# **1. INTRODUCTION**

# “Gaming Aegis” is a web and android based application focused mainly on gamers in Kerala. Now a days gamming has become a profession in the world. The project is mainly focused on players playing competitive matches in multiplayer games. The application thus provides the best platform to improve their skills and game sense. The players can register in the tournaments easily .

Technologies used in “GAMING AGEIS” are JSP, HTML, Android Studio 3.0 as Front end and MySQL as Back end

**JSP** technology is used to create web application just like Servlet technology. It can be thought of as an extension to Servlet because it provides more functionality than servlet such as expression language, JSTL, etc. A JSP page consists of HTML tags and JSP tags. The JSP pages are easier to maintain than Servlet because we can separate designing and development. It provides some additional features such as Expression Language, Custom Tags, etc.

**MySQL** is a freely available open source Relational Database Management System (RDBMS) that uses Structured Query Language (**SQL**). **SQL** is the most popular language for adding, accessing and managing content in a database. It is most noted for its quick processing, proven reliability, ease and flexibility of use.

# MODULES

* Admin Module
* Sub admin (District wase)
* Community
* Players
* Guest

## Admin

* Login with valid user name and password
* Dashboard
* Manage Users
* Admin can view and verify users
* Reports
* Admin can view players list
* Admin can view community lists
* Admin can view Subadmin list
* Admin can view and manage tournament
* Logout

# Sub Admin (District Wase)

* Login with valid username and password
* Homepage
* My Profile
* Edit Profile
* Change Password
* Manage players
* Manage community
* Manage Tournaments
* Logout

## Community

* Login with valid username and password
* Homepage
* My Profile
* Edit Profile
* Change password
* Create tournaments
* Logout

## Players

* Login with valid username and password
* Home page
* My Profile
* Edit Profile
* Change password
* Apply tournament
* Contact other users through direct chatting
* Logout

## Guest

* Login
* Player Registration
* Community Registration
* About us
* Contact us

**2. ABOUT THE ORGANISATION**

PCPL Technologies is a global IT solution organization servicing customers in North America, Europe, Asia and Australia. Our focus on select industry verticals in Banking, Financial Services and Insurance, Travel and Transportation and Retail and Manufacturing helped us create differentiation through specialization.

The company has been assessed at Level 5 of SEMI-CMMI, the highest maturity level as per the Software Engineering Institute’s Capability, Maturity Model as well as Level 5 of the people CMM framework. It has an ISO 9001:2000 certification from KPMG, UK and also confirms to the ISO 27001 information security management certification. The company’s Thailand operations are in confirmation with the international ISO 20000 IT management standard.

PCPL, the brainchild of two, young Indian entrepreneurs, pioneered and nurtured the concept of high quality IT education in India. Set up in 1981, PCPL has trained one out every three professionals in the country and become a beacon in the global IT revolution. From introducing computers to the people of India, to providing advanced IT skills to the student and professional, PCPL has evolved into a training powerhouse. While our special–priced, IT program have enabled ordinary citizens to achieve computer and internet literacy, our career education has shaped the lives of millions of individuals.

The immense contribution of PCPL leaders to the IT education realm lies in their revolutionary, “out-of-the box” thinking, which helped them discern a market need well before its time and their belief in the power of R&D to spur innovation in instructional design and delivery.

**3. SYSTEM ANALYSIS**

System Analysis is a detailed study of the various functions performed by the system and the relationships within and outside of the system. It is the way of studying a system with an eye on solving its problem using computers. It is the most essential part of the project development.

In this phase, the problem is identified and an alternate system solution for solving it is recognized. System Analysis not only includes the process of synthesis, which is a process of putting parts together to form a new product, but also the requirement identification and specification.

During analysis, data are collected on the available files, decision points, and transactions handled by the present system. Training experience and common sense are required for the collection of the information needed to do the analysis Data flow diagrams, interviews, on-site observations and questionnaires are examples.

The System Analysis includes two stages: Preliminary Analysis and Detailed Analysis. Preliminary Analysis includes a quick look at what is needed. Detailed Analysis includes an in-depth look at the system and analyses the costs and benefits. The costs and benefits of each alternative guide the selection of the best system for the job. Cost/Benefit Analysis identifies the costs and benefits of a given system and categorizes them for analysis.

Once analysis is completed, the analyst has a firm understanding of what is to be done. The next step is to decide how the problem might be solved. Thus, in system design, we move from the logical to the physical aspects of the life cycle. To analyses a system one has to study the system work in detail, before designing to the appropriate computer based system that will meet all requirements of the system.

There are a number of different approaches to system analysis. When a computer-based information system is developed, systems analysis (according to the Waterfall model) would constitute the following steps:

* The development of a feasibility study, involving determining whether a project is economically, socially, technologically and organizationally feasible.
* Conducting fact-finding measures, designed to ascertain the requirements of the system's end-users. These typically span interviews, questionnaires, or visual observations of work on the existing system.
* Gauging how the end-users would operate the system (in terms of general experience in using computer hardware or software), what the system would be used for and so on.

Techniques such as interviews, questionnaires etc. can be used for the detailed study of these processes. The data collected by these sources must be scrutinized to arrive at a conclusion. The conclusion is an understanding of how the system functions. This system is called the ***Existing System***. The Existing system is then subjected to close observation and the problem areas are identified. The designer now functions as a problem solver and tries to sort out the difficulties that the enterprise faces. The solutions are given as a proposal which is the ***Proposed System.*** The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is then presented to the user for an endorsement by the user. The proposal is reviewed on user request and suitable changes are made. This is a loop that ends as soon as the user is satisfied with the proposal.

# 3.1 EXISTING SYSTEM

Currently it is hard for a newcomer to find a tournaments in multiplayer games in gamming community. The slot booking for a tournaments through WhatsApp, discord and google forms. Currently no public platform for gamming community

# LMITATIONS OF EXISTING SYSTEM

* Skilled players will not get the opportunity
* Hard to find tournaments to a newcomer
* Registration through WhatsApp make more effort to the organizer

# 3.2 PROPOSED SYSTEM

The aim of the proposed system is to provide the opportunity to a user to find more tournaments and events organized by others community .This system makes it easier for both new comers and the ones who already in gamming community to find the right tournaments or events they want. The system will also act as a community for all those who interested in gamming.

# ADVANTAGES OF PROPOSED SYSTEM

* Easy access: The user can access the details anytime anywhere in the world.
* Multiple user interface: At a time different users can access the proposed system.
* Create opportunities for new comers.
* Makes it easier to find new better tournaments
* Gaming community will become more accessible to commoners.
* New and different tournaments will come.
* User friendly and interactive

**4. SOFTWARE REQUIREMENT SPECIFICATION (SRS)**

A **Software Requirement Specification** (SRS) is a detailed description of a software system to be developed with its functional and nonfunctional requirements. The SRS developed based the agreement between customer and contractors. It may include the use cases of how user is going to interact with software system. The software requirement specification consists of all necessary requirements required for project development. To develop the software system we should have clear understanding of software system. To achieve this we need continuous communication with customers to gather all requirements.

A good SRS defines how software system will interact with all internal modules, hardware, communication with other programs, and human user interactions with wide range of real life scenarios. Using the software requirement specification (SRS) document on QA lead, managers creates test plan. It is very important that testers must be cleared with every details specified in this document in order to avoid faults in test cases and its expected results.

**WHAT IS THE PROBLEM TO BE SOLVED?**

* Speed up the users registration process
* Provision to post complaints
* Minimize the human effort
* Speed up the processing of data

**CUSTOMER REQUIREMENTS**

* The system should have a login screen.
* The system should display the essential details about users
* The system should be able to view, edit and delete.
* The system should have users with unique id.
* The system should display the basic information of users.
* The system should display the category and subcategory of the users
* System should have a provision login form for administrator.
* The system provides the feature of online registration.
* The login page of the system shall allow a provision to reset the password.
* The system shall allow the admin to edit the details of users

**4.1 FUNCTIONAL REQUIREMENTS**

Software requirements are broadly divided classified as functional and nonfunctional requirements. Functional requirements are related to the expectations from the intended software. They describe what the software has to do. They are also called Product features. Sometimes, functional requirements may also specify what the software should not do.

There are four modules in this system.

* Admin Module
* Sub Admin Module
* Community
* Players

**1. Admin**

The main activities of Admin are

* Verification

The Admin can manage the tournaments, community and players that added in these websites. It is to help Administrator can identify the thing about being fake or not.

* Add District

The Admin can add the districts in Kerala.

* Add Place

The Admin can add the details of different places in districts Kerala.

* Add Mode

The Admin can add the types of Modes.

* Add Types

The Admin can add Different types.

* Add Slot

The Admin can add Different Slots.

* Add Player types

The Admin can add Different Player types.

**2. Sub Admin (District Wise)**

* Verification

The Sub Admin can manage the tournaments, community and players that added in these websites. It is to help Administrator can identify the thing about being fake or not.

**3. Community**

* Upload Tournaments

The community can upload different Tournaments.

**4. Player**

* Register Tournaments

The players can register different tournaments

**4.2. NON-FUNCTIONAL REQUIREMENTS**

**Security:**

The most important non-functional requirement, the system can be access by a valid user using their own username and password provided by the administrator.

**Availability:**

The system should be available in 24 hours.

**Usability:**

The system should be easy to understand by all the users.

**Maintainability:**

The system should have the capability to be modified.

**4.3 System Requirements**

**Hardware Specifications**

Processor : I3 or higher

System bus : 64bits

Memory : 8GB RAM or Higher

Hard disk : 250GB or Higher

Monitor : 14” LCD Monitor

Keyboard : 104 keys

Pointing Device : Two or Three Button Mouse

**Software Specifications**

Operating System : Windows7 or above

IDE : Net Beans IDE 8.0

Front End : JSP, HTML, Android Studio 3.0

Scripting Language : JavaScript

Back End : MySQL 5.0

Web Server : Glass Fish/Tomcat

Browser : Mozilla Firefox, Google Chrome

**FEATURES OF TECHNOLOGY**

* **JSP**

Java Server Pages (JSP) is a server-side programming technology that enables the creation of dynamic, platform-independent method for building Web-based applications. JSP have access to the entire family of Java APIs, including the JDBC API to access enterprise databases. Java Server Pages (JSP) is a technology for developing web pages that support dynamic content which helps developers insert java code in HTML pages by making use of special JSP tags, most of which start with <% and end with %>.

A Java Server Pages component is a type of Java servlet that is designed to fulfill the role of a user interface for a Java web application. Web developers write JSPs as text files that combine HTML or XHTML code, XML elements, and embedded JSP actions and commands.

Using JSP, you can collect input from users through web page forms, present records from a database or another source, and create web pages dynamically.

JSP tags can be used for a variety of purposes, such as retrieving information from a database or registering user preferences, accessing JavaBeans components, passing control between pages and sharing information between requests, pages etc.

**Why Use JSP?**

Java Server Pages often serve the same purpose as programs implemented using the Common Gateway Interface (CGI). But JSP offer several advantages in comparison with the CGI.

* Performance is significantly better because JSP allows embedding Dynamic Elements in HTML Pages itself instead of having a separate CGI files.
* JSP are always compiled before it's processed by the server unlike CGI/Perl which requires the server to load an interpreter and the target script each time the page is requested.
* Java Server Pages are built on top of the Java Servlets API, so like Servlets; JSP also has access to all the powerful Enterprise Java APIs, including JDBC, JNDI, EJB, JAXP etc.
* JSP pages can be used in combination with servlets that handle the business logic, the model supported by Java servlet template engines.

Finally, JSP is an integral part of Java EE, a complete platform for enterprise class applications. This means that JSP can play a part in the simplest applications to the most complex and demanding.

## Advantages of JSP:

Following is the list of other advantages of using JSP over other technologies:

* **vs. Active Server Pages (ASP):** The advantages of JSP are twofold. First, the dynamic part is written in Java, not Visual Basic or other MS specific language, so it is more powerful and easier to use. Second, it is portable to other operating systems and non-Microsoft Web servers.
* **vs. Pure Servlets:** It is more convenient to write (and to modify!) regular HTML than to have plenty of println statements that generate the HTML.
* **vs. Server-Side Includes (SSI):** SSI is really only intended for simple inclusions, not for "real" programs that use form data, make database connections, and the like.
* **vs. JavaScript:** JavaScript can generate HTML dynamically on the client but can hardly interact with the web server to perform complex tasks like database access and image processing etc.
* **vs. Static HTML:** Regular HTML, of course, cannot contain dynamic information.

## JSP Processing:

The following steps explain how the web server creates the web page using JSP:

* As with a normal page, your browser sends an HTTP request to the web server.
* The web server recognizes that the HTTP request is for a JSP page and forwards it to a JSP engine. This is done by using the URL or JSP page which ends with **.jsp** instead of .html.
* The JSP engine loads the JSP page from disk and converts it into a servlet content. This conversion is very simple in which all template text is converted to println( ) statements and all JSP elements are converted to Java code that implements the corresponding dynamic behavior of the page.
* The JSP engine compiles the servlet into an executable class and forwards the original request to a servlet engine.
* A part of the web server called the servlet engine loads the Servlet class and executes it. During execution, the servlet produces an output in HTML format, which the servlet engine passes to the web server inside an HTTP response.
* The web server forwards the HTTP response to your browser in terms of static HTML content.
* Finally web browser handles the dynamically generated HTML page inside the HTTP response exactly as if it were a static page.

MySQL is the most popular Open Source Relational SQL database management system. MySQL is one of the best RDBMS being used for developing web-based software applications.

* **Android**

Android is an open source and Linux-based **Operating System** for mobile devices such as smartphones and tablet computers. Android was developed by the *Open Handset Alliance*, led by Google, and other companies.

Android offers a unified approach to application development for mobile devices which means developers need only develop for Android, and their applications should be able to run on different devices powered by Android.

The first beta version of the Android Software Development Kit (SDK) was released by Google in 2007 where as the first commercial version, Android 1.0, was released in September 2008.

On June 27, 2012, at the Google I/O conference, Google announced the next Android version, 4.1 **Jelly Bean**. Jelly Bean is an incremental update, with the primary aim of improving the user interface, both in terms of functionality and performance.

The source code for Android is available under free and open source software licenses. Google publishes most of the code under the Apache License version 2.0 and the rest, Linux kernel changes, under the GNU General Public License version 2.

Android applications are usually developed in the Java language using the Android Software Development Kit.

Once developed, Android applications can be packaged easily and sold out either

through a store such as **Google Play**, **SlideME**, **Opera Mobile Store**, **Mobile Store**, **Mobango**, **F-droid** and the **Amazon Appstore**.

Android powers hundreds of millions of mobile devices in more than 190 countries around the world. It's the largest installed base of any mobile platform and growing fast. Every day more than 1 million new Android devices are activated worldwide.

## MySQL Database

MySQL is a fast, easy-to-use RDBMS being used for many small and big businesses. MySQL is developed, marketed, and supported by MySQL AB, which is a Swedish company. MySQL is becoming so popular because of many good reasons:

* MySQL is released under an open-source license. So you have nothing to pay to use it.
* MySQL is a very powerful program in its own right. It handles a large subset of the functionality of the most expensive and powerful database packages.
* MySQL uses a standard form of the well-known SQL data language.
* MySQL works on many operating systems and with many languages including PHP, PERL, C, C++, JAVA, etc.
* MySQL works very quickly and works well even with large data sets.
* MySQL is very friendly to PHP, the most appreciated language for web development.
* MySQL supports large databases, up to 50 million rows or more in a table. The default file size limit for a table is 4GB, but you can increase this (if your operating system can handle it) to a theoretical limit of 8 million terabytes (TB).
* MySQL is customizable. The open-source GPL license allows programmers to modify the MySQL software to fit their own specific environments.
* **TOMCAT SERVER**

WAMP Server stands for “Windows, Apache, MySQL and PHP”.WAMP is a variation of LAMP for Windows system and is often installed as a software bundle(Apache,MySQL and PHP).it is often used for web development and Internal testing, but may also be used to serve live websites.

The most important part of the WAMP package is Apache (or “Apache HTTP Server) which is used run the web server within windows.by running a local Apache web server on a Windows machine, a web developer can test webpages in a web browser without publishing them on live internet.

WAMP also include MySQL and PHP, which are two of the most common technologies used for creating dynamic websites. MySQL is a high speed database, while JSP is a scripting language that can be used to access data from the database. By installing these two components locally, a developer can build and test a dynamic website before publishing it to a public web server.

While Apache, MySQL and PHP are open source components that can be installed individually, they are usually installed together. One popular package is called “WAMPSERVER”, which provides a user friendly way to install and configure the “AMP” components on Windows.

* **NON FUNCTIONAL REQUIREMENTS**

Nonfunctional requirements are mostly quality requirements that stipulate how well the software does what it has to do. Non-functional requirements that are especially

important to users include specifications of desired performance, availability, reliability, flexibility and usability. Non-functional requirements for developers are maintainability, portability, and testability.

* **QUALITY ATTRIBUTES**

Quality attribute requirements such as those for performance, security, modifiability, reliability and usability have a significant influence on the software architecture of a system. Architects needs to understand their needs in terms of quality attributes. For example, they need to understand whether they will achieve deadlines in real time systems. What kinds of modifications are supported by their design and how the system will respond in the event of a failure? There are large and thriving attribute communities that studies various quality attributes but they each have their own language and sets of concepts. However, architects tend to think in terms of architectural patterns. What the architects needs is a characterization of architectural patterns in terms of factors that affect the various qualities attributes so that a software design can be understood in terms of those quality attributes.

In order to reason about architectural patterns in quality attribute terms, we must first precisely characterize the quality attribute requirements and then give examples of how to reason about architectural patterns. We characterize quality attributes using general scenarios codify architectural as quality attributes design primitives (or attribute primitives). Attribute primitives are the extension of our earlier work on Attribute-Based Architectural Styles.

**5. FEASIBILITY ANALYSIS**

A feasibility study is a test of a system proposal according to its workability, impact on the security of the organization, ability to meet user needs, and effective use of resources. The objective of a feasibility study is not to solve problem but to acquire a sense of its scope .During the study, the problem definition is crystallized and aspects of the problem to be included in the system are determined. The result of the feasibility study is a formal document detailing the nature and scope of the proposed system.

One of the important outcomes of the preliminary investigation is the determination of the feasibility of the system. These are different aspects of the feasibility study in the investigation phase. After reviewing the documents by selected personnel and investigating the various resources, the following are the three feasibilities.

Three key combinations are involved in the feasibility study. They are:

* Economic feasibility
* Technical feasibility
* Operational feasibility

**Economic Feasibility**

Economic analysis is the most frequently used method for evaluating the effectiveness of a candidate system. More commonly known as cost/benefit analysis, the procedure is to determine the benefit and saving that are expected from a candidate system and compare them with the term of time by automating the process of report generation. The system can be developed technically and if installed would still be good for the organization. The cost is found to be lesser compared to the benefits of the existing system. The workload of a user will decrease to half of the current workload. Hence the proposed system is found to be economic feasible “**70mm**”.

**Technical Feasibility**

Technical study is a study of hardware and software requirements. All the technical issues related to the proposed system is dealt during feasibility stage of preliminary investigation produced the following results.

**Operational Feasibility**

The developed system is completely driven and user friendly. Also the system is developed in Visual Basic, which is GUI. There is little need skill for new user to operate the software. Reports will be exactly as per the requirement. At the beginning of preliminary investigation work all the personnel approached responded positively this reduces the chance of resistance to the proposed system. Considering all the issue stated above makes the proposed system feasible.

**6. Data Flow Diagram (DFD)**

Data Flow Diagram is a network that describes the flow of data and processes that change, or transform, data throughout the system. This network is constructed by use a set of symbols that do not imply a physical implementation. It is a graphical tool for structured analysis of the system requirements. DFD models a system by using external entities from which data flows to a process, which transforms the data and creates, output-data-flows which go to other processes or external entities or files. Data in files may also flow to processes as inputs.

There are various symbols used in a DFD. Bubbles represent the processes. Named arrows indicate the data flow. External entities are represented by rectangles. Entities supplying data are known as sources and those that consume data are called sinks. Data are stored in a data store by a process in the system. Each component in a DFD is labelled with a descriptive name. Process names are further identified with a number.

The Data Flow Diagram shows the logical flow of a system and defines the boundaries of the system. For a candidate system, it describes the input (source), outputs (destination), database (files) and procedures (data flow), all in a format that meet the user‟s requirements.

The main merit of DFD is that it can provide an overview of system requirements, what data a system would process, what transformations of data are done, what files are used, and where the results flow.

This network is constructed by use a set of symbols that do not imply a physical implementation. It is a graphical tool for structured analysis of the system requirements. DFD models a system by using external entities from which data flows to a process, which transforms the data and creates, output-data-flows which go to other processes or external entities or files. External entities are represented by rectangles. Entities supplying data are known as sources and those that consume data are called sinks. Data are stored in a data store by a process in the system. It is a graphical tool for structured analysis of the system requirements. DFD models a system by using external entities from which data flows to a process, which transforms the data and creates, output-dataflows which go to other processes or external entities or files. Data in files may also flow to processes as inputs.

## Rules for constructing a Data Flow Diagram

1. Arrows should not cross each other
2. Squares, circles and files must bear names.
3. Decomposed data flow squares and circles can have same time
4. Choose meaningful names for data flow
5. Draw all data flows around the outside of the diagram

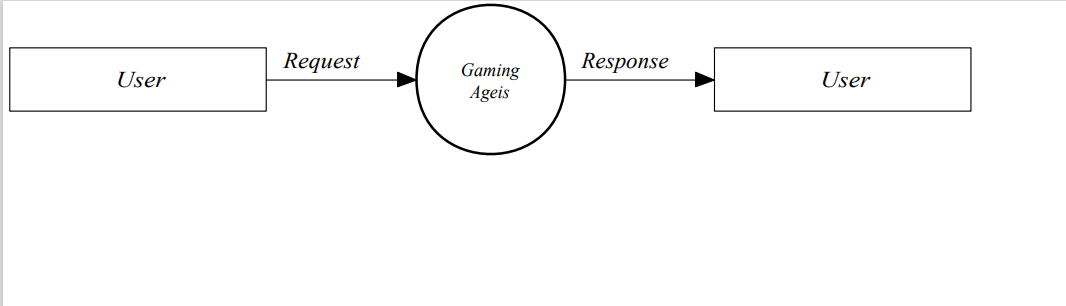
## Basic Data Flow Diagram Symbol

|  |  |
| --- | --- |
|  | A data flow is a route, which enables packets of data to travel from one point to another. Data may flow from a source to a process and from data store or process. An arrow line depicts the flow, with arrow head pointing in the direction of the flow. |
|  | Circles stands for process that converts data in to information. A process represents transformation where incoming data flows are changed into outgoing data flows. |
|  | A data store is a repository of data that is to be stored for use by a one or more process may be as simple as buffer or queue or sophisticated as relational database. They should have clear names. If a process merely uses the content of store and does not alter it, the arrowhead goes only from the store to the process. If a process alters the details in the store, then a doubleheaded arrow is used. |
|  | A source or sink is a person or part of an organization, which enters or receives information from the system, but is considered to be outside the contest of data flow model. |

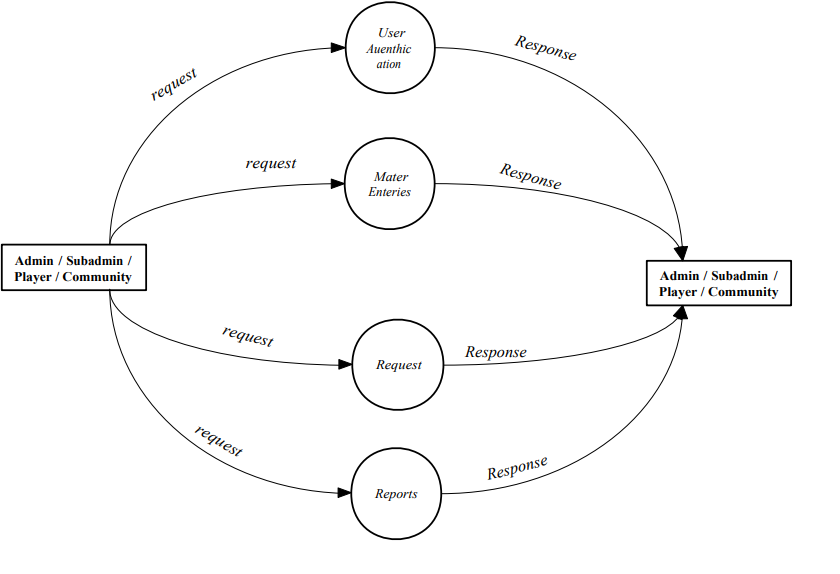
## Data Flow Diagram

Each component in a DFD is labelled with a descriptive name. Process name are further identified with number. Context level DFD is draw first. Then the process is decomposed into several elementary levels and is represented in the order of importance. A DFD describes what data flow (logical) rather than how they are processed, so it does not depend on hardware, software, and data structure or file organization.

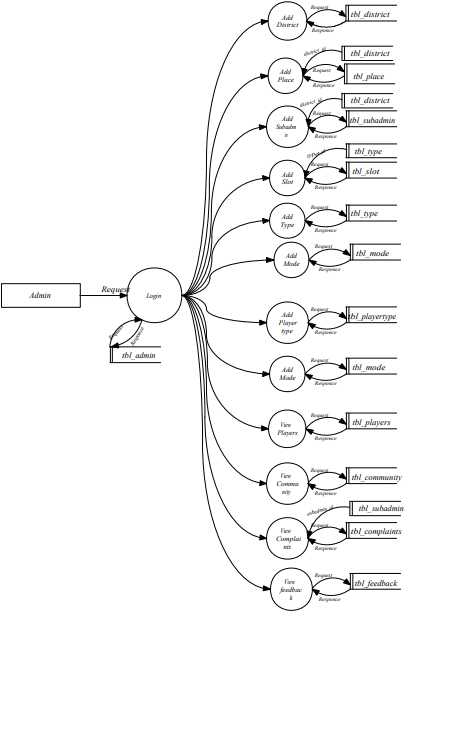
**Zeroth Level DFD**



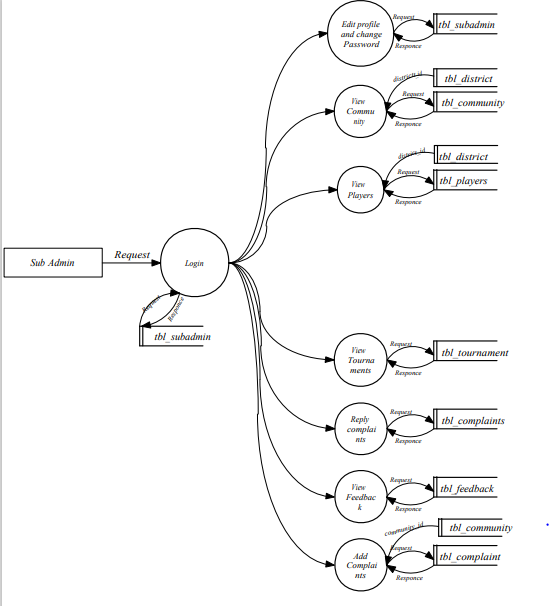
**First Level DFD**



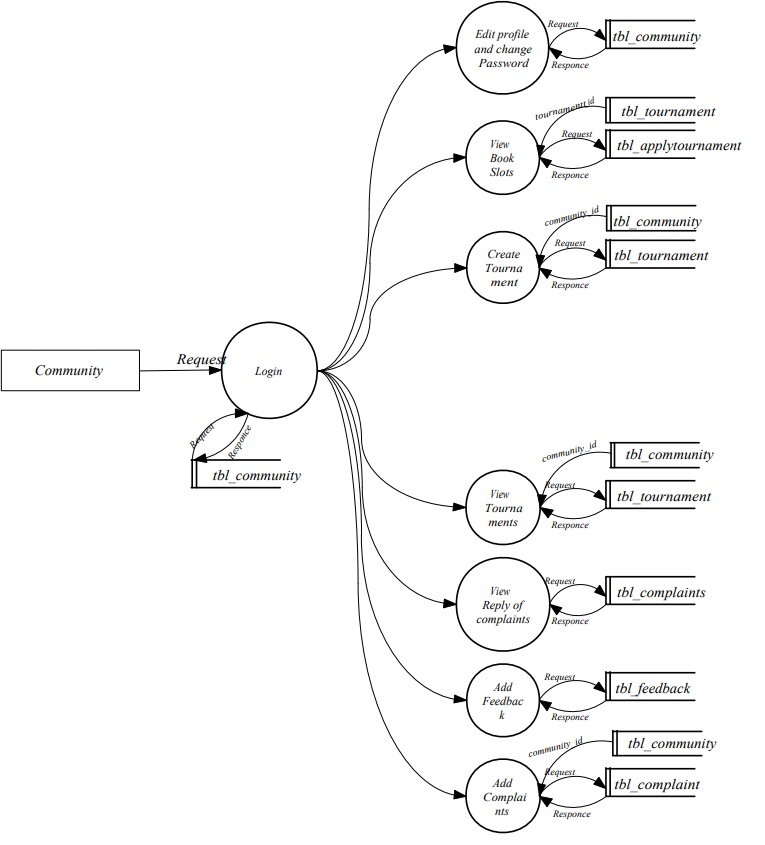
**Second Level DFD for Admin**



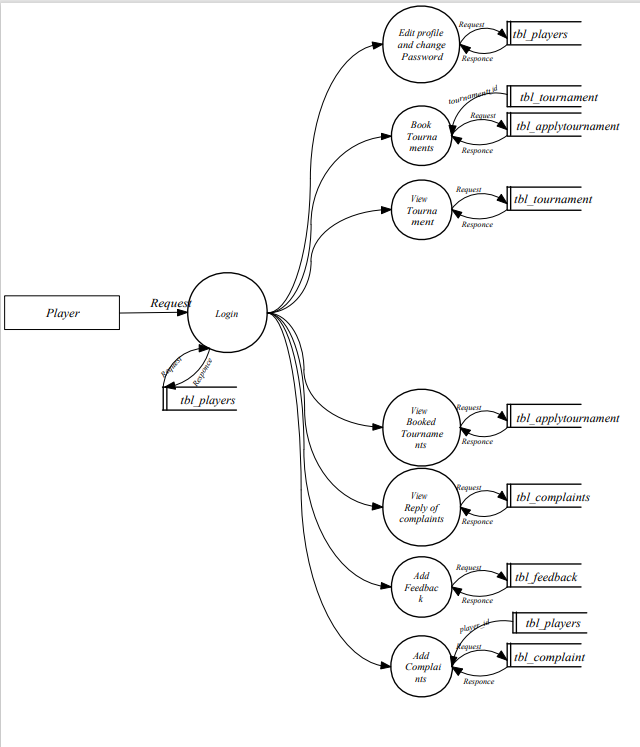
**Second Level DFD for SubAdmin**



**Second Level DFD for Community**



**Second Level DFD for Player**



**6.1 TABLE DESIGN**

Table is a collection of complete details about a particular subject. These data are saved in rows and Columns. The data of each Row are different units. Hence, rows are called RECORDS and Columns of each row are called FIELDS. Data is stored in tables, which is available in the backend the items and data, which are entered in the input, form id directly stored in this table using linking of database. We can link more than one table to input forms. We can collect the details from the different tables to display on the output.

1. **tbl\_district**

Description: This table is used to districts

Primary key: district\_id

Foreign key: Null

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.No** | **Fieldname** | **Data type** | **Constraints** | **Description** |
| 1 | district\_id | Integer | Primary key | Unique id of district |
| 2 | District\_name | Varchar(50) | Not null | Name of the District |

1. **tbl\_place**

Description: To store the places in each district

Primary key: place\_id

Foreign key: district\_id

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.No** | **Fieldname** | **Data type** | **Constraints** | **Description** |
| 1 | place\_id | Integer | Primary key | Unique id of place |
| 2 | place\_name | Varchar(50) | Not null | Name of place |
| 3 | district\_id | Varchar(50) | Foreign key | Id of the district |

1. **tbl\_playertype**

Description: To store the player type

Primary key: playertype\_id

Foreign key: Null

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.No** | **Fieldname** | **Data type** | **Constraints** | **Description** |
| 1 | playertype\_id | Integer | Primary key | Unique id of player type |
| 2 | player\_type | Varchar(50) | Not null | Name of the type |

1. **tbl\_mode**

Description: To store the game mode

Primary key: mode\_id

Foreign key: Null

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.No** | **Fieldname** | **Data type** | **Constraints** | **Description** |
| 1 | mode \_id | Integer | Primary key | Unique id of game mode |
| 2 | mode \_name | Varchar(50) | Not null | Name of the game mode |

1. **tbl\_type**

Description: To store the game type

Primary key:type\_id

Foreign key: Null

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.No** | **Fieldname** | **Data type** | **Constraints** | **Description** |
| 1 | type\_id | Integer | Primary key | Unique id of game type |
| 2 | type\_name | Varchar(50) | Not null | Name of the type |

1. **tbl\_slot**

Description: To store the slot

Primary key:slot \_id

Foreign key: type\_id

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.No** | **Fieldname** | **Data type** | **Constraints** | **Description** |
| 1 | slot \_id | Integer | Primary key | Unique id of slot |
| 2 | slot \_no | Varchar(50) | Not null | Number of the slot |
| 3 | type\_id | integer | Foreign key | Unique id of game type |

1. **tbl\_subadmin**

Description: To store the details of subadmin

Primary key:subadmin\_id

Foreign key: district\_id

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.No** | **Fieldname** | **Data type** | **Constraints** | **Description** |
| 1 | subadmin \_id | Integer | Primary key | Unique id of subadmin |
| 2 | subadmin \_name | Varchar(50) | Not null | Name of the subadmin |
| 3 | subadmin \_contact | Varchar(50) | Not null | Contact number subadmin |
| 4 | subadmin \_email | Varchar(50) | Not null | Email id of subadmin |
| 5 | subadmin \_gender | Varchar(50) | Not null | Gender of the subadmin |
| 6 | subadmin \_address | Varchar(50) | Not null | address of the subadmin |
| 7 | subadmin \_username | Varchar(50) | Not null | User name of the subadmin |
| 8 | subadmin \_password | Varchar(50) | Not null | Password of the subadmin |
| 9 | district\_id | integer | Foreign key | Unique id of district |

1. **tbl\_admin**

Description: To store the details of admin

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 1 | admin \_uname | Varchar(50) | Not null | User name of the admin |
| 2 | admin \_password | Varchar(50) | Not null | Password of the admin |

1. **tbl\_player**

Description: To store the details players

Primary key:player \_id

Foreign key: place\_id,playertype\_id

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.No** | **Fieldname** | **Data type** | **Constraints** | **Description** |
| 1 | player \_id | Integer | Primary key | Unique id of player |
| 2 | player \_name | Varchar(50) | Not null | Name of the tag |
| 3 | player \_contact | Varchar(50) | Not null | Contact number player |
| 4 | player \_email | Varchar(50) | Not null | Email id of player |
| 5 | player \_address | Varchar(50) | Not null | Address if the player |
| 6 | player \_photo | Varchar(50) | Not null | Photo of the player |
| 7 | player \_gender | Varchar(50) | Not null | Gender of the player |
| 8 | player \_uname | Varchar(50) | Not null | User name of the player |
| 9 | player \_password | Varchar(50) | Not null | Password of the player |
| 10 | game\_id | integer | Not null | Unique id of game |
| 11 | player\_dob | integer | Not null | Date of birth of player |
| 12 | place\_id | integer | Foreign key | Unique id of place |
| 13 | player\_status | integer | Not null | Status of the player |
| 14 | playertype \_id | integer | Foreign key | Unique id of player type |

1. **tbl\_community**

Description: To store the details community

Primary key:community \_id

Foreign key: district\_id

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.No** | **Fieldname** | **Data type** | **Constraints** | **Description** |
| 1 | community \_id | Integer | Primary key | Unique id of community |
| 2 | community \_name | Varchar(50) | Not null | Name of the community |
| 3 | community \_contact | Varchar(50) | Not null | Contact number community |
| 4 | community \_email | Varchar(50) | Not null | Email id of community |
| 5 | community \_logo | Varchar(50) | Not null | Logo of the community |
| 6 | community \_liscence | Varchar(50) | Not null | Liscence of the community |
| 7 | community \_username | Varchar(50) | Not null | User name of the community |
| 8 | community \_password | Varchar(50) | Not null | Password of the community |
| 9 | district\_id | integer | Foreign key | Unique id of district |
| 10 | community \_status | integer | Not null | Status of the community |

1. **tbl\_tournament**

Description: To store the details of tournament

Primary key:tournament\_id

Foreign key: mode\_id, type\_id, community\_id

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.No** | **Fieldname** | **Data type** | **Constraints** | **Description** |
| 1 | tournament \_id | Integer | Primary key | Unique id of tournament |
| 2 | tournament \_name | Varchar(50) | Not null | Name of the tournament |
| 3 | tournament \_status | Varchar(50) | Not null | Status of tournament |
| 4 | tournament \_date | Varchar(50) | Not null | date of tournament |
| 5 | tournament \_time | Varchar(50) | Not null | Time of the tournament |
| 6 | tournament \_description | Varchar(50) | Not null | description of the tournament |
| 7 | tournament \_fee | Varchar(50) | Not null | If it is free or paid the tournament |
| 8 | tournament \_link | Varchar(50) | Not null | Youtube link of the tournament |
| 9 | tournament \_fees | Varchar(50) | Not null | fees of the tournament |
| 10 | community\_id | integer | Foreign key | Unique id of community |
| 11 | mode\_id | integer | Foreign key | Unique id of game mode |
| 12 | type\_id | integer | Foreign key | Unique id of game type |

1. **tbl\_applytournament**

Description: To store the applt details of tournament

Primary key: **applytournament\_id**\_id

Foreign key: slot\_id, player\_id, tournament\_id

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.No** | **Fieldname** | **Data type** | **Constraints** | **Description** |
| 1 | applytournament \_id | Integer | Primary key | Unique id of apply tournament |
| 2 | applytournament \_status | Varchar(50) | Not null | Status of the apply tournament |
| 3 | tournament \_id | Integer | Not null | Unique id of tournament |
| 4 | slot\_id | Integer | Not null | Unique id of slot |
| 5 | slot \_status | Varchar(50) | Not null | Status of the slot |
| 6 | player \_id | Integer | Not null | Unique id of player |
| 7 | id | Varchar(50) | Not null | The id of the tournament |
| 8 | pass | Varchar(50) | Not null | The password of the tournament |

1. **tbl\_feedback**

Description: To store the feedback

Primary key: **feedback**\_id

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.No** | **Fieldname** | **Data type** | **Constraints** | **Description** |
| 1 | feedback \_id | Integer | Primary key | Unique id of feedback |
| 2 | feedback\_description | Varchar(50) | Not null | description of the feedback |
| 3 | feedback \_date | Integer | Not null | Date of feedback |

1. **tbl\_complaint**

Description: To store the details of complaints

Primary key:complaint\_id

Foreign key: subadmin\_id, player\_id, community\_id

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl.No** | **Fieldname** | **Data type** | **Constraints** | **Description** |
| 1 | complaint \_id | Integer | Primary key | Unique id of complaint |
| 2 | complaint \_description | Varchar(50) | Not null | description of the complaint |
| 3 | complaint \_reply | Varchar(50) | Not null | reply of complaint |
| 4 | complaint \_date | Varchar(50) | Not null | date of complaint |
| 5 | subadmin \_id | Integer | Not null | Unique id of subadmin |
| 6 | player \_id | Integer | Not null | Unique id of player |
| 7 | community \_id | Integer | Not null | Unique id of community |

1. **SYSTEM DESIGN**

The goal of design process is to produce a model or a representation of a moving  from the problem domain to the solution domain .In top level design focus is on deciding  which modules are needed for the system, the specification of these modules and how  these modules can be interconnected.In this project design technique used is top-down,  object-oriented dynamic modeling technique. A top-down design approach starts by  identifying the major components and iterating until the desired level of details is  achieved. In object oriented design technique, the modules in the design represent data  abstraction.

**7.1 INPUT DESIGN**

Input design is the process of converting user-oriented inputs to a computer-based  format. The quality of system determines the quality of system outputs. All the data entry  screens should be of interactive nature so that the user can directly input data according  to prompt messages.The input design determines whether the user can interact directly  with computer. Interactive input screens ensure the reliability and accuracy of the system.  The goal of designing input data is to make it free from logical errors. The input data is  also used for easy calculation of necessary functions.

Different forms are used for data screens in order to input data into the system.  The screen formats have options like add, edit and update with buttons, which is used for  easy input and retrieving information. All the fields are validated. If the user enters  invalid data appropriate messages are displayed.Usually, an incorrect output is produced  when the users provide incorrect inputs. In this system, the user who want to search the  data and have to provide much inputs. The design in this system is such a way that, the  user can search and select using simple mouse clicks. Then the user can select a record by  mouse click. Then the output screen display the selected record.

**7.2 OUTPUT DESIGN**

One of the most important features of the system for users is the output it  produces. Output design should improve the system relationship with the user and helps  in decision making. Without quality output, the entire system appears to be unnecessary  that users will avoid using. The objective of the output design is to define the control and  formats of all printed, documented, reports and screens that will be produced by the  system. Computer output is the most important and direct source of information to the  user. For many end users output is the main reason for developing the system and the  basis on which they will evaluate the usefulness of the application. Output generally  refers to the results that are generated by the system. The user must give a valid input to  get an accurate output. The outputs are the total number of function points, total effort  required, total time required for development and scheduling to develop that project.

# 8. SYSTEM TESTING AND IMPLEMENTATION

## TESTING

Coding conventions are a set of guidelines for a specific programming language that recommend programming style, practices and methods for each aspect of a piece program written in this language. These conventions usually cover file organization, indentation, comments, declarations, statements, white space, naming conventions, programming practices, programming principles, programming rules of thumb, architectural best practices, etc. These are guidelines for software structural quality. Software programmers are highly recommended to follow these guidelines to help improve the readability of their source code and make software maintenance easier.

## 8.1 TEST CASES

The objective of system testing is to ensure that all individual programs are working as expected, that the programs link together to meet the requirements specified and to ensure that the computer system and the associated clerical and other procedures work together. The initial phase of system testing is the responsibility of the analyst who determines what conditions are to be tested, generates test data, produced a schedule of expected results, runs the tests and compares the computer produced results with the expected results with the expected results. The analyst may also be involved in procedures testing. When the analyst is satisfied that the system is working properly, he hands it over to the users for testing. The importance of system testing by the user must be stressed. Ultimately it is the user must verify the system and give the go-ahead.

During testing, the system is used experimentally to ensure that the software does not fail, i.e., that it will run according to its specifications and in the way users expect it to. Special test data is input for processing (test plan) and the results are examined to locate unexpected results. A limited number of users may also be allowed to use the system so analysts can see whether they try to use it in unexpected ways. It is preferably to find these surprises before the organization implements the system and depends on it.

In many organizations, testing is performed by person other than those who write the original programs. Using persons who do not know how certain parts were designed or programmed ensures more complete and unbiased testing and more reliable software.

Parallel running is often regarded as the final phase of system testing. Since he parallel operation of two systems is very demanding in terms of user resources it should be embarked on only if the user is satisfied with the results of testing, it should not be started if problems are known to exist. Testing is the major quality control measure during software development. Its basic function is to detect errors in the software. Thus the goal of testing is to uncover requirement design and coding errors in the program.

Testing is the process of correcting a program with intends of finding an error.

Different types of testing are,

1. Unit Testing
2. Integrated Testing
3. Validation Testing
4. Output Testing
5. User Acceptance Testing

### Unit Testing

In computer programming, unit testing is a method by which individual units of source code, sets of one or more computer program modules together with associated control data, usage procedures, and operating procedures are tested to determine if they are fit for use. In this testing we test each module individual and integrated the overall system. Unit testing focuses verification efforts on the smaller unit of software design in the module. This is also known as module testing. The modules of the system are tested separately. The testing is carried out during programming stage itself. In this testing step each module is found to working satisfactory as regard to the expected output from the module. There are some validation checks for verifying the data input given by the user which both the formal and validity of the entered. It is very easy to find error debug the system.

These types of tests are usually written by developers as they work on code, to ensure that the specific function is working as expected. Unit testing is a software development process that involves synchronized application of a broad spectrum of defect prevention and detection strategies in order to reduce software development risks, time, and costs. It is performed by the software developer or engineer during the construction phase of the software development lifecycle.

### Integration Testing

Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. Software components may be integrated in an iterative way or all together ("big bang"). Normally the former is considered a better practice since it allows interface issues to be located more quickly and fixed. Data can be lost across an interface; one module can have an adverse effort on the other sub functions when combined by, may not produce the desired major functions. Integrated testing is the systematic testing for constructing the uncover errors within the interface. This testing was done with sample data. The developed system has run success full for this sample data. The need for integrated test is to find the overall system performance.

Integration testing is a logical extension of unit testing. In its simplest form, two units that have already been tested are combined into a component and the interface between them is tested. A component, in this sense, refers to an integrated aggregate of more than one unit. Integration testing identifies problems that occur when units are combined. By using a test plan that requires you to test each unit and ensure the viability of each before combining units, you know that any errors discovered when combining units are likely related to the interface between units. This method reduces the number of possibilities to a far simpler level of analysis. Progressively larger groups of tested software components corresponding to elements of the architectural design are integrated and tested until the software works as a system

### Validation Testing

At the culmination of Black Box testing, software is completely assembled as a package, interface errors have been uncovered and corrected and final series of software tests, Validation tests begins. Validation testing can be defined many was but a simple definition is that validation succeeds when the software functions in a manner that can be reasonably accepted by the customer. After validation test has been conducted one of the two possible conditions exists.

1. The function or performance characteristics confirm to specification and are accepted.
2. A derivation from specification uncovered and a deficiency list is created.

### User Acceptance Testing

Acceptance Testing is a level of the software testing process where a system is tested for acceptability. User Acceptance testing is the software testing process where system tested for acceptability & validates the end to end business flow. Such type of testing executed by client in separate environment & confirms whether system meets the requirements as per requirement specification or not.

UAT is performed after System Testing is done and all or most of the major defects have been fixed. This testing is to be conducted in the final stage of Software Development Life Cycle (SDLC) prior to system being delivered to a live environment. UAT users or end users are concentrating on end to end scenarios & typically involves running a suite of tests on the completed system.

User Acceptance testing also known as Customer Acceptance testing (CAT), if the system is being built or developed by an external supplier. The CAT or UAT are the final confirmation from the client before the system is ready for production. The business customers are the primary owners of these UAT tests. These tests are created by business customers and articulated in Business domain languages. So ideally it is collaboration between business customers, business analysts, testers and developers. It consists of test

suites which involve multiple test cases & each test case contains input data (if required) as well as the expected output. The result of test case is either a pass or fail.

Unit testing is the testing of individual forms, like testing of form seat. Here check save, edit, delete, cancel buttons are work or not, if edit and delete is not visible when we save a value, if save and cancel is not visible when we edit. There is another important matter is that validation checking. Check and confirm we can‟t save same value more than one time.

In integration testing, check the entire project and confirm all forms are integrated. Also test prevention of deletion of values when that value is used in another form.

## 8.2 TEST REPORTS

Test Report provides a summary of the results of test performed. A Test report must contain the following details:-

* **Test Summary**

This must include basic information about what was tested and what happened.

### Test Type

This must include basic information about what type of testing (unit testing, integration testing, validation testing etc.) was done and what happened.

### Test Assessment

It should contain a comprehensive assessment of your interpretation of how adequate the test was in light of how thorough the test plan said it should be? It must also specify what wasn't tested well enough.

### Test Results

Summarize the test results. Include a detailed description of any deviations from the original test plan, design, test case, or expected results. Include any issues or bugs discovered during the test.

### Variances

Describe any variances between the testing that was planned and the testing that actually occurred. Also, provide an assessment of the manner in which the test environment may be different from the operational environment and the effect of this difference on the test results.

### Test Instances

Provide a brief description of the unexpected results, problems, or defects that

occurred during the testing.

## IMPLEMENTATION

The implementation is the final stage and it is an important phase. It involves the individual programming; system testing, user training and the operational running of developed proposed system that constitutes the application subsystems. A major task of preparing for implementation is education of users, which should really have been taken place much earlier in the project when they were being involved in the investigation and design work. During the implementation phase system actually takes physical shape. In order to develop a system implemented planning is very essential.

The implementation phase of the software development is concerned with translating design specialization into source code. The user tests the developed system and changes are made according to their needs. Our system has been successfully implemented

The process of putting the developed system in actual use is called system implementation. This includes all those activities that take place to convert from old system to new system. The system can be implemented only after testing is done and is found to be working to specifications. The implementation stage is a systems project in its own right. The implementation stage involves following tasks:

* careful planning
* investigation of system and constrains
* design of method to achieve change over
* evaluation of the changeover method

# 9. MAINTENANCE

Maintenance activity may require the continuing involvement of a large proportion of computer department resources. For computer installations, which have already developed the basic applications for the organization, the main task may be to adapt existing system in a changing environment. Perhaps a better term to describe this activity is system evolution. All systems are dynamic and subject to constantly changing requirements. Efforts must be devoted to adapting them and design should be flexibly specified so that changes are easily implemented. Most changes arise in two ways. A part of the normal running of the system when errors are found, users ask for improvement or external requirements changes and as result of specific investigation and review of the system’s performance.

# 10. REFERENCES

* Stack over flow
* Geek for greeks
* W3resources
* Github