A PROJECT ON

"ONLINE BUS TICKET RESERVATION & MANAGEMENT SYSTEM"

BY USING JAVA

A Project submitted to Acharya Nagarjuna University, Guntur in the partial fulfillment for the award of degree of

BACHELOR OF COMPUTER APPLICATIONS

Submitted by

CHALLA ASHOK

Reg No: Y175144006

Under the esteemed guidance of

A. Sumanth

Associate Professor

G.R.K DEGREE AND PG COLLEGE

Lawyer pet, NH-5, Ongole, Andhra Pradesh-523001

2019-2020

CERTIFICATE

This is to certify that the Project titled "ONLINE BUS TICKET RESERVATION & MANAGEMENT SYSTEM" is a bonafide work carried out by CHALLA ASHOK, **Reg. no: Y175144006** in partial fulfillment of the requirement for the award of degree of Bachelor of Computer Applications.

PROGECT GUIDE

Sumanth

Associate Professor

HEAD OF THE DEPARTME

John Samuel

Associate Professor & HOD

EXTERNAL EXAMINER

DECLARATION

I hereby declare that the project entitled **"ONLINE BUS TICKET RESERVATION & MANAGEMENT SYSTEM"** submitted by me under the guidance of **A.Sumanth, Associate Professor**, Department of BCA, G.R.K Degree and PG college, Ongole in partial fulfillment of the award of the degree of Bachelor of Computer Applications, is the original work done by me and I have not submitted earlier in part or full to any other university for any other degree or diploma.

CHALLA ASHOK

Y175144006

ACKNOWLEDGEMENT

It is high privilege for me to express my deep sense of gratitude to those entire faculty Members who helped me in the completion of the project, specially my internal guide **Mr. Sumanth** who was always there at hour of need. My special thanks to all other faculty members, Batchmate & Seniors of G.R.K Degree and PG college for helping me in the completion of project work and its report submission.

CHALLA ASHOK

Y175144006

INDEX

ABSTRACT

- **Name of the Project: e-Ticketing**

- **Vision:**

  The Ultimate motto of the project addresses all the problems that were faced by the current traveling agencies. Coming to the feasibility aspect the current project focuses on

  1) Economical Feasibility
  2) Technical Feasibility
  3) Operational Feasibility

- **Deliverables/ Functional Specification:-**

  1. Current Administrators
  2. Current Agents
  3. Customers
  4. Details of the services.
  5. Timings, Source, Destination details.

Note:

The application should be secured with different levels of access control.

INITIAL INVESTIGATION

Project Request and Problem initialization

The first step in System Development Life Cycle (SDLC) is the identification of need. The user request identifies the need for change and authorizes the initial investigation. The objective is to determine whether the request is valid and feasible. The outcome of the initial investigation is the presentation of results called project proposal. An acceptance signature on the project proposal by the authorized person and its acceptance by the MIS department makes it a formal agreement to proceed with the detailed analysis and design of the candidate system.

PROBLEM DEFINITION:

This project is basically about the **"e-Ticketing"**.

The existing procedure of "e-Ticketing" was done locally respective to the travelling agency, looking up and checking the various details of different services in different places was a tedious and cumbersome process. It was even error prone and definitely not a pleasant task to perform.

There are many problems involved in the existing system and has the following limitations-

- Time consuming (All process was done involving manually).
- Lack of integration.
- Difficulty in data processing (Since for each time the customer has to manually involve).
- Changing the decision at the last moment is a tedious task.
- Difficulty in viewing the service details.

These were the chief reasons for the development of the project.

To alleviate the above lacuna and thus achieving better information retrieval, the organization has decided to introduce the use of **"e-Ticketing"**

FORMULATION OF SOLUTION STRATEGIES AND ANALYSIS OF ALTERNATE SOLUTIONS

Organizations today can be benefit from the availability of these alternatives and evaluate how they can best benefit from them in the short to long term.

Full treasury centralization is today more accessible than ever. The traditional centralizing structures are still the preferred options, but payment factories are becoming more critical as the integration layer between treasury and the rest of the organization. Furthermore, strategic outsourcing is lowering the investment and project risk barriers and can significantly reduce the execution time of a centralization initiative if not even leap-frog some of the intermediate phases.

While organizational centralization is a concept that is well understood, its practical application faces many challenges that often lead to a slow progression towards fully centralized management models. Transition can take different forms and can proceed at different speeds depending on the corporate organization. Individual business as well as form-wide initiatives, driven by cost efficiency, process integration or performance visibility, generate new centralization-fostering opportunities

The use of reference to centralization terminology requires some qualifications:

- Strategic coordination – the less intrusive form, relying on policies, procedures and guidelines centrally issued.
- Compliance control – based on a formal and strict compliance and reporting framework, which could extend to central approval for certain activities
- Mandated execution – involving the transfer of some value-adding activities to a central entity
- Functional consolidation – migration and reorganization of entire activities into a new infrastructure

Over the past decade functional centralization has experienced a strong acceleration thanks to rapid developments in application technology and communication protocols. By breaking down some of the barriers to effective exchange, access and circulation of data and information, the functional distribution of tasks and activities can be designed in a more flexible manner. Business applications can be deployed as single global installations and accessed remotely and securely. They can interface more easily with other systems and integrate a number of

independent or standalone processes. The combination of these features takes centralization to a new level of sophistication and at the same time makes it more accessible and appropriate to a broader number of organizations.

While organizational centralization is a concept that is well understood, its practical application faces many challenges that often lead to a slow progression towards fully centralized management models. Transition can take different forms and can proceed at different speeds, depending on the corporate organization.

"Centralization" is commonly associated with a number of strong benefits that range from pure cost savings to control improvement, full compliance with corporate policies, process standardization, increased productivity and expertise consolidation.

H**ardware and software Requirements**

User interface requirements:	Dreamweaver MX.
IDE	**:** Eclipse
Database requirements	**:** Oracle8.1
Server	**:** Tomcat 5.0
Preferred Technologies	**:** JavaScript, Java
	(Jdbc 2.0,Servlets2.1, JSP 1.2 ,Struts)

SOFTWARE DEVELOPMENT PROCESS MODEL

In this project we are using the "**Evolutionary Model**" which is also referred to as the successive versions model and sometimes as the Incremental model.

Comparison of different life cycle models

Although classical waterfall model is the basic model for all the other life cycle models but it cannot be used in practical development projects, since these models supports no mechanism to handle the errors during the phases. This problem is overcome in iterative waterfall model but it is not suitable for very large projects and for projects that are subject to many risks.

This model assumes that the requirements be completely specified before the next of the development activity can start, it cannot be satisfactorily used in projects where only rough requirements are available at the beginning of the project. This model creates blocking states in the system i.e., some team members would have to wait for a phase to be complete before they can start their next activity. This is clearly wastage of resources and such wastages are rarely tolerated in real projects.

In this life cycle model, the software is first broken down into several models (or) functional units, which can be incrementally constructed and delivered. The development team first develops the core modules of the system. This initial product skeleton is refined into increasing levels of capability by adding new functionalities in successive versions. Each evolutionary version may be developed using an iterative waterfall model of development.

Each successive version of the product is fully functioning software capable of performing more useful work than the previous versions. In this model, the user gets a chance to experiment with partially developed software much before the complete version of the system is released. Therefore, the evolutionary model helps to accurately elicit user requirements during the delivery of the different versions of the software, and the change requests therefore after delivery of the complete software are minimized. Also the core modules get tested thoroughly, thereby reducing chances of errors in the core modules of the final product. Further, this model obviates the need to commit large resources in one go for development of the system.

The main disadvantage of the successive versions model is that for most practical problems it is difficult to divide the problem into several functional units, which can be incrementally implemented and delivered. Therefore, the evolutionary model is normally useful for only very large products, where it is easier to find modules for incremental implementation. Often the evolutionary model is especially when the customer prefers to receive the product in increments to be able to start using the different features as and when they are developed rather than waiting for the full product to be developed and delivered.

Evolutionary model is very popular for the object-oriented software development projects, because the system can easily be partitioned into stand-alone units in terms of the objects.

ARCHITECTURAL PATTERN

Model-View-Controller

The Model Layer

The model layer in a Java based web application can be implemented using any Java-based technology, such as EJB, Hibernate, or JDO. In our CoreBanking System. The model is represented as simple JavaBeans containing the data and business logic in a simple data access object. As far as possible, the model objects should be developed so that they have no knowledge of the environment. This allows us to more easily reuse them across environments and applications.

The View Layer

The view layer of most Java based web applications is made up of JavaServer pages. To facilitate the development of the view, Java provides a set of JSP tag libraries. These tag libraries allow us to easily provide fully internationalised user interfaces that interact with the model components of a web application. The vast majority of dynamic Web front ends are based on HTML forms, and users of such applications have come to expect from these applications certain behaviours, such as form validation. With standard JSP, this is a tedious process that involves recording the contents of the form and populating every form element with information from a JavaBean in case of error. Java facilitates this sort of form processing and validation using Custom tags. These, in combination with the JSP tag libraries, make View development with forms really simple and natural.

The Controller Layer

Java includes a Servlet that implements the primary functions of the Controller, which is to map the incoming URL to a model object. The Servlet provides the following functions:

1. Decide what action is required to service a users request
2. Provide view data to the view
3. Decide which view to show next

A Java developer must provide these actions (models) to implement the logic of their application.

What is Model-View-Controller?

. Let's start by looking at how the Model, the View, and the Controller interact with one another:

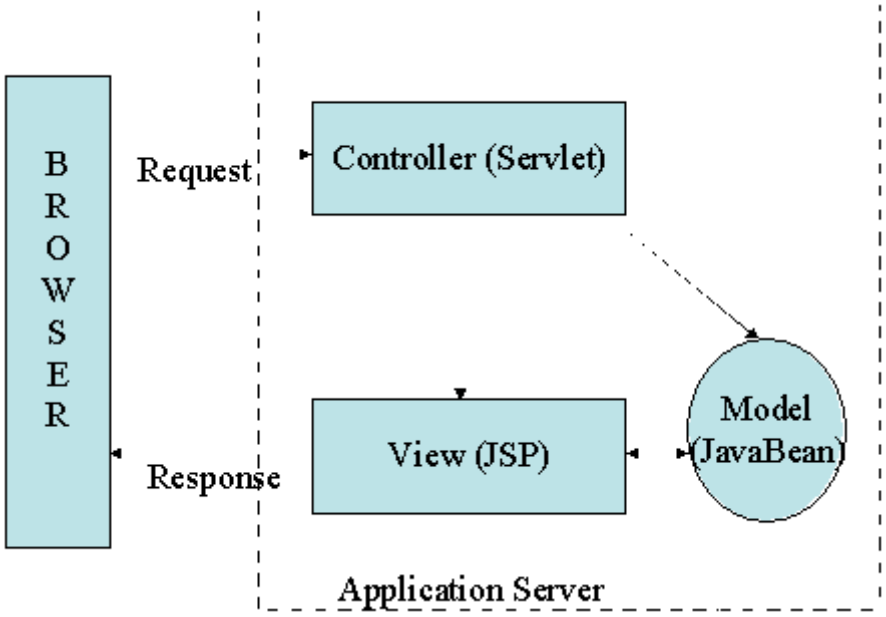

Figure1:Model

Figure2:MVCarchitecture

As you can see from the above diagram, the user interacts with the Controller components (usually represented by Servlets) by submitting requests to them. In turn, the Controller components instantiate Model components (usually represented by JavaBeans or other similar technology), and manipulate them according to the logic of the application. Once the Model is constructed, the Controller decides which View (usually represented by JavaServer Pages) to show to the user next, and this View interacts with the Model to show the relevant data to the user.

Using Java Server Pages or Servlets alone is known as Model 1. Model 2 was not particularly innovative or new; it uses Servlets to resemble Controller and Java ServerPages for resembling views.Many people realised that it follows the well-known MVC pattern that was developed back in the days of Smalltalk. As such, Java programmers tend to use the terms Model 2 and MVC interchangeably.

DESIGN PATTERN

Data Access Object

Context

Access to data varies depending on the source of the data. Access to persistent storage, such as to a database, varies greatly depending on the type of storage (relational databases, object-oriented databases, flat files, and so forth) and the vendor implementation

Problem

Applications can use the JDBC API to access data residing in a relational database management system (RDBMS). The JDBC API enables standard access and manipulation of data in persistent storage, such as a relational database. The JDBC API enables J2EE applications to use SQL statements, which are the standard means for accessing RDBMS tables. However, even within an RDBMS environment, the actual syntax and format of the SQL statements may vary depending on the particular database product.

There is even greater variation with different types of persistent storage. Access mechanisms, supported APIs, and features vary between different types of persistent stores such as RDBMS, object-oriented databases, flat files, and so forth.

Such disparate data sources offer challenges to the application and can potentially create a direct dependency between application code and data access code. When business components-entity beans, session beans, and even presentation components like servlets and helper objects for Java Server Pages (JSP) pages --need to access a data source, they can use the appropriate API to achieve connectivity and manipulate the data source. But including the connectivity and data access code within these components introduces a tight coupling between the components and the data source implementation. Such code dependencies in components make it difficult and tedious to migrate the application from one type of data source to another. When the data source changes, the components need to be changed to handle the new type of data source.

Forces

Portability of the components is directly affected when specific access mechanisms and APIs are included in the components. Components need to be transparent to the actual persistent store or

data source implementation to provide easy migration to different vendor products, different storage types, and different data source types.

Solution

Use a Data Access Object (DAO) to abstract and encapsulate all access to the data source. The DAO manages the connection with the data source to obtain and store data.

The DAO implements the access mechanism required to work with the data source. The data source could be a persistent store like an RDBMS, an external service like a B2B exchange, a repository like an LDAP database, or a business service accessed via CORBA Internet Inter-ORB Protocol (IIOP) or low-level sockets. The business component that relies on the DAO uses the simpler interface exposed by the DAO for its clients. The DAO completely hides the data source implementation details from its clients. Because the interface exposed by the DAO to clients does not change when the underlying data source implementation changes, this pattern allows the DAO to adapt to different storage schemes without affecting its clients or business components. Essentially, the DAO acts as an adapter between the component and the data source.

Participants and Responsibilities

- **Business Object**

  The Business Object represents the data client. It is the object that requires access to the data source to obtain and store data. A Business Object may be implemented as a session bean, entity bean, or some other Java object, in addition to a servlet or helper bean that accesses the data source.

- **Data Access Object**

  The DataAccessObject is the primary object of this pattern. The DataAccessObject abstracts the underlying data access implementation for the Business Object to enable transparent access to the data source. The Business Object also delegates data load and store operations to the DataAccessObject.

- **Transfer Object**

  This represents a Transfer Object used as a data carrier. The DataAccessObject may use a Transfer Object to return data to the client. The DataAccessObject may also receive the data from the client in a Transfer Object to update the data in the data source.

Consequences:

- **Enables Transparency**

  Business objects can use the data source without knowing the specific details of the data source's implementation. Access is transparent because the implementation details are hidden inside the DAO.

- **Enables Easier Migration**

  A layer of DAOs makes it easier for an application to migrate to a different database implementation. The business objects have no knowledge of the underlying data implementation. Thus, the migration involves changes only to the DAO layer. Further, if employing a factory strategy, it is possible to provide a concrete factory implementation for each underlying storage implementation. In this case, migrating to a different storage implementation means providing a new factory implementation to the application.

- **Reduces Code Complexity in Business Objects**

  Because the DAOs manage all the data access complexities, it simplifies the code in the business objects and other data clients that use the DAOs. All implementation-related code (such as SQL statements) is contained in the DAO and not in the business object. This improves code readability and development productivity.

Centralizes All Data Access into a Separate Layer

Because all data access operations are now delegated to the DAOs, the separate data access layer

can be viewed as the layer that can isolate the rest of the application from the data access implementation. This centralization makes the application easier to maintain and manage.

Scope of the Development Project

Database Tier: The concentration is applied by adopting the Oracle 8.1 Enterprise versions. SQL is taken as the standard query language. The overall business rules are designed by using the power of PL/SQL components like stored procedures stored functions and database triggers.

User Tier: The use interface is developed is a browser specific environment to have centralized architecture. The components are designed using Dreamweaver and Java server pages power the dynamic of the page design.

Data Base Connectivity Tier

The communication architecture is designed by concentrated on the standards of servlets and Java Beans. The database connectivity is established using the Java Database connectivity.

Purpose

The generated application is the first version upon the system. The overall system is planned to be in the formal of distributed architecture with homogeneous database platform. The major objective of the overall system is to keep the following components intact.

- ◈ System consistency
- ◈ System integrity
- ◈ Overall security of data
- ◈ Data reliability and Accuracy
- ◈ User friendly name both at administration and user levels
- ◈ Considering the fact of generality and clarity
- ◈ To cross check that the system overcomes the hurdles of the version specific standards

OUTLINE OF ANALYZED PROCESSES

ACTOR: ADMIN

- **Process:1** Registration of new Administrator

Input: Enter all the admin details like first name, last name, date of birth etc.

Process: registration (form)

Output: registration success/ registration fail

- **Process:2** View Pending Agents

Input: All the Pending Agents

Process: view Pending Agents (form)

Output: Accept/Reject

Process:3 View Permanent Agents

Input: All the Permanent Agents

Process: view all the permanent Agent Details

Output: success/fail

- **Process :4** Add a new Bus

Input: Enter Number, Type, Source, and Destination…

Process: Add the details in the database

Output: success/fail

- **Process :5** Add a new Bus Type

Input: BusType, Id…

Process: Insertion of new Type (Id.)

Output: success/fail

- **Process :6** Add Offer

Input: OfferName, Applicable for, time.

Process: adding Offer (form)

Output: success/fail

- **Process :7** Add New Trip Details

Input: Enter all the trip details like Tripid, Locationid, etc.

Process: New Trip (form)

Output: success / fail

- **Process:8** Change Halts

Input: Enter existing source, destination details of the Location

Process: change Halts (form)

Output: success/fail

- **Process:9** Send Messages To Notice Board

Input: Enter Description of the Topic, Applicable to…

Process: sendMessages(adform)

Output: success

- **Process:10** Send Mails

Input: Agents Id, Customer Id

Process: send mails (form)

Output: success

- **Process:11** Create new group

Input: Enter grid and name of the group

Process: insertGroupDetails (GroupDetailsForm gdf)

Output: success

- **Process:12** View all the Buses

Input: View All

Process: get all the details from Database Table

Output: success/fail

- **Process:13** View Offers

Input: View Offers, Time they will be elapsed.

Process: view Offers (table)

Output: success

- **Process:14** view Permanent Agents

Input: view all the existing Permanent Agents

Process: all the permanent Agents (table)

Output: success

- **Process:15** view messages

Input: view All the messages that r sent by Agents, Customers

Process: view Messages (table)

Output: success

- **Process:16** view Buses

Input: view all the existing Buses

Process: view Buses(table)

Output: success

- **Process:17** View TripDetails

Input: view All the TripDetails

Process: Get all the trip details like(Trip Id,Timings..)

Output: success

- **Process:18** view BusTypes

Input: view all the existing BusTypes

Process: View BusTypes(table)

Output: success

ACTOR: AGENT

- **Process:1** Registration

Input: Enter all the Agentdetails like first name, last name, date of birth etc.

Process: registration (form)

Output: registration success/ registration fail

- **Process:2** View Pending Agents

Input: All the Pending Agents

Process: view Pending Agents (form)

Output: Accept/Reject

- **Process:3** View Permanent Agents

Input: All the Permanent Agents

Process: view all the permanent AgentDetails

Output: success/fail

- **Process :4** Add a new Bus

Input: Enter Number, Type, Source, and Destination.

Process: Add the details in the database

Output: success/fail

- **Process :5** Add a new BusType

Input: BusType, Id,

Process: Insertion of new Type (Id.)

Output: success/fail

- **Process :6** AddOffer

Input: OfferName, Applicable for,time..

Process: adding Offer (form)

Output: success/fail

- **Process :7** Add New TripDetails

Input: Enter all the tripdetails like Tripid, Locationid, etc.

Process: NewTrip(form)

Output: success / fail

- **Process:8** Change Halts

Input: Enter existing source, destination details of the Location

Process: change Halts (form)

Output: success/fail

- **Process:9** Send Messages To NoticeBoard

Input: Enter Description of the Topic,Applicable to…

Process: sendMessages(adform)

Output: success

- **Process:10** SendMails

Input: AgentsId, CustomerId

Process: send mails(form)

Output: success

- **Process:11** Creat new group

Input: Enter gid and name of the group

Process: insertGroupDetails (GroupDetailsForm gdf)

Output: success

- **Process:12** View all the Buses

Input: View All

Process: get all the details from Database Table

Output: success/fail

ACTOR: CUSTOMER

- **Process:1** Registration of new Administrator

Input: Enter all the admindetails like first name, last name, date of birth etc.

Process: registration (form)

Output: registration success/ registration fail

- **Process:2** View Pending Agents

Input: All the Pending Agents

Process: view Pending Agents(form)

Output: Accept/Reject

- **Process:3** View Permanent Agents

Input: All the Permanent Agents

Process: view all the permanent Agent Details

Output: success/fail

- **Process :4** Add a new Bus

Input: Enter Number, Type, Source, and Destination.

Process: Add the details in the database

Output: success/fail

- **Process :5** Add a new Bus Type

Input: BusType, Id,

Process: Insertion of new Type (Id.)

Output: success/fail

- **Process :6** AddOffer

Input: OfferName, Applicable for, time.

Process: adding Offer (form)

Output: success/fail

- **Process :7** Add New TripDetails

Input: Enter all the trip details like Tripid, Locationid, etc.

Process: New Trip (form)

Output: success / fail

PROJECT SYNOPSIS

Technical Descriptions

- **Database:** The total number of databases that were identified to build the system is 14. The major parts of the databases are categorized as administration components and customer of based components. The administration components are useful is managing the actual master data that may be necessary to maintain the consistency of the system. These databases purely used for the internal organizational needs and necessities.

  The Administrator, Agent and Customer components are designed to handle to transactional states that arise upon the system whereas customer makes a visit onto the portal for making his transactions faster. The Customer components are scheduled accept parametrical information from the users as per the system necessity.

- **GUI:**

  In the flexibility of the users the interface has been developed a graphics concept in mind, associated through a browsers interface. The GUI'S at the top level have been categorized as

  - ✓ Administration users interface
  - ✓ Agents interface
  - ✓ Customer users interface

  The Administration users interface concentrate on the consistent in that is practically part of organizational actuaries and which needs proper authentication for data collation.
  The Administrator and Agent user interface helps the respective actors in transacting with the actual information as per their necessities with specific to the required services. The GUI's restrict the ordinary users from mismanipulating the systems data, which can make the existing system non-operational. The information with specific to their personal standards and strategies can be changed through proper privileges.

Modules:

1) **Administrator Module**: This module maintains the services related to system administrator who is authenticated upon the system. This module fairly maintains the integration between the modules related to backend database and the functionalities carried out in the whole organization. This module also binds itself with the agent and customer details.
2) **Agent Module**: This module maintains the information related to the customers who have been signed upon to the system as well as the internal information of the organization. The module integrates itself with the other modules like the Administrator module and customer module that are provided by the organization. This module acts as a major integrator with Admin transactions and the requests for approvals that are raised by the customer.
3) **Customer Module**: This module manages and keeps track of the details of the existing services. It has interaction to Agent as well as administrator to keep track of the consistency of information form time to time as they are executed.

1. Actor: Admin

The Admin module consists of the following services:

- Register another administrator.
- View pending agents and Accept or Reject them
- View permanent agents
- Add a new Bus.
- Add New Bus Type
- Add Offer
- Add New Trip Details
- Change Halts
- Send Messages to notice Board
- Send Mails
- View All the Busses
- View All Administrators

- View Offers
- View Permanent Agents
- View Messages
- View Busses
- View Trip details
- View Bus Types

Register another administrator:

In this process, Admin submits the details of another administrator with whom he want to share his responsibilities. The person whom admin appoints as an administrator will have the privilege to do all the responsibilities that are performed by actual Admin only if he is properly authenticated after login.

- **View pending agents and Accept or Reject them:**
  In this process, he can view the pending agents, he may accept the agents or reject. The agents who r accepted will be treated as Permanent agents Those who are rejected, their details will not be updated in the database.

- **ViewPermanentAgents:**
  In this process he can view all the permanent agents along with their complete details. And the details of new Agents which he made from pending to permanent agents.

- **Add New Bus :**
  In this process, if the existing services are not able to meet the requirements of customers, he may add new types to meet the requirements of customer.

- **Add New Bus Type:**
  In this process, if the existing services are not able to meet the requirements of customers, he may add new types to meet the requirements of customer.

- **Add Offer:**
  In this process, a strategy of new offers will be declared by admin to attract customers, and increase his business and thereby withstanding in the competition.

- **Add New Trip Details:**

  In this process, new trip details will be declared by admin as per the agent & customer requirements. By modifying the trip details according to customer and agent requirements they may feel convenient.

- **Change Halts:**

  In this process, admin can change the halts of his buses according to the Requirement of customers and also the agents keeping in view of appropriate halts and timings.

- **Send Messages to notice Board:**

  In this process, admin can display the messages about the bus services, timings, charges, offers, trips&halts, also the details of performance appraisal of his employees to motivate them.

- **Send Mails:**

  In this process, admin can send the messages about the queries that were posted by both agent and customer. A proper feedback must be there for every organization to withstand the competition and to be interactive with customers.

- **View All Administrators:**

  In this process Admin can view all the administrators that are appointed by him for responsibility division.and he can all the details of them completely in this module.

- **View Offers:**

  In this process Admin can view the offers he provided .Because he should delete the offers as and when the time of the particular offer has been elapsed.

- **View Permanent Agents:**

  In this process Admin can view the permanent agents that r under his guidance.By viewing this module he can have the complete idea that who are the new agents added as permanent and also their details.

- **View Messages:**

  In this process Admin can view all the messages that are sent by agents for enhancements and has the privilege to implement the enhancements if the requirements are really

needed. Can view all the messages that are sent by agents for enhancements and has the privilege to implement the enhancements if the requirements are really needed.

- **View Buses:**
  In this process, admin can view the details of the services and their appropriate timings and their halts and also the type of services and all other desired details.

- **ViewTripDetails:**
  In this process, admin can view the details of the services and their appropriate timings and their halts and also the type of services and all other desired details.

- **ViewBusTypes:**
  Instead of sending messages to each and every account with this service Admin can display the message into the notice board and is accessed by every person.

- **Logout:**
  Whenever the Admin wants to quit the application he needs to use this service so that the session will be invalidated so that no one can access his account thus restricting others in accessing the Admin's account.

2. Actor: Agent

- Registration for new License
- Registration
- Add Offer
- View Offers
- Send Messages
- View Messages
- Book Ticket
- Block Ticket
- Add Customer
- View Customers
- Logout

- **Request for new license:**

  In this process, new bus station that is agent want to establish should take a license from the Central agency i.e. Head Office .In the license certificate establishment details. Location, date details will be there.

- **Registration**

  In this process, agent can change his password by submitting the specified fields like agent id, old password, new password, retype new password. If he enters the correct values then his password and he will get new password.

- **Add Offer**

  In this process, offers will be announced by agent office to implement business strategies such that it will improve the throughput and withstand in the competitive environment.

- **View Offers:**

  In this process, the existing offers include and also to delete the offers if the specified offer time has been elapsed.

- **Send Messages:**

  In this process, the total offers include new & old will be maintaining at the agent office. If the correspondent agent office has some permanent customer it can send messages to them.

- **View Messages:**

  In this process, every agent maintain list of messages to which they have sent, the messages how the people (customers r interactive with both agents and Administrators.

- **Book Ticket:**

  In this process, agent can book the tickets as the requirement of the customer approached by verifying the availability of trips, timings, availability seats….

- **Block Ticket**

  In this process, agent office maintains details of customers if any one wants to cancel the tickets immediately they can block the tickets by assigning to others who r ready that trip.

- **Add Customers:**

  In this process, if any customer wants to register with the agent he has the privilege to add the customer and make the services available to this new Customer.

- **View Customers:**

  In this process, every agent office maintains data about the customers, and their complete details regarding the journey along with his journey details. And also the services he was using.

- **Logout:**

  Whenever the Agent wants to quit the application he needs to use this service so that the session will be invalidated so that no one can access his account thus restricting others in accessing the Agent's account.

3. Actor: Customer

- Register
- View Offers
- View Messages
- Send Messages to Agent
- Send Messages to Administrator
- View Bus Services
- View Bus Trip details like timings …
- Send Request to agent for booking a ticket
- Logout

- **Register :**

  In this process the Customer who wants the intended services provided by the agency. He should submit the details required, and if they r valid then only he will be given with one unique ID, Password which he should submit while login phase. If he forgets the password or Id he will be provided with an option to regain its Uid And password, but he should submit some details correctly with the data which he was submitted during registration phase.

- **View Offers:**
  In this process, customer see offers according to that he can register to view the offers. If he is eligible for that offer i.e. time is not elapsed he may bargain that offer.

- **View Messages:**
  In this process, customer sees all messages that are given by Customers, Agents, and Administrators and may get the required information.

- **Send Message to Agent:**
  In this process, Customer Registration modifications (i.e. ticket cancellation, buy new tickets, dates postponement) occur. If the customer has any problem-sending message to agent can solve him.

- **Send Message to Administrators:**
  If the agents could not solve problems of the customers, meesage are sent to the administrators to solve those problems. Such they may get the accurate information from the adminstrator or send their valuable suggestions to implement by the Organization.

- **View Bus Services:**
  This process helps to get overall information about bus services i.e. Bus timings, routes such that he may get the services for his desired timings And may travel accordingly.

- **View Bus Trip details like timings …**
  This process helps to give overall information about bus services And No. of Trips, Bus timings, routes, availability at his desired timings.

- **Send Request to agent for a ticket booking:**
  In this process customer can directly reserve the tickets in nearer agents or from far place. by submitting the source & destination details along with the fare details i.e. the way of amount to be paid.

- **Logout:**
  Whenever the Customer wants to quit the application he needs to use. This service so that the session will be invalidated so that no one can access his Account.

IMPLEMENTATION

Program Design Language

The program design language is also called as structured English or pseudopodia. PDL is a generic reference for a design language PDL looks like a modern language. The difference between PDL and real programming language lies in the narrative text embedded directly within PDL statements.

The characteristics required by a design language are:

- A fixed system of keywords that provide for all structured constructs date declaration and modularity characteristics.
- A free syntax of natural language that describes processing features.
- Subprogram definition and calling techniques that support various nodes of interface description.

PDL syntax should include constructs for subprogram definition, interface description date declaration techniques for structuring, conditions constructs, repetition constructs and I/O constructs.

PDL can be extended to include keywords for multitasking and/or concurrent processing interrupt handling, interposes synchronization the application design for which PDL is to be used should dictate the final form for the design language.

Testing Objectives:

The main objective of testing is to uncover a host of errors, systematically and with minimum effort and time. Stating formally, we can say,

- Testing is a process of executing a program with the intent of finding an error.
- A successful test is one that uncovers an as yet undiscovered error.
- The tests are inadequate to detect possibly present errors.
- The software more or less confirms to the quality and reliable standards.

Unit Testing :

- The purpose of the coding and unit testing phase of software development is to translate the software design into source code.Each component of the design is implemented as a program module.The end-product of this phase is a set of program modules that have been individually tested.To enable the engineers to write good quality programs, every software development organization normally formulates its own coding standards that suits itself.A coding standard addresses issues such as the standard ways of laying out the program codes, the template for laying out the function and module headers, commenting guidelines, variable and function naming conventions, the maximum number of source lines permitted in each module, and so forth.

- During this phase, each module is unit tested to determine the correct working of all the individual modules.It involves testing each module in isolation as this is the most efficient way to debug the errors identified at this stage.Another reason behind testing a module in isolation is that the other modules, with which this module has to be interfaced, may not be ready.

Integration and System Testing

Integration of different modules is undertaken once they have been coded and uni tested.During the integration and syste3m testing phase, the modules are integrated in a planned manner.The different modules making up a software product are almost never integrated in one shot.Integration is normally carried out incrementally over a number of steps.During each integration step, the partially integrated system is tested and a set of previously planned modules are added to it.Finally, when all the modules have been successfully integrated and tested, system testing is carried out. The goal of system testing is to ensure that the developed system conforms to its requirements laid out in the SRS document.

Our project is integrated and tested by using an activity by name ∝- testing. ∝- testing is the system testing performed by the development team.

MAINTENANCE

Maintenance is any work done to change the system after it is in operational. The term maintenance is used to describe activities that occur following the delivery of the product to the customer. The maintenance phase of the software life cycle is the time period in which a software product performs useful work.

Maintenance activities involve making enhancements to products, adapting products to new environments, correcting problems.

In this be retrieve the data from the database design by searching the database. So, for maintaining data our project has a backup facility so that there is an additional copy of data, which needs to be maintained.

More over our project would update the annual data on to a CD, which could be used for later reference.

CONCLUSION

- **WORK DONE:**

The **"e-Ticketing"** was successfully designed and is tested for accuracy and quality.

During this project we have accomplished all the objectives and this project meets the needs of the organization .The developed will be used in searching, retrieving and generating information for the concerned requests.

- **GOALS**
  - ✓ Reduced entry work.
  - ✓ Easy retrieval of information
  - ✓ Reduced errors due to human intervention
  - ✓ User friendly screens to enter the data
  - ✓ Portable and flexible for further enhancement
  - ✓ Web enabled.
  - ✓ Fast finding of information requested

BIBLIOGRAPHY

Reference Books:

1. **The Complete Reference**

   ----------Patrik Naughton, Herbert Schildt

2. **Java Servlet Programming**

   ----------Orielly

3. **Html Black Book**

   ----------Steven Hozner

4. **The Programming Language**

   ----------Ivan Bayross

5. **Software Engineering**

   ----------James

Websites:

1. **http://www.java.sun.com**
2. **http://www.sunsoft.com**
3. **http://www.javasoft.com**
4. **http://www.apress.com**
5. **http://www.oracle.com**
6. **http://www.jspin.com**