

TABLE OF CONTENTS

1. Introduction.....	3
1.1 Introduction	3
1.2 Objectives.....	3
2. Software Requirement Specification.....	4
2.1 Purpose and Scope.....	
2.2 Functional Requirements.....	4
2.3 Non Functional Requirements.....	4
2.4 Diagrams.....	
2.4.1 Usecase Diagram.....	
2.4.2 Dataflow Diagram	
2.4.3 Sequential Diagram	
3. Software Designing Document.....	13
3.1 High Level Design.....	13
3.2 Detailed Design Document.....	14
3.2.1 Architecture and Component Level Design.....	3.3
3.2.1.1 Architecture Overview	
3.2.1.2.1 Class Diagram '	
3.2.2 Data Architecture.....	
3.2.2.1 Entity-Relation Diagram.....	
3.2.2 Quality assurance.....	
3.2.3 Graphical User Interface.....	
4. Implementation and Results.....	20
4.1 Detailed Test plan for auto_driver.....	21
4.2 Detailed Test plan for passenger.....	20
5. Software Test Documentation.....	21
5.1 Home	21
6. Summary and Conclusion.....	
7. References.....	

8. Appendices.....

1. Introduction

2. Software Requirement Specification

2.1 Purpose and Scope

Purpose :

- The main purpose of this project is to provide customer friendly travelling system.
- Nowadays in many villages people are facing the problems with their journey(like waiting for the vehicles).
- So, we are providing solution for it by implementing Auto Reservation System.
- The Auto Reservation System is a system which is used by the auto drivers and the passengers to make travelling easy.
- The system provides the autos information to the passengers and passengers information to the auto drivers.
- The system will be updated all the time so that the passengers can get the updated information all the time.

Scope :

- The main intension of creating this is to provide reservations for autos by making an application for reservation.
- The application will be access via a internet on a PC at any place.
- It can be used by educated as well as uneducated people.
- As we know India is a country with 70% people lives in villages, we can circulate our application throughout the country.
- And we know that there is no application for booking autos in India except metropolitan cities. So, it is better to implement this in villages as well as smaller cities.

- In our application ,we will provide some features like live status, availability check ,advance booking etc.
 - By using live status passengers can identify the current location.
 - Availability check is used to check whether autos are present or not in particular location and at that particular time.
 - In advance booking people can book their autos in advance so that they can easily avoid the waiting time.
-
- In villages ,they need more transport system because they don't have RTC services and train services there. So, they can easily use this service.
 - We'll provide fare details too, So, no need of arguing or barging with drivers, passengers can easily estimate their journey cost.
- By this we can say that simply we are providing a interface between autos and passengers so that it has a wide range scope.

2.2 Functional Requirements

Registration :

Auto driver has to register to provide his service.

Use case name	Registration
Actors	Auto driver & Admin
Brief Discription	This use case is to register the auto driver.
Goals	To create an account to driver and to store the details of the driver into the database.
Triggers	Driver clicks the register button.
Pre-Condition	Driver should be in register page.
Post-Condition	After verifying account will be created and stored.
Basic Flow	Driver clicks on the register button and enters required details.
Alternate Flow	In the login page an option for register will be provided for the driver.
Exception	Exception will handled if he enters wrong details.
Quality	Completion time will depend on the internet speed.

Login :

To know the passengers that auto is available if the driver logged in to his account.

Use case name	Login
Actor	Driver & Admin
Brief Discription	This use case allows the driver to his account.
Goals	To log into particular account and to provide service.
Triggers	Driver clicks the Login option.
Pre-Condition	Driver should be in login page.
Post-Condition	
Basic Flow	Driver clicks the login button by submitting the username and password.
Alternate Flow	No alternative flow.
Exception	Exception is handled if the driver enters incorrect details.
Quality	Login time is very less.

Live status :

To check the current location of auto.

Use case name	Live status
Actor	Passenger
Brief Discription	This use case allows the passenger to see about the current location.
Goals	To estimate the journey time.
Triggers	Driver clicks the live status option.
Pre-Condition	Driver should have registration id.
Post-Condition	Passenger can get the expected journey time.
Basic Flow	Driver clicks the live status button by submitting the registered id.
Alternate Flow	No alternative flow.
Exception	Exception is handled if the driver enters incorrect details.
Quality	Login time is very less.

Verify :

To verify the driver whether he is registered or not.

Use case name	Verify
Actor	Driver & Admin
Brief Discription	This is used to check the driver information in database.
Goals	To keep track of all driver.
Triggers	When the driver clicks on the submit button.
Pre-Condition	Driver should be in login page.
Post-Condition	Validation will be done.
Basic Flow	Driver clicks the login button by submitting the username and password.
Alternate Flow	No alternative flow.
Exception	Exception is handled if the driver enters incorrect details.
Quality	Login time is very less.

Feedback :

The passenger can give the feedback about auto drivers and about their journey.

Use case name	Feedback
Actor	Passenger & Driver
Brief Discription	Passenger can give the feedback on driver and driver can see that feedback.
Goals	To get the feedback on driver.
Triggers	Passenger clicks on feedback button.
Pre-Condition	Passenger had to driven in that auto.
Post-Condition	
Alternate Flow	None.
Exception	If passenger didn't travelled in that auto then exception can be handled by the admin.
Quality	Depending upon the internet speed.

Complaints :

If passengers loose their belongings(mobile,bags,etc) they can give complaint.

Use case name	Complaints
Actor	Passenger & Admin
Brief Discription	Passenger gives the complaint on driver or if passenger lose anything then passenger can give the complaint.
Goals	To better service and help in finding the belongings of passengers.
Triggers	By clicking on complaints button .
Pre-Condition	Need to travel before filing the complaint.
Post-Condition	Complaint ID can be sent to passenger
Alternate Flow	None.
Basic flow	Filing complaint and getting solution
Exception	Without travelling on auto, if someone tries to give complaint then admin can reject that.
Quality	Based on the complaint

Availability check :

The passenger can check available autos.

Use case name	Availability check
Actor	Passenger & Admin
Brief Discription	Passengers can check the availability of autos between source and destination locations.
Goals	To show the available autos to passengers
Triggers	Passenger clicks on Availability check button.
Pre-Condition	Passenger have to give the source and destination details
Post-Condition	Application have to show the available autos after entering source and destination details.
Alternate Flow	None.
Basic flow	After selection of source and destination it will show the available autos.
Exception	Without entering any of one in source and destination error may occur and it can be handled by admin.
Quality	Based on availability and internet speed.

Status update :

If some passenger books an auto then automatically the status of that auto will be updated as busy.

Use case name	Status update
Actor	Passenger&Admin
Brief Discription	To modify the status of auto either busy or available.
Goals	To check the status of auto.
Pre-Condition	Driver should be in the application.
Post-Condition	Get the details of available autos.
Alternate Flow	None.
Exception	None.
Quality	Depending upon the net speed.

Book auto :

The passenger can book an auto.

Use case name	Book auto
Actor	Passenger & Admin
Brief Discription	Passenger can book auto by checking it's availability.
Goals	To book auto.
Triggers	By clicking on book auto button
Pre-Condition	Passenger need to check wether auto is available or not.
Post-Condition	Booking information will be sent to auto driver and passenger.
Alternate Flow	None.
Basic flow	Passenger can book auto and travel.
Quality	Based on the internet speed.

Fare details :

We will provide perfect fare details so that we need current location and destination location. We will specify the costs according to the situation.

Use case name	Fare details
Actor	passenger ,Driver&Admin
Brief Discription	This use case allows passengers to know fare details between two places.
Goals	To know the fare details from source to destination
Triggers	User clicks fare details option.
Pre-Condition	Passenger should enter source and destination places.
Post-Condition	Display the fare details.
Alternate Flow	None.
Exception	Handled if he enters wrong details.
Quality	Depending upon the net speed.

2.2 Non Functional Requirements

2.2.1 Performance Requirements :

- The system should be capable of storing and retrieving details of auto drivers and passengers as much as very fast.

2.2.2 Resource requirements :

- Every auto must be having a GPS connection to keep in track.

2.2.3 Verification requirements :

- Driver must have a driving licence to register in ARS.
- Auto must have all required certificates to certify that the auto is capable of transport.
- Driver must register with their personal details like mobile number.

2.2.4 Usability:

- The software allows user to view live status of auto and fare details.
- Nearby auto can be found in our software once registered for auto. It allows user to give feedback about autodrivers.

2.2.5 Security :

- We need to make sure that every driver logs in to his account while he is on duty.
- This will help to maintain a limited no. of passengers in autos to avoid the accidents caused because of overload.
- We will provide security to the luggage of passengers.

2.2.6 Database :

- We have to maintain a database for autos details and passengers details.
- For that we will use one server and stay connected with all passengers and auto drivers.

2.2.7 Maintenance :

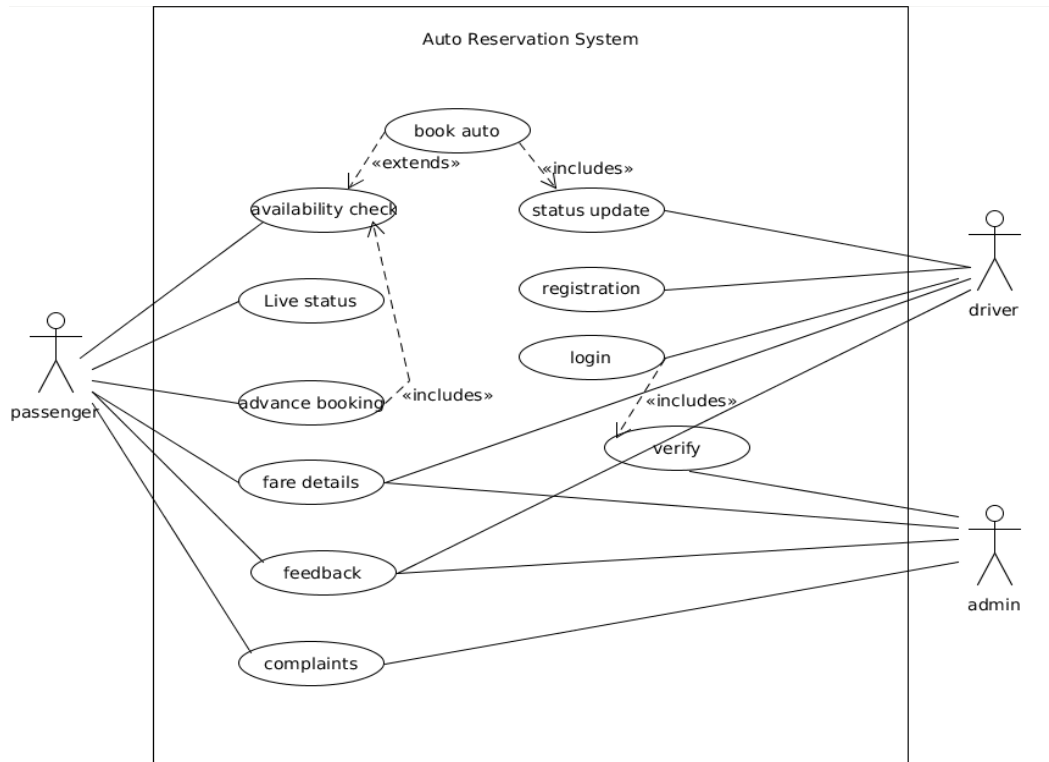
- For easy maintenance we use one admin method which means only admin can change the data.
- For users we'll provide read access only.
- For drivers we will provide some functionalities like updating their status, their location, etc.

2.2.8 Compatibility :

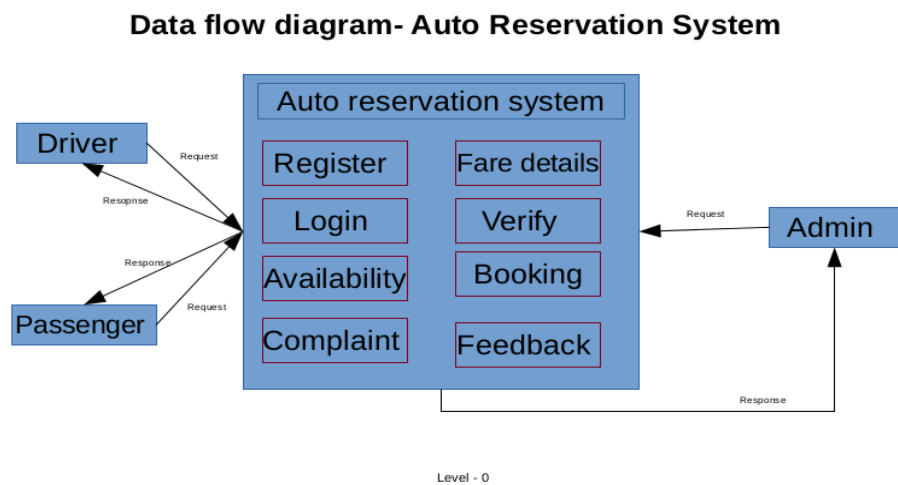
- We will provide an application which can be used in mobiles and PCs also.
- We will provide one web page to follow us and to get updated.

2.4 Diagrams

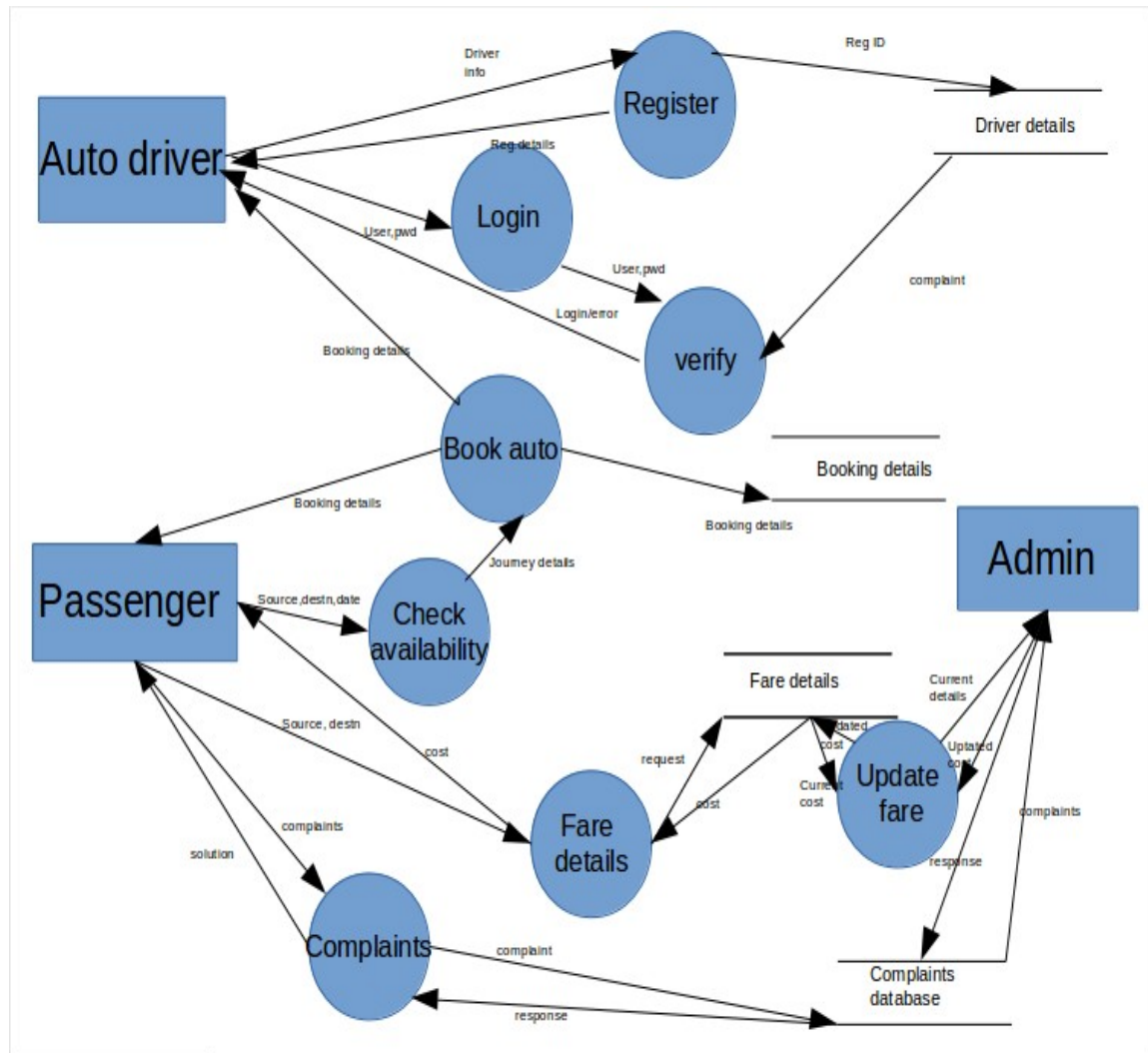
2.4.1 Usecase Diagram



2.4.2 Dataflow Diagram : level 0

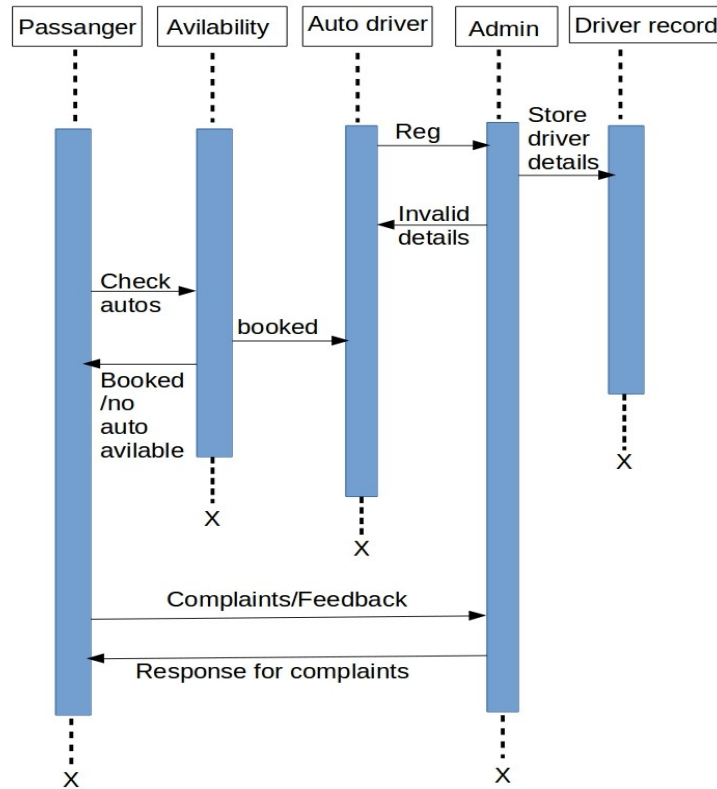


2.4.2 Dataflow Diagram : Level – 1



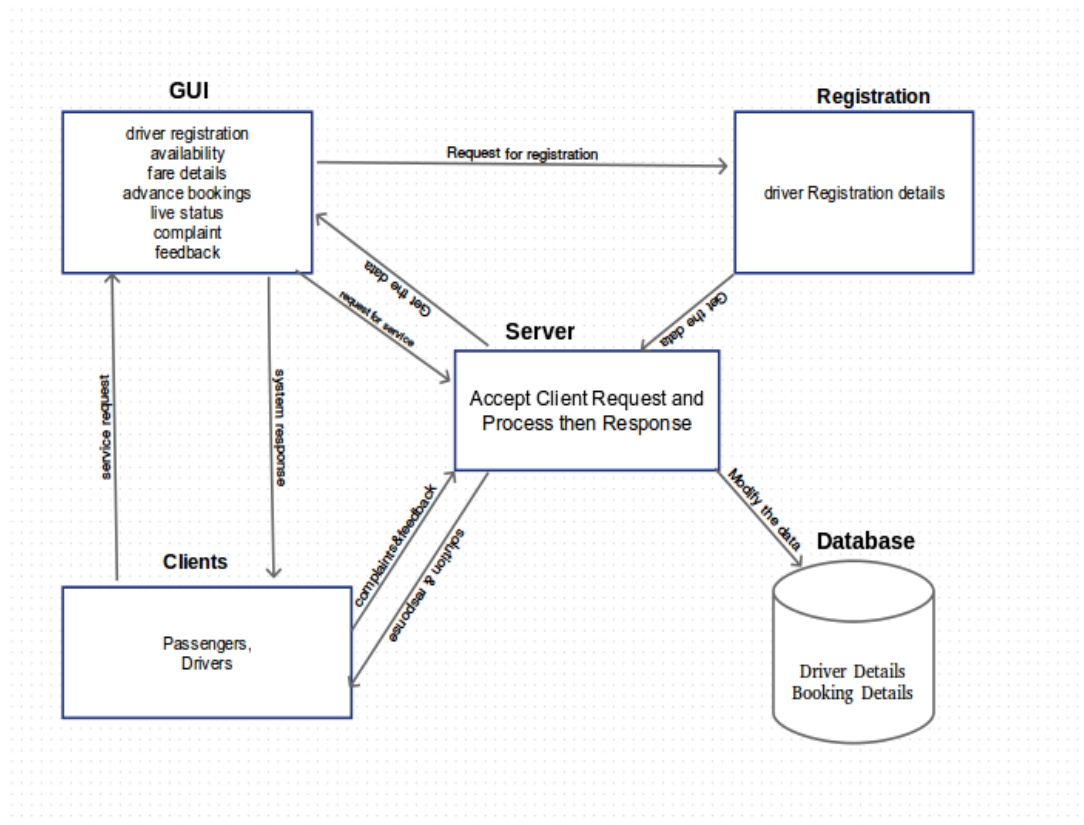
2.4.3 Sequential Diagram

Sequential diagram for Auto Reservation System



3. Software Designing Document

3.1 High Level Design



3.2 Detailed Design Document

Goals and Objectives

This project is to facilitate passengers to find a auto immediately and for auto drivers it will find the customers. With this software people can find easy transport and they can reduce the waiting time.

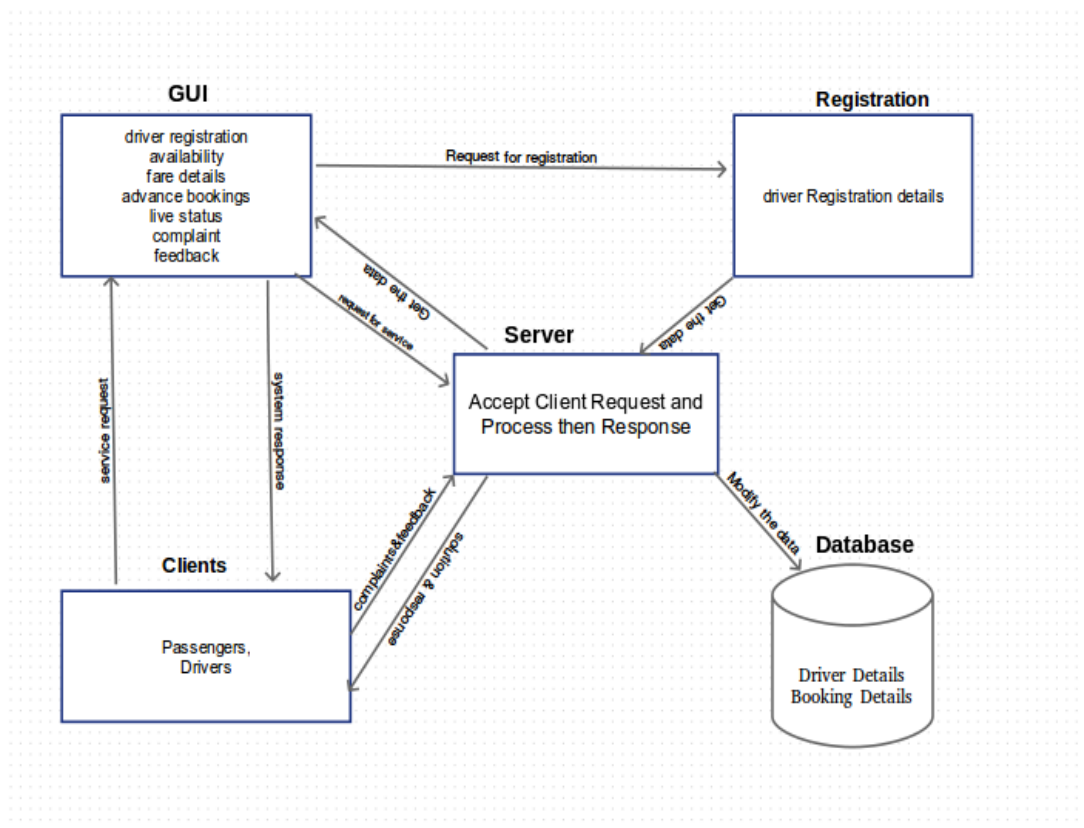
Scope

The scope of this project includes creating a communication between passenger and driver. The software stores each and every information about the journey. And it will also stores the entire details of a driver so that we can prevent many security problems.

Many rural areas are not having any transport facility so it could be so helpful for people who are living remote areas.

3.2.1 Architecture and Component Level Design

3.2.1.1 Architecture Overview



3.2.1.2 Description of Components

We divided our project into 3 modules. Those are

1. Server
2. Database
3. Graphical user interface

3.2.1.2.1 Server

In our project we will use php serverside scripting language because it executes in server along with maximum all available web servers like Apache, IIS(Internet Information Server) etc., .It is a Pre Process Hypertext, we could do many things on server by using PHP on server and co-ordinate with DB server for CURD(Create, Update, Read, Delete) actions.

Algorithms of server side :

Login:

```
login(username,password)
{
    connect to the mysql server
    select database
    sql=select * from driver_details where user_name=username and
                                           password=password;
    result=execute the query(sql);
    if(result==1)
        return 1;
    else
        return 0;
    close database;
}
```

Registration:

```
Registration(name,username,password,c_password,mobile_no,auto_reg_no,auto_no,address)
{
    connect to the mysql server;
    select database;
    if(password==c_password)
    {
        sql=insert into driver_details
            values(name, username, password, c_password, mobile_no,
                auto_reg_no, auto_no, address);
        result=execute the query(sql);
        if(result==1)
            return 1;
        else
            return 0;
    }
    else
    {
        display("try again with correct password");
    }
}
```

ForgotPassword:

```
forgotPassword(username,phonenumber,auto_reg_no,auto_no)
{
    connect to the mysql server
    select database
    sql=select * from driver_details where username=user_name and
    phonenumber=phonenumber and auto_reg_no=auto_reg_no and auto_no=auto_no;
    result=execute the query(sql);
    if(result==1)
        return result;
    else
        return 0;    }
```

ChangePassword:

```
changePassword(username,newpassword)
{
    connect to the mysql server
    select database
    sql=update driver_details set password=newpassword where
user_name=username;
    result=execute the query(sql);
    if(result==1)
        return 1;
    else
        return 0;
}
```

Booking :

```
booking(source,destination,user_mob,no_of_people,fare,time,auto_no)
{
    connect to the mysql server
    select database
    booking_id=generate booking id by time stamp;
    sql = insert into booking values(booking_id, auto_no, user_mob, source,
        destination, no_of_people,fare,time)
    result = execute the query(sql);
    if(result==1)
        return 1;
    else
        return 0;
}
```

Feedback :

```

feedback(booking_id,mobile_no,feedback_data)
{
    connect to the mysql server
    select database
    sql=insert into feedback(booking_id,user_mob,feedback)
        values(booking_id, mobile_no,feedback_data)
    result=execute the query(sql);
    if(result==1)
        return 1;
    else
        return 0;
}

```

Complaints :

```

complaint(booking_id,mobile_no,complaint_data)
{
    connect to the mysql server
    select database
    sql=insert into feedback(booking_id,user_mob,complaint)
        values(booking_id, mobile_no,complaint_data)
    result=execute the query(sql);
    if(result==1)
        return 1;
    else
        return 0;
}

```

Faredetails:

```

faredetails(source,destination)
{
    connect to the mysql server
    select database
    sql=select fare from fare_dist where source=source and
        destination=destination;
    result=execute the query(sql);
    if(result==1)
        return 1;
    else
        return 0;
}

```

Live Status :

```

livestatus(booking_id)
{
    connect to the mysql server
    select database
    sql = select location from live_status where booking_id=booking_id;
    result=execute the query(sql);
    if(result==1)
        return 1;
    else
        return 0;
}

```

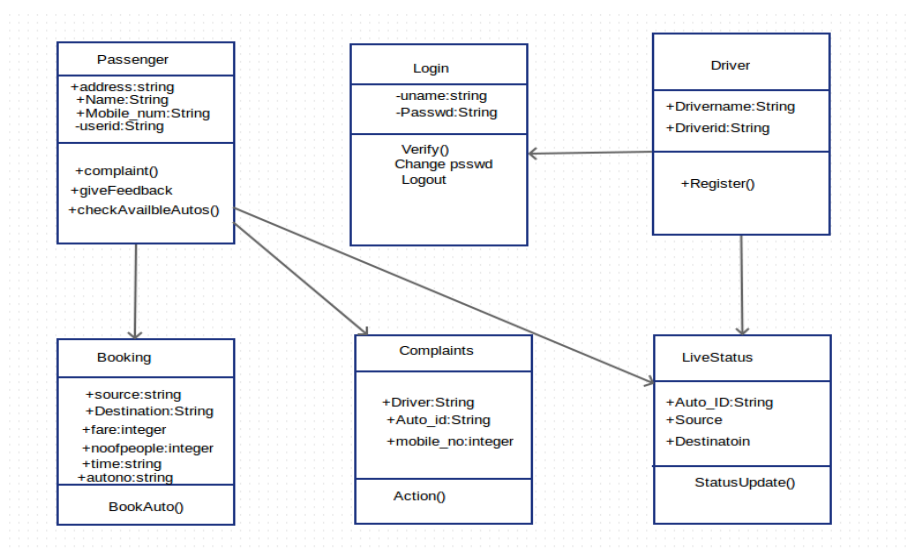
Availability :

```

availability(soure,destination,time)
{
    connect to the mysql server
    select database
    sql = select * from availability where source=source and
                                                destination = destination and time=time;
    result=execute the query(sql);
    if(result==1)
        return 1;
    else
        return 0;
}

```

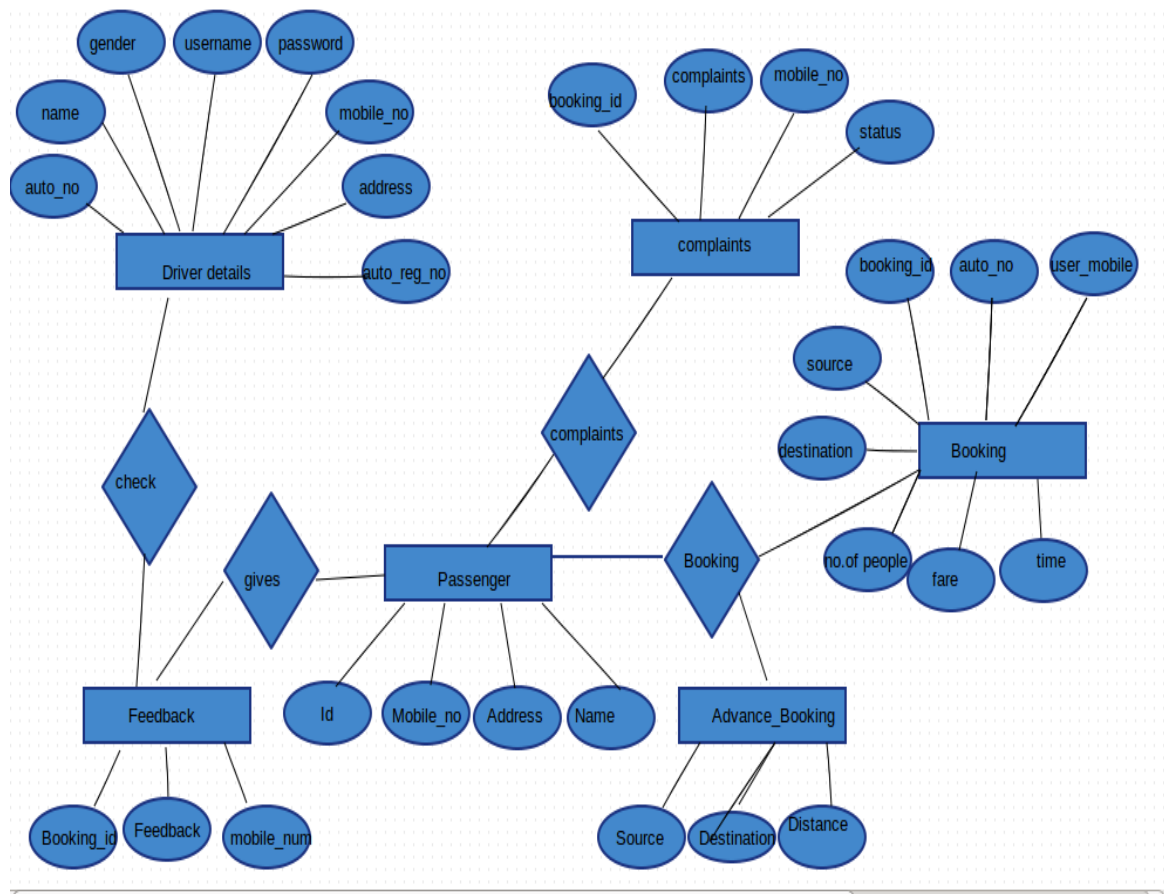
3.2.1.2.1 Class Diagram :



3.2.2 Data

Architecture :

3.2.2.1 Entity-Relation Diagram :



3.2.2.2 Database Architecture :

Driver_details:

Field	Type	Null	Key	Default
s.no	int(11)	No	PRIMARY	
name	varchar(30)	No		NULL
gender	varchar(10)	No		NULL
user_name	varchar(30)	No	UNIQUE	
password	varchar(30)	No		NULL
mobile_no	bigint(13)	No	UNIQUE	
auto_reg_no	varchar(20)	No	UNIQUE	
auto_no	varchar(20)	No	UNIQUE	
address	varchar(200)	No		NULL

booking:

Field	Type	Null	Key	Default
booking_id	varchar(20)	No	PRIMARY	
auto_no	varchar(20)	No	FOREIGN	
user_mob	bigint(13)	No		NULL
source	varchar(20)	No		NULL
destination	varchar(20)	No		NULL
no_of_people	int(5)	No		NULL
fare	bigint(20)	No		NULL
time	varchar(20)	No		NULL

Complaints:

Field	Type	Null	Key	Default
booking_id	varchar(20)	No	FOREIGN	
complaint	varchar(500)	No		NULL
user_mob	bigint(13)	No	FOREIGN	
status	varchar(20)	No		Not solved

Feedback:

Field	Type	Null	Key	Default
booking_id	varchar(20)	No	FOREIGN	
user_mob	bigint(13)	No	FOREIGN	
feedback	varchar(500)	No		NULL

Fare_dist:

Field	Type	Null	Key	Default
s.no	Int(11)	No	PRIMARY	
source	varchar(30)	No	FOREIGN	
destination	varchar(30)	No	FOREIGN	
distane	int(20)	No		NULL
fare	int(20)	No		NULL

live_status:

Field	Type	Null	Key	Default
booking_id	varchar(20)	No	FOREIGN	
location	varchar(20)	No		NULL

login_details:

Field	Type	Null	Key	Default
user_name	varchar(30)	No	FOREIGN	
source	varchar(30)	No		NULL
destination	varchar(30)	No		NULL
fromtime	varchar(30)	No		NULL
totime	varchar(30)	No		NULL
fromdate	varchar(30)	No		NULL
todate	varchar(30)	No		NULL

Availability :

Field	Type	Null	Key	Default
s.no	int(11)	No	PRIMARY	
auto_no	varchar(20)	No	FOREIGN	
available	varchar(10)	No		NULL
fairperkm	int(10)	No		NULL
fromtime	varchar(30)	No		NULL
totime	varchar(30)	No		NULL
fromdate	varchar(30)	No		NULL
todate	varchar(30)	No		NULL

3.2.2 Quality assurance

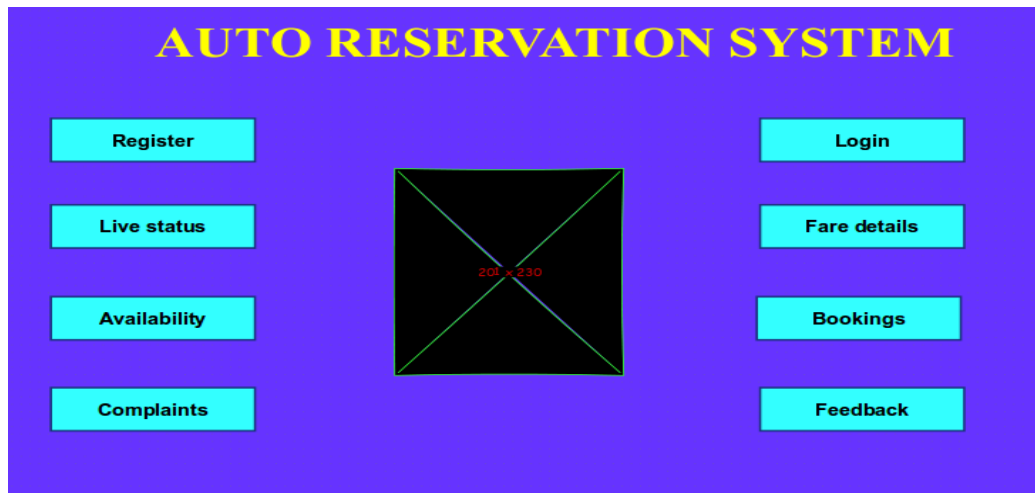
3.2.2.1 Detailed Test plan for auto_driver:

	Test Conditions	Expected Output
Register	Auto driver has to enter his/her data.	Driver registration should completed successfully if he/she given valid data.
Login	Driver has to enter valid username and password.	Driver should loggedin successfully if he/she entered valid data.
Show_feedbacks	Already logged in and click on show feedback button.	Shows the feedbacks of his/her bookings if he logged in.
Fare details	Has to enter the source and destination	Displays the fare details between the source and destination.

3.2.3 Graphical User Interface

3.2.3.1 Home

It's a home page. It consists the details of the services which can be provided by our Auto Reservation System. And it consists a logo in between services.



Attributes:

Register
Login
Live status
Fare details
Availability
Bookings
Complaints
Feedback

Methods:

Precondition : User must have internet facility to view our website.

Postcondition : User can see the home page of our reservation .

Algorithm :
If url in user browser == our website url and then
user can see our website.

Error Checking :

If user don't enter correct url he will have error below mentinoed way.

Please check address or Server not found.

Security:

Nobody can get to know the who are the users of our website.

3.2.3.2 Login :

By this service auto driver can log into his account and he will be available for booking.

HOME

DRIVER LOGIN

USER ID : XXXXXXXX

PASSWORD : XXXXXXXXX

LOGIN

Attributes:

User ID
Password

Methods:

Precondition : Driver should have to register before trying to login.

Postcondition : Driver can be logged in if he enters correct details otherwise he will be shown with some error message.

Algorithm :

if (url in browser == our website login page url) :
login page

Error Checking :

if driver won't enter correct url he will have error below mentioned ways.

Page not found.

3.2.3..3 Live status :

It's a interface to show the live status of a auto.

HOME

Live Status

Booking ID : XXXXXXXX

Locate

Clear

Location Details

Current Location : XX LOCATION

Remaining Distance : X KM

Estimated Time : X min

Attributes:

Booking ID
Current location
Remaining time
Estimated time

Methods:

Precondition : User must be registered an auto and must have booking id with him.

Postcondition : Passenger can get the current location, remaining distance and expected time to complete his journey.

Algorithm :

```
if booking_id is in database:
    show live_status
else :
    incorrect booking_id
```

Error Checking :

If someone tries to check live status without a perfect booking id it prints error like this.

Please check your booking ID.

Security: This is so secure because we are dealing with perfect booking database.

3.2.3.4 Fare Details:

It's main purpose is to provide the passenger perfect details about their journey cost estimation. In this passenger will get the fare details according to distance.



The screenshot shows a web form titled "Fare Details" on a blue background. In the top left corner, there is a "HOME" button. The form contains three input fields: "Source", "Destination", and "No. of People", each with a dropdown arrow. Below these fields are two buttons: "FARE" and "Clear".

Attributes:

Source
Destination
Date
Time
No. of People

Methods:

Precondition : Passenger must have the full details of his journey and he have o be in fare details page.

Postcondition : Passenger will get the complete information about fare details between to places.

Algorithm : It checks whether source and destination are perfect or not and if perfect then it will show corresponding fare.

```
if (source in database) and (destination in database) :  
    r : read fare_data from database  
    fare : calculate r*no_of_people  
    display fare
```

```
else :  
    display incorrect details
```

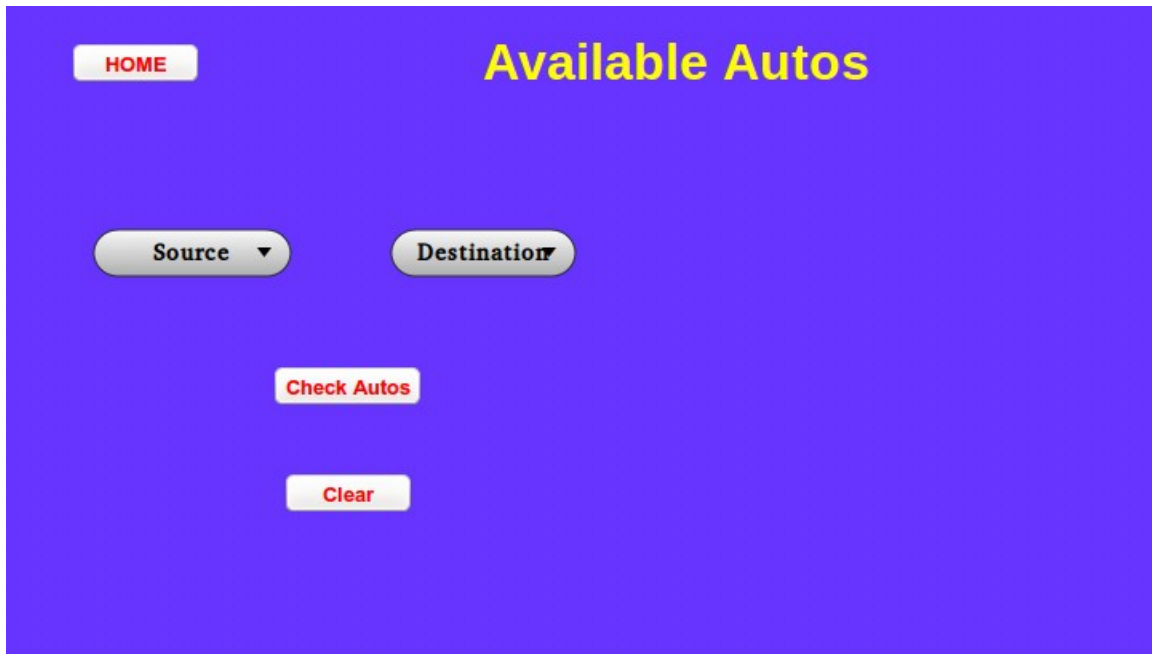
Error Checking : If someone tries to check unknown places fare details it prints error like this.

Please check the place you entered.

Security: Secure. Because we are dealing with perfect fare database.

3.2.3.5 Availability of Autos :

It is intended to check whether autos are present or not.



Attributes:

Source
Destination

Methods:

Precondition : Passenger must have the full details of his journey and he have to be in fare details page.

Postcondition : Passenger can get the list of autos as output and redirected to booking page.

Algorithm :

```
if (source == auto_source) and (destination == auto_destination) :  
    display auto_details  
else :  
    sorry no autos available, try again later.
```

Error Checking :

If there are no autos it will show warning message like you don't have any autos now.

Security:

Secure but we have to take care about cyber attacks because hackers can change the data in database.

3.2.3.6 Booking :

It's a interface to book an auto in advance.

S.No.	Auto ID	Source	Destination	Booking
1	TS06 XXXX	XXXX	XXXX	BOOK
2	TS06 XXXX	XXXX	XXXX	BOOK
3	TS06 XXXX	XXXX	XXXX	BOOK
4	TS06 XXXX	XXXX	XXXX	BOOK

Attributes :

Source
Destination
No. of People
Book

Methods:

Precondition : Passenger must have full details about his journey, he/she should be in advance booking page and there must be atleast one auto available.

Postcondition : Passenger will be redirected to confirmation page by clicking book button.

Algorithm :

```
if (source in database) and (destination in database) :  
    g[] : get available auto details  
    r : read fare_data from database  
    fare : calculate r*no_of_people  
    display auto details and fare details  
    click on booking :  
        go to booking page  
else :  
    display incorrect details
```

Error Checking : If user enters any mistake in input data. Then it will report it.

Security: This is so secure because we are not showing all auto details or passenger details to anyone and we are just showing just nearest autos details.

3.2.3.7 Confirm booking :

Here we can confirm our booking.

The screenshot shows a web form titled 'Confirm Booking' on a blue background. In the top left corner, there is a 'HOME' button. The form contains two columns of input fields separated by colons. The left column fields are: 'Auto ID', 'Source', 'Destination', 'Date', 'No. of People', and 'Fare'. The right column fields are: 'Auto ID', 'XX LOCATION', 'YY LOCATION', 'DD/MM/YYYY', 'XX', and 'X RS'. At the bottom of the form, there are two buttons: 'CONFIRM BOOKING' and 'CANCEL'.

Attributes:

Total journey details
Booking ID

Methods:

Precondition : User must fill the correct data and he have to be in confirmation stage.

Postcondition : Passenger can get the booking details that shows he had registered for journey. And a info will be sent to driver.

Algorithm :

```
if user click on confirm booking :  
    store booking details in database;  
else :  
    go to previous page.
```

Error Checking :

If someone tries to cancel their journey then they can cancel it now also.

Security:

This is so secure because we confirming a journey by checking two times.

3.2.3.8 Complaints :

In this page passengers can register a complaint regarding their journey.

The screenshot shows a web interface for registering a complaint. It has a blue background. At the top left is a 'HOME' button. In the center is the title 'Complaints' in large yellow font. Below the title are three input fields: 'Booking ID' with a value of 'XXXXXXXX', 'Mobile Number' with a value of 'XXXXXXXXXX', and 'Complaint' with a placeholder 'Write here'. Each field is preceded by a colon. At the bottom center is a 'Submit' button.

Attributes:

Booking ID

Date

Complaint

Methods:

Precondition :

User must travelled on that auto on or before of registration date.

Postcondition :

Complaint will be registered.

Algorithm :

```
if booking_id in booking details table:
    store the complaint in database;
else:
    display booking_id not valid.
```

Error Checking :

If someone tries to register a complaint without travelling on it, the error will be like this.

Please travel on this auto before registering complaint.

Security:

The main purpose of making this is to avoid security issues.

3.2.3.9 Feedback :

In this page passengers can give feedback regarding our auto service.

HOME

Feedback

Booking ID : XXXXXXXX

Mobile Number : XXXXXXXX

Feedback : Write here

Submit

Attributes:

Booking ID

Feedback

Methods:

Precondition : User must travelled on that auto.

Postcondition : Feedback will be stored on database.

Algorithm :

- if booking_id in booking details table:
 - store the feedback in database;
- else:
 - display booking_id not valid.

Error Checking : If someone tries to give negative feedback without travelling on it, the error will be like this.

Please travel on this auto before giving feedback.

Security:

The main purpose of making this is to avoid security issues.