity, All loops are executed on their boundaries.

#### **Conditional Testing**

In this part of the testing each of the conditions were tested to both true and false aspects. And all the resulting paths were tested. So that each path that may be generate on particular condition is traced to uncover any possible errors.

## **Data Flow Testing**

This type of testing selects the path of the program according to the location of definition and use of variables. This kind of testing was used only when some local variable were declared. The *definition-use chain* method was used in this type of testing. These were particularly useful in nested statements.

## **Loop Testing**

In this type of testing all the loops are tested to all the limits possible. The following exercise was adopted for all loops:

- All the loops were tested at their limits, just above them and just below them.
- All the loops were skipped at least once.
- For nested loops test the inner most loop first and then work outwards.
- For concatenated loops the values of dependent loops were set with the help of connected loop.
- Unstructured loops were resolved into nested loops or concatenated loops and tested as above.
- Each unit has been separately tested by the development team itself and all the input have been validated.

## **CLIENT SIDE VALIDATION**

Various client side validations are used to ensure on the client side that only valid data is entered. Client side validation saves server time and load to handle invalid data. Some checks imposed are:

- VBScript is used to ensure those required fields are filled with suitable data only. Maximum lengths of the fields of the forms are appropriately defined.
- Forms cannot be submitted without filling up the mandatory data so that manual mistakes of submitting empty fields that are mandatory can be sorted out at the client side to save the server time and load.
- Tab-indexes are set according to the need and taking into account the ease of user while working with the system.

## **SERVER SIDE VALIDATION**

Some checks cannot be applied at client side. Server side checks are necessary to save the system from failing and intimating the user that some invalid operation has been performed or the performed operation is restricted. Some of the server side checks imposed is:

- Server side constraint has been imposed to check for the validity of primary key and foreign key. A primary key value cannot be duplicated. Any attempt to duplicate the primary value results into a message intimating the user about those values through the forms using foreign key can be updated only of the existing foreign key values.
- User is intimating through appropriate messages about the successful operations or exceptions occurring at server side.
- Various Access Control Mechanisms have been built so that one user may not agitate upon another. Access permissions to various types of users are controlled according to the organizational structure. Only permitted users can log on to the system and can have access according to their category. User- name, passwords and permissions are controlled o the server side.
- Using server side validation, constraints on several restricted operations are imposed.

## **7.2 IMPLEMENTATION**

A software application in general is implemented after navigating the complete life cycle method of a project. Various life cycle processes such as requirement analysis, design phase, verification, testing and finally followed by the implementation phase results in a successful project management. The software application which is basically a web based application has been successfully implemented after passing various life cycle processes mentioned above.

As the software is to be implemented in a high standard industrial sector, various factors such as application environment, user management, security, reliability and finally performance are taken as key factors throughout the design phase. These factors are analyzed step by step and the positive as well as negative outcomes are noted down before the final implementation.

Security and authentication is maintained in both user level as well as the management level. The data is stored in MySQL, which is highly reliable and simpler to use, the user level security is managed with the help of password options and sessions, which finally ensures that all the transactions are made securely. The forms are designed in such a way that it is attractive, convenient and informative. Forms are designed in PHP with various features, which make the console output more pleasing. As the outputs are the most important sources of information to the users, better design will improve the system's convenience and effectiveness.

The application's validations are made, taken into account of the entry levels available in various modules. Possible restrictions like number formatting, date formatting and confirmations for both save and update options ensures the correct data to be fed into the database. Thus all the aspects are charted out and the complete project study is practically implemented successfully for the end users.

## **8. CONCLUSION AND FUTURE WORK**

In this project we have proposed a system which will help the visually impaired people to access email services efficiently. This system will help in overcoming some drawbacks that were earlier faced by the blind people in accessing emails. We have eliminated the concept of using keyboard shortcuts along with screen readers which will help reducing the cognitive load of remembering keyboard shortcuts. Also any naive user who does not know the location of keys on the keyboard need not worry as keyboard usage is eliminated. The user only needs to follow the instructions given by the IVR and use mouse clicks accordingly to get the respective services offered. Other than this the user might need to feed in information through voice inputs when specified.

## 9. APPENDIX

### 9.1 SOURCE CODE

```
<html>
  <head>
    <link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/css/bootstrap.min.css">
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.0/jquery.min.js"></script>

    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.0/js/bootstrap.min.js"></script>
    <title>Welocome</title>
    <style>
      .btn-primary{
        height: 100vh;
        width: 100vw;
      }
    </style>
  </head>
  <body>
    <div id="player">
      <audio autoplay hidden loop="true">
        <source src="Click_anywhere_file.mp3" type="audio/mpeg">
        If you're reading this, audio isn't supported.
      </audio>
    </div>

    <a class="btn btn-primary" href="file:///C:/Users/
/Desktop/college/email/google_signin.html" role="button"></a>
  </body>
</html>
```

```

<!DOCTYPE html>
<html>
    <head>
        <title>Form to send your email</title>
    </head>
    <body>
        <form method = "post" action="send_mail.php">
            To :
            <br>
            <input type="text" name="mail_to">
            <br>
            Subject :
            <br>
            <input type="text" name="sub">
            <br>
            Message :
            <br>
            <input type="text" name="mes">
            <br>
            <br>
            <input type="submit" value=send_email>
            <br>
        </form>
    </body>
</html>

<?php
$mailto = $_POST['mail_to'];
$mailSub = $_POST['sub'];
$mailMsg = $_POST['mes'];

    require 'PHPMailer-master/PHPMailerAutoload.php';
$mail = new PHPMailer();
$mail ->IsSmt();

```



```

$mail ->SMTPDebug = 0;
$mail ->SMTPAuth = true;
$mail ->SMTPSecure = 'tls';
$mail ->Host = "smtp.gmail.com";
$mail ->Port = 587; // or 587
$mail ->IsHTML(true);
$mail ->Username = "mansimransingh.anand2016@vitstudent.ac.in";
$mail ->Password = "Hi@Vel17Msa_vit";
$mail ->SetFrom("mansimransingh.anand2016@vitstudent.ac.in");
$mail ->Subject = $mailSub;
$mail ->Body = $mailMsg;
$mail ->AddAddress($mailto);

if(!$mail->Send())
{
    echo "Mail Not Sent";
}
else
{
    echo "Mail Sent";
}
?>
<html>

<head>
    <title>Sending HTML email using PHP</title>
</head>

<body>

    <?php

        $to = "mansimransinghanand@gmail.com";
        $subject = "This is subject";

```

```
$message = "<b>This is HTML message.</b>";
```

```
$message .= "<h1>This is headline.</h1>";
```

```
$header = "From:mansimransinghanand@gmail.com\r\n";
```

```
$header .= "Cc:mansimransingh.anand2016@vitstudent.ac.in\r\n";
```

```
$header .= "MIME-Version: 1.0\r\n";
```

```
$header .= "Content-type: text/html\r\n";
```

```
$retval = mail ($to,$subject,$message,$header);
```

```
if( $retval == true ) {
```

```
    echo "Message sent successfully...";
```

```
}else {
```

```
    echo "Message could not be sent...";
```

```
}
```

```
?>
```

```
</body>
```

```
</html>
```

```
<?php
```

```
/**
```

```
 * PHPMailer SPL autoloader.
```

```
 * @param string $classname The name of the class to load
```

```
 */
```

```
function PHPMailerAutoload($classname)
```

```
{
```

```
    //Can't use __DIR__ as it's only in PHP 5.3+
```

```
    $filename =
```

```
dirname(__FILE__).DIRECTORY_SEPARATOR.'class.'.strtolower($classname).'.ph
```

```
p';
```

```
    if (is_readable($filename)) {
```

```
        require $filename;
```

```
    }
```

```

}

if (version_compare(PHP_VERSION, '5.1.2', '>=')) {
    //SPL autoloading was introduced in PHP 5.1.2
    if (version_compare(PHP_VERSION, '5.3.0', '>=')) {
        spl_autoload_register('PHPMailerAutoload', true, true);
    } else {
        spl_autoload_register('PHPMailerAutoload');
    }
} else {
    /**
     * Fall back to traditional autoload for old PHP versions
     * @param string $classname The name of the class to load
     */

    # removed line 51-54 by mansimran because of error

    #Deprecated: __autoload() is deprecated, use spl_autoload_register() instead in
    C:\xampp\htdocs\email_1\Voice-based-E-mail\PHPMailer-
    master\PHPMailerAutoload.php on line 45

    /**function __autoload($classname)
    {
        PHPMailerAutoload($classname);
    }

    */
}

<?php
class PHPMailerOAuth extends PHPMailer
{
    /**

```

```

    * The OAuth user's email address
    * @var string
    */
    public $oauthUserEmail = "";

    /**
     * The OAuth refresh token
     * @var string
     */
    public $oauthRefreshToken = "";

    /**
     * The OAuth client ID
     * @var string
     */
    public $oauthClientId = "";

    /**
     * The OAuth client secret
     * @var string
     */
    public $oauthClientSecret = "";

    /**
     * An instance of the PHPMailerOAuthGoogle class.
     * @var PHPMailerOAuthGoogle
     * @access protected
     */
    protected $oauth = null;

    /**
     * Get a PHPMailerOAuthGoogle instance to use.
     * @return PHPMailerOAuthGoogle
     */

```

```

public function getOAUTHInstance()
{
    if (!is_object($this->oauth)) {
        $this->oauth = new PHPMailerOAuthGoogle(
            $this->oauthUserEmail,
            $this->oauthClientSecret,
            $this->oauthClientId,
            $this->oauthRefreshToken
        );
    }
    return $this->oauth;
}

/**
 * Initiate a connection to an SMTP server.
 * Overrides the original smtpConnect method to add support for OAuth.
 * @param array $options An array of options compatible with
stream_context_create()
 * @uses SMTP
 * @access public
 * @return bool
 */
public function smtpConnect($options = array())
{
    if (is_null($this->smtp)) {
        $this->smtp = $this->getSMTPInstance();
    }

    if (is_null($this->oauth)) {
        $this->oauth = $this->getOAUTHInstance();
    }

    // Already connected?
    if ($this->smtp->connected()) {

```

```

        return true;
    }

    $this->smtp->setTimeout($this->Timeout);
    $this->smtp->setDebugLevel($this->SMTPDebug);
    $this->smtp->setDebugOutput($this->Debugoutput);
    $this->smtp->setVerp($this->do_verp);
    $hosts = explode(';', $this->Host);
    $lastexception = null;

    foreach ($hosts as $hostentry) {
        $hostinfo = array();
        if (!preg_match('/^((ssl|tls):\\\/\\\/)*([a-zA-Z0-9\\.-]*)?:?([0-9]*)$/',
            trim($hostentry), $hostinfo)) {
            // Not a valid host entry
            continue;
        }
        // $hostinfo[2]: optional ssl or tls prefix
        // $hostinfo[3]: the hostname
        // $hostinfo[4]: optional port number
        // The host string prefix can temporarily override the current setting for
SMTPSecure
        // If it's not specified, the default value is used
        $prefix = "";
        $secure = $this->SMTPSecure;
        $tls = ($this->SMTPSecure == 'tls');
        if ('ssl' == $hostinfo[2] or (" == $hostinfo[2] and 'ssl' == $this->SMTPSecure))
        {
            $prefix = 'ssl:\/';
            $tls = false; // Can't have SSL and TLS at the same time
            $secure = 'ssl';
        } elseif ($hostinfo[2] == 'tls') {
            $tls = true;
            // tls doesn't use a prefix

```

```

        $secure = 'tls';
    }

    //Do we need the OpenSSL extension?
    $sslext = defined('OPENSSL_ALGO_SHA1');
    if ('tls' === $secure or 'ssl' === $secure) {

        //Check for an OpenSSL constant rather than using extension_loaded, which
        is sometimes disabled
        if (!$sslext) {
            throw new phpmailerException($this-
>lang('extension_missing').'openssl', self::STOP_CRITICAL);
        }
    }

    $host = $hostinfo[3];
    $port = $this->Port;
    $tport = (integer)$hostinfo[4];
    if ($tport > 0 and $tport < 65536) {
        $port = $tport;
    }

    if ($this->smtp->connect($prefix . $host, $port, $this->Timeout, $options)) {
        try {
            if ($this->Helo) {
                $hello = $this->Helo;
            } else {
                $hello = $this->serverHostname();
            }
            $this->smtp->hello($hello);
            //Automatically enable TLS encryption if:
            // * it's not disabled
            // * we have openssl extension
            // * we are not already using SSL
            // * the server offers STARTTLS
            if ($this->SMTPAutoTLS and $sslext and $secure != 'ssl' and $this-
>smtp->getServerExt('STARTTLS')) {
                $tls = true;
            }
        }
    }

```

```

    }
    if ($tls) {
        if (!$this->smtp->startTLS()) {
            throw new phpmailerException($this->lang('connect_host'));
        }
        // We must resend HELO after tls negotiation
        $this->smtp->hello($hello);
    }
    if ($this->SMTPAuth) {
        if (!$this->smtp->authenticate(
            $this->Username,
            $this->Password,
            $this->AuthType,
            $this->Realm,
            $this->Workstation,
            $this->oauth
        )
        ) {
            throw new phpmailerException($this->lang('authenticate'));
        }
    }
    return true;
} catch (phpmailerException $exc) {
    $lastexception = $exc;
    $this->edebg($exc->getMessage());
    // We must have connected, but then failed TLS or Auth, so close
connection nicely
    $this->smtp->quit();
}
}

// If we get here, all connection attempts have failed, so close connection hard
$this->smtp->close();
// As we've caught all exceptions, just report whatever the last one was

```



```

        if ($this->exceptions and !is_null($lastexception)) {
            throw $lastexception;
        }
        return false;
    }
}

<?php
namespace League\OAuth2\Client\Provider;
require 'vendor/autoload.php';

use League\OAuth2\Client\Provider\Exception\IdentityProviderException;
use League\OAuth2\Client\Token\AccessToken;
use League\OAuth2\Client\Tool\BearerAuthorizationTrait;
use Psr\Http\Message\ResponseInterface;

session_start();

//If this automatic URL doesn't work, set it yourself manually
$redirectUri = isset($_SERVER['HTTPS']) ? 'https://' : 'http://' .
$_SERVER['HTTP_HOST'] . $_SERVER['PHP_SELF'];
//$redirectUri = 'http://localhost/phpmailer/get_oauth_token.php';

//These details obtained are by setting up app in Google developer console.
$clientId = 'RANDOMCHARS-----duv1n2.apps.googleusercontent.com';
$clientSecret = 'RANDOMCHARS-----lGyjPcRtvP';

class Google extends AbstractProvider
{
    use BearerAuthorizationTrait;

    const ACCESS_TOKEN_RESOURCE_OWNER_ID = 'id';
    protected $accessToken;
    protected $hostedDomain;
    protected $scope;

```

```

public function getBaseAuthorizationUrl()
{
    return 'https://accounts.google.com/o/oauth2/auth';
}

public function getBaseAccessTokenUrl(array $params)
{
    return 'https://accounts.google.com/o/oauth2/token';
}

public function getResourceOwnerDetailsUrl(AccessToken $token)
{
    return '';
}

protected function getAuthorizationParameters(array $options)
{
    if (is_array($this->scope)) {
        $separator = $this->getScopeSeparator();
        $this->scope = implode($separator, $this->scope);
    }

    $params = array_merge(
        parent::getAuthorizationParameters($options),
        array_filter([
            'hd'      => $this->hostedDomain,
            'access_type' => $this->accessType,
            'scope'    => $this->scope,
            // if the user is logged in with more than one account ask which one to use
            // for the login!
            'authuser' => '-1'
        ])
    );
}

```

```

        return $params;
    }

```

```

protected function getDefaultScopes()
{
    return [
        'email',
        'openid',
        'profile',
    ];
}

```

```

protected function getScopeSeparator()
{
    return ' ';
}

```

```

protected function checkResponse(ResponseInterface $response, $data)
{
    if (!empty($data['error'])) {
        $code = 0;
        $error = $data['error'];

        if (is_array($error)) {
            $code = $error['code'];
            $error = $error['message'];
        }

        throw new IdentityProviderException($error, $code, $data);
    }
}

```

```

protected function createResourceOwner(array $response, AccessToken $token)
{

```

```

        return new GoogleUser($response);
    }
}

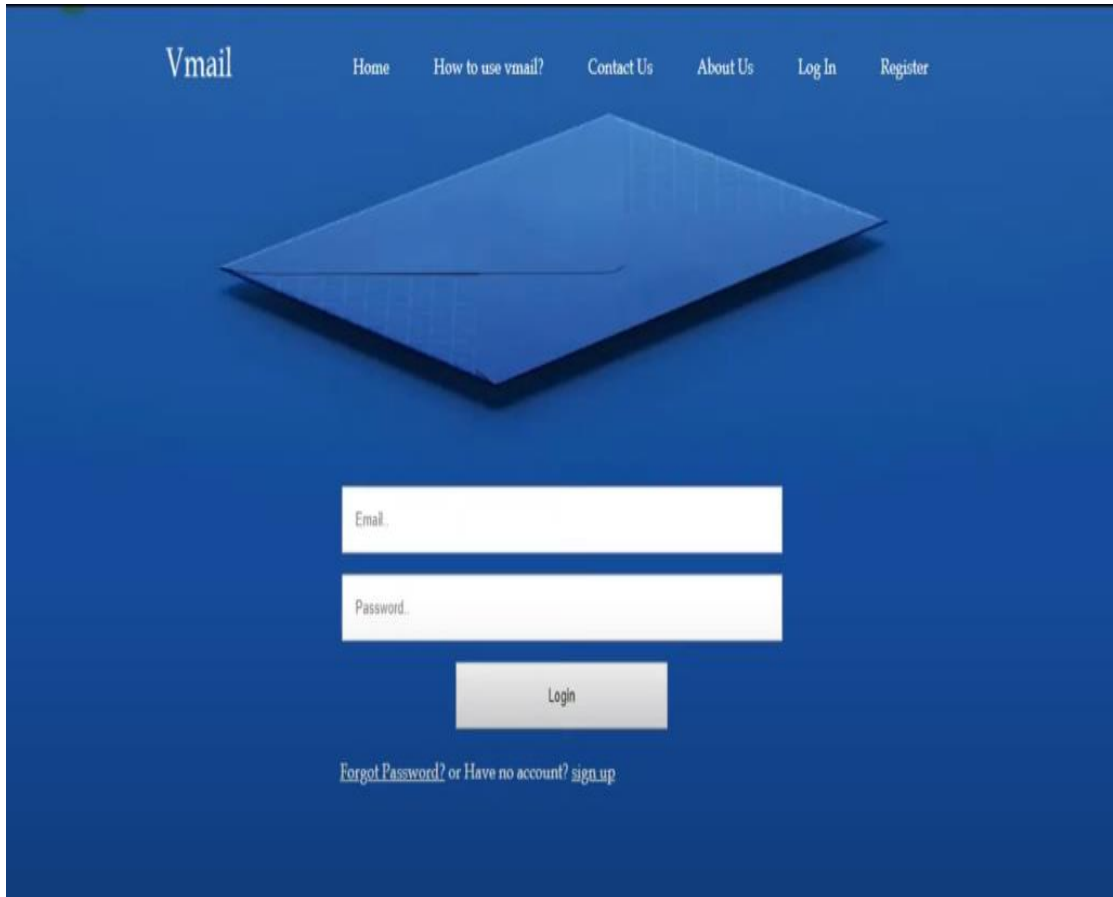
//Set Redirect URI in Developer Console as
[https/http]://<yourdomain>/<folder>/get_oauth_token.php
$provider = new Google(
    array(
        'clientId' => $clientId,
        'clientSecret' => $clientSecret,
        'redirectUri' => $redirectUri,
        'scope' => array('https://mail.google.com/'),
        'accessType' => 'offline'
    )
);

if (!isset($_GET['code'])) {
    // If we don't have an authorization code then get one
    $authUrl = $provider->getAuthorizationUrl();
    $_SESSION['oauth2state'] = $provider->getState();
    header('Location: ' . $authUrl);
    exit;
    // Check given state against previously stored one to mitigate CSRF attack
} elseif (empty($_GET['state']) || ($_GET['state'] !== $_SESSION['oauth2state'])) {
    unset($_SESSION['oauth2state']);
    exit('Invalid state');
} else {
    // Try to get an access token (using the authorization code grant)
    $token = $provider->getAccessToken(
        'authorization_code',
        array(
            'code' => $_GET['code']
        )
    )

```

```
);  
  
// Use this to get a new access token if the old one expires  
echo 'Refresh Token: ' . $token->getRefreshToken();  
}
```

## 9.2 SCREEN SHOTS




The screenshot shows the Vmail login interface. At the top, the 'Vmail' logo is on the left, and a navigation menu with links for 'Home', 'How to use vmail?', 'Contact Us', 'About Us', 'Log In', and 'Register' is on the right. The background is a solid blue color. In the center, there is a large, 3D-rendered blue envelope icon. Below the envelope, there are two white input fields: the first is labeled 'Email' and the second is labeled 'Password'. A grey 'Login' button is positioned below the password field. At the bottom of the form area, there is a link that reads 'Forgot Password? or Have no account? [sign up](#)'.

Reload this page

Vmail

[Home](#)[How to use vmail?](#)[Contact Us](#)[About Us](#)[Log In](#)[Register](#)



First Name.

Last Name.

Email.

Password.


Confirm Password.

Phone Number.

Birthdate.

Gender.

Sign Up



abc xyz


abc@gmail.com \*\*\*\*\*

\*\*\*\*\* 0 192 1178 300


12 July 1995 male

Sign Up

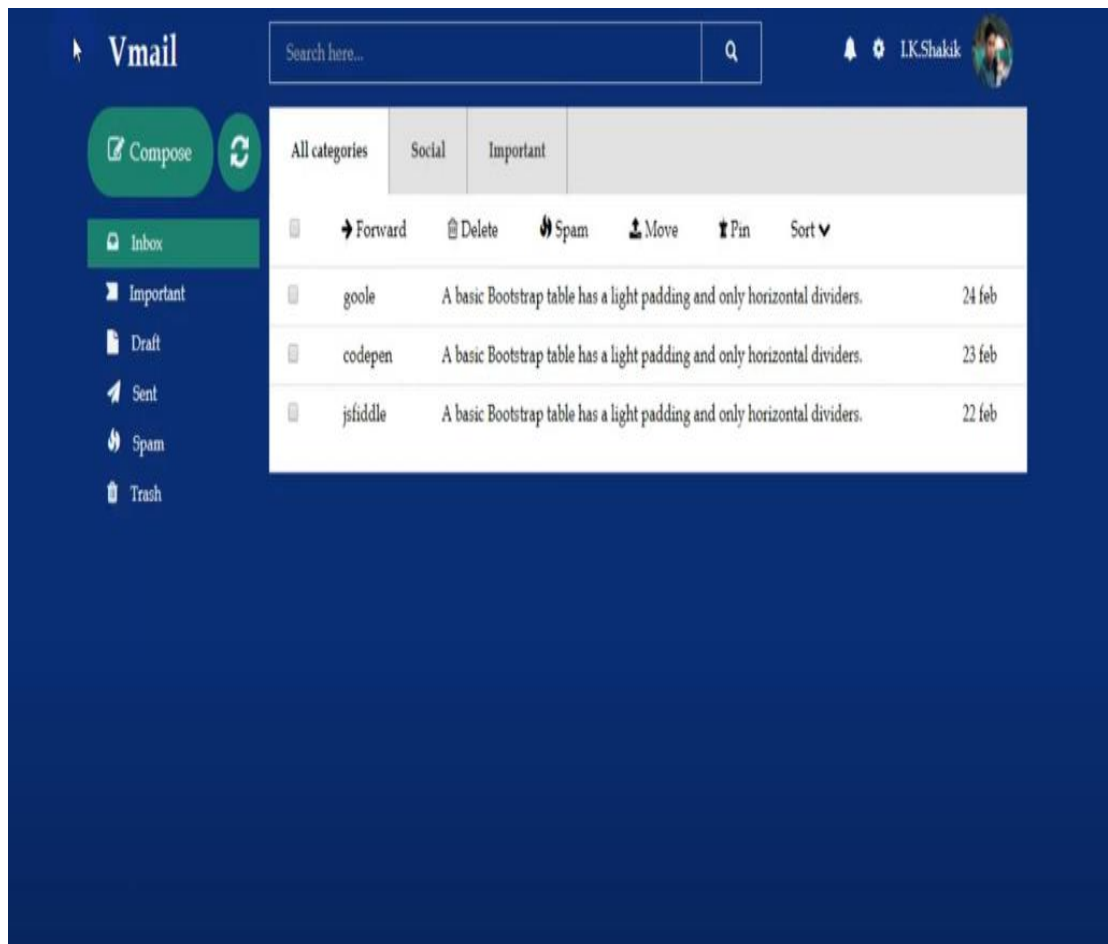


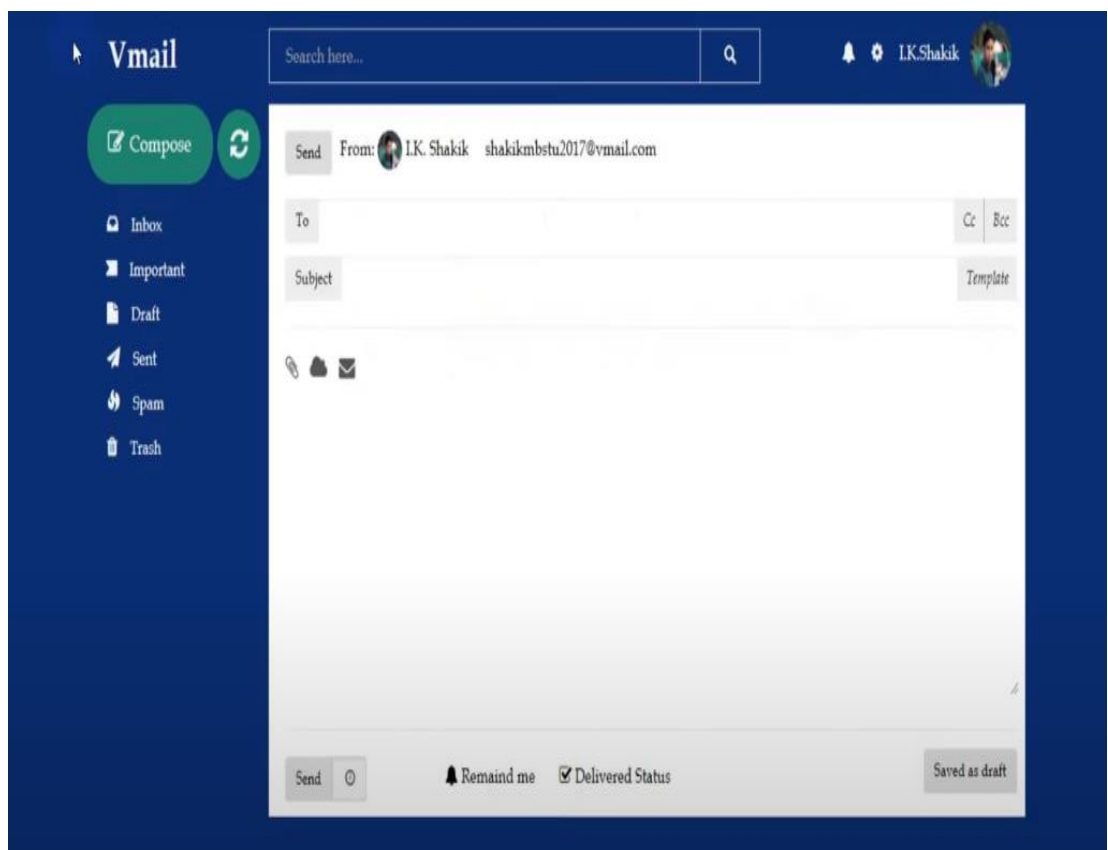
Vmail

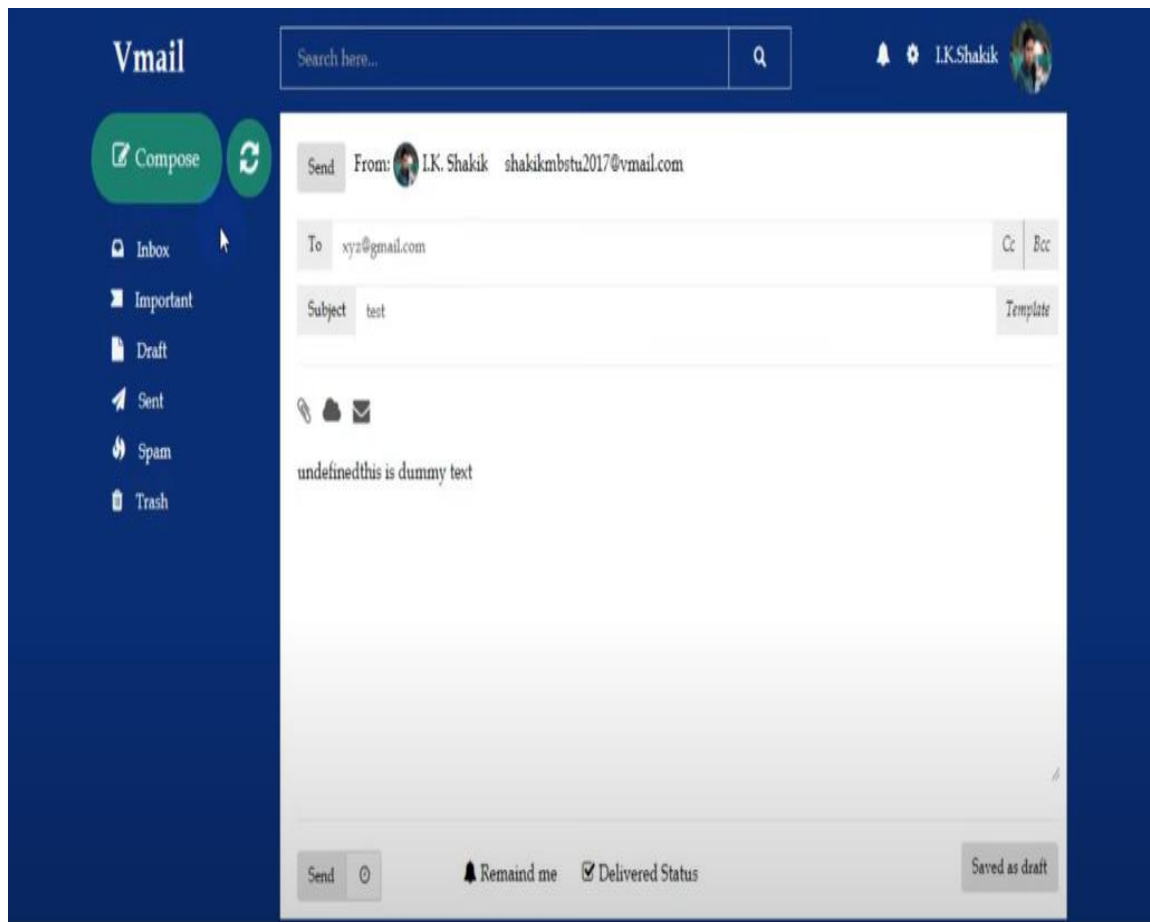
[Home](#) [How to use vmail?](#) [Contact Us](#) [About Us](#) [Log In](#) [Resister](#)

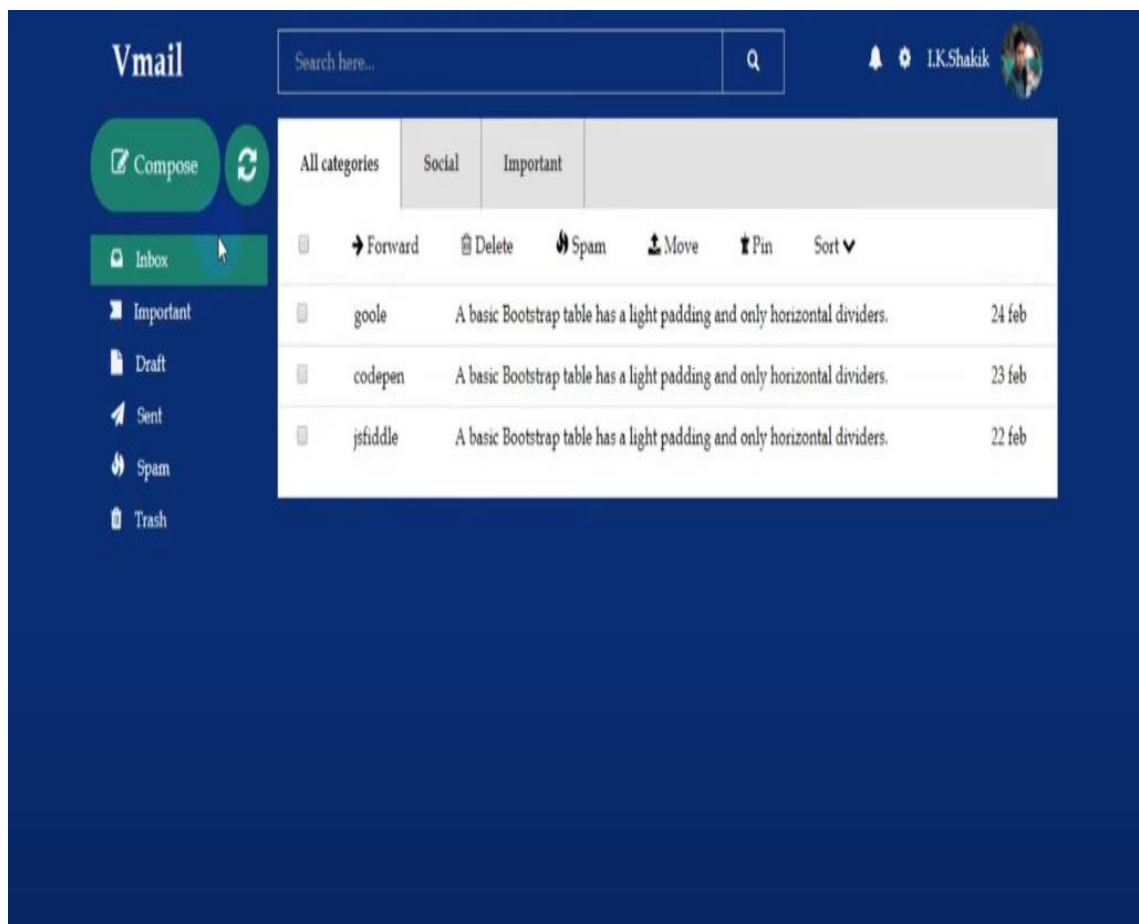


[Forgot Password?](#) or [Have no account? sign up](#)









## **10. BIBLIOGRAPHY**

### **BOOKS**

1. PHP Objects, Patterns, and Practice (Paperback) by Matt Zandstra
2. PHP and MySQL Web Development (Developer's Library) by Luke Welling
3. Programming PHP (Paperback) by RasmusLerdorf published 2002
4. PHP Object-Oriented Solutions (Paperback) by David Powers published 2008
5. PHP Cookbook (Paperback) by Adam Trachtenberg published 2002
6. Modern PHP: New Features and Good Practices (Paperback) by Josh Lockhart published 2015
7. Head First PHP & MySQL (Paperback) by Lynn Beighley
8. Essential PHP Security (Paperback) by Chris Shiflett published 2005
9. Learning PHP Design Patterns (Paperback) by William Sanders published 2013
10. Professional PHP Programming (Paperback) by Sascha Schumann published 1999
11. PHP Master (Paperback) by Davey Shafik published 2011
12. Web Database Applications with PHP and MySQL (Paperback) by Hugh E. Williams

### **WEBSITES**

1. <https://www.w3schools.com/php>
2. <http://php.net/manual/en/tutorial.php>
3. <https://www.thoughtco.com/create-links-in-php-2693950>
4. <https://www.tutorialrepublic.com/html-tutorial/html-links.php>
5. <https://ryanstutorials.net/html-tutorial/html-links.php>