1. INTRODUCTION:

1.1 Project Overview:

The purpose of this project is to provide a platform for users and rental product(s) owners in an effective and efficient manner.

Online Renting System is a one stop rental portal. It provides services such as Hiring Motor Vehicles, Service Apartments, Hotels, Guest Houses, Meeting & Conference Halls, Audio visuals, Party rentals, Computers and Other Products. It provides the facility to make online orders and get everything done before you reach the destination.

1.2 Project Description:

This web application would be implementable irrespective of the location. Instead of providing products from only one rental show room the application is acting as interface between different users (borrowers and lenders). There is no restriction of rental showrooms for our website. Means any rental showroom owners want to display their products on our site then they simply register in our site by providing personal information and product description(s).

This web application is providing some additional features for vendors to edit or delete their products. Customers need to register with our site as they search and place orders in our site. We are responsible for communication between customer and the vendor and maintain the database. It also providing an extra module that it accepts feedback from the customers. The application provides an extra feature that is “Relocation Services”.

There are many rental systems are available in online. But, they are not providing all products at one place. Also many of them restricted to only one city. That means car rental system in online deals only with cars. Also many of them are not providing effective communication between customer and the vendor. Also present rental systems restricted to only one vendor means products are supplied only from one rental show room.

1.3. MODULES:

The system after careful analysis has been identified to be presented with the following modules:

Online Renting System deals with the following modules

- **Registration (User/Vendor) :** The Vendor/User fills the registration form by giving the personal information and successfully registers with the website.

- **Product (Description/Images/Status) :** After entering into vendor's homepage vendor will add his products by filling add product form, by providing sufficient details about product such as product id, available dates, rent etc and upload the image of that product.

  It consists of following sub modules:

  1. Update Product: Vendor can update the existing product details such as rent, available dates, etc. by entering the product id of that product.
  2. Delete Product: Vendor can delete his products by entering the product id.

- **Data base maintenance:** The data provided by the Vendors such as product details, personal details, etc. and data provided by the customer such as feedback and booking details will be maintained in a data base by the website administrator.

- **Searching and Booking the product:** The Customer after accessing the site searches for products, if he/she finds the required product then he/she need to fill the booking form and submit to the database.

- **Authentication:** Authentication is nothing but providing security to the system. Here everyone must enter into the system through login page. The login page will restrict the unauthorized users. A user must provide his credential like user Id and password for log into the system. For that the system maintains data for all users.Whenever a user enters his user id and password, it checks in the database for user existence.If the user exists he can be treated as a valid user. Otherwise the request will throw back.

- **Tracking System:** A vehicle tracking system is an electronic device installed in a vehicle to enable the owner or a third party to track the vehicle's location. Most modern vehicle tracking systems use Global Positioning Systems (GPS) modules for accurate location of the vehicle. Many systems also combine a communications component such as cellular or satellite transmitters to communicate the vehicle's location to a remote user. Vehicle information can be viewed on electronic maps via the Internet or specialized software.

  Current vehicle tracking systems have their roots in the shipping industry. Corporations with large fleets of vehicles required some sort of system to determine where each vehicle was at any given time. Vehicle tracking systems can now also be found in consumers vehicles as a theft prevention and retrieval device. Police can follow the signal emitted by the tracking system to locate a stolen vehicle.

  Our system uses:

  It is difficult to configure our web application for each and every GPS tracking device, as there are more than 130 GPS communication protocols and more than 800 models of GPS tracking devices.
  But for the security purpose we can make use of " traccar server" which provides live vehicles and asset tracking, reports and notifications regarding the GPS enabled products, if user/customer is willing to go for it and we will add it to his profile for providing live updates of the GPS enabled products at an extra cost. Or He/she can also make use of third party apps like ExtremTracEX300 , which he/she has to install in his/her personal computer to track the GPS enabled product(s).
  The ExtremTrac EX300 is a PC based GPS tracking software with Google earth maps, Google maps, and Microsoft MapPoint map 2004/2006/2009 versions. EX300 can track unlimited vehicles in real time. Realized the idea of control and track vehicles on a personal computer, with easy operation.

1.4 Hardware and Software Requirements:

1.4.1 Hardware Specifications for Development Environment:

Hardware to be used:

- **Processor:** Intel Pentium or More
- **Ram:** 512 MB Ram
- **Hard Disk:** PC with 20GB

1.4.2 Software requirements for Development Environment:

- **Operating System Server:** Windows XP or later.
- **Database Server:** MY SQL.
- **Tools:** XAMPP, Sublime Text, Web Browser.
- **User Interface:** HTML, CSS and JavaScript.
- **Code Behind:** PHP.

1.5.3 Client Requirements for using this web application:

- Web Browser.
- Internet Connection.

2. LITERATURE SURVEY

2.1. Existing System:

If anybody want to rent a product from a particular city from their own home, how it is possible? If one person is going to another city, but if he want products for rent before he reach is destination, then how it possible? So answer to these questions is our web site.

There are many rental systems which are available online. But, they are not providing all products at one place. Also many of them restricted to only one city. That means car rental system in online deals only with cars. Also many of them are not providing effective communication between customer and the vendor. Also present rental systems restricted to only one vendor means products are supplied only from one rental show room.

Drawbacks:

- They limited to only one product and limited cities.
- No effective communication between user and the vendor.
- Products limited to only one rental show room.

2.2 Proposed System:

The proposed system is a web based system which can be accessed by customer from anywhere around the world. The system can offer number of products from vendor in different locations.

A vendor directly registers into this system using this system user interface without any manual approach.

The proposed system can accept any type of product for rental, this system interface support to the vendors to upload their product image into the system. A customer directly interacts with this product image and gets necessary information regarding the rental products.

The proposed system accepts an online request from the customers to reserve any rental system product for his own purpose. Administration play vital role here. Administrator can able to communicate the reservation information of any product to that particular vendor using this system.

2.3 Architectural Design:

- This Web Application (Online Renting System) is based on 3-tier Architecture:
- Data Service layer.
- Business/ application layer.
- User/presentation layer.

In the architecture based on 3-tier, functions are split in 3 layers, which are presentation layer, business application layer and the data layer. Three-tiered model also called as multitier architecture came up in order to conquer limitations that were prevailing in 2-tier architecture model. In 3-tier architecture model, middle tier was introduced in between the user interface client environment and the database system server environment. The middle tier can be implemented in many ways like transaction processing monitors, message servers or else application servers. Middle tier has the ability to perform the queuing process on the application execution as well as database staging. Such architecture provides more versatility and also supports the changes made in business layer, technology layer as well as data layer. The 3-tier architectural model is described as the client system computer, the application system server and the data server.

3rd-tier in the 3-tiered architecture furnishes functionality in the database management and is committed to the data as well as file services, which is optimized without the use of any kind of proprietary DBMS languages. Its data management part which ensures the consistency of data in distributed environment, with the usage of certain features like data locking process, consistency and the replication process. Depending on the request by the user for the services and the data, the process of connectivity in between the tiers can show a dynamic change.

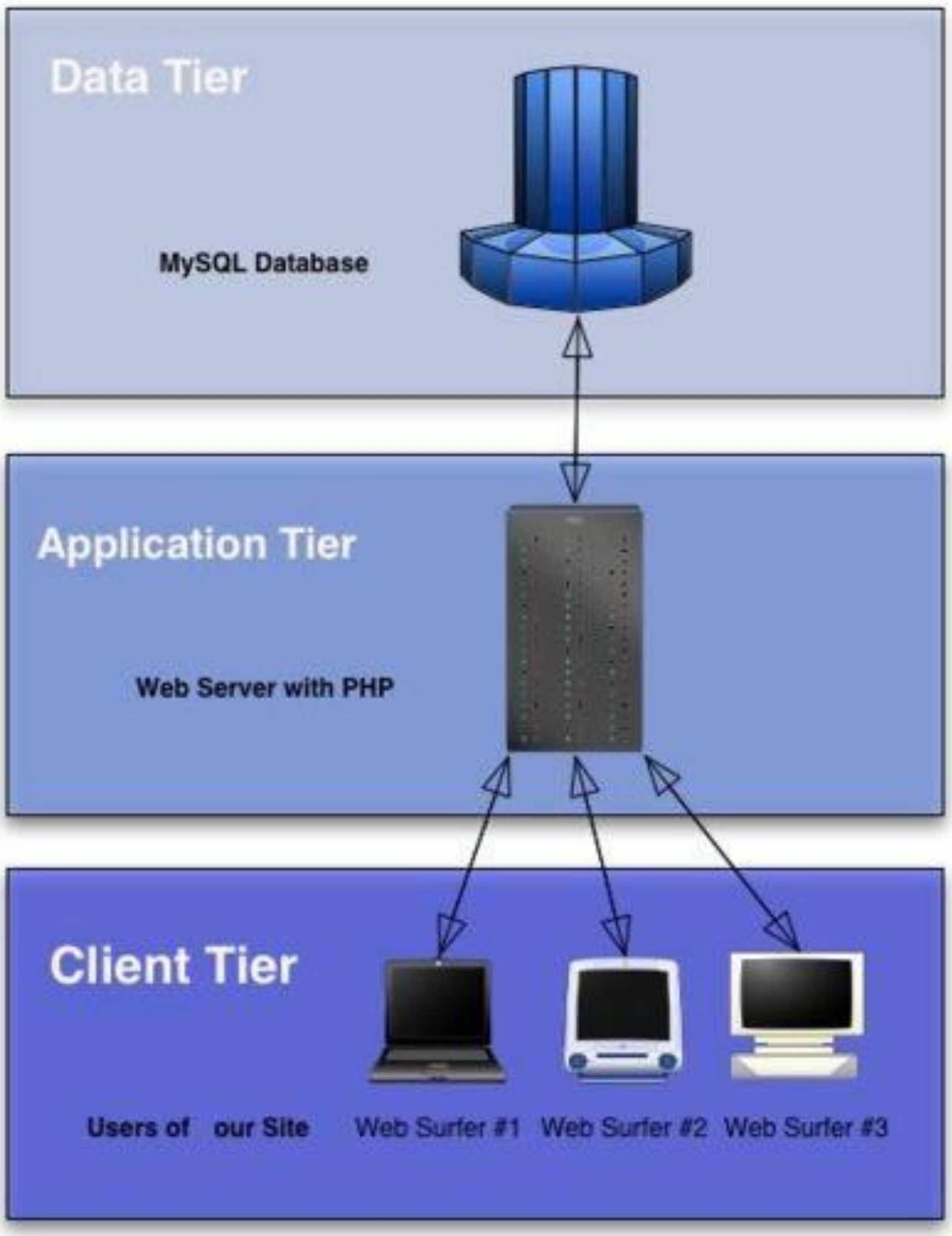


FIGURE 2.3: Three Tier Architecture

2.3.1 Client tier:

It’s the responsibility of the client-tier to present data, receive the user events and management of user interface.

2.3.2 Application Server tire:

Its application server actually, where the business logic resides. It’s the responsibility of the business application layer to implement the business rules. It’s in this layer, where the business objects actually reside, which is not the case in 2-tier model. Data in this layer is not in any case vulnerable, as compared to data in 2-tier architectural model.

2.3.3 Data Server tier:

This layer allows the storing of data. Moreover, the prevalent RDBMS and legacy databases are usually reused over here.

In 3-tier architecture all tiers can run on just single machine, since there exist logical boundaries in between the three tiers. The most important significance lies in the efficiently structured application. Today it’s the three tier model that is given preference over the 2-tier model. It’s because of the benefits provided by 3 tier (or multi-tier) architectural models, which are as follows:

- Since application layer is a separate layer over here, it provides better versatility and load distribution.
- Since the middle-tier is distributed among several servers, it eradicates the problem of fast recovery.
- As the applications are managed on server side, the system administration process is quite easy.
- Enhanced modularity assures the software reusability along with integration of older application.

When the web application based on large scale is thought of, and then the 3-tier architectural model is considered as more appropriate than the 2-tier architectural model. This is due to the following reasons:

- Due to portioning, the designing and the development process is quick as well as easy.
- It’s quite easy to update or make changes in any part of application, which would in no case effect the working of other tiers.
- The tier performance can be optimized and easily monitored.

Considering the above stated facts, the 3-tier architectural model is thus preferred for this specific application.

2.4 Feasibility Study:

A feasibility study looks at the viability of an idea with an emphasis on identifying potential problems and attempts to answer one main question: Will the idea work and should you proceed with it?
Feasibility studies address things like where and how the system will operate. They provide in-depth details about the system to determine if and how it can succeed, and serve as a valuable tool for developing a winning system plan.
A feasibility study is carried out to select the best system that meets performance requirements. The main aim of the feasibility study activity is to determine whether it would be financially and technically feasible to develop the product. The feasibility study activity involves the analysis of the problem and collection of all relevant information relating to the product such as the different data items which would be input to the system, the processing required to be carried out on these data, the output data required to be produced by the system as well as various constraints on the behavior of the system.

The information you gather and present in your feasibility study will help you:

- ✓ List in detail all the things are needed to make the system work.
- ✓ Identify logistical and other system-related problems and solutions.
- ✓ Develop marketing strategies to convince an investor that your system is worth considering as an investment.
- ✓ Serve as a solid foundation for developing your system plan.

There are three measures of feasibility analysis which must be considered while selecting a particular solution for the specified problem. They are as:

Technical Feasibility:

It is concerned with specifying equipment and software that will successfully satisfy the user requirement. The technical needs of the system may vary considerably, but might include:

a) The facility to produce outputs in a given time.

b) Response time under certain conditions.

c) Ability to process a certain volume of transaction at a particular speed.

d) Facility to communicate data to distant locations.

In examining technical feasibility, configuration of the system is given more importance than the actual hardware. The configuration should give the complete picture about the system's requirements: How many workstations are required, how these units are interconnected so that they could operate and communicate smoothly. What speeds of input and output should be achieved at particular quality of printing.

Economic_Feasibility:

Economic analysis is the most frequently used technique for evaluating the effectiveness of a proposed system. More commonly known as Cost / Benefit analysis, the procedure is to determine the benefits and savings that are expected from a proposed system and compare them with costs. If benefits outweigh costs, a decision is taken to design and Implement the system. Otherwise, further justification or alternative in the proposed system will have to be made if it is to have a chance of being approved. This is an outgoing effort that improves in accuracy at each phase of the system life cycle.

Operational_Feasibility:

Operational feasibility study tests the operational scope of the software to be developed. The points to be considered are:

a. What changes will be brought with the system?
b. What organizational structure is disturbed?
c. What new skills will be required? Do the existing staff members have these skills?

Proposed projects are beneficial only if they can be turned into information system that will meet the operating requirements of the organization. This test of feasibility asks if the system will work when it is developed and installed.

3. SYSTEM ANALYSIS AND DESIGN:

3.1 Requirement Specification:

Software engineering is a process of discovery, refinement, modeling and specification (a set of activities that is often referred to as analysis model of required data, information and control flow) for the creation of operational behavior. Alternative solutions are analyzed and a complete analysis model is created. Software requirements engineering process is described in the following way:

"Requirements engineering is the systematic use of proven principles, techniques, languages and tools for the cost effective analysis, documentation and ongoing evolution of user needs and the specification of external behavior of a system to satisfy those needs."

Both the software engineers and customers take an active role in the software requirement analysis. The customer attempts to reformulate sometimes-nebulous system level description of data, function and behavior into concrete detail. The developer acts as interrogator, consultant, or. Requirement analysis is a software engineering task that bridges the gap between the system level requirements engineering and software design. Requirements Engineering activities result in the specification of the software's operational characteristics (data, function and behavior) indicates software interface with other system elements and establishes constraints that software must need. Requirement analysis allows the analyst to refine the software allocation and build models of data, functional and behavioral domains that will be treated by software. Requirement analysis provides the software designer with a representation of information, Function and behavior that can be translated to data, architectural, interface, and component level Design.

Finally the requirement specification provides the developer and the customer with the means to assess quality once the software is built.

The project is based on Agile Unified Process:

3.2 Agile Approach:

Agile development methodology provides opportunities to assess the direction of a project throughout the development lifecycle. This is achieved through regular cadences of work, known as sprints or iterations, at the end of which teams must present a potentially shippable product increment. By focusing on the repetition of abbreviated work cycles as well as the functional product they yield, agile methodology is described as “iterative” and “incremental.” In waterfall, development teams only have one chance to get each aspect of a project right. In an agile paradigm, every aspect of development requirements, design, etc. is continually revisited throughout the lifecycle. When a team stops and re-evaluates the direction of a project every two weeks, there’s always time to steer it in another direction.

The results of this “inspect-and-adapt” approach to development greatly reduce both development costs and time to market. Because teams can develop software at the same time they’re gathering requirements, the phenomenon known as “analysis paralysis” is less likely to impede a team from making progress. Agile development methodology helps companies build the right product. Instead of committing to market a piece of software that hasn’t even been written yet, agile empowers teams to continuously re-plan their release to optimize its value throughout development, allowing them to be as competitive as possible in the marketplace. Development using an agile methodology preserves a product’s critical market relevance and ensures a team’s work doesn’t wind up on a shelf, never released.

The project is based on Object Oriented Approach using Unified Process (UP). The UP undergoes under following phases:

- ☞ **Inception Phase:** Approximate vision, business case, scope, vague estimates.

- ☞ **Elaboration Phase:** Refined vision, iterative implementation of the core Architecture, resolution of high risks, identification of most requirements and scope, more realistic estimates.

☞ **Construction Phase:** Iterative implementation of the remaining lower risk and Easier elements and preparation for deployment.

☞ **Transition Phase :** Beta tests, deployment.

3.2.1 Steps:

- **Inception:**
  The intent was to build a basic understanding of the problem, the people who want the solution, the nature solution that is desired and the effectiveness of preliminary communication and collaboration.

- **Elaboration:**
  We expanded and refined the information obtained during inception and elaboration. We focused developing a refined technical model of software functions, features and constraints.

- **Specification:**
  Here we produced a written document specifying all the captured requirements in a Consistent and therefore more understandable manner. This document served as the Foundation subsequent software engineering activities.

- **Validation:**
  The product produced as a consequence of requirements engineering was assessed for quality during this step, in order to ensure that all software requirements have been stated unambiguously; that inconsistencies, omissions and errors have been detected and corrected; and that the work product conforms to the standard established for the process the project and the product.

- **Requirements Management:**
  It consists of a set of activities that helped us to identify, control and track requirements and changes to requirements at any time as the project proceeded.

3.3 Analysis Phase:

Although the analysis phase greatly resembles the Preliminary Investigation phase, any activities previously executed in the preliminary Investigation phase will now be executed in more depth. The completion of the Preliminary Investigation phase narrows the scope of the activities in the Analysis phase so that efforts will focus on the chosen solution. Now the models for the current system and proposed system can be fleshed out with more detailed specifications. Because the steering committee has approved the new system, work can begin in earnest.

The Analysis phase has six basic activities:

1) Study the existing system.

2) Review the conclusions obtained by the preliminary Investigation phase recommended solution, feasibility issues, and rejection of alternative solutions.

3) Prepare the model of the new system.

4) Revise the preliminary design.

5) Devise the detailed schedule for project implementation.

6) Prepare the report on the Analysis phase for review by management.

<u>Preliminary investigation</u>

This phase commences with discussion on the request made by the user. The request can be for a new system or for modifying the existing one. An estimate is made of whether the identified users' needs can be satisfied or not. Preliminary investigation verifies the problem and understands the need for the required system. It considers whether the proposed system will be cost effective from the business point of view and whether it can be developed within existing budgetary constraints. In addition, the time factor, which determines the duration of the project, is also considered. Preliminary investigation should be quick and cost effective. The output of preliminary investigation decides whether the new system should be developed or not.

3.4 Design Phase:

The activities in the Design phase are executed in order to furnish the details needed for the coding of the system. The appropriate hardware and software platforms for the system's implementation are selected; these basic issues are depicted although these issues are listed in the Design phase, they were under consideration throughout the earlier phases. The users also actively participate in the Design Phase by evaluating the analyst's proposals for the user interfaces, such as screen designs and report formats. The format of the computer files is defined in this phase along with the specifications for test data needed for program testing activities in the next phase. The formats for input data and output data are also specified, the medium for the system's inputs and outputs is chosen, and any paper forms form manually recorded data are drafted, testing procedures are formulated in a preliminary manner.

The training of the users is considered in this phase, although final plans are typically deferred to the implementation phase. The computer programs are described in a detailed fashion, typically by a diagramming technique known as structure charts. After this phase is completed, the details of the system have been fully specified. The system is now ready for programming to begin. Preliminary plans are also drawn up for the installation of the new system; final plans will be produced during the Implementation phase. Before moving into the next phase, the checkpoint occurs: Management decides whether the activities to date have progressed satisfactorily and, if so, approves continuation of the project.

4. METHODOLOGY:

In software engineering, a software development methodology (also known as a system development methodology, software development life cycle, software development process, software process) is a splitting of software development work into distinct phases (or stages) containing activities with the intent of better planning and management. It is often considered a subset of the systems development life cycle.

In our project we will use the spiral model. It is a risk-driven process model generator for software projects. Based on the unique risk patterns of a given project, the spiral model guides a team to adopt elements of one or more process models, such as incremental, waterfall, or evolutionary prototyping.

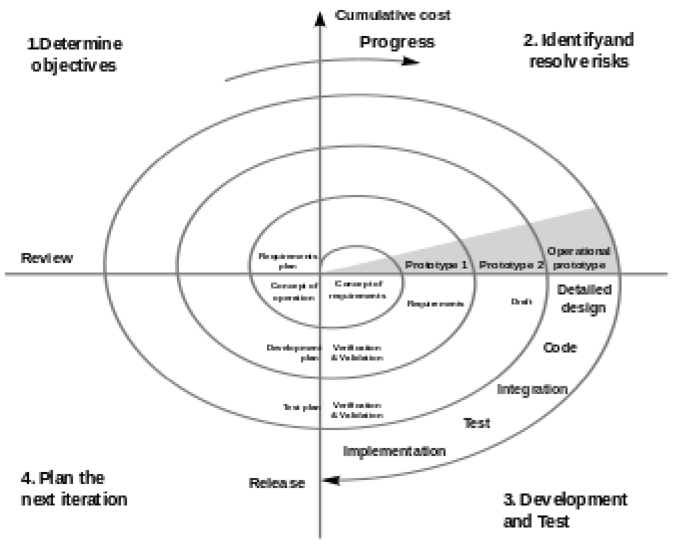


FIGURE 4: SPIRAL MODEL

The Six Invariants

Authentic applications of the spiral model are driven by cycles that always display six characteristics. Boehm illustrates each with an example of a "hazardous spiral look-alike" that violates the invariant.

1. Define artifacts concurrently

Sequentially defining the key artifacts for a project often lowers the possibility of developing a system that meets stakeholder "win conditions" (objectives and constraints).

This invariant excludes "hazardous spiral look-alike" processes that use a sequence of incremental waterfall passes in settings where the underlying assumptions of the waterfall model do not apply. Boehm lists these assumptions as follows:

1. The requirements are known in advance of implementation.
2. The requirements have no unresolved, high-risk implications, such as risks due to cost, schedule, performance, safety, security, user interfaces, organizational impacts, etc.
3. The nature of the requirements will not change very much during development or evolution.
4. The requirements are compatible with all the key system stakeholders' expectations, including users, customer, developers, maintainers, and investors.
5. The right architecture for implementing the requirements is well understood.
6. There is enough calendar time to proceed sequentially.

In situations where these assumptions do apply, it is a project risk not to specify the requirements and proceed sequentially. The waterfall model thus becomes a risk-driven special case of the spiral model.

2. Perform four basic activities in every cycle

This invariant identifies the four basic activities that must occur in each cycle of the spiral model:

1. Consider the win conditions of all success-critical stakeholders.
2. Identify and evaluate alternative approaches for satisfying the win conditions.
3. Identify and resolve risks that stem from the selected approaches.
4. Obtain approval from all success-critical stakeholders, plus commitment to pursue the next cycle.

Project cycles that omit or shortchange any of these activities risk wasting effort by pursuing options that are unacceptable to key stakeholders, or are too risky.

Some "hazardous spiral look-alike" processes violate this invariant by excluding key stakeholders from certain sequential phases or cycles. For example, system maintainers and administrators might not be invited to participate in definition and development of the system. As a result, the system is at risk of failing to satisfy their win conditions.

3. Risk determines level of effort

For any project activity (e.g., requirements analysis, design, prototyping, testing), the project team must decide how much effort is enough. In authentic spiral process cycles, these decisions are made by minimizing overall risk.

For example, investing additional time testing a software product often reduces the risk due to the marketplace rejecting a shoddy product. However, additional testing time might increase the risk due to a competitor's early market entry. From a spiral model perspective, testing should be performed until the total risk is minimized, and no further.

"Hazardous spiral look-alikes" that violate this invariant include evolutionary processes that ignore risk due to scalability issues, and incremental processes that invest heavily in a technical architecture that must be redesigned or replaced to accommodate future increments of the product.

4. Risk determines degree of detail

For any project artifact (e.g., requirements specification, design document, test plan), the project team must decide how much detail is enough. In authentic spiral process cycles, these decisions are made by minimizing overall risk.

Considering requirements specification as an example, the project should precisely specify those features where risk is reduced through precise specification (e.g., interfaces between hardware and software, interfaces between prime and sub-contractors). Conversely, the project should not precisely specify those features where precise specification increases risk (e.g., graphical screen layouts, behavior of off-the-shelf components).

5. Use anchor point milestones

Boehm's original description of the spiral model did not include any process milestones. In later refinements, he introduces three anchor point milestones that serve as progress indicators and points of commitment. These anchor point milestones can be characterized by key questions.

1. Life Cycle Objectives. Is there a sufficient definition of a technical and management approach to satisfying everyone's win conditions? If the stakeholders agree that the answer is "Yes", then the project has cleared this LCO milestone. Otherwise, the project can be abandoned, or the stakeholders can commit to another cycle to try to get to "Yes."
2. Life Cycle Architecture. Is there a sufficient definition of the preferred approach to satisfying everyone's win conditions, and are all significant risks eliminated or mitigated? If the stakeholders agree that the answer is "Yes", then the project has cleared this LCA milestone. Otherwise, the project can be abandoned, or the stakeholders can commit to another cycle to try to get to "Yes."
3. Initial Operational Capability. Is there sufficient preparation of the software, site, users, operators, and maintainers to satisfy everyone's win conditions by launching the system? If the stakeholders agree that the answer is "Yes", then the project has cleared the IOC milestone and is launched. Otherwise, the

project can be abandoned, or the stakeholders can commit to another cycle to try to get to "Yes."

"Hazardous spiral look-alikes" that violate this invariant include evolutionary and incremental processes that commit significant resources to implementing a solution with a poorly defined architecture.

The three anchor point milestones fit easily into the Rational Unified Process (RUP), with LCO marking the boundary between RUP's Inception and Elaboration phases, LCA marking the boundary between Elaboration and Construction phases, and IOC marking the boundary between Construction and Transition phases.

6. Focus on the system and its life cycle

This invariant highlights the importance of the overall system and the long-term concerns spanning its entire life cycle. It excludes "hazardous spiral look-alikes" that focus too much on initial development of software code. These processes can result from following published approaches to object-oriented or structured software analysis and design, while neglecting other aspects of the project's process needs.

4.1 DATA FLOW DIAGRAMS:

A data flow diagram is graphical tool used to describe and analyze movement of data through a system. These are the central tool and the basis from which the other components are developed. The transformation of data from input to output, through processed, may be described logically and independently of physical components associated with the system. These are known as the logical data flow diagrams. The physical data flow diagrams show the actual implements and movement of data between people, departments and workstations. A full description of a system actually consists of a set of data flow diagrams. Using two familiar notations Yourdon, Gane and Sarson notation develops the data flow diagrams. Each component in a DFD is labeled with a descriptive name. Process is further identified with a number that will be used for identification purpose. The development of DFD'S is done in several levels. Each process in lower level diagrams can be broken down into a more detailed DFD in the next level. The lop-level diagram is often called context diagram. It consists a single process bit, which plays vital role in studying the current system. The process in the context level diagram is exploded into other process at the first level DFD.

The idea behind the explosion of a process into more process is that understanding at one level of detail is exploded into greater detail at the next level. This is done until further explosion is necessary and an adequate amount of detail is described for analyst to understand the process.

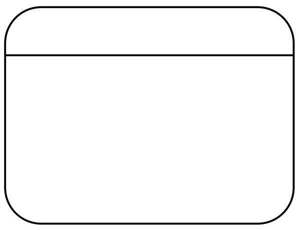
Larry Constantine first developed the DFD as a way of expressing system requirements in a graphical from, this lead to the modular design.

A DFD is also known as a "bubble Chart" has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. So it is the starting point of the design to the lowest level of detail. A DFD consists of a series of bubbles joined by data flows in the system.

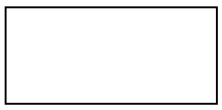
4.1.1 DFD SYMBOLS:

In the DFD, there are four symbols

1. A square defines a source(originator) or destination of system data
2. An arrow identifies data flow. It is the pipeline through which the information flows
3. A circle or a bubble represents a process that transforms incoming data flow into outgoing data flows.
4. An open rectangle is a data store, data at rest or a temporary repository of data.

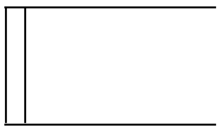


Process that transforms data flow.



Source or Destination of data

Data flow



Data Store

4.1.2 CONSTRUCTING A DFD :

Several rules of thumb are used in drawing DFD'S:

1. Process should be named and numbered for an easy reference. Each name should be representative of the process.
2. The direction of flow is from top to bottom and from left to right. Data traditionally flow from source to the destination although they may flow back to the source. One way to indicate this is to draw long flow line back to a source. An alternative way is to repeat the source symbol as a destination. Since it is used more than once in the DFD it is marked with a short diagonal.
3. When a process is exploded into lower level details, they are numbered.
4. The names of data stores and destinations are written in capital letters. Process and dataflow names have the first letter of each work capitalized

A DFD typically shows the minimum contents of data store. Each data store should contain all the data elements that flow in and out.

4.1.3 SAILENT FEATURES OF DFD'S

1. The DFD shows flow of data, not of control loops and decision are controlled considerations do not appear on a DFD.
2. The DFD does not indicate the time factor involved in any process whether the dataflow take place daily, weekly, monthly or yearly.
3. The sequence of events is not brought out on the DFD.

4.1.4 TYPES OF DATA FLOW DIAGRAMS :

1. Current Physical
2. Current Logical
3. New Logical
4. New Physical

1. Current Physical :

In Current Physical DFD process label include the name of people or their positions or the names of computer systems that might provide some of the overall system-processing label includes an identification of the technology used to process the data. Similarly data flows and data stores are often labels with the names of the actual physical media on which data are stored such as file folders, computer files, business forms or computer tapes.

2. Current Logical:

The physical aspects at the system are removed as much as possible so that the current system is reduced to its essence to the data and the processors that transform them regardless of actual physical form.

3. New Logical:

This is exactly like a current logical model if the user were completely happy with the user were completely happy with the functionality of the current system but had problems with how it was implemented typically through the new logical model will differ from current logical model while having additional functions, absolute function removal and inefficient flows recognized.

4. New Practical:

The new physical represents only the physical implementation of the new system.

4.1.5. RULES GOVERNING THE DFD'S:

PROCESS

1) No process can have only outputs.
2) No process can have only inputs. If an object has only inputs than it must be a sink.
3) A process has a verb phrase label.

DATA STORE :

1) Data cannot move directly from one data store to another data store, a process must move data.
2) Data cannot move directly from an outside source to a data store, a process, which receives, must move data from the source and place the data into data store
3) A data store has a noun phrase label.

SOURCE OR SINK:

The origin and /or destination of data.

1) Data cannot move direly from a source to sink it must be moved by a process
2) A source and /or sink has a noun phrase land.

DATA FLOW:

1) A Data Flow has only one direction of flow between symbols. It may flow in both directions between a process and a data store to show a read before an update. The later is usually indicated however by two separate arrows since these happen at different type.
2) A join in DFD means that exactly the same data comes from any of two or more different processes data store or sink to a common location.
3) A data flow cannot go directly back to the same process it leads. There must be at least one other process that handles the data flow produce some other data flow returns the original data into the beginning process.
4) A Data flow to a data store means update (delete or change).
5) A data Flow from a data store means retrieve or use.

A data flow has a noun phrase label more than one data flow noun phrase can appear on a single arrow as long as all of the flows on the same arrow move together as one package.

4.1.6 Level 0 Data Flow Diagram:

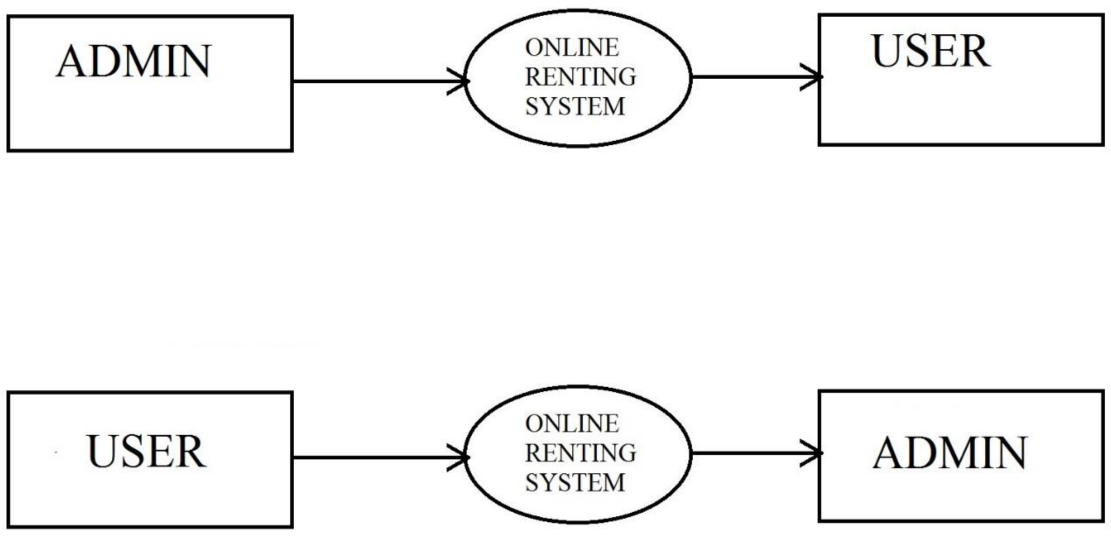


FIGURE 4.1.6(1): LEVEL 0 DFD

Login DFD Diagram:

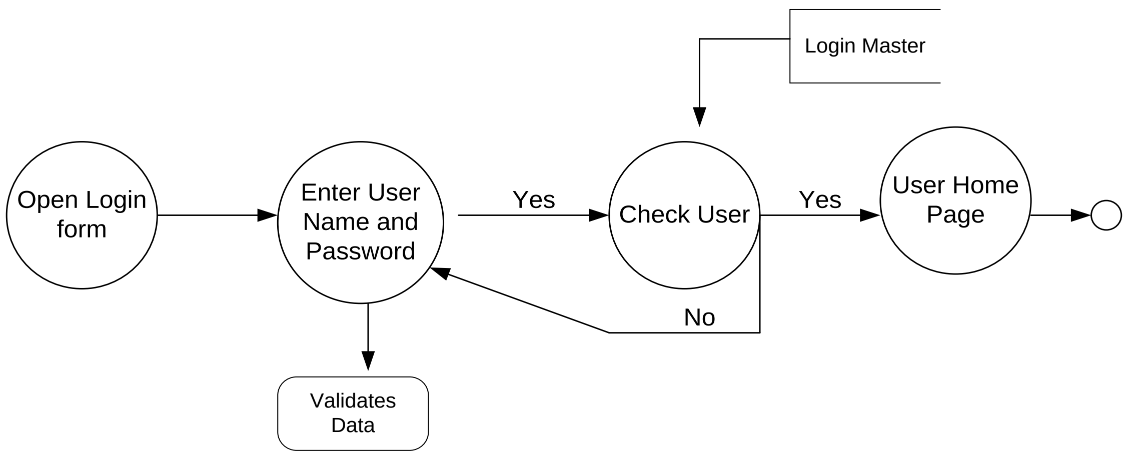


FIGURE 4.1.6(2): Login Data Flow Diagram

Level 1 DFD:

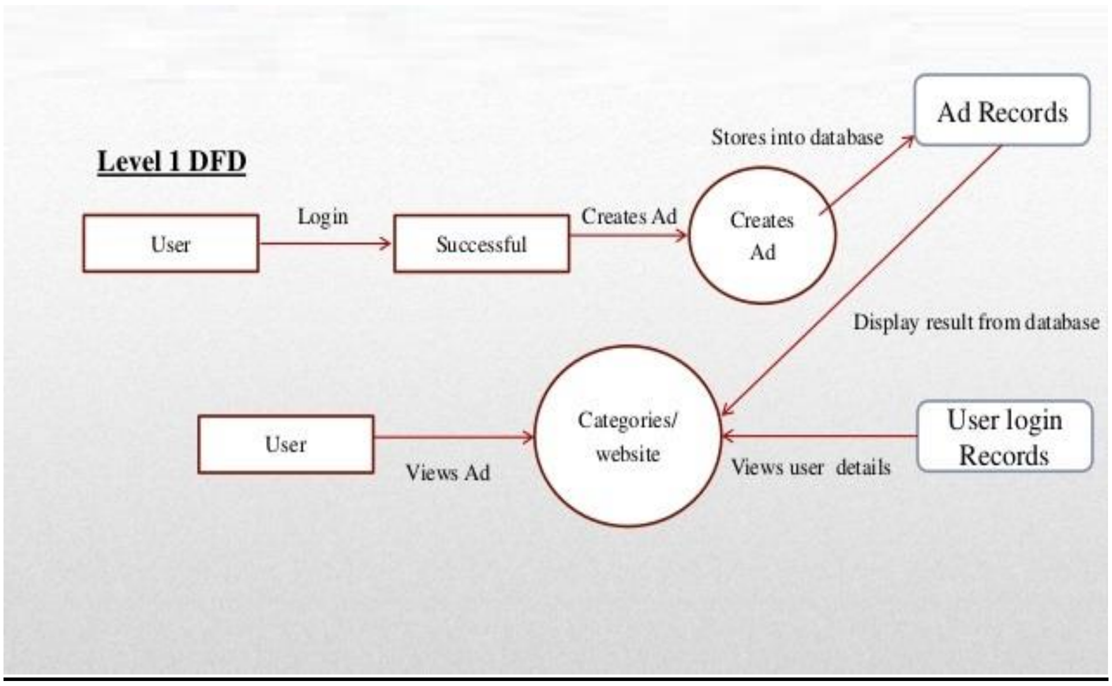


FIGURE 4.1.6(3): Level 1 Data Flow Diagram

4.2 Use case Diagram in our system:

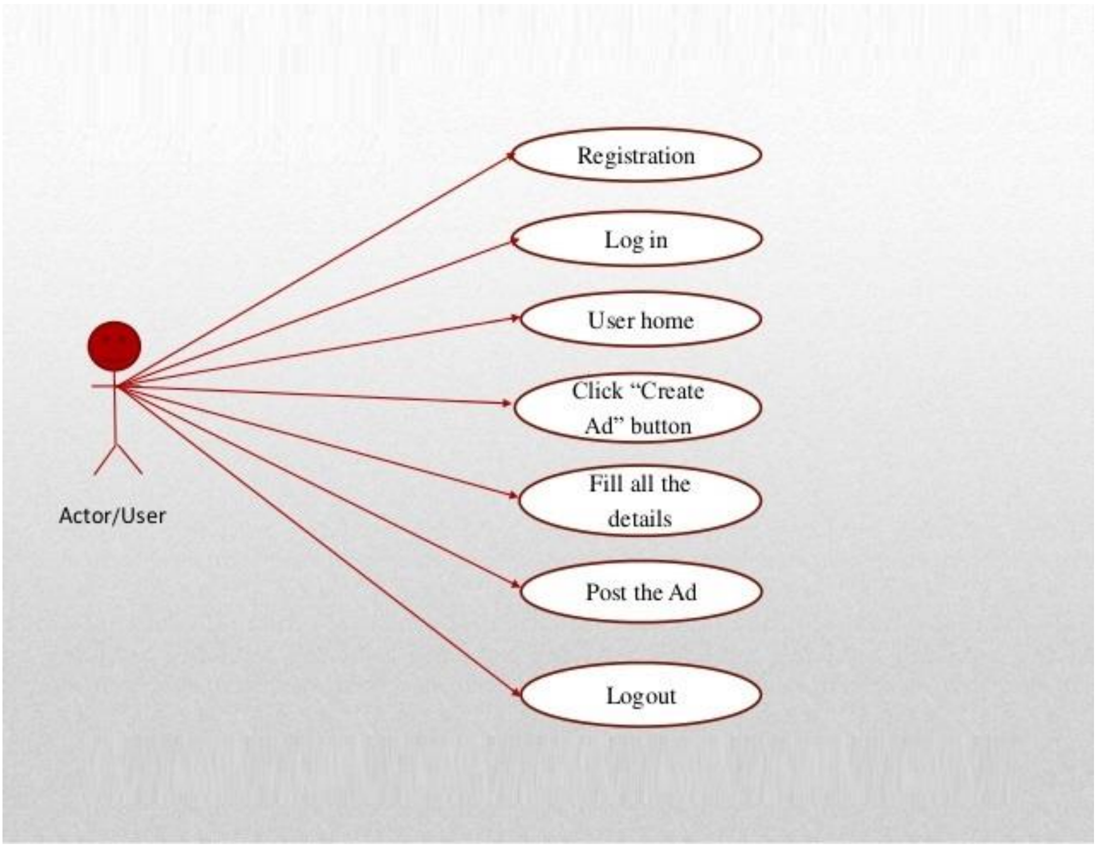


FIGURE 4.2: USE CASE DIAGRAM

4.3 Extended Use case Diagram in our system:

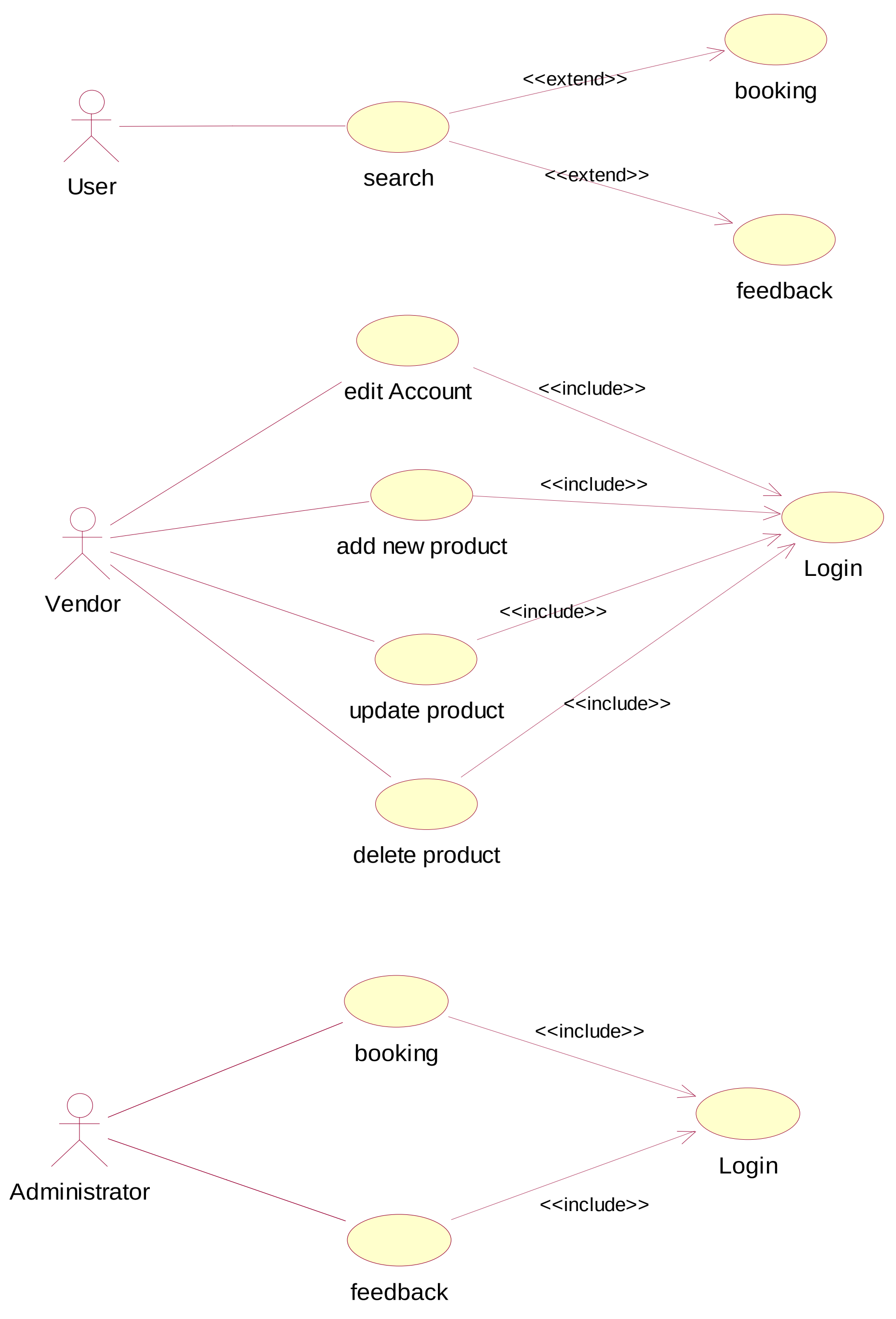
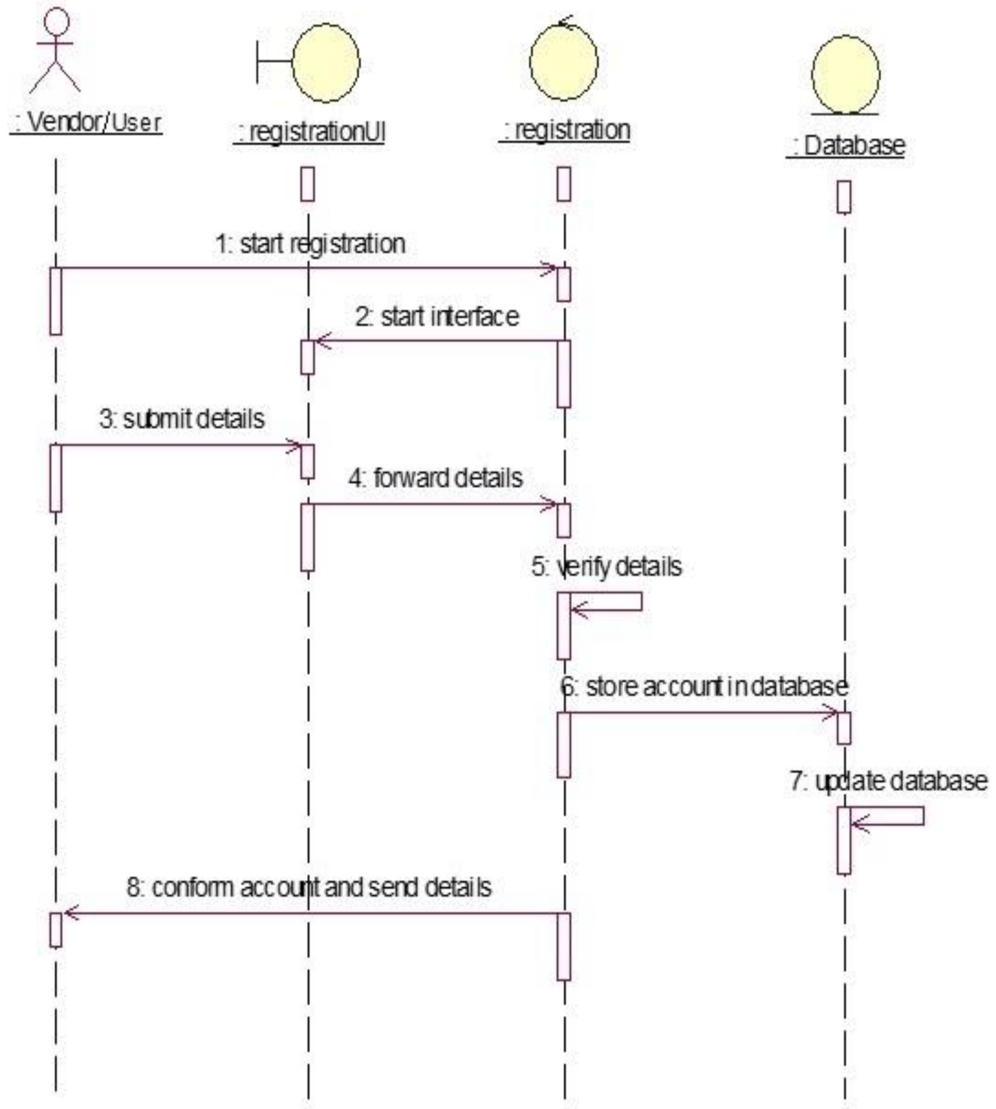


FIGURE 4.3: EXTENDED USE CASE DIAGRAM

4.4 Sequence Diagrams:



4.4.1 SEQUENCE DIAGRAM FOR REGISRTRATION

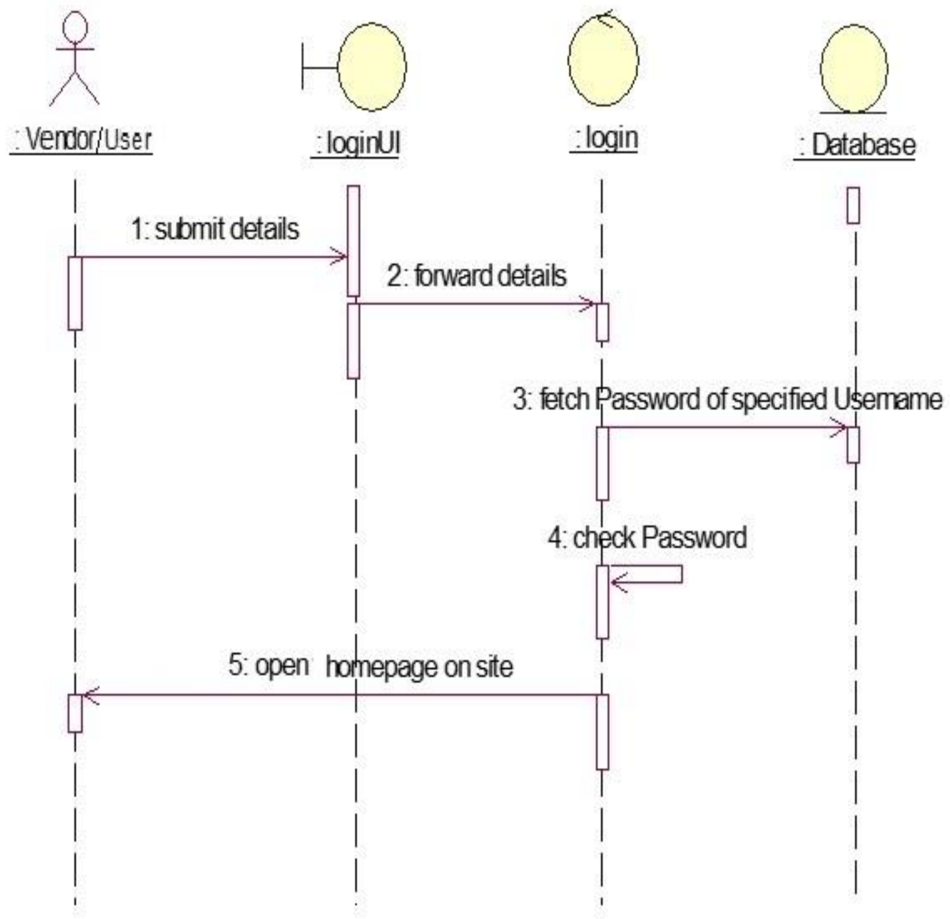


FIGURE 4.4.2: SEQUENCE DIAGRAM FOR LOGIN

4.4.3 Sequence Diagram for Vendor to add product:

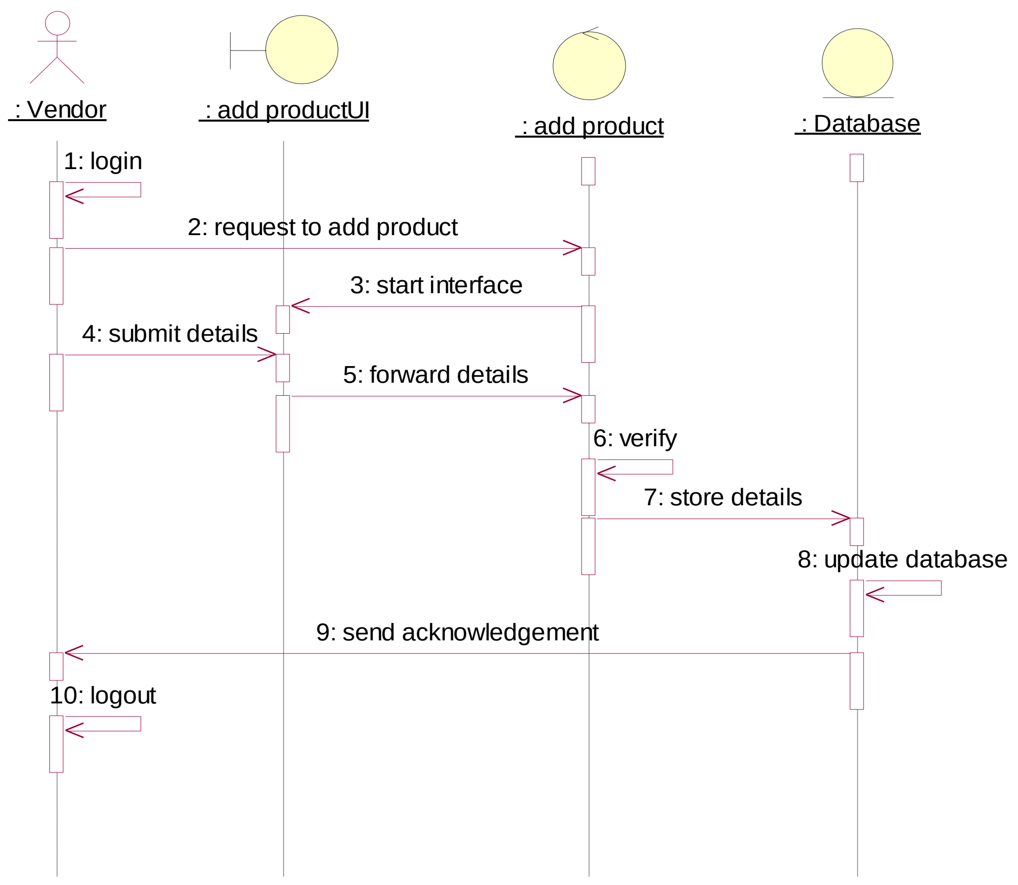


FIGURE 4.4.3: SEQUENCE DIAGRAM FOR VENDOR

4.4.4 Sequence Diagram for Vendor to update product:

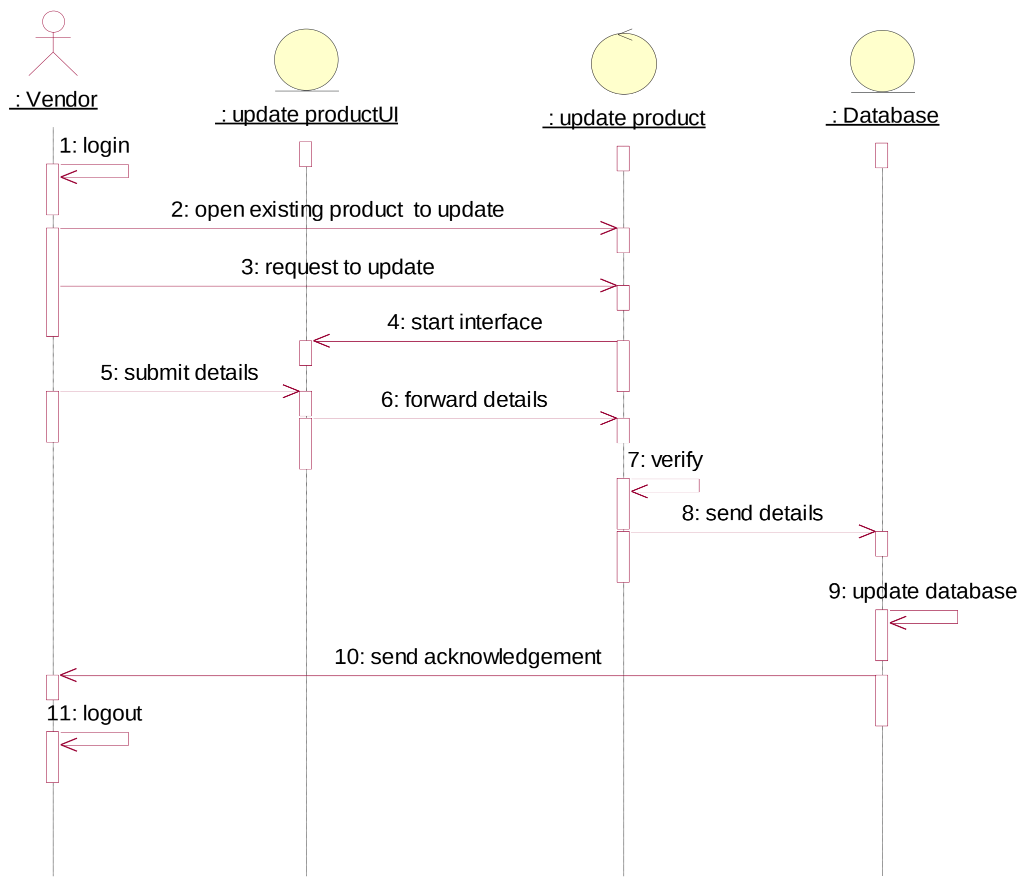


FIGURE 4.4.4: SEQUENCE DIAGRAM FOR VENDOR TO UPDATE PRODUCT

4.4.5 Sequence Diagram for Vendor to delete product:

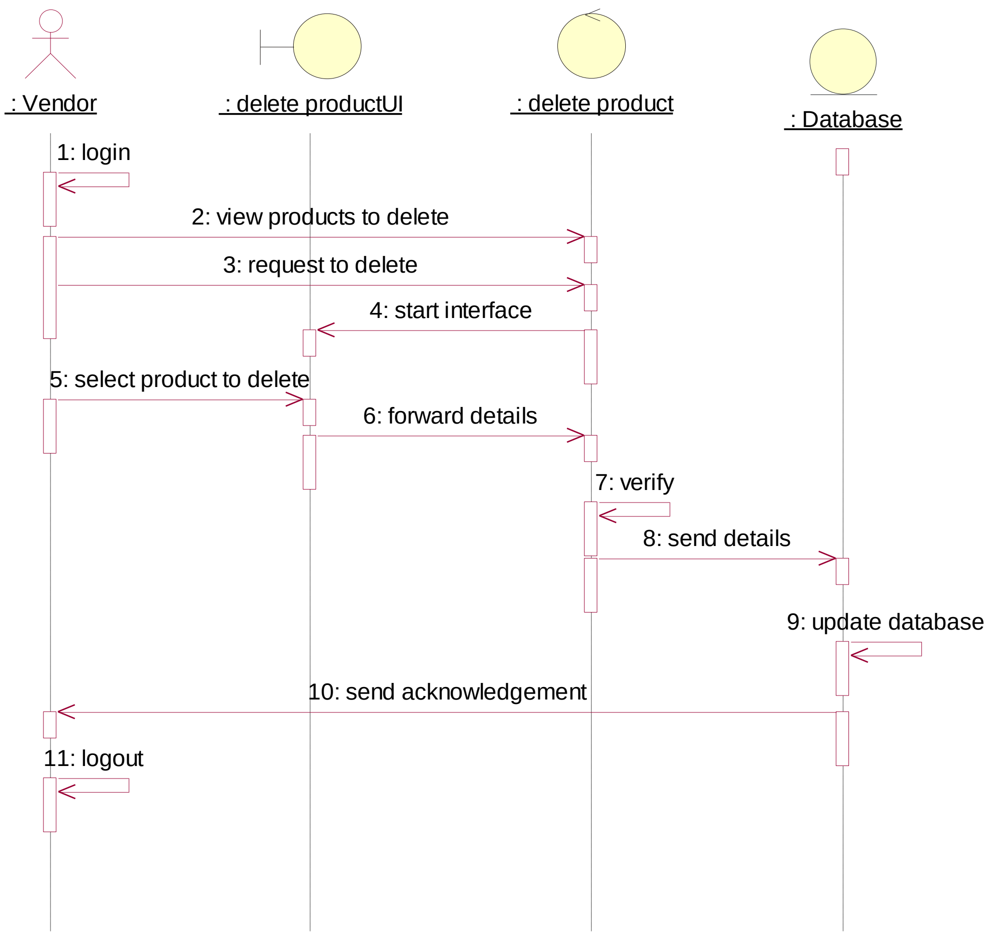


FIGURE 4.4.5 SEQUENCE DIAGRAM FOR VENDOR TO DELETE PRODUCT

4.4.6 Sequence Diagram for Vendor to edit account:

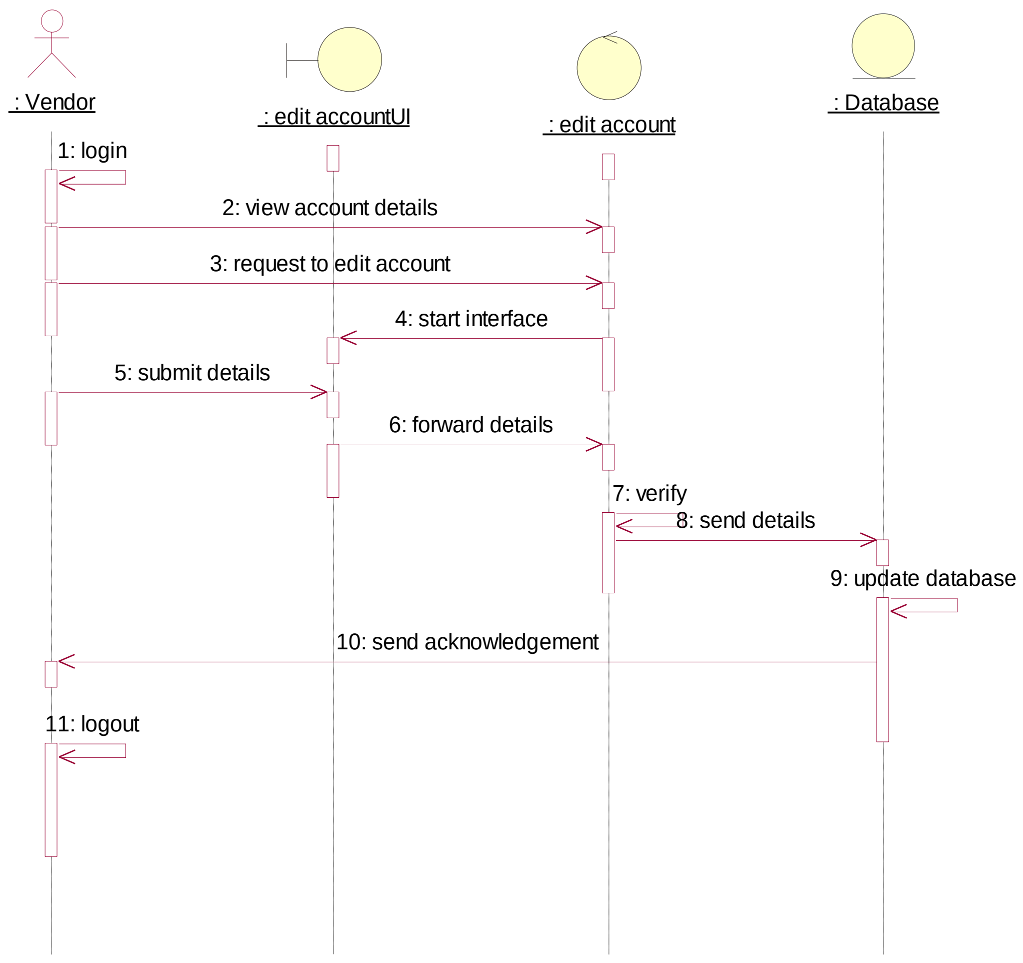


FIGURE 4.4.6 Sequence Diagram for Vendor to edit account

4.5 COLLABORATION DIAGRAMS:

A COLLABORATION DIAGRAM is an alternate way to show a scenario. This type of diagram shows object interactions organized around the objects and their links to each other. A collaboration diagram contains

- Objects drawn as rectangles
- Links between objects shown as lines connecting the linked objects
- Messages shown as text and an arrow that points from the client to the supplier

There are two types of numbering schemes:

1. Flat numbered sequence

   In this messages are numbered as 1, 2, 3…….
2. Decimal numbered sequence

   In this messages are given numbers as 1.1, 2.2, 1.3 …

Purpose:

1. Collaboration diagrams are very useful for drawing class diagrams.
2. These diagrams give the interactions between the classes and objects.
3. To find the operations of the classes.
4. Using command we can draw sequence diagram directly from the collaboration diagrams.

Differences between Sequence and Collaboration Diagrams:

1. Sequence diagrams are time oriented where as collaboration diagrams shows the static connections among the objects.
2. Sequence diagram is very easy to read where as collaboration diagram is some what complex to read.

4.5.1 Collaboration Diagram for Vendor Registration:

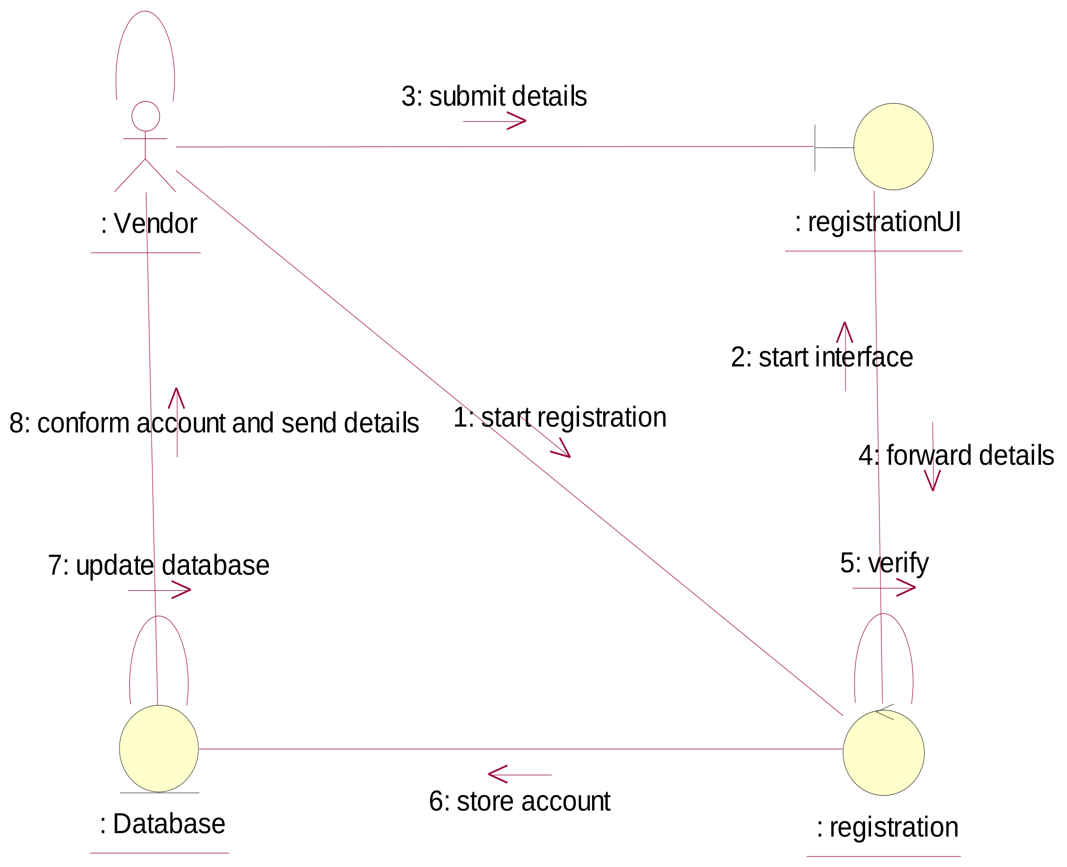


FIGURE 4.5.1 Collaboration Diagram for Vendor Registration

4.5.2 Collaboration Diagram for Vendor Login:

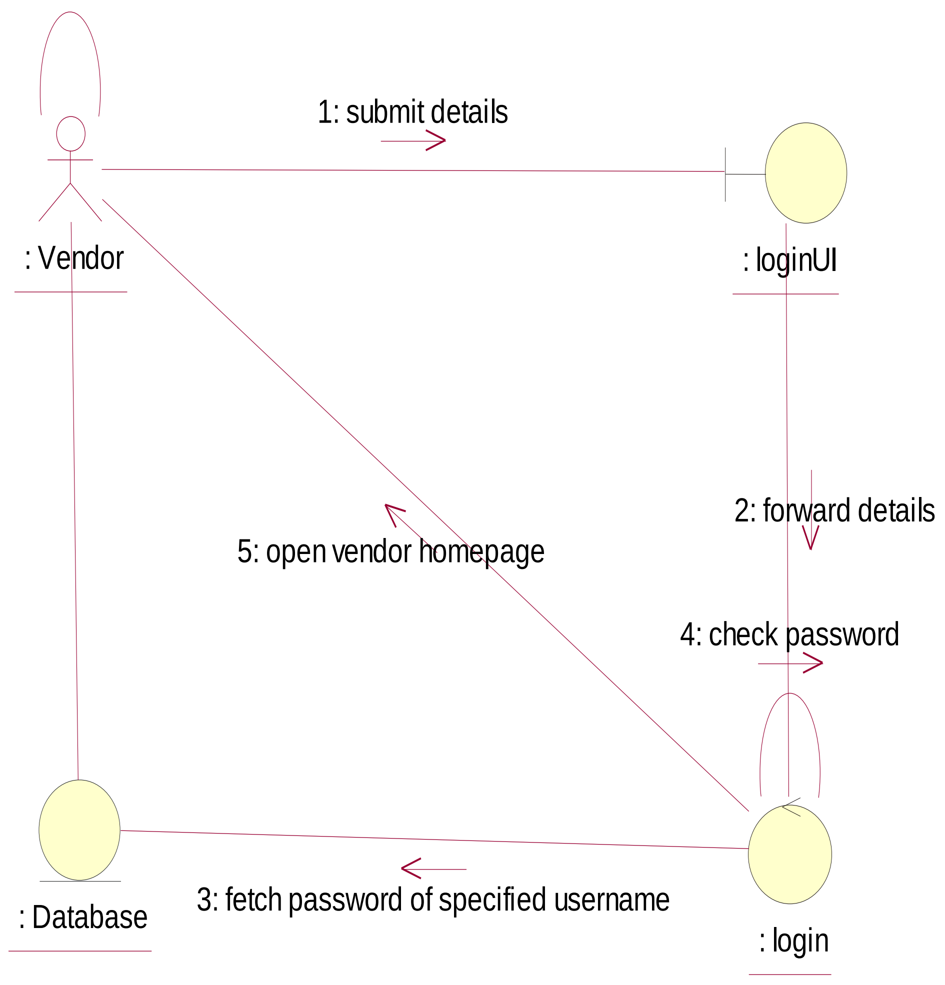


FIGURE 4.5.2 Collaboration diagram for Vendor Login

4.5.3 Collaboration Diagram for Vendor to add product:

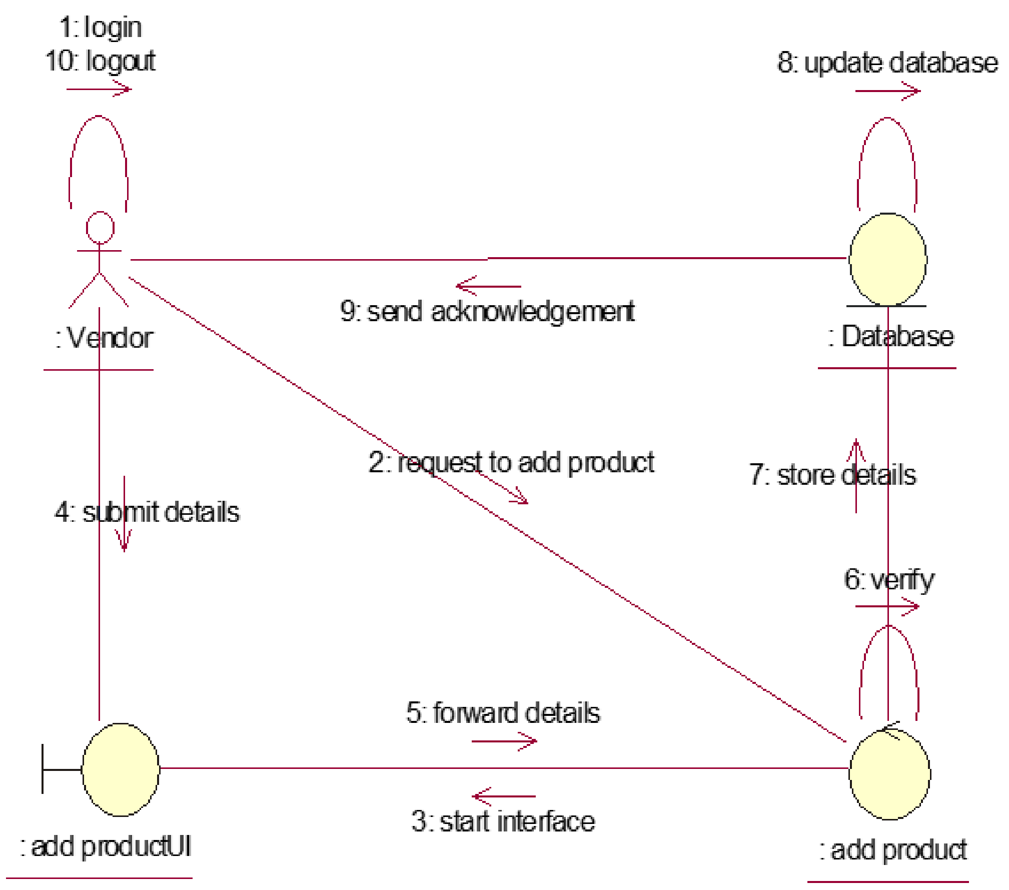


FIGURE4.5.3 Collaboration Diagram for Vendor to add product:

4.5.4 Collaboration Diagram for Vendor to update product:

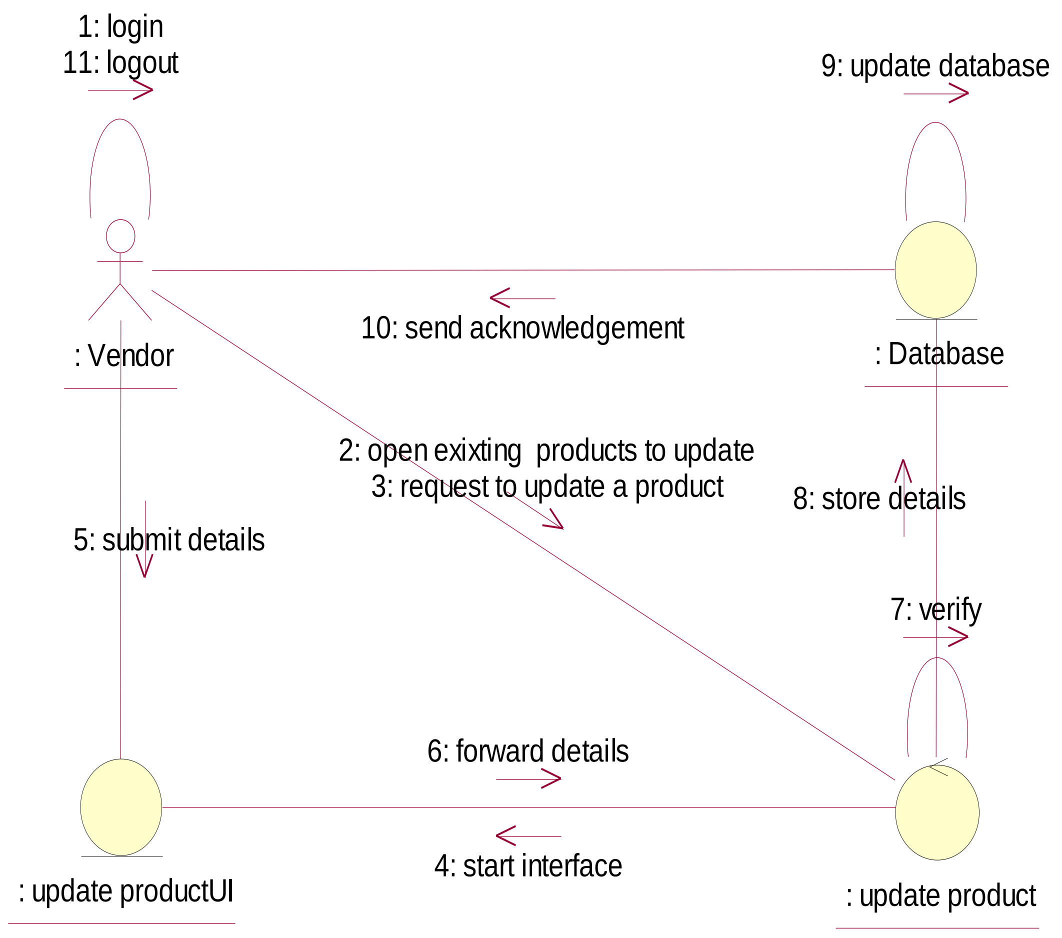


FIGURE 4.5.4 Collaboration Diagram for Vendor to update product

4.5.5 Collaboration Diagram for Vendor to delete product:

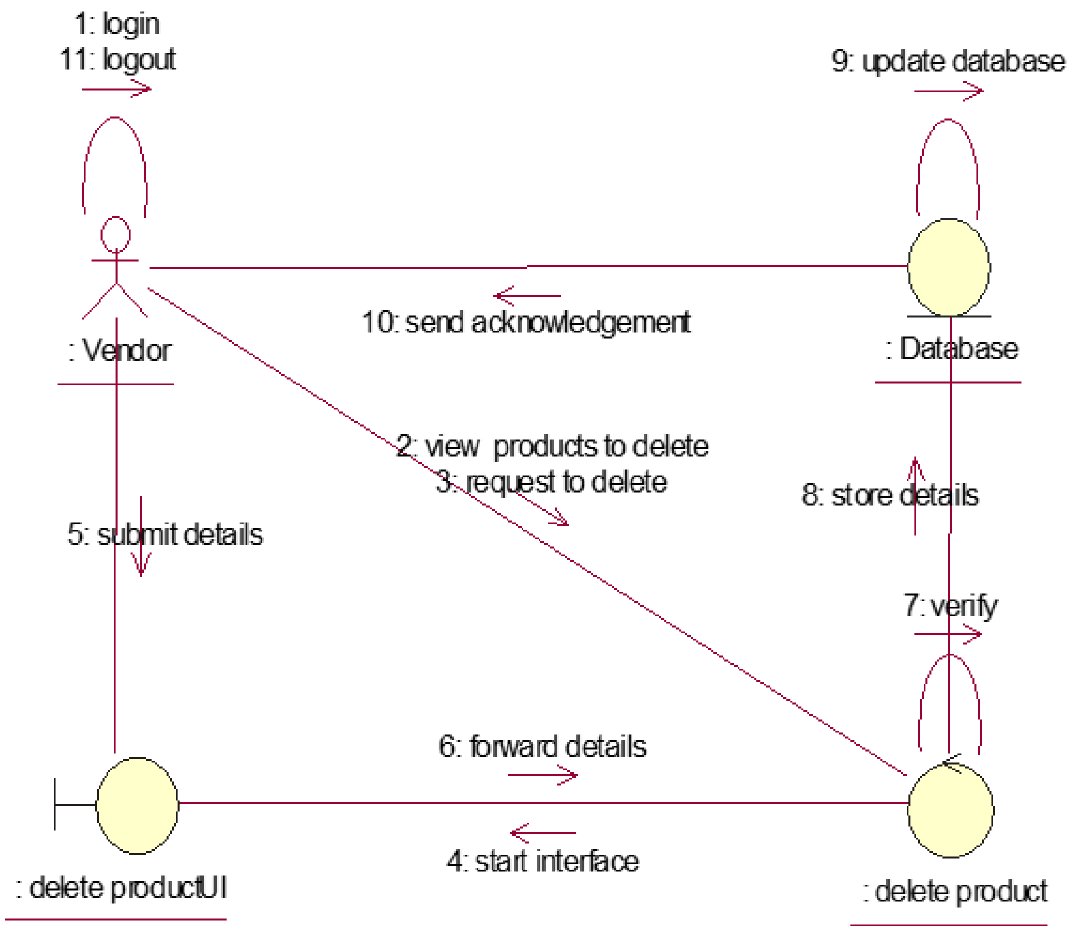


FIGURE 4.5.5 Collaboration Diagram for Vendor to delete product

4.5.6 Collaboration Diagram for Vendor to edit account:

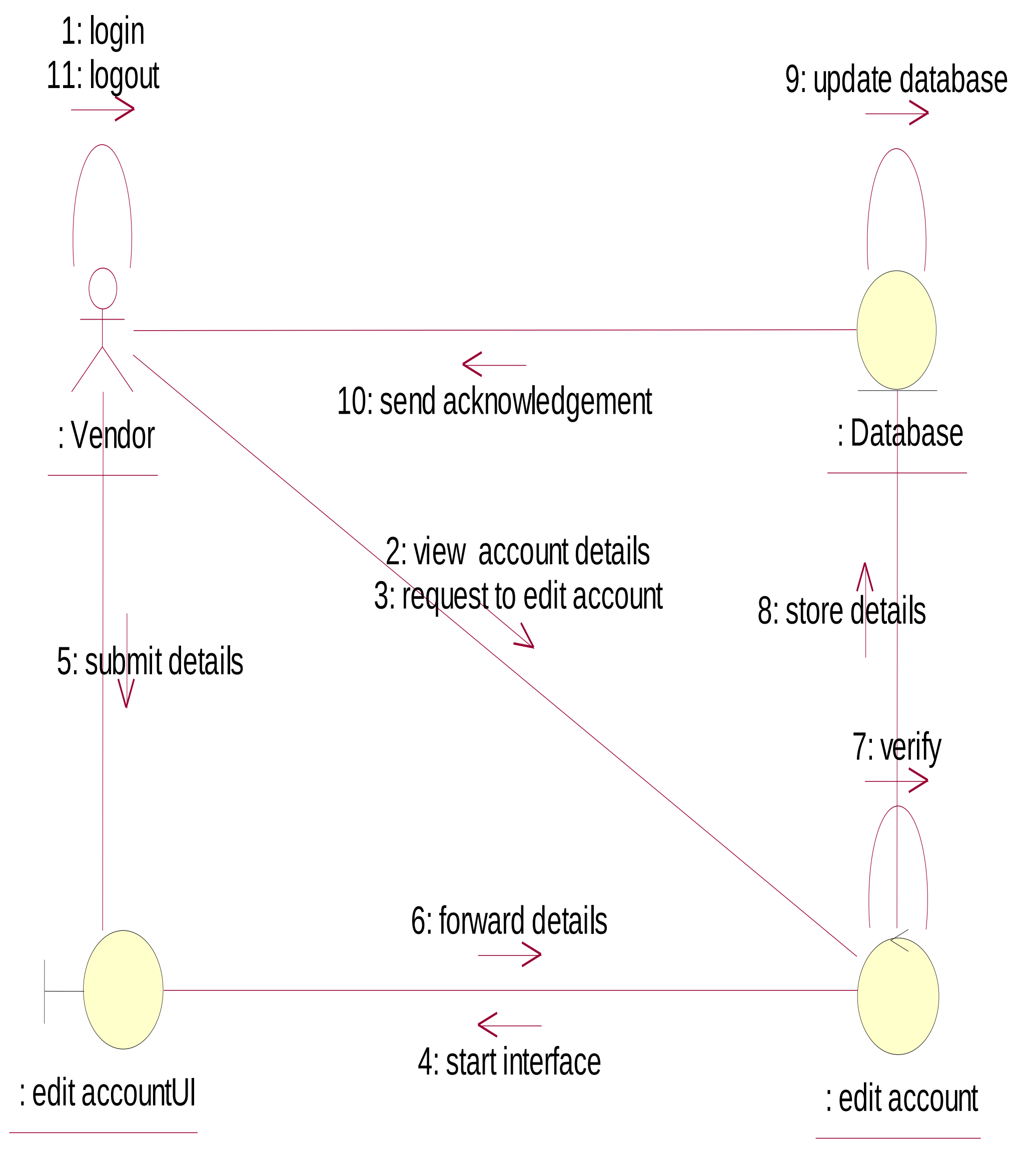


FIGURE 4.5.6 Collaboration Diagram for Vendor to edit account

4.5.7Collaboration Diagram for Customer for booking products:

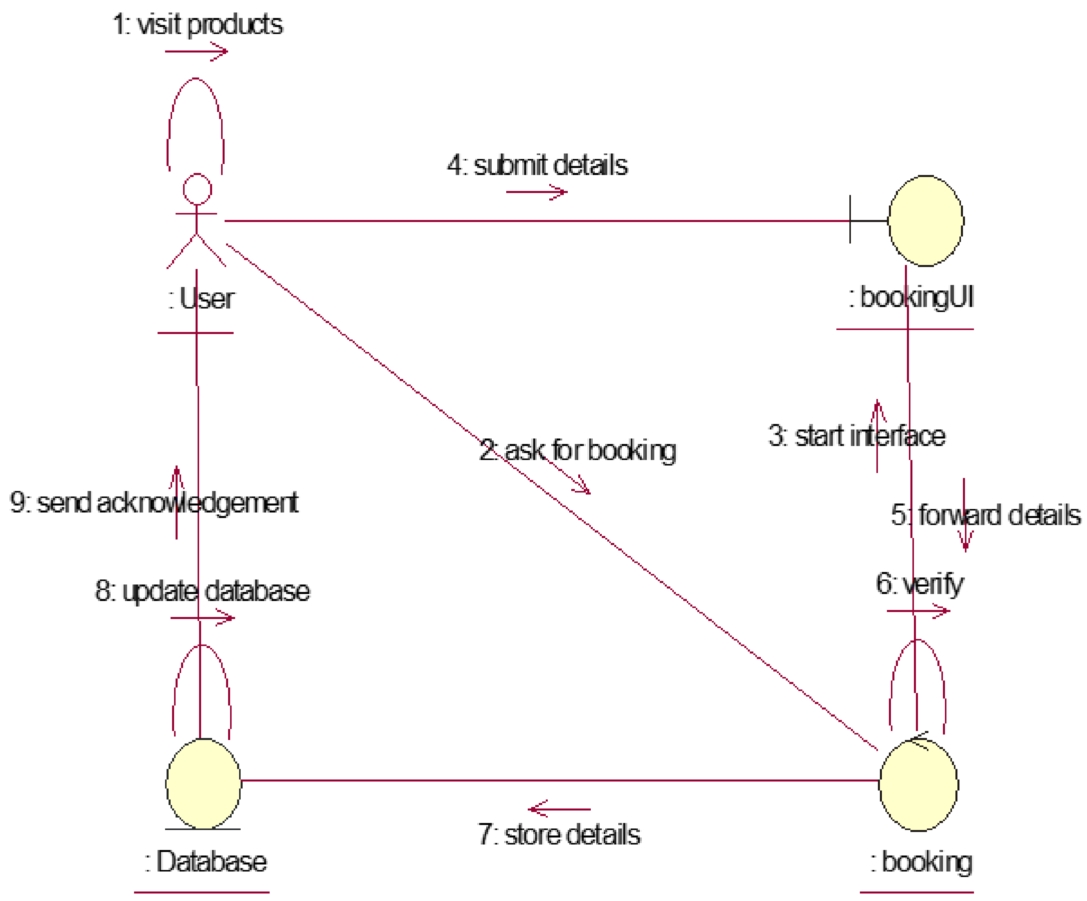


FIGURE 4.5.7 Collaboration Diagram for Customer for booking products

4.6 ACTIVITY DIAGRAMS

4.6.1 Activity diagram for Vendor Perspective:

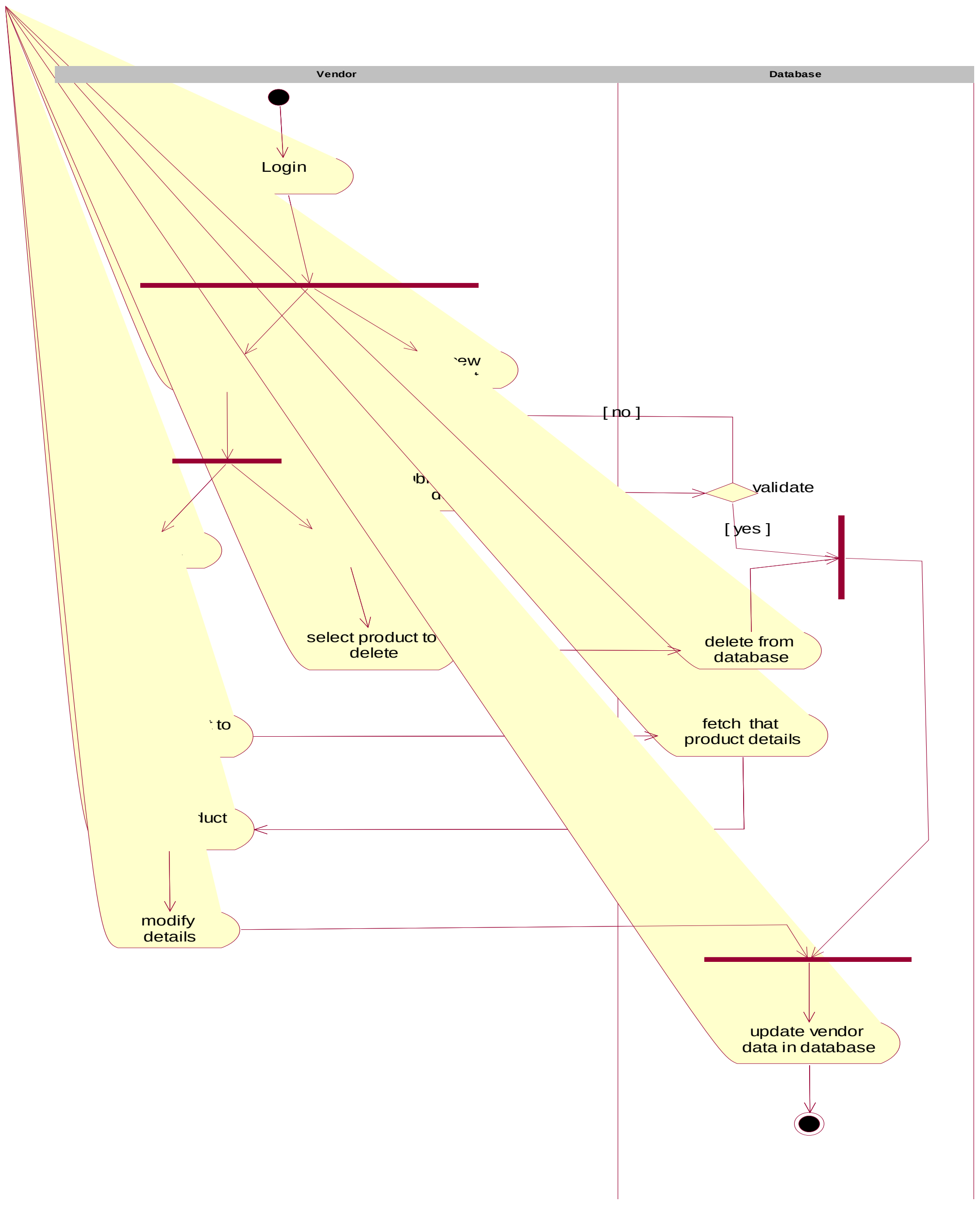


FIGURE 4.6.1 Activity diagram for Vendor Perspective

4.6.2 Activity Diagram for user prospective:

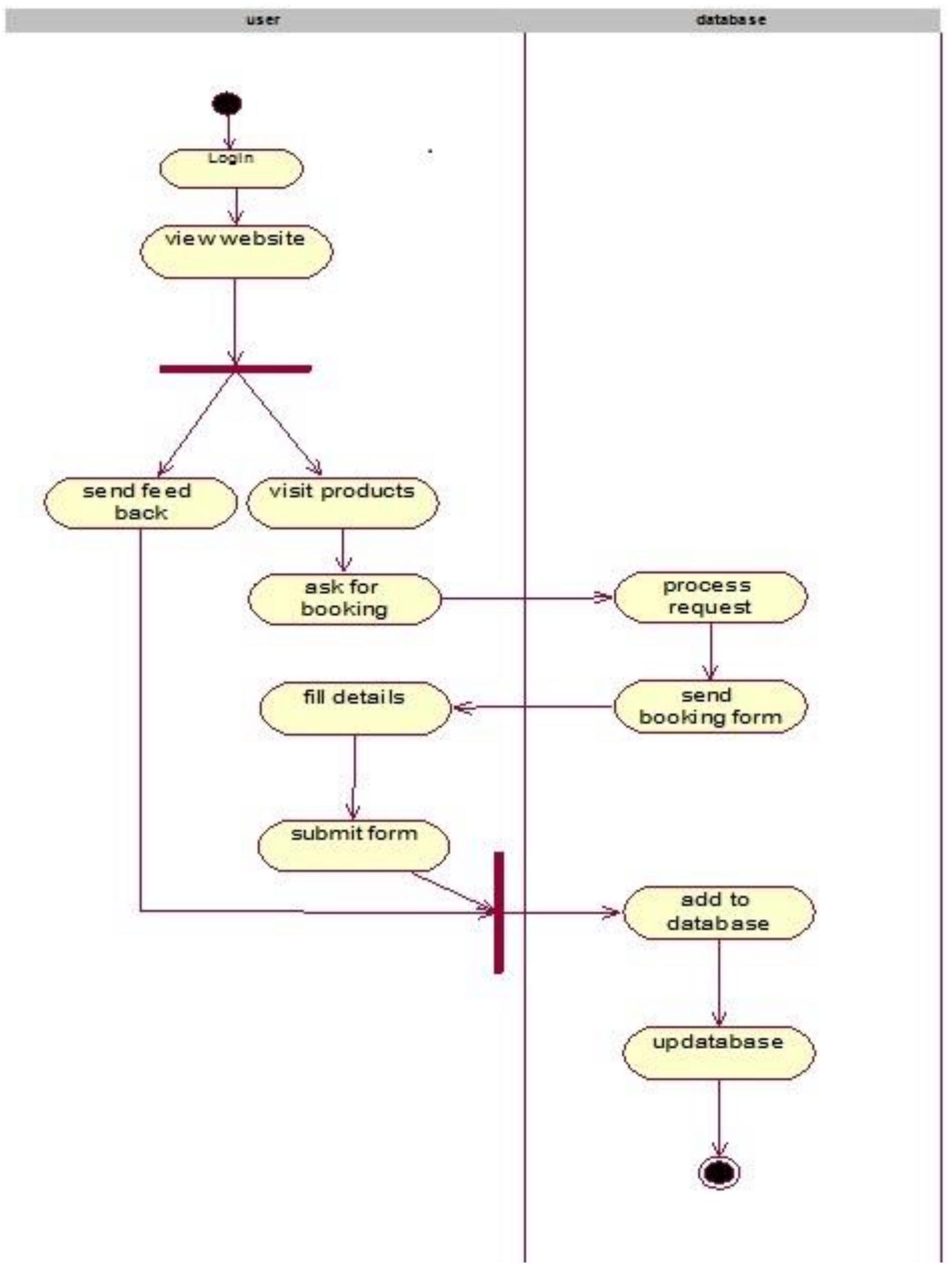


FIGURE 4.6.2 Activity Diagram for user prospective

4.6.3 Activity Diagram for Administrator Prospective:

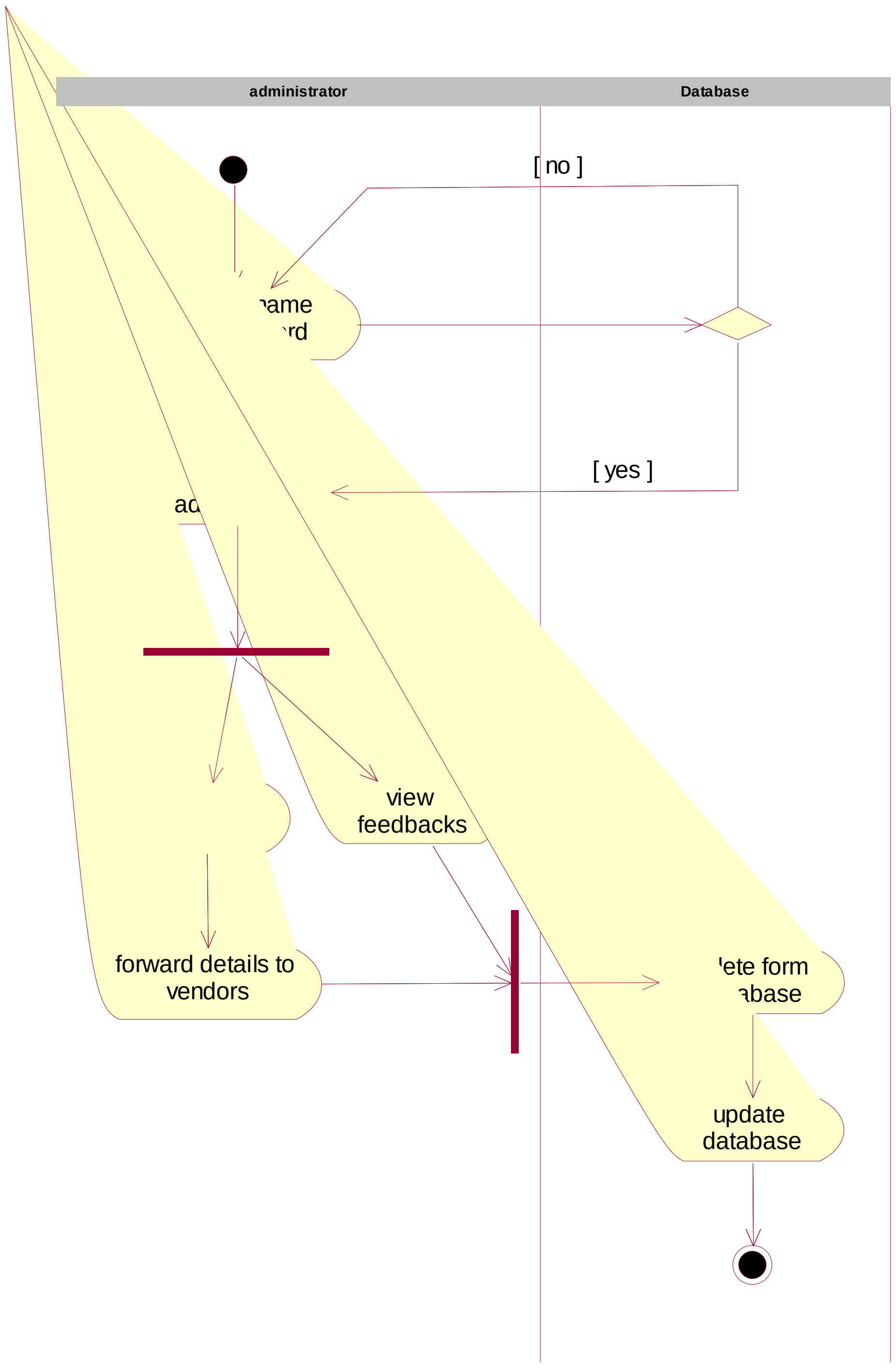


FIGURE 4.6.3 Activity Diagram for Administrator Prospective

4.7. STANDARDS AND TECHNOLOGIES USED:

Language Used	:	PHP
Database	:	MY SQL Server
Design Tools	:	HTML 5, CSS 3, Java Script

4.7.1 MY SQL:

MySQL is an open-source relational database management system (RDBMS).Its name is a combination of "My", the name of co-founder Michael Widenius' daughter, and "SQL", the abbreviation for Structured Query Language. 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. For proprietary use, several paid editions are available, and offer additional functionality. MySQL is a central component of the LAMP open source web application software stack (and other "AMP" stacks). LAMP is an acronym for "Linux, Apache, MySQL, Perl/PHP/Python". Applications that use the MySQL database include: TYPO3, MODx, Joomla, WordPress, phpBB, MyBB, and Drupal. MySQL is also used in many high-profile, large-scale websites, including Google, Facebook, Twitter, Flickr, and YouTube.

MySQL is written in C and C++. Its SQL parser is written in yacc, but it uses a home-brewed lexical analyzer. MySQL works on many system platforms, including AIX, BSDi, FreeBSD, HP-UX, eComStation, i5/OS, IRIX, Linux, macOS, Microsoft Windows, NetBSD, Novell NetWare, OpenBSD, OpenSolaris, OS/2 Warp, QNX, Oracle Solaris, Symbian, SunOS, SCO OpenServer, SCO UnixWare, Sanos and Tru64. A port of MySQL to OpenVMS also exists. The MySQL server software itself and the client libraries use dual-licensing distribution. They are offered under GPL version 2, beginning from 28 June 2000 (which in 2009 has been extended with a FLOSS License Exception) or to use a proprietary license. Support can be obtained from the official manual.Free support additionally is available in different IRC channels and forums. Oracle offers paid support via its MySQL Enterprise products. They differ in the scope of services and in price. Additionally, a number of third party organisations exist to provide support and services, including MariaDB and Percona. MySQL has received positive reviews, and reviewers noticed it "performs extremely well in the average case" and that the "developer interfaces are there, and the documentation (not to mention feedback in the real world via Web sites and the like) is very, very good". It has also been tested to be a "fast, stable and true multiuser, multi-threaded sql database server".

4.7.2 PHP

It is a server-side scripting language designed primarily for web development but also used as a general-purpose programming language. Originally created by Rasmus Lerdorf in 1994, the PHP reference implementation is now produced by The PHP Development Team. PHP originally stood for *Personal Home Page*, but it now stands for the recursive acronym *PHP: Hypertext Preprocessor*.

PHP code may be embedded into HTML or HTML5 markup, or it can be used in combination with various web template systems, web content management systems and web frameworks. PHP code is usually processed by a PHP interpreter implemented as a module in the web server or as a Common Gateway Interface (CGI) executable. The web server software combines the results of the interpreted and executed PHP code, which may be any type of data, including images, with the generated web page. PHP code may also be executed with a command-line interface (CLI) and can be used to implement standalone graphical applications.

The standard 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.

The PHP language evolved without a written formal specification or standard until 2014, leaving the canonical PHP interpreter as a *de facto* standard. Since 2014 work has gone on to create a formal PHP specification.

PHP 7:

During 2014 and 2015, a new major PHP version was developed, which was numbered PHP 7. The numbering of this version involved some debate. While the PHP 6 Unicode experiment had never been released, several articles and book titles referenced the PHP 6 name, which might have caused confusion if a new release were to reuse the name.] After a vote, the name PHP 7 was chosen.

The foundation of PHP 7 is a PHP branch that was originally dubbed *PHP next generation* (*phpng*). It was authored by Dmitry Stogov, Xinchen Hui and Nikita Popov, and aimed to optimize PHP performance by refactoring the Zend Engine to use more compact data structures with improved cache locality while retaining near-complete language compatibility. As of 14 July 2014, WordPress-based benchmarks, which served as the main benchmark suite for the phpng project, showed an almost 100% increase in performance. Changes from phpng are also expected to make it easier to improve performance in the future, as more compact data structures and other changes are seen as better suited for a successful migration to a just-in-time (JIT) compiler. Because of the significant changes, the reworked Zend Engine is called *Zend Engine 3*, succeeding Zend Engine 2 used in PHP 5.

Because of major internal changes in phpng, it must receive a new major version number of PHP, rather than a minor PHP 5 release, according to PHP's release process. Major versions of PHP are allowed to break backward-compatibility of code and therefore PHP 7 presented an opportunity for other improvements beyond phpng that require backward-compatibility breaks, including wider use of exceptions, reworking variable syntax to be more consistent and complete, and the deprecation or removal of various legacy features.

PHP 7 also introduced new language features, including return type declarations for functions, which complement the existing parameter type declarations, and support for the scalar types (integer, float, string, and boolean) in parameter and return type declarations.

4.7.3 HTML 5

HTML 5 is a markup language used for structuring and presenting content on the World Wide Web. It is the fifth and current version of the HTML standard.

It was published in October 2014 by the World Wide Web Consortium (W3C) to improve the language with support for the latest multimedia, while keeping it both easily readable by humans and consistently understood by computers and devices such as web browsers, parsers, etc. HTML5 is intended to subsume not only HTML 4, but also XHTML 1 and DOM Level 2 HTML.

HTML5 includes detailed processing models to encourage more interoperable implementations; it extends, improves and rationalizes the markup available for documents, and introduces markup and application programming interfaces (APIs) for complex web applications. For the same reasons, HTML5 is also a candidate for cross-platform mobile applications, because it includes features designed with low-powered devices in mind.

Many new syntactic features are included. To natively include and handle , the new <video>, <audio> and <canvas> elements were added, and support for scalable vector graphics (SVG) content and MathML for mathematical formulas. To enrich the semantic content of documents, new page structure elements such as <main>, <section>, <article>, <header>, <footer>, <aside>, <nav> and <figu re>, are added. New attributes are introduced, some elements and attributes have been removed, and others such as <a>, <cite> and <menu> have been changed, redefined or standardized.

The APIs and Document Object Model (DOM) are now fundamental parts of the HTML5 specification] and HTML5 also better defines the processing for any invalid documents.

4.7.4 CSS 3:

Cascading Style Sheets (**CSS**) is a style sheet language used for describing the presentation of a document written in a markup language. Although most often used to set the visual style of web pages and user interfaces written in HTML and XHTML, the language can be applied to any XML document, including plain XML, SVG and XUL, and is applicable to rendering in speech, or on other media. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications.

CSS is designed primarily to enable the separation of presentation and content, including aspects such as the layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate .css file, and reduce complexity and repetition in the structural content.

Separation of formatting and content makes it possible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. It can also display the web page differently depending on the screen size or viewing device. Readers can also specify a different style sheet, such as a CSS file stored on their own computer, to override the one the author specified.

Changes to the graphic design of a document (or hundreds of documents) can be applied quickly and easily, by editing a few lines in the CSS file they use, rather than by changing markup in the documents.

The CSS specification describes a priority scheme to determine which style rules apply if more than one rule matches against a particular element. In this so-called *cascade*, priorities (or *weights*) are calculated and assigned to rules, so that the results are predictable.

The CSS specifications are maintained by the World Wide Web Consortium (W3C). Internet media type (MIME type) `text/css` is registered for use with CSS by RFC 2318 (March 1998). The W3C operates a free CSS validation service for CSS documents.

4.7.5 JAVASCRIPT:

JavaScript is one of the three core technologies of World Wide Web content production; the majority of websites employ it, and all modern Web browsers support it without the need for plug-ins. JavaScript is a multi-paradigm language, since it supports prototype-based with first-class functions, imperative, and functional programming paradigms. It has an API for working with text, arrays, dates, regular expressions, and basic manipulation of the DOM, but does not include network, storage, or sophisticated graphics APIs, relying instead upon APIs made available by its host environment.

Although there are strong outward similarities between JavaScript and Java, including language name, syntax, and respective standard libraries, the two languages are distinct and differ greatly in design; JavaScript was influenced by programming languages such as Self and Scheme.

JavaScript is also used in environments that are not Web-based, such as PDF documents, site-specific browsers, and desktop widgets. Newer and faster JavaScript virtual machines (VMs) and platforms built upon them have also increased the popularity of JavaScript for server-side Web applications. On the client side, developers have traditionally implemented JavaScript as an interpreted language, but more recent browsers perform just-in-time compilation. Programmers also use JavaScript in video-game development and in desktop and mobile applications.

4.7.6 BOOTSTRAP:

Bootstrap is a free and open-source front-end web framework for designing websites and web applications. It contains HTML- and CSS-based design templates for typography, forms, buttons, navigation and other interface components, as well as optional JavaScript extensions. Unlike many web frameworks, it concerns itself with front-end development only.

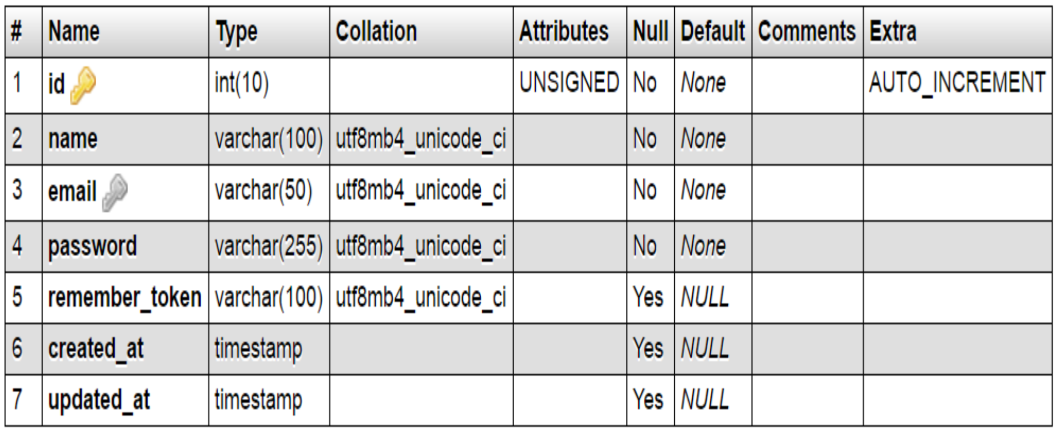
Bootstrap is the second most-starred project on GitHub, with more than 107,000 stars and 48,000 forks. Bootstrap 3 supports the latest versions of the Google Chrome, Firefox, Internet Explorer, Opera, and Safari (except on Windows). It additionally supports back to IE8 and the latest Firefox Extended Support Release (ESR).

Since 2.0, Bootstrap supports responsive web design. This means the layout of web pages adjusts dynamically, taking into account the characteristics of the device used (desktop, tablet, mobile phone).

Starting with version 3.0, Bootstrap adopted a mobile-first design philosophy, emphasizing responsive design by default.

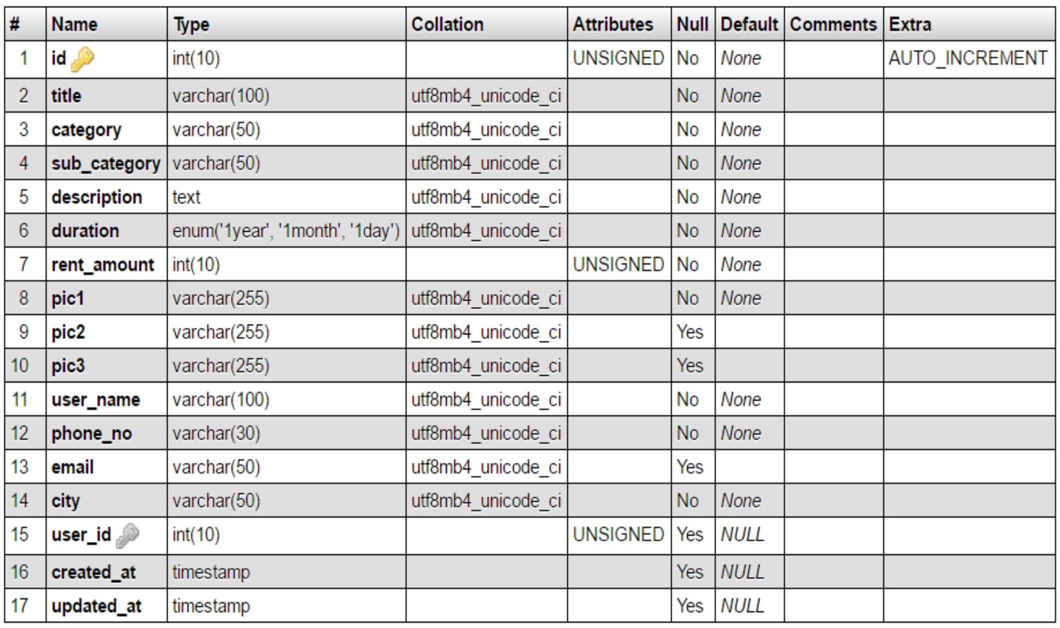
The version 4.0 alpha release added Sass and flexbox support.

4.8 TABLES USED IN THE SYSTEM:



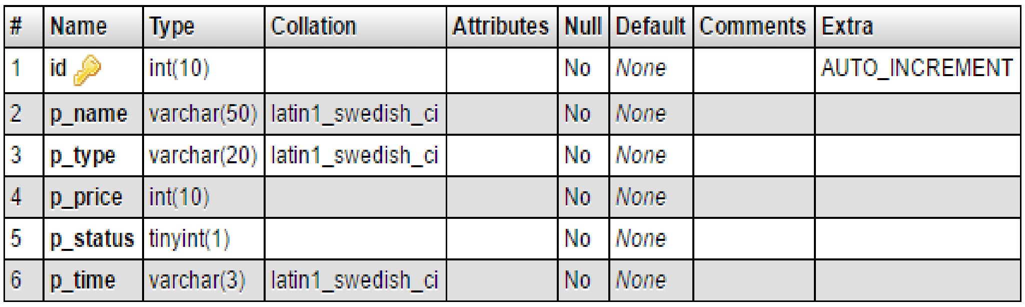
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	int(10)		UNSIGNED	No	*None*		AUTO_INCREMENT
2	name	varchar(100)	utf8mb4_unicode_ci		No	*None*		
3	email	varchar(50)	utf8mb4_unicode_ci		No	*None*		
4	password	varchar(255)	utf8mb4_unicode_ci		No	*None*		
5	remember_token	varchar(100)	utf8mb4_unicode_ci		Yes	*NULL*		
6	created_at	timestamp			Yes	*NULL*		
7	updated_at	timestamp			Yes	*NULL*		

1. User Database Table



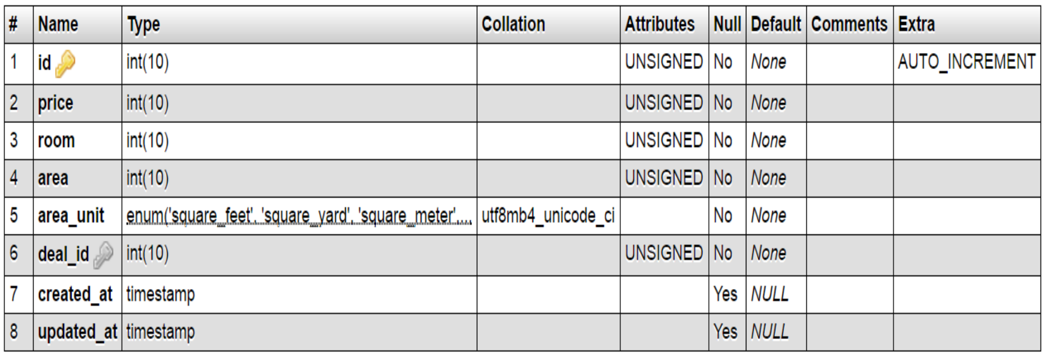
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	int(10)		UNSIGNED	No	*None*		AUTO_INCREMENT
2	title	varchar(100)	utf8mb4_unicode_ci		No	*None*		
3	category	varchar(50)	utf8mb4_unicode_ci		No	*None*		
4	sub_category	varchar(50)	utf8mb4_unicode_ci		No	*None*		
5	description	text	utf8mb4_unicode_ci		No	*None*		
6	duration	enum('1year', '1month', '1day')	utf8mb4_unicode_ci		No	*None*		
7	rent_amount	int(10)		UNSIGNED	No	*None*		
8	pic1	varchar(255)	utf8mb4_unicode_ci		No	*None*		
9	pic2	varchar(255)	utf8mb4_unicode_ci		Yes			
10	pic3	varchar(255)	utf8mb4_unicode_ci		Yes			
11	user_name	varchar(100)	utf8mb4_unicode_ci		No	*None*		
12	phone_no	varchar(30)	utf8mb4_unicode_ci		No	*None*		
13	email	varchar(50)	utf8mb4_unicode_ci		Yes			
14	city	varchar(50)	utf8mb4_unicode_ci		No	*None*		
15	user_id	int(10)		UNSIGNED	Yes	*NULL*		
16	created_at	timestamp			Yes	*NULL*		
17	updated_at	timestamp			Yes	*NULL*		

2. Ads Database Table



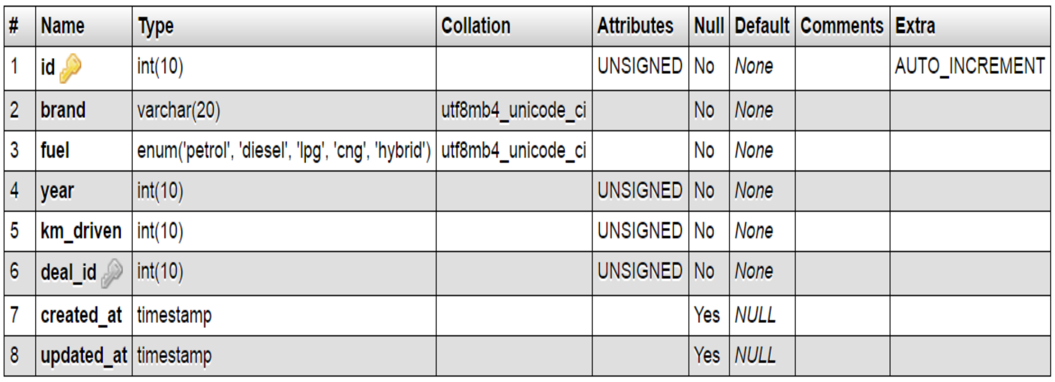
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	int(10)			No	*None*		AUTO_INCREMENT
2	p_name	varchar(50)	latin1_swedish_ci		No	*None*		
3	p_type	varchar(20)	latin1_swedish_ci		No	*None*		
4	p_price	int(10)			No	*None*		
5	p_status	tinyint(1)			No	*None*		
6	p_time	varchar(3)	latin1_swedish_ci		No	*None*		

3. Product Database Table



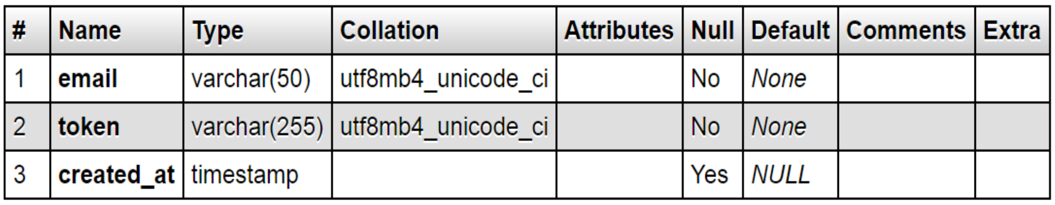
#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	int(10)		UNSIGNED	No	*None*		AUTO_INCREMENT
2	price	int(10)		UNSIGNED	No	*None*		
3	room	int(10)		UNSIGNED	No	*None*		
4	area	int(10)		UNSIGNED	No	*None*		
5	area_unit	enum('square_feet', 'square_yard', 'square_meter',...	utf8mb4_unicode_ci		No	*None*		
6	deal_id	int(10)		UNSIGNED	No	*None*		
7	created_at	timestamp			Yes	*NULL*		
8	updated_at	timestamp			Yes	*NULL*		

4. Properties Database Table



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	id	int(10)		UNSIGNED	No	*None*		AUTO_INCREMENT
2	brand	varchar(20)	utf8mb4_unicode_ci		No	*None*		
3	fuel	enum('petrol', 'diesel', 'lpg', 'cng', 'hybrid')	utf8mb4_unicode_ci		No	*None*		
4	year	int(10)		UNSIGNED	No	*None*		
5	km_driven	int(10)		UNSIGNED	No	*None*		
6	deal_id	int(10)		UNSIGNED	No	*None*		
7	created_at	timestamp			Yes	*NULL*		
8	updated_at	timestamp			Yes	*NULL*		

5. Vehicle Database Table



#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra
1	**email**	varchar(50)	utf8mb4_unicode_ci		No	*None*		
2	**token**	varchar(255)	utf8mb4_unicode_ci		No	*None*		
3	**created_at**	timestamp			Yes	*NULL*		

6. Password Reset Database Table

4.8.1 DATABASE SCHEMA:

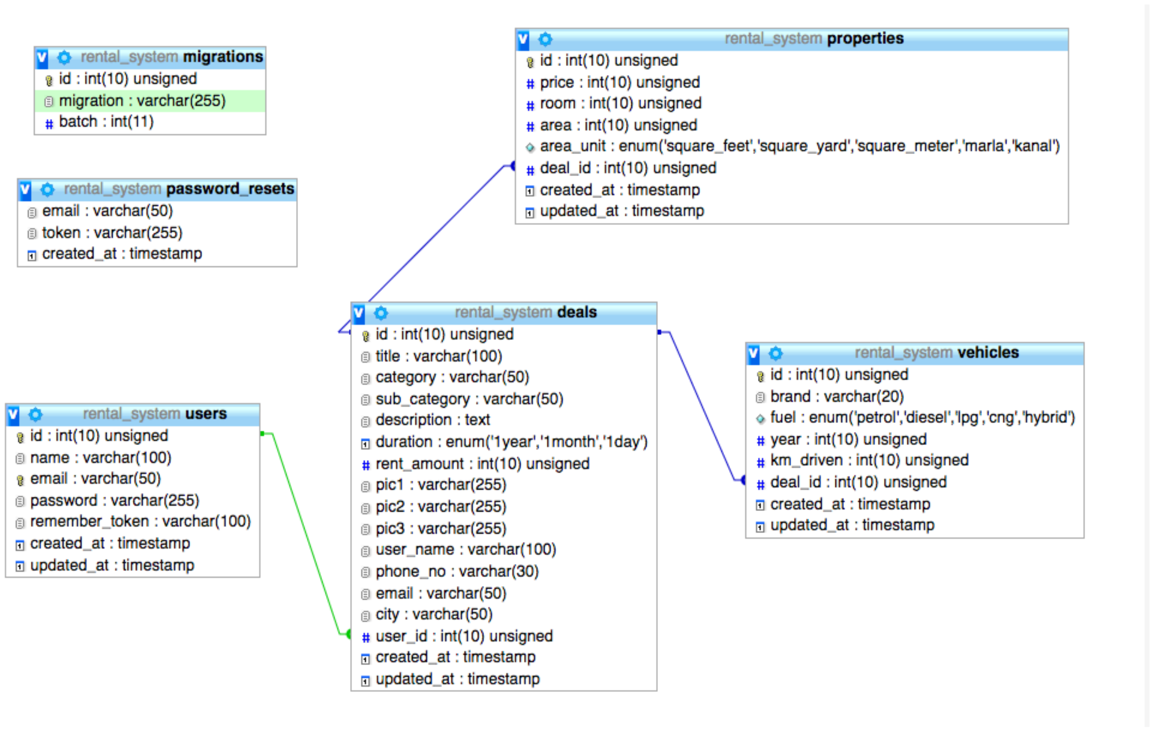


FIGURE 4.8.1: DATABASE SCHEMA OF ORS

4.9 ER DIAGRAM:

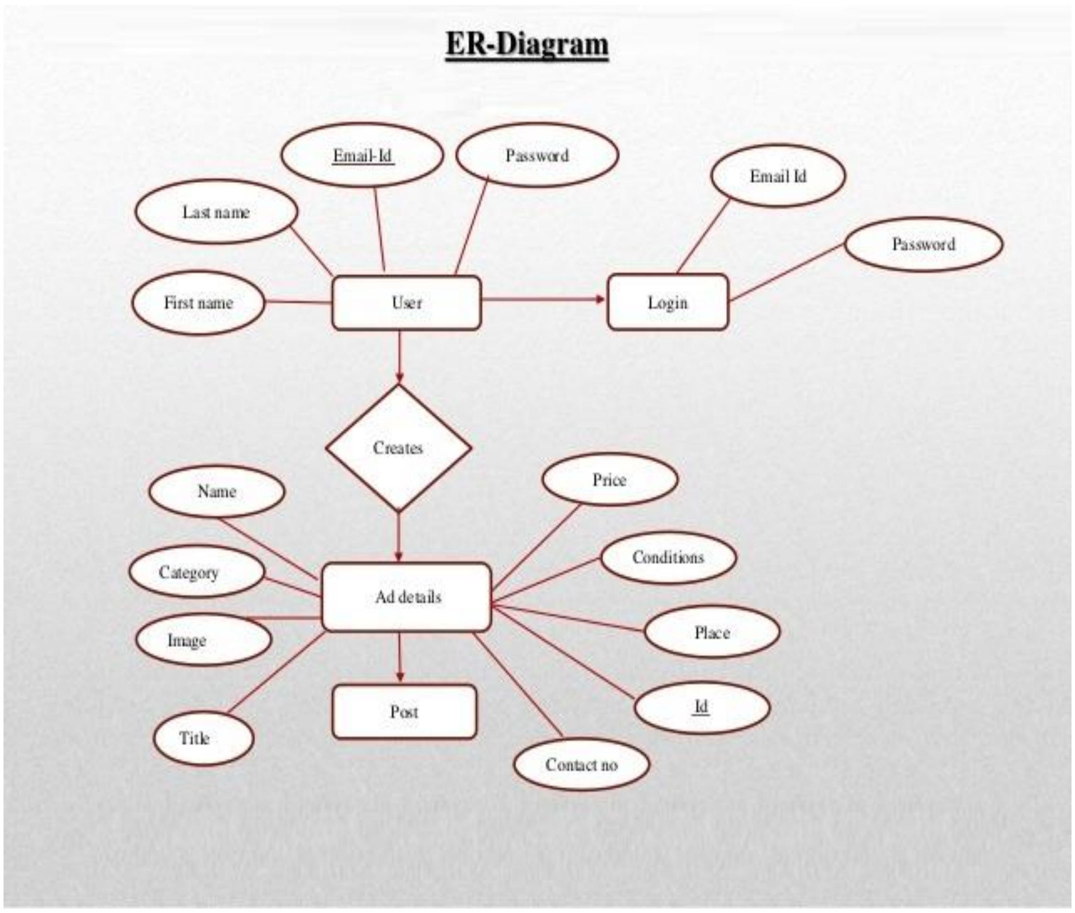


FIGURE 4.9: ER Diagram of ORS

4.10 PROJECT CODE:

4.10.1: For Layout of ORS:

1. App.php :

```
<!DOCTYPE html>

<html lang="{{ config('app.locale') }}">

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">



<!-- CSRF Token -->

<meta name="csrf-token" content="{{ csrf_token() }}">



<title>{{ config('app.name', 'Laravel') }}</title>



<!-- Styles -->

<link href="{{ asset('css/app.css') }}" rel="stylesheet">

<link href="{{ asset('css/mystyle.css') }}" rel="stylesheet">



<!-- Scripts -->

<script>

window.Laravel = {!! json_encode([
'csrfToken' => csrf_token(),

]) !!};

</script>

</head>

<body>

<div id="app">

<nav class="navbar navbar-default navbar-static-top">

<div class="container">

<div class="navbar-header">



<!-- Collapsed Hamburger -->

<button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-
target="#app-navbar-collapse">

<span class="sr-only">Toggle Navigation</span>

<span class="icon-bar"></span>

<span class="icon-bar"></span>

<span class="icon-bar"></span>

</button>



<!-- Branding Image -->

<a class="navbar-brand" href="{{ url('/') }}">

{{ config('app.name', 'Laravel') }}

</a>
```

```
</div>


<div class="collapse navbar-collapse" id="app-navbar-collapse">
<!-- Left Side Of Navbar -->
<ul class="nav navbar-nav">
&nbsp;
</ul>


<!-- Right Side Of Navbar -->
<ul class="nav navbar-nav navbar-right">
<!-- Authentication Links -->
@if (Auth::guest())
<li><a href="{{ route('login') }}">Login</a></li>
<li><a href="{{ route('register') }}">Register</a></li>
@else
<li class="dropdown">
<a href="#" class="dropdown-toggle" data-toggle="dropdown" role="button" aria-
expanded="false">
{{ Auth::user()->name }} <span class="caret"></span>
</a>


<ul class="dropdown-menu" role="menu">
<li>
```

```
<a href="{{ route('logout') }}"

onclick="event.preventDefault();

document.getElementById('logout-form').submit();">

Logout

</a>



<form id="logout-form" action="{{ route('logout') }}" method="POST"
style="display: none;">

{{ csrf_field() }}

</form>

</li>

</ul>

</li>

@endif

</ul>

</div>

</div>

</nav>



@yield('content')

</div>



<!-- Scripts -->
```

```
<script src="{{ asset('js/app.js') }}"></script>

</body>

</html>
```

2. Header.php :

```
<!DOCTYPE html>

<html lang="en">

<head>

<title>Online Renting System</title>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

{{--<link rel="stylesheet"
href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">

<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>

<script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>

--}}

<link href="{{ asset('css/app.css') }}" rel="stylesheet">

<link href="{{ asset('css/bootstrap.min.css') }}" rel="stylesheet">

<link href="{{ asset('css/font-awesome.min.css') }}" rel="stylesheet">


<link href="{{ asset('css/mystyle.css') }}" rel="stylesheet">
```

```
<script src="{{ asset('js/bootstrap.min.js') }}"></script>

<script src="{{ asset('js/jquery.min.js') }}"></script>


<style>

/* Remove the navbar's default rounded borders and increase the bottom margin */

.navbar {

margin-bottom: 50px;

border-radius: 0;

}


/* Remove the jumbotron's default bottom margin */

.jumbotron {

margin-bottom: 0;

}


/* Add a gray background color and some padding to the footer */

footer {

background-color: #f2f2f2;

padding: 25px;

}

</style>

</head>

<body>
```

```
<div class="jumbotron">
<div class="container text-center">
<h1>Online Renting System</h1>
<p>India's Best Renting System For Renting And Borrowing Products</p>
</div>
</div>

<nav class="navbar navbar-inverse">
<div class="container-fluid">
<div class="navbar-header">
<button type="button" class="navbar-toggle" data-toggle="collapse" data-
target="#myNavbar">
<span class="icon-bar"></span>
<span class="icon-bar"></span>
<span class="icon-bar"></span>
</button>
<a class="navbar-brand" href="{{ route('add.index') }}">ORS</a>
</div>
<div class="collapse navbar-collapse" id="myNavbar">
<ul class="nav navbar-nav">
<li class="active"><a href="{{ route('add.index') }}">Home</a></li>
<li><a href="{{ route('add.create') }}">Submit an Ad</a></li>
```

```
</ul>

<ul class="nav navbar-nav navbar-right">

@if(\Illuminate\Support\Facades\Auth::check())

<li><a href="{{ route('home') }}"><span class="glyphicon glyphicon-user"></span>
My Adds</a></li>

<li><a href="{{ route('logout') }}" onclick="event.preventDefault();
document.getElementById('logout-form').submit();"><span class="glyphicon
glyphicon-user"></span> Logout </a></li>

<form id="logout-form" action="{{ route('logout') }}" method="POST"
style="display: none;">

{{ csrf_field() }}

</form>

@else

<li><a href="{{ route('login') }}"><span class="glyphicon glyphicon-user"></span>
Login</a></li>

<li><a href="{{ route('register') }}"> Register</a></li>

@endif

</ul>

</div>

</div>

</nav>


@yield('content')
```

```
<br><br>


<footer class="container-fluid text-center">

<p>Online Renting System</p>

{{--<form class="form-inline">Get deals:

<input type="email" class="form-control" size="50" placeholder="Email Address">

<button type="button" class="btn btn-danger">Sign Up</button>

</form>--}}

</footer>


</body>

</html>
```

4.10.2 For Home Page :

1. Home.php:

```
@extends('layouts.header')


@section('content')

<div class="container">

<div class="row">


<div class="table-responsive">

<table class="table">

<thead>

<tr>

<th> Date </th>

<th> Title </th>

<th> Price </th>

<th> Actions </th>

</tr>

</thead>

<tbody>


@foreach($adds as $add)

<tr>

<td> {{ $add->created_at }}</td>
```

```
<td> {{ $add->title }}</td>

<td> {{ $add->rent_amount }} / {{ $add->duration }}</td>

<td>

<a class="no-text-decoration" href="{{ route('add.show', $add) }}"><span
class="glyphicon glyphicon-user"></span>preview</a>

<a class="no-text-decoration" href="{{ route('add.edit', $add) }}"><span
class="glyphicon glyphicon-user"></span>edit</a>

{{--<a class="no-text-decoration" href="{{ route('add.delete', $add) }}">Delete</a>

--}}

</td>

</tr>

@endforeach



</tbody>

</table>

</div>

</div>

</div>

@endsection
```

2. Welcome.php:

```
@extends('layouts.header')

@section('content')



<div class="container">

<div class="row">

<form action="" method="post" class="form-horizontal">



<?php $cities = config('myconfig.cities'); ?>

<div class="col-sm-4">

<select class="form-control" id="city" name="city">

<option value="">All Cities</option>

@foreach($cities as $city)

<option value="{{ $city }}">{{ strtoupper($city)}}</option>

@endforeach

</select>

</div>

<div class="col-sm-4">

<select class="form-control" id="category" name="category">

<option>-- Select a Category --</option>

<option>Yearly Products</option>

<option>Monthly Products</option>

<option>Daily Products</option>
```

```
</select>

</div>



<?php $vehicles = config('myconfig.category.vehicles');

$properties = config('myconfig.category.property');?>

<div class="col-sm-4">

<select class="form-control" id="sub_category" name="sub_category">

<option>-- Select a Sub Category --</option>

@foreach($vehicles as $vehicle)

<option value="{{ $vehicle }}">{{ strtoupper($vehicle)}}</option>

@endforeach

</select>

</div>

</form>

</div>

</div><br>



<?php $totalRecords = count($adds); ?>

@for($i = 0; $i< $totalRecords; )

<div class="container">

<div class="row">

@for($j = 0; $j < 3 && $i < $totalRecords; $j++, $i++)

<div class="col-sm-4">
```

```
<a class="no-text-decoration" href="{{ route('add.show', [$adds[$i]['id']]) }}">

<div class="panel panel-primary">

<div class="panel-heading">{{ $adds[$i]['title'] }}</div>

<div class="panel-body div-images"><img src="{{
asset('uploads').'/'.$adds[$i]['pic1']}}" class="img-responsive image-height"
alt="Image"></div>

<div class="panel-footer">Rent price : {{ $adds[$i]['rent_amount'] }} / {{
$adds[$i]['duration'] }}</div>

</div>

</a>

</div>

@endfor

</div>

</div>

<br>

@endfor

@endsection
```

4.10.3: For Login and Register:

1. Login.php :

```
@extends('layouts.header')

@section('content')

<div class="container">

<div class="row">

<div class="col-md-8 col-md-offset-2">

<div class="panel panel-default">

<div class="panel-heading">Login</div>

<div class="panel-body">

<form class="form-horizontal" role="form" method="POST" action="{{
route('login') }}">

{{ csrf_field() }}


<div class="form-group{{ $errors->has('email') ? ' has-error' : '' }}">

<label for="email" class="col-md-4 control-label">E-Mail Address</label>


<div class="col-md-6">

<input id="email" type="email" class="form-control" name="email" value="{{
old('email') }}" required autofocus>


@if ($errors->has('email'))

<span class="help-block">
```

```
<strong>{{ $errors->first('email') }}</strong>

</span>

@endif

</div>

</div>


<div class="form-group{{ $errors->has('password') ? ' has-error' : '' }}">

<label for="password" class="col-md-4 control-label">Password</label>


<div class="col-md-6">

<input id="password" type="password" class="form-control" name="password"
required>


@if ($errors->has('password'))

<span class="help-block">

<strong>{{ $errors->first('password') }}</strong>

</span>

@endif

</div>

</div>


<div class="form-group">

<div class="col-md-6 col-md-offset-4">
```

```
<div class="checkbox">

<label>

<input type="checkbox" name="remember" {{ old('remember') ? 'checked' : '' }}>
Remember Me

</label>

</div>

</div>

</div>



<div class="form-group">

<div class="col-md-8 col-md-offset-4">

<button type="submit" class="btn btn-primary">

Login

</button>

</div>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

@endsection
```

2. Register.php:

```
@extends('layouts.header')


@section('content')

<div class="container">

<div class="row">

<div class="col-md-8 col-md-offset-2">

<div class="panel panel-default">

<div class="panel-heading">Register</div>

<div class="panel-body">

<form class="form-horizontal" role="form" method="POST" action="{{
route('register') }}">

{{ csrf_field() }}


<div class="form-group{{ $errors->has('name') ? ' has-error' : '' }}">

<label for="name" class="col-md-4 control-label">Name</label>


<div class="col-md-6">

<input id="name" type="text" class="form-control" name="name" value="{{
old('name') }}" required autofocus>


@if ($errors->has('name'))

<span class="help-block">
```

```
<strong>{{ $errors->first('name') }}</strong>

</span>

@endif

</div>

</div>


<div class="form-group{{ $errors->has('email') ? ' has-error' : '' }}">

<label for="email" class="col-md-4 control-label">E-Mail Address</label>


<div class="col-md-6">

<input id="email" type="email" class="form-control" name="email" value="{{
old('email') }}" required>


@if ($errors->has('email'))

<span class="help-block">

<strong>{{ $errors->first('email') }}</strong>

</span>

@endif

</div>

</div>


<div class="form-group{{ $errors->has('password') ? ' has-error' : '' }}">

<label for="password" class="col-md-4 control-label">Password</label>
```

```
<div class="col-md-6">

<input id="password" type="password" class="form-control" name="password"
required>


@if ($errors->has('password'))

<span class="help-block">

<strong>{{ $errors->first('password') }}</strong>

</span>

@endif

</div>

</div>


<div class="form-group">

<label for="password-confirm" class="col-md-4 control-label">Confirm
Password</label>


<div class="col-md-6">

<input id="password-confirm" type="password" class="form-control"
name="password_confirmation" required>

</div>

</div>


<div class="form-group">
```

```
<div class="col-md-6 col-md-offset-4">

<button type="submit" class="btn btn-primary">

Register

</button>

</div>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

@endsection
```

3. Email.php:

```
@extends('layouts.app')


@section('content')

<div class="container">

<div class="row">

<div class="col-md-8 col-md-offset-2">

<div class="panel panel-default">

<div class="panel-heading">Reset Password</div>

<div class="panel-body">

@if (session('status'))

<div class="alert alert-success">

{{ session('status') }}

</div>

@endif


<form class="form-horizontal" role="form" method="POST" action="{{
route('password.email') }}">

{{ csrf_field() }}


<div class="form-group{{ $errors->has('email') ? ' has-error' : '' }}">

<label for="email" class="col-md-4 control-label">E-Mail Address</label>
```

```
<div class="col-md-6">

<input id="email" type="email" class="form-control" name="email" value="{{
old('email') }}" required>



@if ($errors->has('email'))

<span class="help-block">

<strong>{{ $errors->first('email') }}</strong>

</span>

@endif

</div>

</div>



<div class="form-group">

<div class="col-md-6 col-md-offset-4">

<button type="submit" class="btn btn-primary">

Send Password Reset Link

</button>

</div>

</div>

</form>

</div>

</div>

</div>
```

```
</div>

</div>

@endsection
```

4. Reset.php:

```
@extends('layouts.app')



@section('content')

<div class="container">

<div class="row">

<div class="col-md-8 col-md-offset-2">

<div class="panel panel-default">

<div class="panel-heading">Reset Password</div>



<div class="panel-body">

@if (session('status'))

<div class="alert alert-success">

{{ session('status') }}

</div>

@endif



<form class="form-horizontal" role="form" method="POST" action="{{
route('password.request') }}">
```

```
{{ csrf_field() }}

<input type="hidden" name="token" value="{{ $token }}">

<div class="form-group{{ $errors->has('email') ? ' has-error' : '' }}">
<label for="email" class="col-md-4 control-label">E-Mail Address</label>

<div class="col-md-6">
<input id="email" type="email" class="form-control" name="email" value="{{
$email or old('email') }}" required autofocus>

@if ($errors->has('email'))
<span class="help-block">
<strong>{{ $errors->first('email') }}</strong>
</span>
@endif
</div>
</div>

<div class="form-group{{ $errors->has('password') ? ' has-error' : '' }}">
<label for="password" class="col-md-4 control-label">Password</label>

<div class="col-md-6">
```

```
<input id="password" type="password" class="form-control" name="password"
required>


@if ($errors->has('password'))

<span class="help-block">

<strong>{{ $errors->first('password') }}</strong>

</span>

@endif

</div>

</div>


<div class="form-group{{ $errors->has('password_confirmation') ? ' has-error' : ''
}}">

<label for="password-confirm" class="col-md-4 control-label">Confirm
Password</label>

<div class="col-md-6">

<input id="password-confirm" type="password" class="form-control"
name="password_confirmation" required>


@if ($errors->has('password_confirmation'))

<span class="help-block">

<strong>{{ $errors->first('password_confirmation') }}</strong>

</span>

@endif
```

```
</div>

</div>



<div class="form-group">

<div class="col-md-6 col-md-offset-4">

<button type="submit" class="btn btn-primary">

Reset Password

</button>

</div>

</div>

</form>

</div>

</div>

</div>

</div>

</div>

@endsection
```

4.10.3 For Products :

1. Create.php:

```
@extends('layouts.header')

@section('content')

<div class="container">

<div class="row">

<div class="col-md-8 col-md-offset-2">
<div class="panel panel-default">
<div class="panel-heading">
Submit an Ad
</div>
<div class="panel-body">
<form class="form-horizontal" role="form" method="POST" action="{{
($add) ? route('add.update', [$add]) : route('add.store') }} "
enctype="multipart/form-data">
{{ csrf_field() }}

{{ ($add) ? method_field('PUT') : '' }}



<div class="form-group{{ $errors->has('title') ? ' has-error' : '' }}">
<label for="title" class="col-md-4 control-label">Ad. Title<em
class="required-asterick">*</em></label>

<div class="col-md-6">
<input id="title" type="text" class="form-control" name="title" }}
value="{{ old('title') ? old('title') : (($add) ? $add->title : '') }}" >

@if ($errors->has('title'))
<span class="help-block">
```

```
<strong>{{ $errors->first('title') }}</strong>
</span>
@endif
</div>
</div>

<div class="form-group{{ $errors->has('category') ? ' has-error' : '' }}">
<label for="category" class="col-md-4 control-label">Category<em
class="required-asterick">*</em></label>

<?php $category = old('category') ? old('category') : (($add) ? $add->category
: '') ?>
<div class="col-md-6">
<select id="category" class="form-control" name="category" {{ ($add) ?
'disabled' : '' }} required>
<option value=""> -- Select A Category -- </option>
<option {{ ($category == 'vehicles') ? 'selected' : '' }}
value="vehicles">Vehicles</option>
<option {{ ($category == 'property') ? 'selected' : '' }}
value="property">Property</option>
</select>

@if ($errors->has('category'))
<span class="help-block">
<strong>{{ $errors->first('category') }}</strong>
</span>
@endif
</div>
</div>

<div class="form-group{{ $errors->has('sub_category') ? ' has-error' : '' }}">
<label for="sub_category" class="col-md-4 control-label">Sub Category<em
class="required-asterick">*</em></label>
<?php $vehicles = config('myconfig.category.vehicles');
```

```
$properties = config('myconfig.category.property');
$subCategory = old('sub_category') ? old('sub_category') : (($add) ? $add-
>sub_category : '') ?>

<div class="col-md-6">
<select id="sub_category" class="form-control" name="sub_category" {{
($add) ? 'disabled' : '' }} required>
<option>-- Select a Sub Category --</option>
@foreach($vehicles as $vehicle)
<option {{ ($subCategory == $vehicle) ? 'selected' : '' }} value="{{ $vehicle
}}">{{ strtoupper($vehicle)}}</option>
@endforeach
</select>

@if ($errors->has('sub_category'))
<span class="help-block">
<strong>{{ $errors->first('sub_category') }}</strong>
</span>
@endif
</div>
</div>



<div class="form-group{{ $errors->has('description') ? ' has-error' : '' }}">
<label for="description" class="col-md-4 control-label">Ad. Description<em
class="required-asterick">*</em></label>

<div class="col-md-6">
<textarea id="description" class="form-control" name="description"
rows="10" required> {{old('description') ? old('description') : (($add) ? $add-
>description : '') }}</textarea>

@if ($errors->has('description'))
<span class="help-block">
```

```
<strong>{{ $errors->first('description') }}</strong>
</span>
@endif
</div>
</div>

<div class="form-group{{ $errors->has('duration') ? ' has-error' : '' }}">
<label for="duration" class="col-md-4 control-label">Rent Time Unit<em
class="required-asterick">*</em></label>

<?php $duration = old('duration') ? old('duration') : (($add) ? $add->duration :
''); ?>
<div class="col-md-6">
<select id="duration" class="form-control" name="duration">
<option {{ ($duration == '1day') ? 'selected' : '' }} value="1day"> 1 Day
</option>
<option {{ ($duration == '1month') ? 'selected' : '' }} value="1month"> 1
Month </option>
<option {{ ($duration == '1year') ? 'selected' : '' }} value="1year"> 1
Year</option>
</select>
@if ($errors->has('duration'))
<span class="help-block">
<strong>{{ $errors->first('duration') }}</strong>
</span>
@endif
</div>
</div>

<div class="form-group{{ $errors->has('rent_amount') ? ' has-error' : '' }}">
<label for="rent_amount" class="col-md-4 control-label">Rent Amount<em
class="required-asterick">*</em></label>

<div class="col-md-6">
```

```
<input id="rent_amount" type="number" class="form-control"
name="rent_amount"
value="{{ old('rent_amount') ? old('rent_amount') : (($add) ? $add-
>rent_amount : '') }}" min="0" required>

@if ($errors->has('rent_amount'))
<span class="help-block">
<strong>{{ $errors->first('rent_amount') }}</strong>
</span>
@endif
</div>
</div>

<div class="form-group{{ $errors->has('pic1') ? ' has-error' : '' }}">
<label for="pic1" class="col-md-4 control-label">Image 1<em
class="required-asterick">*</em></label>

<div class="col-md-6">
<input id="pic1" type="file" class="form-control" name="pic1"
accept="image/*" {{ (($add) ? $add->pic1 : 'required') }} >
<label>uploaded : {{ (($add) ? $add->pic1 : '') }}</label>
@if ($errors->has('pic1'))
<span class="help-block">
<strong>{{ $errors->first('pic1') }}</strong>
</span>
@endif
</div>
</div>



<div class="form-group{{ $errors->has('pic2') ? ' has-error' : '' }}">
<label for="pic2" class="col-md-4 control-label">Image 2</label>

<div class="col-md-6">
```

```
<input id="pic2" type="file" class="form-control" name="pic2"
accept="image/*" value="{{ old('pic2') ? old('pic2') : (($add) ? $add->pic2 : '')
}}">

@if ($errors->has('pic2'))
<span class="help-block">
<strong>{{ $errors->first('pic2') }}</strong>
</span>
@endif
</div>
</div>



<div class="form-group{{ $errors->has('pic3') ? ' has-error' : '' }}">
<label for="pic3" class="col-md-4 control-label">Image 3</label>

<div class="col-md-6">
<input id="pic3" type="file" class="form-control" name="pic3"
accept="image/*" value="{{ old('pic3') ? old('pic3') : (($add) ? $add->pic3 : '')
}}">

@if ($errors->has('pic3'))
<span class="help-block">
<strong>{{ $errors->first('pic3') }}</strong>
</span>
@endif
</div>
</div>

<div class="form-group{{ $errors->has('user_name') ? ' has-error' : '' }}">
<label for="user_name" class="col-md-4 control-label">User Name<em
class="required-asterick">*</em></label>

<div class="col-md-6">
```

```
<input id="user_name" type="text" class="form-control" name="user_name"
value="{{ old('user_name') ? old('user_name') : (($add) ? $add->user_name :
'') }}" required>

@if ($errors->has('user_name'))
<span class="help-block">
<strong>{{ $errors->first('user_name') }}</strong>
</span>
@endif
</div>
</div>



<div class="form-group{{ $errors->has('phone_no') ? ' has-error' : '' }}">
<label for="phone_no" class="col-md-4 control-label">Phone No.<em
class="required-asterick">*</em></label>

<div class="col-md-6">
<input id="phone_no" type="text" class="form-control" name="phone_no"
value="{{ old('phone_no') ? old('phone_no') : (($add) ? $add->phone_no : '')
}}" required>

@if ($errors->has('phone_no'))
<span class="help-block">
<strong>{{ $errors->first('phone_no') }}</strong>
</span>
@endif
</div>
</div>

<div class="form-group{{ $errors->has('email') ? ' has-error' : '' }}">
<label for="email" class="col-md-4 control-label">Email</label>

<div class="col-md-6">
```

```
<input id="email" type="email" class="form-control" name="email"
value="{{ old('email') ? old('email') : (($add) ? $add->email : '') }}">

@if ($errors->has('email'))
<span class="help-block">
<strong>{{ $errors->first('email') }}</strong>
</span>
@endif
</div>
</div>

<div class="form-group{{ $errors->has('city') ? ' has-error' : '' }}">
<label for="city" class="col-md-4 control-label">City<em class="required-
asterick">*</em></label>

<?php $cities = config('myconfig.cities');
$oldCity = old('city') ? old('city') : (($add) ? $add->city : '') ?>

<div class="col-md-6">
<select class="form-control" id="city" name="city">

<option value="">All Cities</option>
@foreach($cities as $city)
<option {{ ($oldCity == $city) ? 'selected' : '' }} value="{{ $city }}">{{
strtoupper($city)}}</option>
@endforeach
</select>

@if ($errors->has('city'))
<span class="help-block">
<strong>{{ $errors->first('city') }}</strong>
</span>
@endif
</div>
```

```
</div>

<div class="form-group">
<div class="col-md-8 col-md-offset-4">
<button type="submit" class="btn btn-primary"> {{ ($add) ? 'Update' :
'Submit' }} Ad.</button>
<a type="button" href="{{ route('home') }}" class="btn btn-default"> Cancel
</a>
</div>
</div>
</form>
</div>
</div>
</div>
</div>
</div>
@endsection
```

3. Add.php:

```
@extends('layouts.header')

@section('content')

<div class="container">

<div class="row">

<div class="col-md-8 col-md-offset-2">

<div class="panel panel-default">

<div class="panel-heading">

<h3>{{ $add->title }}</h3>

</div>

<div class="panel-body">

<p class="pull-left"><fa class="fa fa-lg fa-2x fa-map-marker"></fa>  {{$add-
>city}}, India

<p class="pull-right">Add added at <strong>{{ $add->created_at
}}</strong></p>

<div class="col-sm-12">

@if($add->pic2 && $add->pic3)

<div class="col-sm-4">

<a target="_blank" href="{{asset('uploads').'/'.$add->pic1}}">

<img class="img-responsive show-images" src="{{asset('uploads').'/'.$add-
>pic1}}">

</a>

</div>

<div class="col-sm-4">
```

```
<a target="_blank" href="{{asset('uploads').'/'.$add->pic2}}">
<img class="img-responsive show-images" src="{{asset('uploads').'/'.$add-
>pic2}}">
</a>
</div>
<div class="col-sm-4">
<a target="_blank" href="{{asset('uploads').'/'.$add->pic3}}">
<img class="img-responsive show-images" src="{{asset('uploads').'/'.$add-
>pic3}}">
</a>
</div>
@else
<div class="col-sm-6">
<a target="_blank" href="{{asset('uploads').'/'.$add->pic1}}">
<img class="img-responsive show-images" src="{{asset('uploads').'/'.$add-
>pic1}}">
</a>
</div>
<div class="col-sm-6">
<?php $pic = ($add->pic2) ? $add->pic2 : (($add->pic3) ? $add->pic3 : '')?>
@if($pic)
<a target="_blank" href="{{asset('uploads').'/'.$pic}}">
<img class="img-responsive show-images" src="{{asset('uploads').'/'.$pic}}">
</a>
```

```
@endif

</div>

@endif

</div><br><br>

<div class="col-sm-12 description-div">

<p>{{$add->description}}</p>

</div>

<div class="col-sm-12 description-div">

<p>You can Contact this Seller.</p>

<p><strong>User Name : {{$add->user_name}}</strong></p>

<p><strong>Phone No. : {{$add->phone_no}}</strong></p>

@if($add->email)

<p><strong>Email : {{$add->email}}</strong></p>

@endif

</div>

</div>

</div>

</div>

</div>

</div>

@endsection
```

4.11 Testing:

Testing is the critical element of any software quality assurance and represents the ultimate review of specification, design and code generation.

Software testing has a dual function; it is used to establish the presence of defects in program and it is used to help judge whether or not the program is usable in practice. Thus software testing is used for validation and verification, which ensure that software conforms to its specification and meets the need of the software customer.

Developer resorted Alpha testing, which usually comes in after the basic design of the program has been completed. The project scientist will look over the program and give suggestions and ideas to improve or to correct the design. They also report and give after a program has been created.

1. Testability:

Software Testability is simply how easily a computer program can be tasted. The check list that follows provides a set of characteristics that lead to testable software.

- ✓ Operability
- ✓ Observables
- ✓ Controllability
- ✓ Decomposability
- ✓ Simplicity
- ✓ Stability
- ✓ Understandability

Following are the attributes of the Good Test:

- ✓ A good test has a high probability of finding an error.
- ✓ A good test is not redundant.
- ✓ A good test should be “Best of Breed”.
- ✓ A good test would be neither too simple nor too complex.

2. Compartmentalization: In this step we divide the project into number of manageable activities and task like

- ✓ Selection Module
- ✓ System Admin data module
- ✓ Dept. Admin data module
- ✓ Store and assign rendered module
- ✓ Employee data module
- ✓ Task creation data module
- ✓ Task allocation and reply data module
- ✓ Insert category and pwd Authority module
- ✓ Testing Module
- ✓ Documentation Module

3. Interdependency: Interdependencies of each compartmentalized activity were then found out. Some tasks must occur in sequence while there are many tasks, which can occur in parallel.

4. Software Inspection: Analyze and check system representations such as the requirements document, design, diagrams and the program source code. They may be applied at all the stages of process.

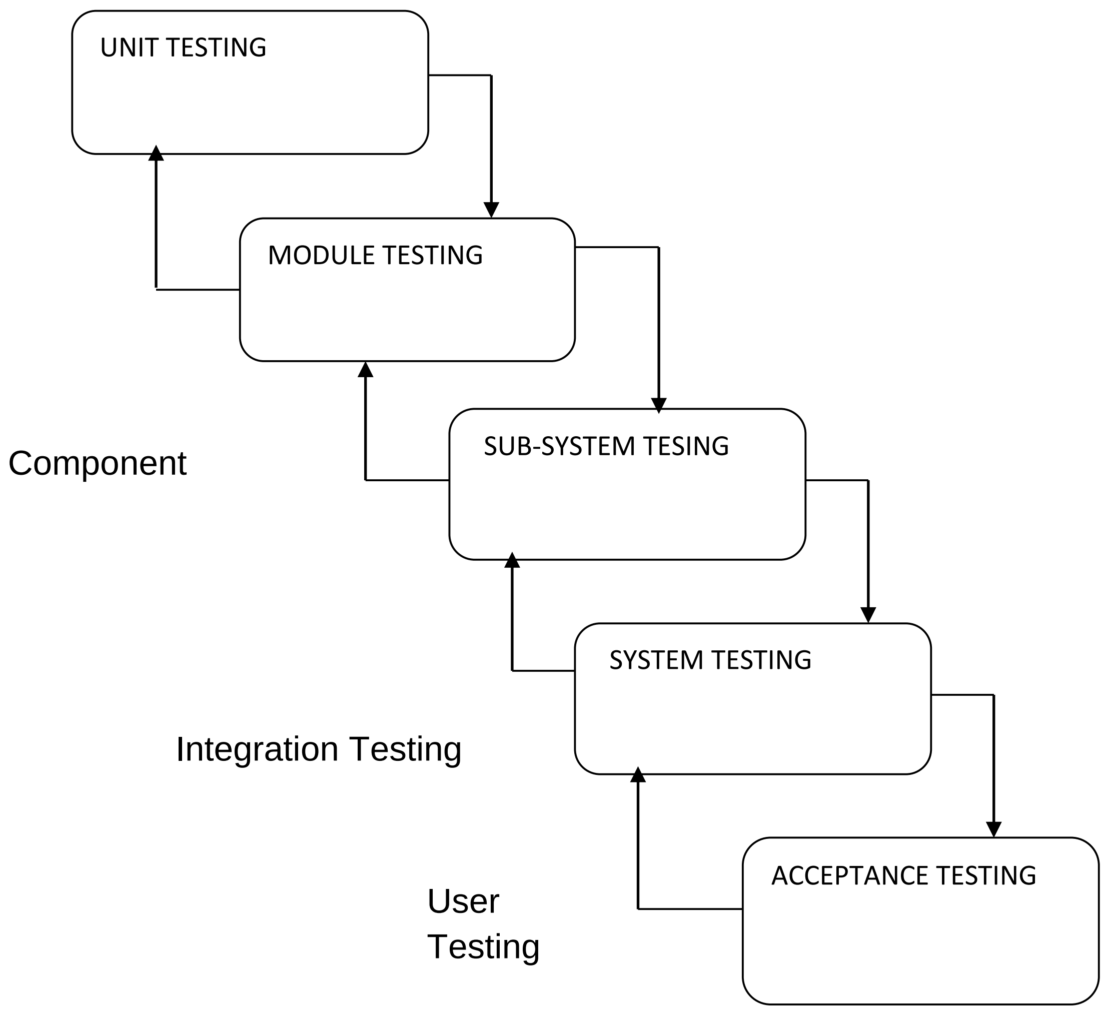


Figure 4.11: Testing Process

4.11.1 Test Plan:

The Testing Process:

Developer tests the software process activity such as design, implementation and the requirement engineering. Because, design errors are very costly to repair when the system has been started to operate. Therefore, it is quite obvious to repair them at early stage of the system. So, analysis is the most important process of any project.

Requirements Tractability:

As most interested portion is whether the system is meeting its requirements or not, for that testing should be planned so that all requirements are individually tested. Developer checked the output of certain combinations of input, which gives desirable results, or not. Strictly stick to our requirements specifications, give you the path to get desirable results from the system.

Tested Items:

Our tested items are like:

- Data fetching from the database
- Data insertion, updating and deleting in the database
- Form access to particular login

Testing Schedule:

Developer has tested each procedure back to back so that errors and omissions can be found as earliest as possible. Once the system has been developed by fully developer tested it on other machines, which differs in configuration.

4.11.2 Testing Methods:

Software testing involves executing an implementation of the software which tests data and examining the outputs of the software and its operational behavior to check that it is performing as required.

☞ **STATISTICAL TESTING:**

Statistical testing is used to test the program’s performance and reliability and to check how it works under operational conditions. Tests are designed to reflect the actual user inputs and their frequency.

The stages involved in the static analysis for this system are follows:

- Control Flow Analysis
    - Unreachable code
    - Unconditional branches into loops
- Data Use analysis
    - Variable used before initialization
    - Variables declared but never used
    - Variables assigned twice but never used between assignments
    - Possible array bound violations
    - Declared variables
- Interface Analysis
    - Parameter type mismatches
    - Parameter number mismatches
    - Non-usage of the results of function
    - Uncalled functions and procedures
- Storage Management Faults
    - Data not stored in proper tables
    - Data cannot be fetched from proper table

☞ **DEFECT TESTING:**

Defect testing is intended to find inconsistencies between a program and its specifications. These inconsistencies are usually due to the program faults or defects.

- **UNIT TESTING:**

  The Developer carries out unit testing in order to check if the particular module or unit of code is working fine. The unit testing comes at the very basic level as it is carried out as and when the unit of the code is developed or a particular functionality is built.

  In this application we test one most important module as task allocation which is as follows:

- **LOOP TESTING:**

  Tester has tested some condition in a code of application. So they test the looping in source code of application for finding miss route or any error or wrong direction of flow in code.

- **BLACK-BOX TESTING**:

In black box testing or functional testing, the developer is concerned about the output of the module and software, i.e. whether the software gives proper output as per the requirements or not. In another words, these testing aims to test a program behavior against specification without making any reference to the internal structure of the program or the algorithms used. Therefore, the source code is not needed, and so even purchased modules can be tested. The program just gets a certain input and its functionality is examined by observing the output.

This can be done in the following way:

- Input Interface
- Processing
- Output Interface

The tested program gets certain inputs. Then the program does its job and generates a certain output, which is collected by a second interface. This result is then compared to the excepted output, which has been determined before the test.

☞ **WHITEBOX TESTING**:

It is also called ‘GLASS BOX’ or ‘STRUCTURAL’ testing. Tester has access to the system design.

a) Simple Loops
b) Nested Loops
c) Concatenated Loops
d) Unstructured Loops
e) Continuous Loops

They can:

a) Examine the design document
b) View source code
c) Individual path examine.
d) Logical path examine one time
e) Logical decision on their true and false

☞ **STRUCTURE TESTING**:

Developer has done his path testing to exercise every independent execution path through a component or program. If every independent path is executed, then all statements in the components must have been executed at least once. The structure of our program is also cheeked.

☞ **INTEGRATION TESTING:**

After our individual modules Developer tested out Developer go to the integrated to create a complete system. This integration process involves building the system for problems that arise from component interactions. Developer has applied top-down strategy to validate high-level components of a system before design and implementations have been completed. Because, our development process started with high-level components and Developer worked down the component hierarchy.

☞ **PERFORMANCE TESTING**:

Performance testing is designed to test the runtime performance of the system within the context of the system. These tests Developer performed as module level as developer as system level. Individual modules developers tested for required performance.

☞ **CONDITION TESTING**:

Condition testing is a test case design method that exercises the logical conditions contained in a program module. If the condition is incorrect, then as least one part of the condition is incorrect. It may include:

- Boolean variable error
- String index out of Bound error
- Null pointer Assignment
- Input Output Connection Exceptions
- Arithmetic expression error
- Parsing (conversion) errors
- Image unloaded errors

☞ **INTERFACE TESTING:**

Interface testing is integral part of integration testing. Therefore, developer checked for the following.

- Interface misuse
- Interface misunderstanding

Developer examined the code to be tested and explicitly list each call to an external component. In the system, standard tests for various modules have been performed, which are follows. All the menu functions and sub menu functions have been checked.

- Validations for all inputs are done.
- All required fields are not left blank.

☞ **OBJECT TESTING**:

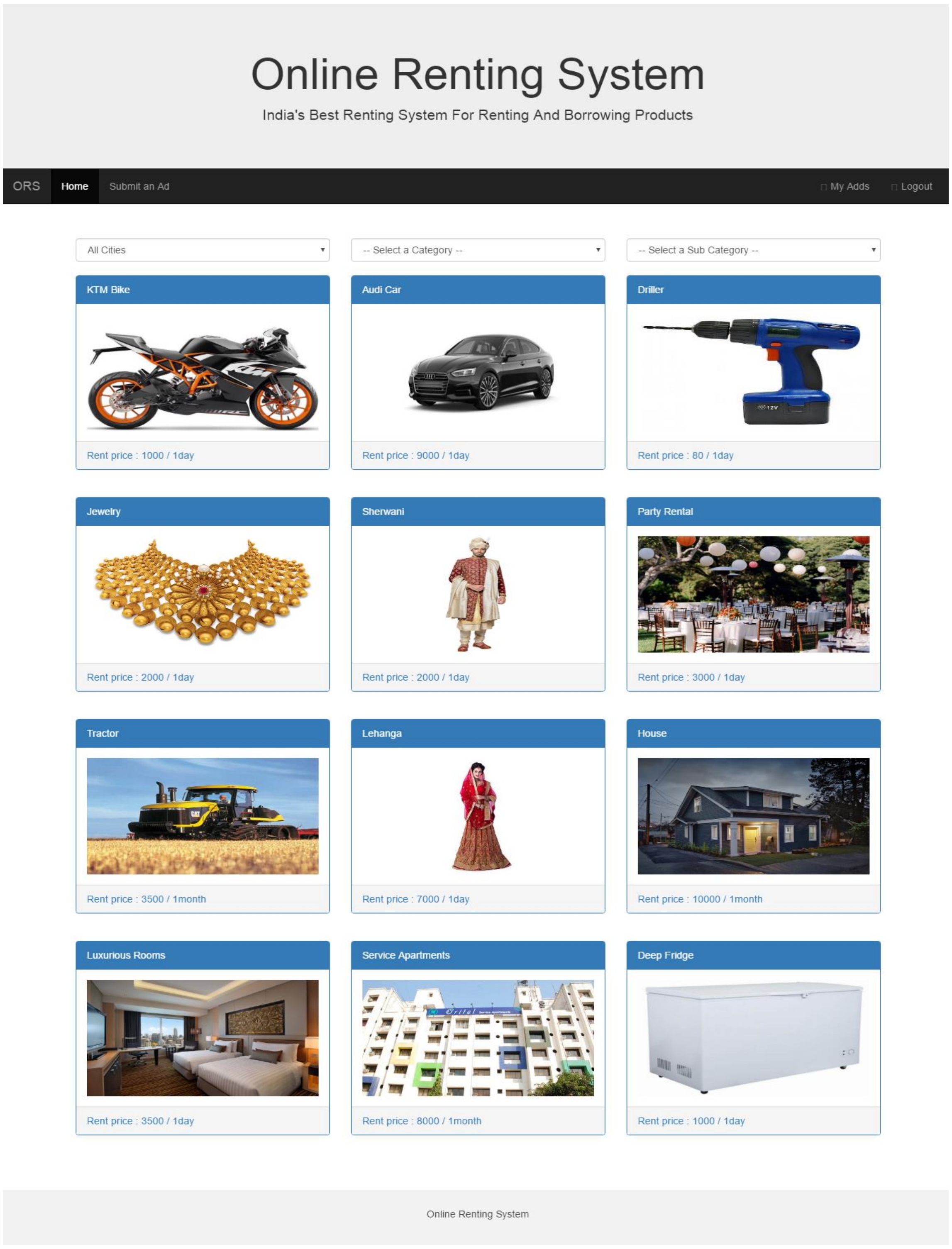
Object testing is to test object as individual components, which are often larger than single function. Here following activities have taken place:

- Testing the individual operations associated with object.
- Testing individual object classes.
- Testing cluster of objects.
- Testing object oriented systems.

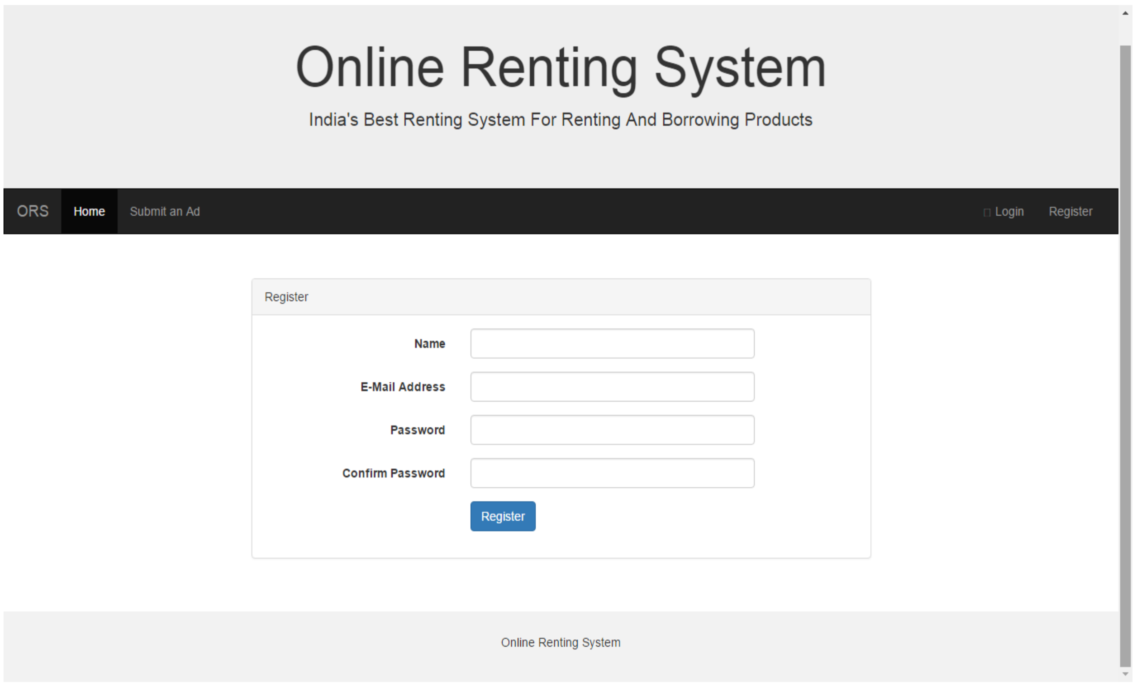
4.11.3 Test Cases:

Test cases Form name	Testing name	Description	Results	Remarks
User register	Unit testing	Username=xyz@yahoo.com Password=123goodpro	Login failed	Wrong username /password
New user	Unit testing	Enter password and confirm password different	Invalid email/password	Not allowed
Verification	Unit testing	Username= SRS123@gmail.com Password=Pass@word1.	Login successfully	Viewed Successfully.

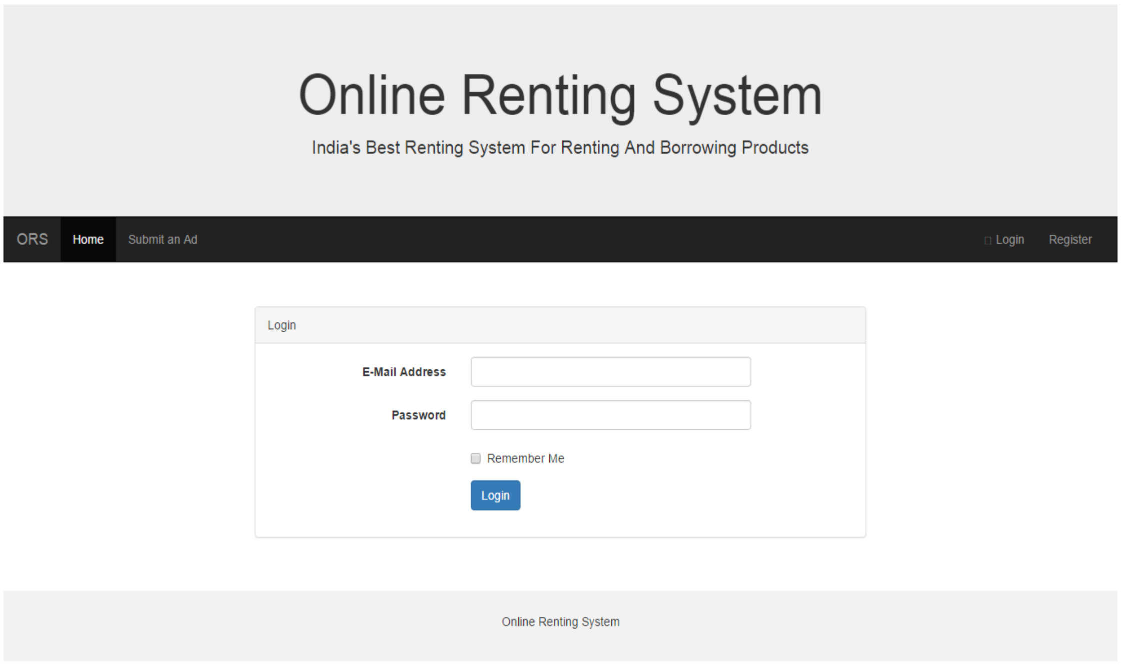
5. OUTPUT SCREENS:



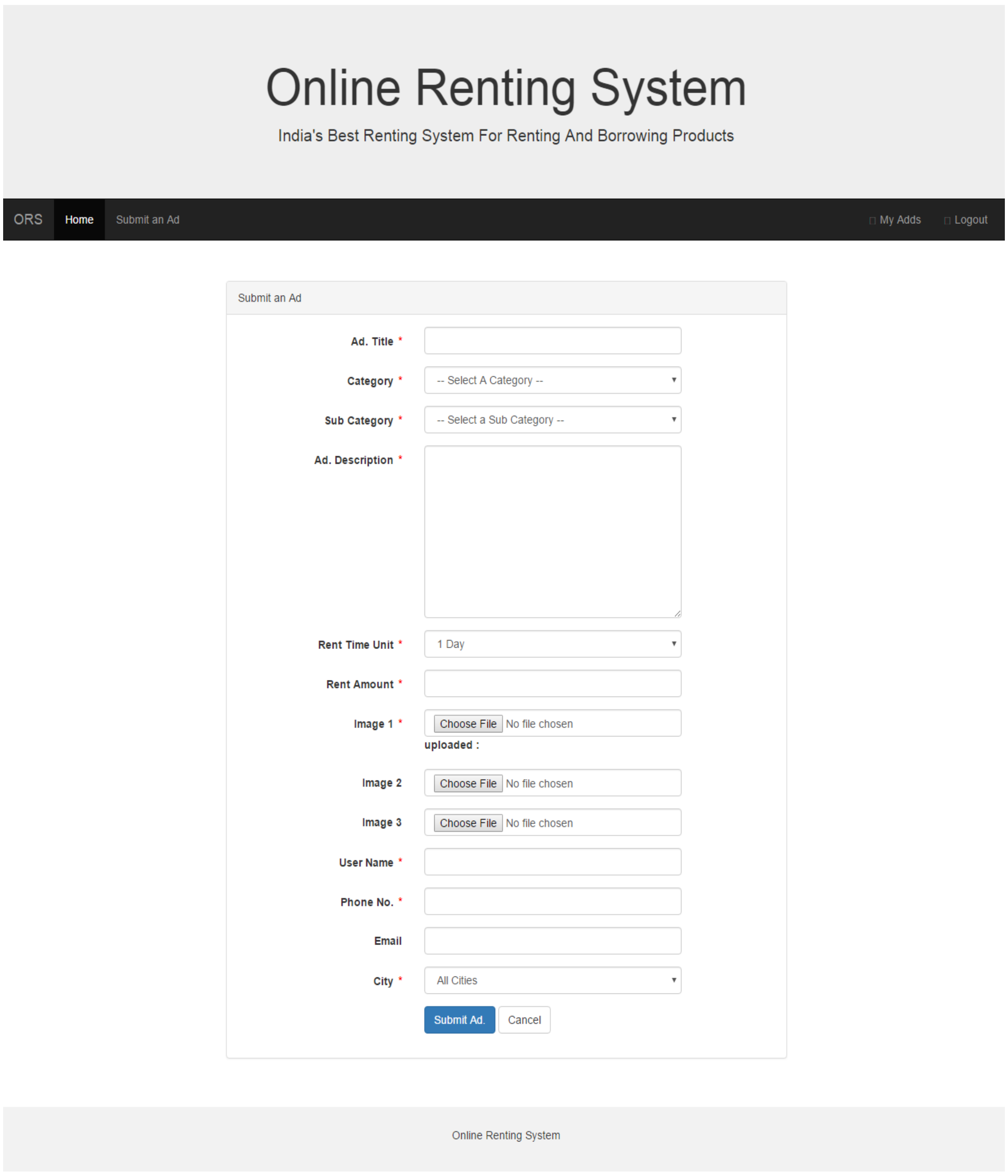
1. Home Page of ORS



2. Registration/Sign up Page of ORS



3. Login Page of ORS

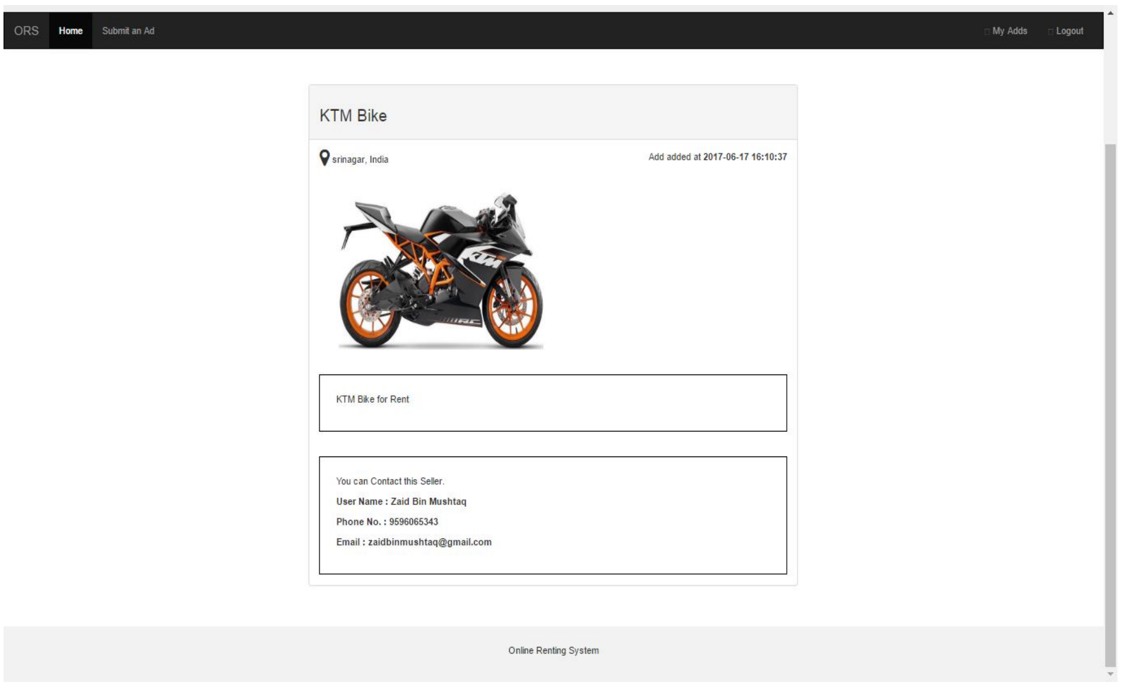

Online Renting System

India's Best Renting System For Renting And Borrowing Products

Submit an Ad

Ad. Title *	
Category *	-- Select A Category --
Sub Category *	-- Select a Sub Category --
Ad. Description *	
Rent Time Unit *	1 Day
Rent Amount *	
Image 1 *	Choose File No file chosen uploaded :
Image 2	Choose File No file chosen
Image 3	Choose File No file chosen
User Name *	
Phone No. *	
Email	
City *	All Cities

4. Submit Ad Page of ORS



KTM Bike

srinagar, India

Add added at **2017-06-17 16:10:37**

KTM Bike for Rent

You can Contact this Seller.

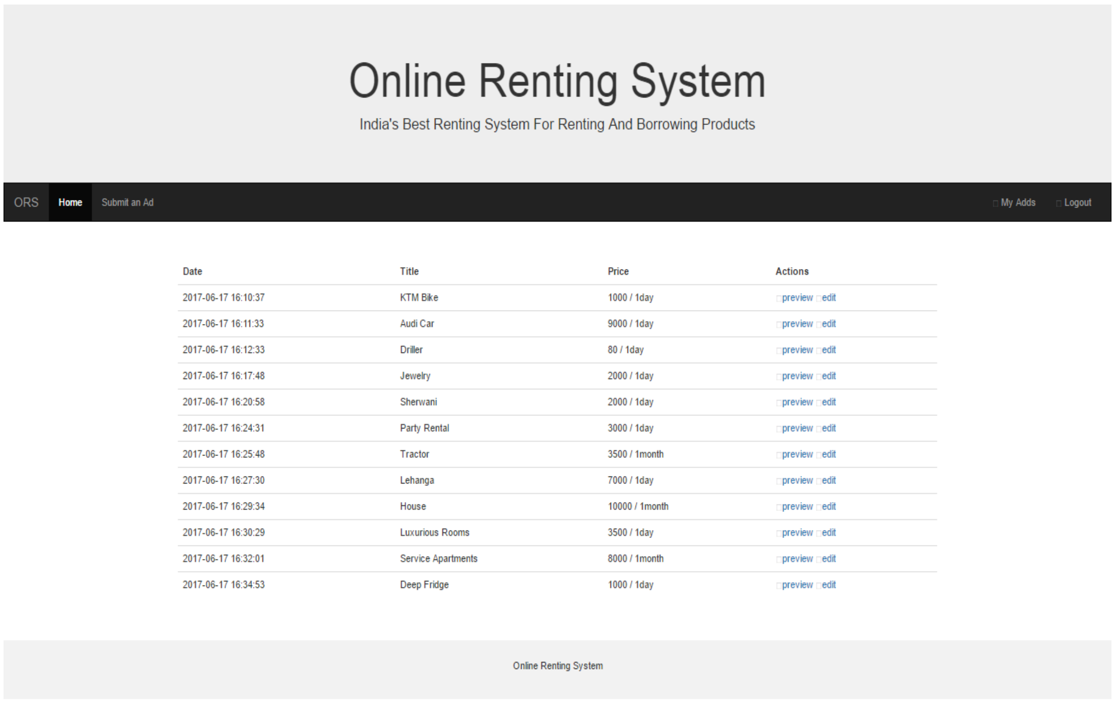
User Name : Zaid Bin Mushtaq

Phone No. : 9596065343

Email : zaidbinmushtaq@gmail.com

Online Renting System

5. View Ad Page of RTO (after clicking on the ad)



Online Renting System

India's Best Renting System For Renting And Borrowing Products

Date	Title	Price	Actions
2017-06-17 16:10:37	KTM Bike	1000 / 1day	preview edit
2017-06-17 16:11:33	Audi Car	9000 / 1day	preview edit
2017-06-17 16:12:33	Driller	80 / 1day	preview edit
2017-06-17 16:17:48	Jewelry	2000 / 1day	preview edit
2017-06-17 16:20:58	Sherwani	2000 / 1day	preview edit
2017-06-17 16:24:31	Party Rental	3000 / 1day	preview edit
2017-06-17 16:25:48	Tractor	3500 / 1month	preview edit
2017-06-17 16:27:30	Lehanga	7000 / 1day	preview edit
2017-06-17 16:29:34	House	10000 / 1month	preview edit
2017-06-17 16:30:29	Luxurious Rooms	3500 / 1day	preview edit
2017-06-17 16:32:01	Service Apartments	8000 / 1month	preview edit
2017-06-17 16:34:53	Deep Fridge	1000 / 1day	preview edit

Online Renting System

6. My Ads Page of ORS after Logging In

6. CONCLUSION:

The main purpose of Online Renting System is Vendors/users to advertise their available products in our web site and also to provide available products booking for the customers/users. Here in the system we are providing a great flexibility for the vendor/user to add, delete or update his products in our web site. They are able to update or delete their personal account. We are maintaining good communication between vendors and the customers. There are many rental systems are available in online. But, they are not providing all products at one place. Also many of them restricted to only one city. That means car rental system in online deals only with cars. Also many of them are not providing effective communication between customer and the vendor. Also present rental systems restricted to only one vendor means products are supplied only from one rental show room. Online Renting System is a one stop rental portal. It provides services such as Hiring Motor Vehicles, Service Apartments, Hotels, Guest Houses, Meeting & Conference Halls, Audio visuals, Party rentals, Computers and Other Products. It provides the facility to make online orders and get everything done before you reach the destination. This web application provides a platform for users and rental product(s) owners/venders, in an effective and efficient manner.

7. FUTURE SCOPE:

Every Edition of a book comes with new topics and modifications if any errors are present. In the similar way, in near future, our application will overcome the flaws if occurred, and attains new features offered to users for the Flexible and ease of access. In future we will add automated mobile alerts for customers and vendor's means if any user places an order then alerts will automatically go to the customer mobile. Also if any vendor signs up into our site then his user name and password sends to his mobile. Any product updates will also sends to his mobile. If any new discounts or offers also sends to vendors mobile.

Now we are maintaining vendor, user communication through email id, but in future we will implement it with mobile alerts by sending all customers booking and personal information automatically to vendor's mobile immediately. We also extend our website too many cities and also with much more cities. Also we will provide automated mail communication between vendor and administrator.

8. Bibliography:

1. http://www.makkan.com/

2. http://www.indianproperty.com/

3. http://www.w3schools.com/

4. Software Engineering by Roger Pressman