**Index**

|  |  |  |
| --- | --- | --- |
| **Sr no** | **Contents** | **Page No** |
| **1** | **INTRODUCTION**  **1.1** Company Profile  **1.2** Existing System and Need for System  **1.3** Scope of Work and Objectives  **1.4** Operating Environment – Hardware and Software | 8  8  8 |
| **2** | **SYSTEM ANALYSIS**  **2.1** System Analysis  **2.2** Proposed System Study  **2.3** Requirement Analysis  **2.4** Feasibility Study | 35  35 |
| **3** | **SYSTEM DESIGN**  **3.1**  Dataflow Diagram  3.2 E-R Diagram  **3.3** Class Diagram  **3.4** Use Case Diagrams  **3.5** Activity Diagram  3.6 Sequence Diagram  3.7 Collaboration Diagram  **3.8**  Component Diagram  **3.9** Deployment Diagram ( in case of Web Deployment )  **3.10** Module Specifications  **3.11** User Interface Design ( Screens etc. )  **3.12** Table specifications ( in case back end is a database ) | 42  42  45  45  46  46  48  51  51  52  52  56  67 |
| **4** | **USER MANUAL** |  |
| **5** | Drawbacks and Limitations |  |
| **6** | **Proposed Enhancements** |  |
| **7** | **Conclusions** |  |
| **8** | **Bibliography** | 110 |
| **9** | **ANNEXURES**  **ANNEXURE 1 : INPUT FORMS WITH DATA**  **ANNEXURE 2 : OUTPUT REPORTS WITH DATA**  **ANNEXURE 3 : SAMPLE CODE** | 119 |

1. **INTRODUCTION**
   1. **Company Profile**

**Name:** VEDANT IT ACADEMY

**Description :**

* Vedant IT Academy is one of the service providers in India, offering a range of web designing and development, Software development and Internet marketing services. It's our constant endeavour to continue providing our clients, customized web designing, web development, customized software development, SEO and content solutions and services.
* Vedant IT Academy was founded to address the need of cost effective yet powerful and user friendly systems. We have employed high qualified technical team with engineering and computer application education that can understand the customer requirements and convert their vision into reality.
* Vedant IT Academy has a pool of highly dedicated team of software professionals and technocrats with adequate experience and exposure to project development.
* Vedant IT Academy believes in a unique advance approach and ties a business relationship with the customers which differentiate us from other web developers. At Vedant IT Academy., we build technology solutions around core business strategies. is here to serve your business. We serve according to your convenience and time frames, with quality control methods enforced at every level in our organization.
* Specialties :

Website Design/Development

Application/Software Development

Internet Marketing (SEO)

Android Development

Hardware Sales and Services.

* 1. **Existing System and Need for System**
* Dish Is Wish is provide an online platform which gives list for all hotels and recipies in all the regions online. Nowadays, internet has reached almost to everywhere. So it is necessary to utilize the available resources efficiently and for the benefits of all the users.
* We should able to get all the things in one look .There will be the city wise listing of all hotels and kitchen compass with their contact numbers and address.It give Facility to add into the wish list.It Will display all brands and their products .
* So that users will get all information at once anywhere so that it saves time.

* 1. **Scope of Work And Objectives**
* To provide an online platform which gives list for all hotels and recipies in all the regions online.
* It gives citywise description of all hotels and restaurants online and provide facility to add your favourate dish into wishlist.It help to the user to see their favourate dish whenever they login.No need to users search their favourate dish after they add their favourate dish into wishlist.
* In kitchen compass provides all shops and malls information where found all materials which want to make recipies.
* In Exhibition provides all products listed according to their brands.It helps to the users to know brand of all products and they buy that products according to the brand
* The main objective of the project is to design and develop a user friendly, easy to use and an efficient web system.
* Computerization can be helpful in reducing the manual workload.
* Computerization can be viewed as a means of saving time and cost.
* To provide better Graphical User Interface (GUI).
* Perform necessary validations & minimizes errors.
* It can be suitably modified & expanded in future for further needs.
  1. **Operating Environment - Hardware and Software**
     1. **Overview of Front End**

**Technology:** PHP (Yii framework)

**Language:** HTML, CSS3, JAVASCRIPT, JQUERY, AJAX.

**About Front end Technologies :**

**Features:**

PHP is programming language which is widely used nowadays it has many features as easy to work on and simple deployment PHP 5.0 is the next generation of PHP 4.0 it is widely used new technology for server-side scripting and also a powerful tool for creating dynamic and interactive web pages. PHP has better language support, a large set of new controls, XML-based components, and better user authentication. Along with PHP many other front end technologies are used for the designing or the validations purpose like jquery, ajax, javascript, html-css.

* OOP on the Internet
* Web Services
* Great XML Support
* Class Library
* Complete Compatibility
* Higher scalability
* Increased performance - Compiled code
* Easier configuration and deployment

**CSS**

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation semantics (the look and formatting) of a document written in a markup language. Its most common application is to style web pages written in HTML and XHTML, but the language can also be applied to any kind of XML document, including plain XML, SVG and XUL.CSS is designed primarily to enable the separation of document content (written in HTML or a similar markup language) from document presentation, including elements such as the layout, colors, and fonts.

Unlike CSS 2, which is a large single specification defining various features, CSS 3 is divided into several separate documents called "modules". Each module adds new capabilities or extends features defined in CSS 2, preserving backward compatibility. Work on CSS level 3 started around the time of publication of the original CSS 2 recommendation. The earliest CSS 3 drafts were published in June 1999 to the modularization, different modules have different stability and statuses. As of June 2012, there are over fifty CSS modules published from the CSS Working Group  and four of these have been published as formal recommendations.

This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple pages to share formatting, and reduce complexity and repetition in the structural content (such as by allowing for table less web design). CSS can also allow the same markup page to be presented in different styles for different rendering methods, such as on-screen, in print, by voice (when read out by a speech-based browser or screen reader) and on Braille-based,tactile devices. It can also be used to allow the web page to display differently depending on the screen size or device on which it is being viewed.

**HTML5**

HTML5 is a core technology markup language of the Internet used for structuring and presenting content for the World Wide Web. It is the fifth revision of the HTML standard (created in 1990 and standardized as HTML 4 as of 1997) and, as of December 2012, is a candidate recommendation of the World Wide Web Consortium (W3C). Its core aims have been to improve the language with support for the latest multimedia while keeping it easily readable by humans and consistently understood by computers and devices (web browsers, parsers, etc.). Following its immediate predecessors HTML 4.01 and XHTML 1.1, HTML5 is a response to the observation that the HTML and XHTML in common use on the World Wide Web are a mixture of features introduced by various specifications, along with those introduced by software products such as web browsers, those established by common practice, and the many syntax errors in existing web documents. It is also an attempt to define a single markup language that can be written in either HTML or XHTML syntax

**JAVASCRIPT**

JavaScript is the programming language of the Web.

All modern HTML pages are using JavaScript.

JavaScript is easy to learn.

**Ajax**

AJAX is about updating parts of a web page, without reloading the whole page.

AJAX = Asynchronous JavaScript and XML.

AJAX is a technique for creating fast and dynamic web pages.

AJAX allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes. This means that it is possible to update parts of a web page, without reloading the whole page.

Classic web pages, (which do not use AJAX) must reload the entire page if the content should change.

Examples of applications using AJAX: Google Maps, Gmail, Youtube, and Facebook tabs.

How AJAX Works.

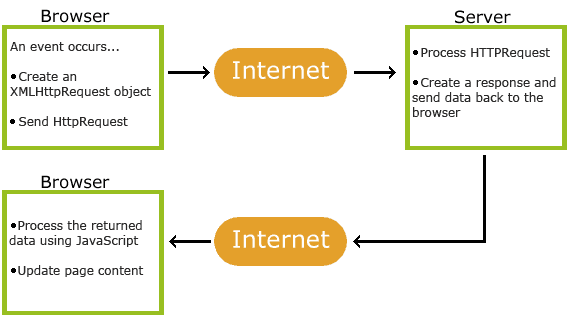


fig. 1.1 : AJAX Working

AJAX is based on internet standards, and uses a combination of:

* XMLHttpRequest object (to exchange data asynchronously with a server)
* JavaScript/DOM (to display/interact with the information)
* CSS (to style the data)
* XML (often used as the format for transferring data)
  + 1. **Overview of Back End**

**MYSQL:**

MYSQL delivers on the data platform vision by enabling organizations to run their most mission-critical applications while lowering the cost of managing the data infrastructure and delivering insights and information to all users. This platform has the following qualities:

**Trusted**—Enables organizations to run their most critical applications with very high levels of security, reliability, and scalability.

**Productive**—Enables organizations to reduce the time and cost required to develop and manage their data infrastructure.

**Intelligent**—Provides a comprehensive platform that delivers insights and information where your users want it.

**Features:**

* Transparent Data Encryption
* External Key Management
* Enhanced Auditing
* Enhanced Database Mirroring
* Performance data collection
* Extended Events
* Backup compression
* Data compression
* Resource Governor

1. **system Analysis**

In System Analysis and Design phase, the whole system development process, the overall system structure and its outlay are defined. Analysis and Design are very important in the whole development cycle process. Any fault in the design phase could be very expensive to solve in the software development process. In this phase, the logical system of the product is developed.

**System Analysis**

The phase is detailed appraisal of existing system. This appraisal includes how the system works and what it does. It also include finding out in more detail what are the problems with system and what user requires from a new system or any new changes in system. The output of this phase results in model of system. The model describes the system function and data and system information flow. The phase also contains the detail set of user requirement and these requirements are used to set objectives for a new system.

A system analysis should be the first undertaking of a feasibility study as it clearly defines the project outline and the clients' requirements. Once these questions have been answered the person/s undertaking the feasibility study will have outlined the project needs definition. The following questions need to be asked to define the project needs definition: What is the end deliverable? What purpose will it serve? What are the environmental effects? What are the rules and regulations? What standards will we be measured against? What are the quality requirements? What is the minimal quality requirements allowed? How much do we need to outsource and in source?

**Proposed System Study**

It is always necessary to study and recognize the problem of existing, which will help in finding out the requirements for the new system. System study helps in finding different alternative for better solution. The project study basically deals with different operation and steps

* Data Gathering
* Study of Existing System
* Analyzing problem
* Studying various documents
* Feasibility study for further improvement

**Requirement Analysis**

Requirement analysis is to find out requirements of the system that the proposed system has to fulfill.

Here, the existing system is totally manual. Proposed system will automate all the processing of the system. Proposed system should have the solutions on the problems mentioned in the problem definition.

Requirements of the system are:-

* System should be user friendly and feasible.
* Data integrity should be maintained.
* Data redundancy and inconsistency should be avoided.
* System should help the decision making process.
* System should be platform independent.
* Security should be maintained.
* Processing time should be less and speed should be high.
* There should be required connections between the related data.

**Feasibility study**

The feasibility study proposes one or more conceptual solution to determine whether the development project has a reasonable chance of success.

It helps us to determine the input and the output of the system. The following are the criteria that are considered to confirm the project feasibility.

**Technical Feasibility:**

At first it is necessary to check that the proposed system is technically feasible or not and to determine the technology and skill necessary to carry out the project. If they are not available then find out the solution to obtain them.

**Operational Feasibility:**

The operational feasibility is obtained by consulting with the system user .Check that proposed solution satisfies the user need or not. There is no resistance from the employees since the new system is helpful.

**System Planning and Schedule**

**Gantt Chart**

Gantt charts are a project-planning tool that can be used to represent the timing of tasks required to complete a project. Because Gantt charts are simple to understand and easy to construct, they are used by most project managers for all but the most complex projects.

In a Gantt chart, each task takes up one row. Dates run along the top in increments of days, weeks or months, depending on the total length of the project. The expected time for each task is represented by a horizontal bar whose left end marks the expected beginning of the task and whose right end marks the expected completion date. Tasks may run sequentially, in parallel or overlapping.

As the project progresses, the chart is updated by filling in the bars to a length proportional to the fraction of work that has been accomplished on the task. This way, one can get a quick reading of project progress by drawing a vertical line through the chart at the current date.

Completed tasks lie to the left of the line and are completely filled in. Current tasks cross the line and are behind schedule if their filled-in section is to the left of the line and ahead of schedule if the filled-in section stops to the right of the line. Future tasks lie completely to the right of the line.

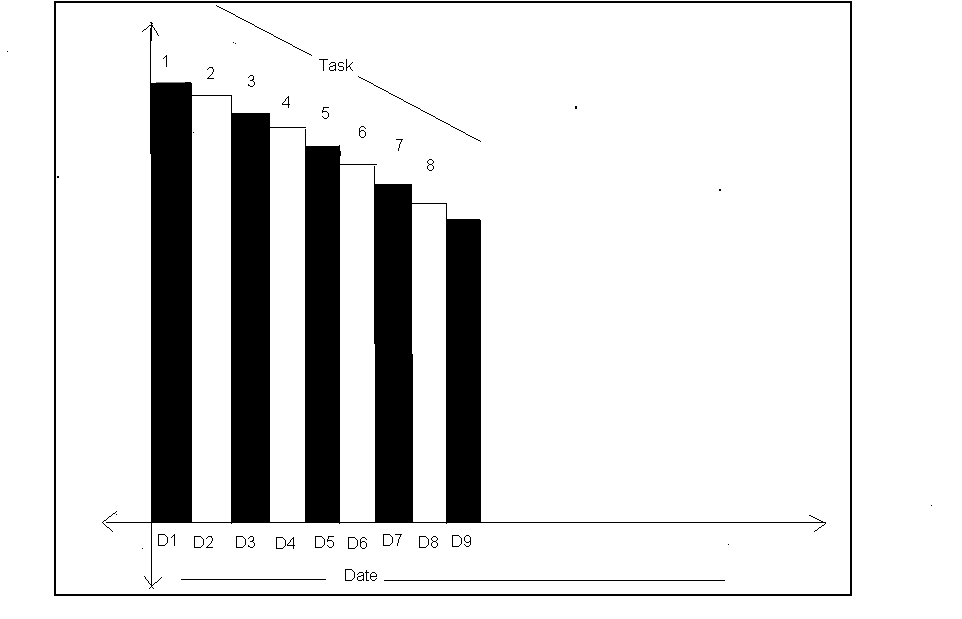
In constructing a Gantt chart, keep the tasks to a manageable number (no more than 15 or 20) so that the chart fits on a single page. More complex projects may require subordinate charts which detail the timing of all the subtasks which make up one of the main tasks.

For team projects, it often helps to have an additional column containing numbers or initials, which identify that on the team is responsible for the task.

We planned our project using according to the Gantt chart as shown as follows:-

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Task** | **Days** | **Date** |
| **1** | **College website** | **100 days** | **July 5** |
| 2 | System study and analysis | 5 days | Jan 23 |
| 3 | Understanding Current database structure | 3 days | Jan 26 |
| 4 | Understanding current system architecture | 5 days | Jan 28 |
| **5** | **System design** | **15 days** | **Dec30** |
|  | Database design | 5 days | Feb 2 |
|  | Form design | 5 days | Feb 6 |
| **6** | **Coding** | **35 days** | **Feb 8**  **To** |
|  | For all forms | 30 days | Mar12 |
| **7** | **Testing** | **15 days** | **Mar27** |
|  | Test | 10 days | - |
|  | Test with user data | 5 days | - |
| **8** | **Documentation** | **7 days** | **April 6** |
| **9** | **Implementation** | **4 days** | **April 10** |

**Fig(1):System Development Table**



### D=date.

**Number(1,2,….9)=task which perform.**

1. **System Design**
   1. **Dataflow Diagram**

* The context level diagrams are initially drawn followed by the levels of DFD’s. A context diagram is a top level (also known as 0) data flow diagram.
* It only contains one process node (process 0) that generates the function of the entire system in relationship to external entities.
* The first level DFD shows the main process can be broken into further processes until you reach pseudo code.
* Symbols used for DFD’s:

|  |  |  |
| --- | --- | --- |
| Symbols | Meaning | Description |
|  | External Agent | An external entity is a source or destination of a data flow which is outside the area of study. |
|  | Data Flow | A data flow shows the flow of information from its source to its destination. A data flow is represented by a line, with arrowheads showing the direction of flow. Information always flows to or from a process and may be written, verbal or electronic. |
|  | Process | A process shows a transformation or manipulation of data flows within the system. |
|  | Data Store | A data store is a holding place for information within the system. It is represented by an open ended narrow rectangle. |

**Context Level Diagram**

fig. 3.1.1 : Context Level Diagram

**1st Level DFD : (Admin)**

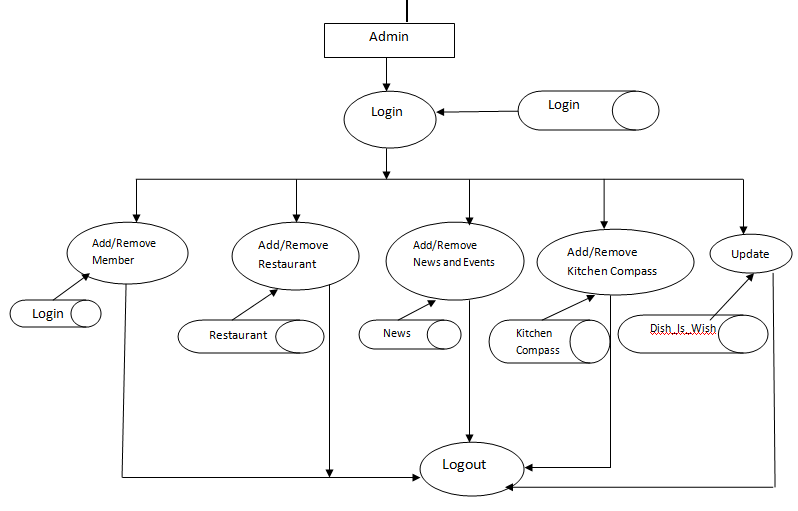
****

fig. 3.1.2 : 1st Level DFD : (Admin)

**(Visitor)**

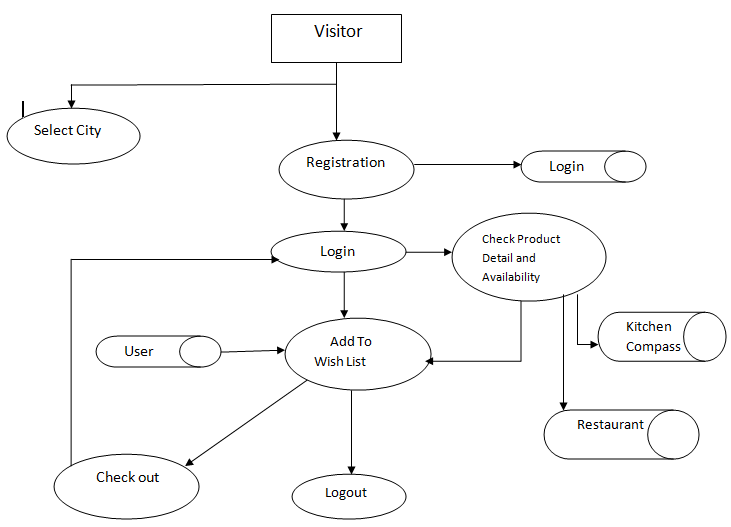
****

fig. 3.1.3 : DFD For (Admin)

* 1. **E-R Diagram**
* An entity-relationship (ER) diagram is a specialized graphic that illustrates the interrelationships between entities in a database. ER diagrams often use symbols to represent three different types of information. Boxes are commonly used to represent entities. Diamonds are normally used to represent relationships and ovals are used to represent attributes.
* There are three basic elements in ER models:
* Entities are the "things" about which we seek information.
* Attributes are the data we collect about the entities.
* Relationships provide the structure needed to draw information from multiple entities.
* Lines linking attribute to entity sets and entity sets to relationship sets
* **Entities** : these are usually nouns used in descriptions of the system, in the discussion of business rules, or in documentation; identified in the narrative
* **Relationships**: these are usually verbs used in descriptions of the system or in discussion of the business rules (entity \_\_\_\_\_\_ entity); identified in the narrative
* **Attributes to the relations**: these are determined by the queries, and may also suggest new entities, e.g. grade; or they may suggest the need for keys or identifiers.
* The *cardinality* defines the relationship between the entities in terms of numbers. An entity may be *optional*: for example, a sales rep could have no customers or could have one or many customers; or *mandatory*: for example, there must be at least one product listed in an order. There are several different types of cardinality notation; *crow's foot notation*, used here, is a common one. In crow's foot notation, a single bar indicates *one*, a double bar indicates *one and only one* (for example, a single instance of a product can only be stored in one warehouse), a *circle* indicates zero, and a *crow's foot* indicates many. The three main cardinal relationships are: one-to-one, expressed as 1:1; one-to-many, expressed as 1: M; and many-to-many, expressed as M: N.

**(User)**

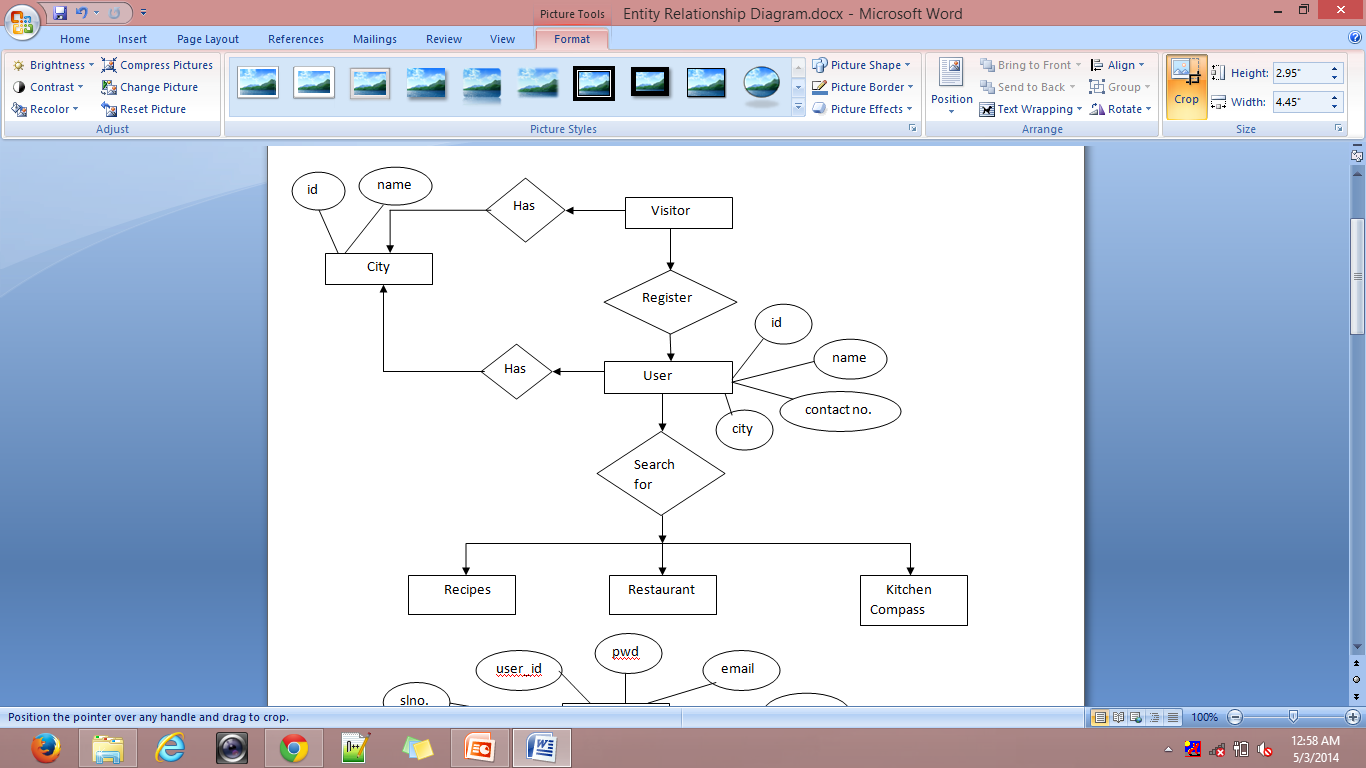
****

fig. 3.2.1 : e-r Diagram For User

**(Admin)**

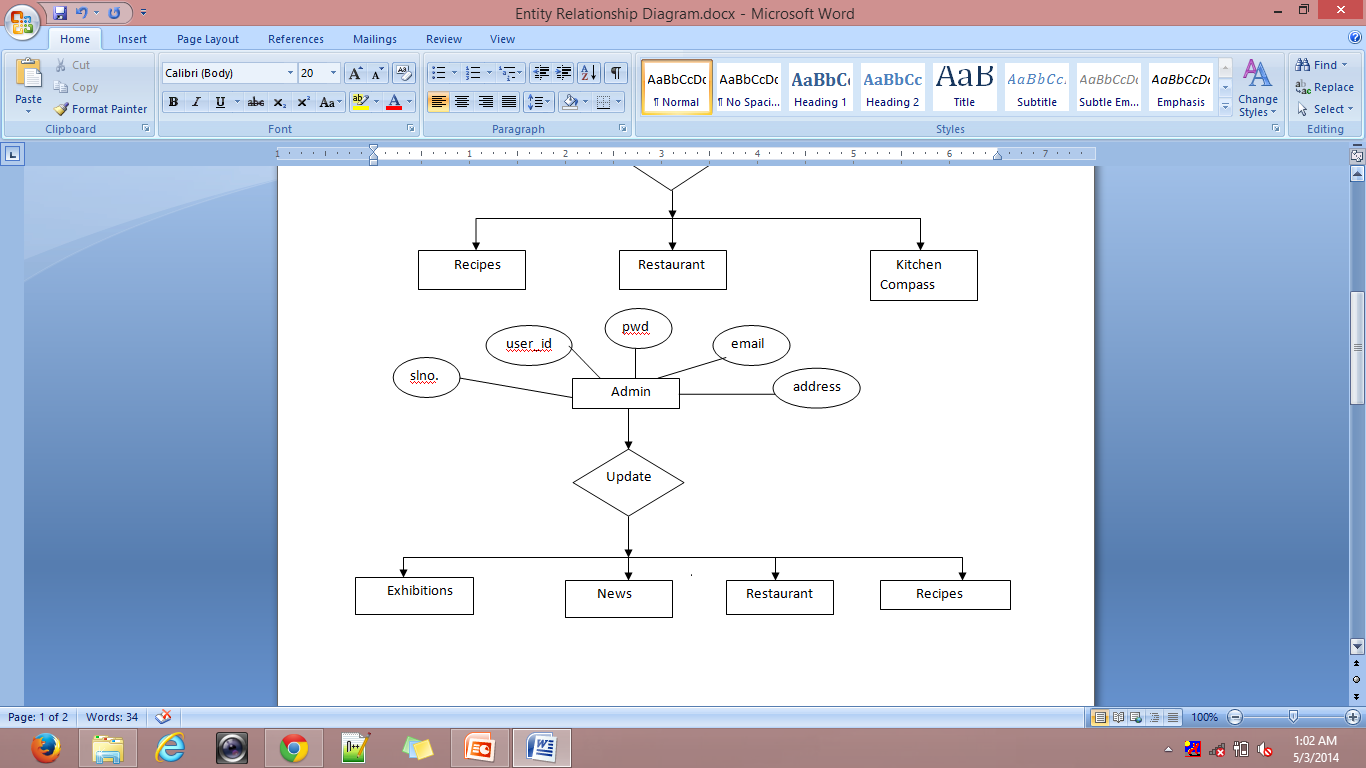
****

fig. 3.2.2 : E-R Diagram For Admin

* 1. **Class Diagram**

A **class diagram** in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes.

**Notations used in Class Diagram are as follows:**

|  |  |  |
| --- | --- | --- |
| Tool Name | Notation | Description |
| Class |  | A *class* defines a collection of similar instances. It exists at compilation time and serves as a type. It defines the interface and implementation of its instances. |
| Object |  | An object is a particular instance of a class. Each object represents a particular instance of something in the problem or solution domain and is created as needed. |
| Association |  | An association is a relationship between classes in a class diagram. |
| Aggregation |  | The aggregation relationship is a special form of association used to model the whole-part or part-of relationship. |
| Specialization & Generalization |  | Specialization is a relationship between classes. Specialization is the is-a-kind-of relationship, in which the specialization is the subclass, or subtype and the generalization is the super class or super type. |

****

fig. 3.3.1 : Class Diagram

* 1. **Usecase Diagram**

A use case diagram depicts participation relationships between actors and use cases. This diagram contains three basic components:

* System
* Actor
* Use case

**Notations used in Use Case diagram:**

|  |  |  |
| --- | --- | --- |
| Tool Name | Notation | Description |
| System |  | System boundary where all use cases will reside. |
| Actor |  | Actor which uses the system. |
| Use Case |  | It is generally function or process in the system. |
| Association |  | It shows direct association between actor and use cases. |
| Dependency |  | It shows dependency between two use cases. |
| Includes |  | It means one use case includes another use case. |

**( visitor )**

****

fig. 3.4.1 : Usecase For Visitor

**(Members)**

****

fig. 3.4.2 : Usecase For Member

* 1. **Activity Diagram**

An activity diagram is a diagram that shows activities and actions to describe workflows. In the Unified Modeling Language an activity diagram represents the business and operational step-by-step workflows of components in the system. An activity diagram shows the overall flow of control. Activity diagram are typically used for business process modeling. They consist of:

* Initial node
* Activity final node
* Activities

The starting point of the diagram is the initial node and the activity final node is the ending. An activity diagram can have zero or more activity final nodes. In between activities are represented by rounded rectangles.

**Notations used in Activity diagram:**

|  |  |  |
| --- | --- | --- |
| Tool Name | Notation | Description |
| Initial State |  | The initial state of the activity. |
| Action State |  | Represents the activity. |
| Decision |  | One activity conditionally follows another activity. |
| Synchronization |  | Multiple activities either follow or precedes synchronization bar. |
| Transition |  | Shows the flow between activities. |
| Signal send state |  | Used to send the signals. |
| Flow final |  | Shows the final flow of activity. |
| Final state |  | Shows the final state i.e. end of activity. |

**(Member)**

****

fig. 3.5.1 : Activity Diagram For Member

**(Visitor)**

****

fig. 3.5.2 : Activity Diagram For Visitor

* 1. **Sequence Diagram**
* An interaction diagram is graphical representation of how objects interact with one another in a scenario. Objects communicate in an interaction diagram by sending messages. A Sequence diagram is organized temporally with the focus on the order in which messages are sent between objects.
* **Notations used in Sequence diagram are as follows:**

|  |  |  |
| --- | --- | --- |
| Tool Name | Notation | Description |
| Object |  | Object can be an actor, class or physical entity. |
| Synchronous message |  | When one object sends this message & waits for the response or completion of process. |
| Asynchronous message |  | When one object sends this message & does not wait for response or completion of process. |
| Return message |  | It is used to give response for a message call. |
| Found message |  | It is used when the sender object is not known or not significant. |
| Lost message |  | It is used when the receiver object is not known or not significant. |
| Create message | <<create>> | When new object is created in the process. |
| Destroy message | <<destroy>> | When an object is destroyed in the process. |
| Conditional message | [if book<2]issuebook | Message along with the condition. |



fig. 3.6.1 : Sequence diagram

* 1. **Collaboration Diagram**

****

fig. 3.7.1 : Collaboration diagem