

A Project report on
**“MODELLING HUMAN DRIVING BEHAVIOUR
THROUGH GAIL”**

Submitted in partial fulfillment of the requirement

for the award of the Degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

M.SUNILKUMAR GOUD

21701A05J9

S.TABASUM

21701A05K7

S.SAMEERA SULTANA

22705A0527

C.VENKATABHARADWAJA

21701A05M6

Under the Guidance of

Mr.D.S.CHANDRA MOULI M.Tech,(Ph.D)

Assistant Professor in CSE, AITS.



Submitted to

Department of Computer Science and Engineering

Annamacharya Institute of Technology and Sciences

(An Autonomous Institution)

(Approved by AICTE, New-Delhi and affiliated to J.N.T.U, Anantapur)

(Accredited by NBA &NAAC)

New Boyanapalli, Rajampet, Annamaiah(Dt),A.P-516126

2024-2025

Department of Computer Science and Engineering

Annamacharya Institute of Technology and Sciences

(An Autonomous Institution)

(Approved by AICTE, New-Delhi and affiliated to J.N.T.U, Anantapur)

(Accredited by NBA & NAAC)

New Boyanapalli, Rajampet, Annamaiah(Dt),A.P-516126



CERTIFICATE

This is to certify that the project report entitled **“MODELLING HUMAN DRIVING BEHAVIOUR THROUGH GAIL”** is submitted by

M.SUNILKUMAR GOUD

21701A05J9

S.TABASUM

21701A05K7

S.SAMEERA SULTANA

22705A0527

C.VENKATABHARADWAJA

21701A05M6

In partial fulfillment of the requirements for the award of Degree of **Bachelor of Technology** in **“Computer Science and Engineering”** for the academic year 2024-25.

Signature of Guide:

Mr.D.S.Chandra Mouli, M.Tech.,(Ph.D)
Assistant Professor in CSE,
AITS, Rajampet.

Signature of HOD:

Dr.M.SubbaRao M.Tech,Ph.D.,
Professor &Head,Dept.of CSE,
Dean of Student Affairs,
AITS, Rajampet.

Department of Computer Science and Engineering

Annamacharya Institute of Technology and Sciences

(An Autonomous Institution)

(Approved by AICTE, New-Delhi and affiliated to J.N.T.U, Anantapur)

(Accredited by NBA & NAAC)

New Boyanapalli, Rajampet, Annamaiah(Dt), A.P-516126



CERTIFICATE

This is to certify that the project report entitled “**MODELLING HUMAN DRIVING BEHAVIOUR THROUGH GAIL**” is submitted by

M.SUNIL KUMAR GOUD

21701A05J9

S.TABASUM

21701A05K7

S.SAMEERA SULTANA

22705A0527

C.VENKATABHARADWAJA

21701A05M6

In partial fulfillment of the requirements for the award of Degree of **Bachelor of Technology** in “**Computer Science and Engineering**” is a record of bonafide work carried out by him during the academic year 2024-25.

Project viva-voce held on : _____

Internal Examiner:

External Examiner:

Department of Computer Science and Engineering

Annamacharya Institute of Technology and Sciences

(An Autonomous Institution)

(Approved by AICTE, New-Delhi and affiliated to J.N.T.U, Anantapur)

(Accredited by NBA & NAAC)

New Boyanapalli, Rajampet, Annamaiah(Dt), A.P-516126



ANTI-PLAGIARISM CERTIFICATE

This is to certify that the project report entitled “**MODELLING HUMAN DRIVING BEHAVIOUR THROUGH GAIL**” is submitted by
in partial fulfillment of the requirements for the award of Degree of **Bachelor of Technology**
in “**Computer Science and Engineering**”. Course contains the plagiarism of ____ % which is
within the acceptable limits.

M.SUNIL KUMAR GOUD
S.TABASUM
S.SAMEERA SULTANA
C.VENKATABHARADWAJA

21701A05J9
21701A05K7
22705A0527
21701A05M6

Date:

Dean/Coordinator

Research & Development Cell

Date:

DECLARATION

We hereby declare that the project report entitled “**MODELLING HUMAN DRIVING BEHAVIOUR THROUGH GAIL**” under the guidance of **Mr.D.S.Chandra Mouli M.Tech,(Ph.D) Assistant Professor in CSE**, Department of Computer Science and Engineering is submitted in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Computer Science and Engineering.

This is a cord of Bonafide work carried out by me/us and the results embodied in this project report have not been reproduced or copied from any source. The results embodied in this project report have been submitted to any other University or institute for the Award of any other Degree or Diploma.

PROJECT ASSOCIATES
M.SUNIL KUMAR GOUD
S. TABASUM
S. SAMEERA SULTANA
C. VENKATA BHARADWAJA

ACKNOWLEDGMENT

We endeavor of a long period can be successful only with the advice of many well-wishers. We take this opportunity to express my deep gratitude and appreciation to all those who encouraged me for the successful completion of the project work.

*Our heartfelt thanks to our Guide, **Mr.D.S.ChandraMouli**^{M.Tech,(Ph.D)}Assistant Professor in Department of Computer Science and Engineering, Annamacharya Institute of Technology and Sciences, Rajampet, for his valuable guidance and suggestions in analyzing and testing throughout the period, till the end of the project work completion.*

*We wish to express sincere thanks and gratitude to **Dr. M. Subba Rao**, Head of the Department of Computer Science and Engineering, for his encouragement and facilities that were offered to us for carrying out this project.*

*We take this opportunity to offer gratefulness to our Principal **Dr.S.M.V.Narayana**, for providing all sorts of help during the project work.*

*We are very much thankful to **Dr. C. Gangi Reddy**, Honorary Secretary of the Annamacharya Educational Trust, for his help in providing good facilities in our college.*

*We would express our sincere thanks to all faculty members of **Computer Science and Engineering Department, batch-mates, friends and lab-technicians**, who have helped us to complete the project work successfully.*

*Finally ,we express our sincere thanks to **our Parent** swhohas provided their heartfelt support and encouragement in the accomplishment to complete this project successfully.*

PROJECT ASSOCIATES

M. SUNIL KUMAR GOUD

S. TABASUM

S. SAMEERA SULTANA

C. VENKATA BHARADWAJA

CONTENTS

CHAPTER	PAGENO
ABSTRACT	
1.INTRODUCTION	1-2
2.LITERATURE SURVEY	3-5
3.SYSTEM DESIGN	
3.1 Existing System	6
3.1.1 Disadvantages	6
3.2 Proposed System	6
3.2.1 Advantages	6
3.3 Modules in Proposed System	
3.3.1 Server	7
3.3.2.View and authorize user	7
3.3.3.End User	7
3.3.4.Algorithms Used	
3.3.4.1. GAIL	7
3.3.4.2. PS-GAIL	8
4.SYSTEM REQUIREMENTS SPECIFICATIONS	
4.1.Software Requirements	9
4.2.Hardware Requirements	9
4.3.Feasibility Study	9
4.3.1.Economic Feasibility	9
4.3.2.Technical Feasibility	10
4.3.3.Behavioral Feasibility	10
4.3.4.Benefits of Doing Feasibility Study	10
4.4.Functional and Non-Functional Requirements	
4.4.1.Functional Requirements	10
4.4.2. Non-Functional Requirements	11
5.SYSTEM DESIGN	
5.1. Architecture Design	12
5.2. Introduction to UML Diagrams	12
5.2.1. Goals	13
5.3.UML Notations	14-15

5.4.UML Diagrams	
5.4.1. Use Case Diagram	16
5.4.2. Class Diagram	17
5.4.3. Sequence Diagram	18
5.4.4. Collaboration Diagram	19
5.4.5. Deployment Diagram	20
5.4.6. Activity Diagram	21
5.4.7. Component Diagram	22
5.4.8. State Chart Diagram	23
6.SYSTEM CODING AND IMPLEMENTATION	
6.1. Introduction to Java Programming Language	24
6.1.1. Importance of Python	24
6.2. Java Virtual Machine	25
6.2.1. Java Script	26-27
6.2.2. HTML	28
6.2.3. JDBC	29-33
6.3. Sample Code	34-42
7.SYSTEM TESTING	
7.1.Software Testing Techniques	43
7.1.1. Testing Objectives	43
7.1.2. Test Case Design	43
7.1.3. White Box Testing	43
7.1.4. Black Box Testing	43
7.2. Software Testing Strategies	44
7.2.1. Unit Testing	44
7.2.2. Integration Testing	45
7.2.3. Validation Testing	45
7.2.4. System Testing	46
7.2.5. Security Testing	46
7.2.6. Performance Testing	46
8.RESULTS	48-54
9.CONCLUSION AND FUTURE ENHANCEMENTS	55
BIBLIOGRAPHY	56-57
PLAGARISM REPORT	
JOURNAL PUBLICATION	

LIST OF FIGURES & TABLES

Fig. No.	Figures	Page No
5.1.	Architecture Diagram	12
5.3.	UML Notations	14-16
5.4.1.	Use Case Diagram	17
5.4.2.	Class Diagram	18
5.4.3	Sequence Diagram	19
5.4.4	Collaboration Diagram	20
5.4.5.	Deployment Diagram	21
5.4.6	Activity Diagram	22
5.4.7	Component Diagram	23
5.4.8	State Chart Diagram	24
6.1.1	Working Of Java Program	25
6.2.1.	Java Virtual Machine	26
6.2.2.	Process of Java Program	26
6.2.3.1	Apache Tomcat Server	33

ABSTRACT

The modelling of human driving behavior using Generative Adversarial Imitation Learning (GAIL). GAIL leverages the power of generative adversarial networks to mimic and replicate complex human driving patterns by training a generative model to imitate the behavior of expert drivers. The study involves designing and implementing a GAIL framework that includes a generator to produce driving actions and a discriminator to assess the realism of these actions compared to real-world driving data. By integrating these components, the model learns to replicate nuanced driving behaviors, such as lane changes, speed adjustments, and reaction to road conditions. The research includes the collection and preprocessing of driving datasets, model training, and evaluation using metrics such as imitation accuracy, behavioral fidelity, and policy performance. The results demonstrate the potential of GAIL in creating realistic and adaptable driving behavior models, which can enhance autonomous vehicle systems, driver assistance technologies, and transportation simulations by providing a more accurate representation of human driving tendencies.

Keywords: *Generative Adversarial Imitation Learning, Human Driving Behavior, Autonomous Vehicles, Generative Models, Imitation Learning, Driving Patterns, Machine Learning, Behavioral Modeling, Simulation and Driver Assistance Technologies.*

CHAPTER-1

INTRODUCTION

1.INTRODUCTION

Automated vehicles can provide numerous and diverse benefits such as safety, efficiency, and equality of access. However, automated driving systems should undergo an exhaustive safety validation procedure before being made available to the public. Simulations are likely to greatly complement the acceptance of automated driving systems since they have lower risk, more limited testing time, and cost significantly less than real-world tests. For these simulations to make reasonable estimates of system performance, they must include human driver behavior as a representative model. In addition, while planning in an uncertain environment, autonomous vehicles will depend on robust models of human driving behavior.

Human behavior creates the observable vehicle trajectories. Our models must be able to accurately replicate the characteristics of these trajectories as observed in reality. Yet, the task of precisely modeling human driving behavior is complicated by various factors. First, human behavior is stochastic; it has varying phenomena in different contexts and at different instants, even if all other conditions are the same. One of the central challenges in human-robot interaction (HRI) is to deal with this intrinsic stochasticity of human behavior. Second, human behavior is multimodal. Several actions can be chosen by a human driver even if the intended behavior is more or less clear. Third, such attributes give rise to the generation of human behavior such as intent or invisible characteristics. Fourth, the behavior is expressed in terms of high-dimensional states and nonlinearities of the dynamics.

Driving is seen as a sequential decision-making process under uncertainty, and this work describes it in terms of Markov Decision Processes (MDPs). Solving an MDP requires cost function definition, but cost function was not available and is hard to realize for driving. Because of unavailability of a cost function, imitation learning, which is also called learning from demonstration, is very much promising. In imitation learning, expert demonstrations are used to induce a policy that behaves similarly to that of the experts on the unknown cost function. There are three main limitations of the conventional imitation learning methods.

Firstly, they usually fail in mimicking the expert owing to the limited class of cost functions. Second, the class of cost functions is often given as a span of a basis function requiring manual definition rather than learned from data. Thirdly, these tend to involve reinforcement learning re-

applications at high compute cost. Generative Adversarial Imitation Learning (GAIL) is one such attempt to remove such limitations. Via GANs, GAIL learns the cost using expressive function approximators like neural networks, as opposed to restricting the cost to being a member of a very narrowly defined class of functions. Additionally, with the help of Trust Region Policy Optimization (TRPO), GAIL directly develops policies rather than the intermediate value functions. In this work, we apply GAIL to learn models of human driving behavior from real-world driving data.

CHAPTER-2

LITERATURE SURVEY

2. LITERATURE SURVEY

1.Petrovic, D., Mijailovic, R. and Pesic, D.: "Accidents of vehicles with autonomy: varieties of collisions, maneuvers and faults of ordinary drivers." Transp. Res. Proc. 45, 161-168, Jan. 2020, doi:10.1016/j.trpro.2020.03.003.

The potential net reduction of traffic accidents may be achieved through the future application of autonomous vehicles. The reason attributed to this phenomenon is the absence of drivers in the causation of the accidents. Testing of autonomous vehicles in artificially simulated conditions has been done all over the globe; for the selected period, the authors reviewed autonomous vehicle accidents which occurred within the confines of the US state of California between 2015 and 2017. For a better appreciation of traffic accidents with autonomous vehicles, the authors did a comparative study of accidents involving traditional vehicles at those locations where there have been accidents between traditional vehicles and autonomous vehicles.

2.H. U. Ahmed, Y. Huang, and P. Lu, " With regard to the car-following model and modelling tools in the context of human and autonomous-ready driving behaviours in microsimulation," Smart Cities, vol. 4, no. 1, pp.314-335, Mar. 2021, doi:10.3390/smartcities4010019.

The platform of an experimental microtraffic simulation can be used to probe the driving behavior of vehicles along a road system. It is expected that the car-following behavior of robot-driven autonomous vehicles (AVs) and connected autonomous vehicles (CAVs) will have significant differences compared to the conventional human driven cars and will indeed need modelling of a different nature. Therefore, this paper targets at conducting a comprehensive literature review on car-following models used in common micro-simulation tools for human-driven and robot-driven vehicles. This discussion is, in particular, on the car-following logics like the Wiedemann model and adaptive cruise control technology based on dynamic behaviour and driving of the vehicle.

3.S. Pruekprasert, X. Zhang, J. Dubut, C. Huang, and M. Kishida, "Decision Making for Autonomous Vehicles at Unsignalized Intersection in Presence of Malicious Vehicles," in Proceedings of IEEE Intelligent Transportation Systems Conference (ITSC), 2019.

Such gatherings will look at decision-making for self-driving vehicles at unsignalized

intersections in the presence of malicious automobiles. Malicious vehicles break all traffic laws, putting themselves first, while non-malicious vehicles obey the rules. Each vehicle determines its control input from a Nash equilibrium, taking into account its priority belief. Finally, numerical simulations are their way of verifying the excellence of the method against a variety of malicious vehicle scenarios.

4.D.Isele, "Autonomous Driving in Overloaded Traffic", IEEE Intelligent Transportation Systems Conference (ITSC), 2019

The safety movements of any such vehicle, especially with merges whose gaps are smaller than the ego-vehicle, part consideration are on how autonomous cars can get through high-density urban traffic. A game-theoretic approach for interaction and deadlock management is also suggested by the authors. Among contributions is to outline how game-tree decision making could be implemented by autonomous cars through computationally tractable approximations. The authors further develop a stochastic rule-based traffic agent for simulating interactive behaviors for benchmarking in a crowded merge situation.

5.S. Panwai and H. Dia, "Neural agent car-following models," IEEE Transactions on Intelligent Transportation Systems, vol. 8, no. 1, pp. 60–70, 2007.

This paper proposes a car-following model that uses neural networks to relate observations to actions. The model can be classified as a desired separation model but is devoid of any reaction time and behavioral considerations. Its performances were checked against field data and neural network model had been found superior when compared to Gipps and psychophysical models. The validation at micro and macro levels depicts a reasonably good match with field data and stability analyses have proved that the model is robust against disturbances introduced by traffic.

6. N.M. Negash and J. Yang "Anticipation-Based Autonomous Platoon Control Strategy with Minimum Parameter Learning Adaptive Radial Basis Function Neural Network Sliding Mode Control" SAE Int. J. Vehicle Dyn., Stability, NVH, vol. 6, no. 3, pp. 247–265, Apr. 2022, doi: 10.4271/10-06-03-0017.:

A radial basis function neural network (RBFNN) based adaptive sliding mode controller is presented in this paper to cater for a 3-DOF robot manipulator with time-dependent uncertainties and disturbance. RBF is one of the most popular intelligent methods to

approximate uncertainties due to its simple structure and fast learning capacity. By choosing a proper Lyapunov function, the stability of the controller can be proven and the update laws of the RBFN can be derived easily. Simulation test has been conducted to verify the effectiveness of the controller. The result shows that the controller has successfully compensate the time-varying uncertainties and disturbances with error less than 0.001 rad.

7. M. H. Arshad, M. Bilal, and A. Gani, “Human activity recognition: Review, taxonomy and open challenges,” *Sensors*, vol. 22, no. 17, p. 6463, Aug. 2022, doi:10.3390/s22176463:

In this paper , Human Activity Recognition (HAR) is being widely used in a variety of domains, and vision and sensor-based data enable cutting-edge technologies to detect, recognize, and monitor human activities. Several reviews and surveys on HAR have already been published, but due to the constantly growing literature, the status of HAR literature needed to be updated. Hence, this review aims to provide insights on the current state of the literature on HAR published since 2018. The ninety-five articles reviewed in this study are classified to highlight application areas, data sources, techniques, and open research challenges in HAR.

8. N. Jiang, B. Yu, F. Cao, P. Dang, and S. Cui, “An extended visual angle car-following model considering the vehicle types in the adjacent lane,” *Phys. A, Stat. Mech. Appl.*, vol. 566, Mar. 2021, Art. no. 125665, doi:10.1016/j.physa.2020.125665:

In the real two-lane traffic environment without isolation belt, changes in vehicle type in the adjacent lane will have an impact on the driver’s following behavior in the lane. Due to changes in the vehicle type in the adjacent lane, we improve the full velocity difference model and present an extended visual angle car-following model. From the perspective of stimulus–response framework, the model takes the visual angle and its rate of change in front of the same lane and adjacent lane as stimulus factors, the stability condition is obtained through linear stability theory. The results indicate that the vehicle type in the adjacent lane has great impact on the stability of traffic flow.

CHAPTER-3

SYSTEM ANALYSIS

3. SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

Existing methodologies for Multi-agent simulation of human driving behaviour are as follows: closed-loop forward simulation, game-theoretic prediction, and open-loop independent trajectory prediction. The open-loop independent trajectory prediction paradigm has its injunctive models predicting each agent's trajectory in the scene. The model includes the multitude of simple models for the mentioned paradigm, based on the different mixtures of steering angle, yaw rate, acceleration, and constant velocity.

3.1.1 Disadvantages

- Tendency to Ignore Interaction
- Decreased Accuracy
- Lack of Ability to Mimic Complicated Behavior
- Lower Adaptability
- Inability to Manage Uncertainty

3.2 PROPOSED SYSTEM

We learn neural network policies modeling human driving behavior from demonstrations through automatic training rather than through features that are hand-designed. We have identified changes to GAIL, which are beneficial for driver modeling tasks: parameter sharing for multi-agent imitation, reward augmentation to provide domain knowledge, and the maximization of mutual information to extract individual driving patterns. Our driver modelling is demonstrated using real-world driving demonstrations from the NGSIM driving dataset.

3.2.1. Advantages

The proposed system replicates not only a single car but a complete ensemble of traffic scenes mimicking the driving behavior as it comes from agents' interactions.

- This can be regarded as a multicast control problem where multiple agents have to coordinate in driving under the same road scenario.
- During testing, the learned policy from a single experience with another agent observes that there exist cars around behaving differently than trained and again makes some minor mistakes that accumulate over time.

3.3. MODULES IN PROPOSED SYSTEM

3.3.1 SERVER:

The Service Provider in this module logs in with a valid username and password to perform certain operations such as Login, View All Users, View All Datasets, View All Datasets By GAIL, View Drinking Type Results, View Age Count Results and View City.

3.3.2. VIEW AND AUTHORIZE USERS:

In this module, the administrator can view the complete list of users - that is everyone who has registered. The administrator is able to see all the user information such as username, email address, and residential address. An administrator will then authorize the user.

3.3.3. END USER:

The module now accommodates a total of n users. Before performing any functions within this module, the user must register. On registration, user information is then stored in a database. After the successful completion of registering, the user is now to log in with the authorized username and password. After a successful Login, user will do some operations like Register and Login, View Profile, Upload Datasets, Find Driving Type Results, Find Driving Type Results By Hash code.

3.4. ALGORITHMS USED:

Below are the Machine Learning algorithms that have been utilized to train and test the sample dataset.

3.4.1 GENERATIVE ADVERSARIAL IMITATION LEARNING

Maximum Causal Entropy IRL (MaxEntIRL), a distinct imitation learning model, is the foundation of Generative Adversarial Imitation Learning. MaxEntIRL seeks to identify an expense function in which the specialist is the unique optimum, whereas apprenticeship learning seeks to identify an approach that is at least as excellent as the expertise on cost functions. This second goal turns out to be identical to identifying a policy with an occupation distribution equal to the expert under specific assumptions. This section explains how to derive

such a link, the resulting imitation learning method, and how it relates to Generative Adversarial Networks.

- GAIL integrates Generative Adversarial Networks and imitation learning so that the agent's generator will be the driving behavior it aims to learn while the discriminator distinguishes the agent's behavior from the expert's driving data.
- The agent (generator) will seek to create driving actions that will deceive the discriminator into believing it is replicating human driving.
- The discriminator is given the expert's driving data and the generated driving data of the agent and attempts to identify which data is authentic (human) and which is counterfeit (generated).
- The agent (generator) is trained to replicate the expert's behavior by gaining feedback from the discriminator.

3.4.2 PS-GAIL ALGORITHM

The PS-GAIL (Probabilistic Sampling Generative Adversarial Imitation Learning) algorithm is basically an expansion of the GAIL framework (Generative Adversarial Imitation Learning) and integrates probabilistic sampling into the process of learning for the sake of greater learning efficiency and imitation learning robustness.

- The concept of PS-GAIL is to apply a probabilistic method to sample expert states or trajectories in a more knowledgeable way, which could be beneficial when enhancing the performance of GAIL, particularly in situations with high-dimensional action or state spaces, for example, autonomous driving.

CHAPTER-4

SYSTEM REQUIREMENTS SPECIFICATION

4.SYSTEM REQUIREMENTS SPECIFICATION

Software requirements specifications (SRS), which are also referred to as software system requirements specifications, provide a complete description of the tasks a system has to perform. The use cases here provide an indication of how the software will deal with its users. The SRS also has non-functional specifications apart from usage case. Non-functional specifications are requirements that constrain design or implementation (e.g., performance engineering requirements, quality or design restrictions).

4.1. SOFTWARE REQUIREMENTS

- ☐ Operating System : Windows XP
- ☐ Coding Language : Java/J2EE(JSP,Servlet)
- ☐ Front End : J2EE
- ☐ Back End : MYSQL

4.2. HARDWARE REQUIREMENTS

- Processor : Pentium Dual Core
- RAM : 4 GB
- Hard Disk : 120 GB

4.3. FEASIBILITY STUDY

Identifying the best solution to satisfy performance needs is the objective of a feasibility study. They encompass a description of identification, an evaluation of possible system candidates, and the selection of the best candidate.

- Economic Feasibility
- Technical Feasibility
- Behavioural Feasibility

4.3.1. Economic Feasibility:

The most common method for deciding if an ongoing system is worthwhile is economic analysis. The procedure, better known as cost/benefit analysis, involves figuring up savings and benefits to determine if they exceed costs. If they do, the choice of designing and implementing the system is then made. If the system is to include an addition that requires

approval, additional justification or modifications must be done.

4.3.2. Technical Feasibility:

The capability of the current computer system to support the anticipated growth is the center of the technical examination (hardware, software, etc.). For technical progress to be permitted, financial issues should exist. The project becomes unfeasible if money is an extreme limitation.

4.3.3. Behavioral Feasibility:

Strength of the expected opposition of the user staff to making a computerised system should be estimated. To introduce a system that is possibly there requires an additional effort towards informing, influencing, and education of the prevailing ways of conceiving business. It is amply known that computer installations relate to knowledge somehow.

4.3.4. Benefits of Doing a Feasibility Study:

The following list outlines some of the advantages of conducting a feasibility study.

- Analysis part of this study, which is being designed as the first phase of the software development cycle, helps in carefully analyzing the system requirements.
- Helps to identify the risk variables involved in developing and applying the system.
- Risk analysis planning is facilitated by the feasibility study.
- Feasibility studies make it possible for cost-benefit analysis that facilitates efficient operation of the system and organisation.
- Feasibility studies assist in training developers planning to implement the system.

4.4. FUNCTIONAL AND NON-FUNCTIONAL REQUIREMENTS

The analysis of requirements is a critical activity to determine whether a system or software project will succeed. The two major classes of requirements are functional requirements and non-functional requirements.

4.4.1. Functional Requirements:

These are the requirements that have to be offered by the system if it is to fulfill the individual requests of the end user for primary comforts. The agreement has to inevitably include that each one of these functionalities be incorporated into the system. They are defined or represented as input to be given to the system, an action to be performed, and a resulting output. Unlike non-functional requirements, they are really the user-reported requirements that can be

observed straight away in the completed product.

An example of a functional requirement is:

- 1)Whenever there is a login to the system, the user has to authenticate.
- 2)In case of a cyberattack, turn off the system.
- 3)When a user is registering for the first time on a software system, an automatic verification email goes to him/her.

4.4.2. Not Functional Requirements

They generally represent the quality standards to be met by the system as per the agreement with the project. There might be various aspects assigned different levels of priority or enactment based on the project. They are also called non-behavioral requirements.

They cover mainly such as: Security, Portability, Maintainability, Reliability, Scalability, Performance, Re-use, Flexibility.

Examples of non-functional needs are :

- 1) Regarding that activity, emails must be sent within 12 hours maximum.
- 2) Serve every request in less than ten seconds.
- 3) When more than 10.000 users are simultaneous, the site should load within 3 seconds.

CHAPTER-5

SYSTEM DESIGN

5. SYSTEM DESIGN

5.1. ARCHITECTURE DESIGN

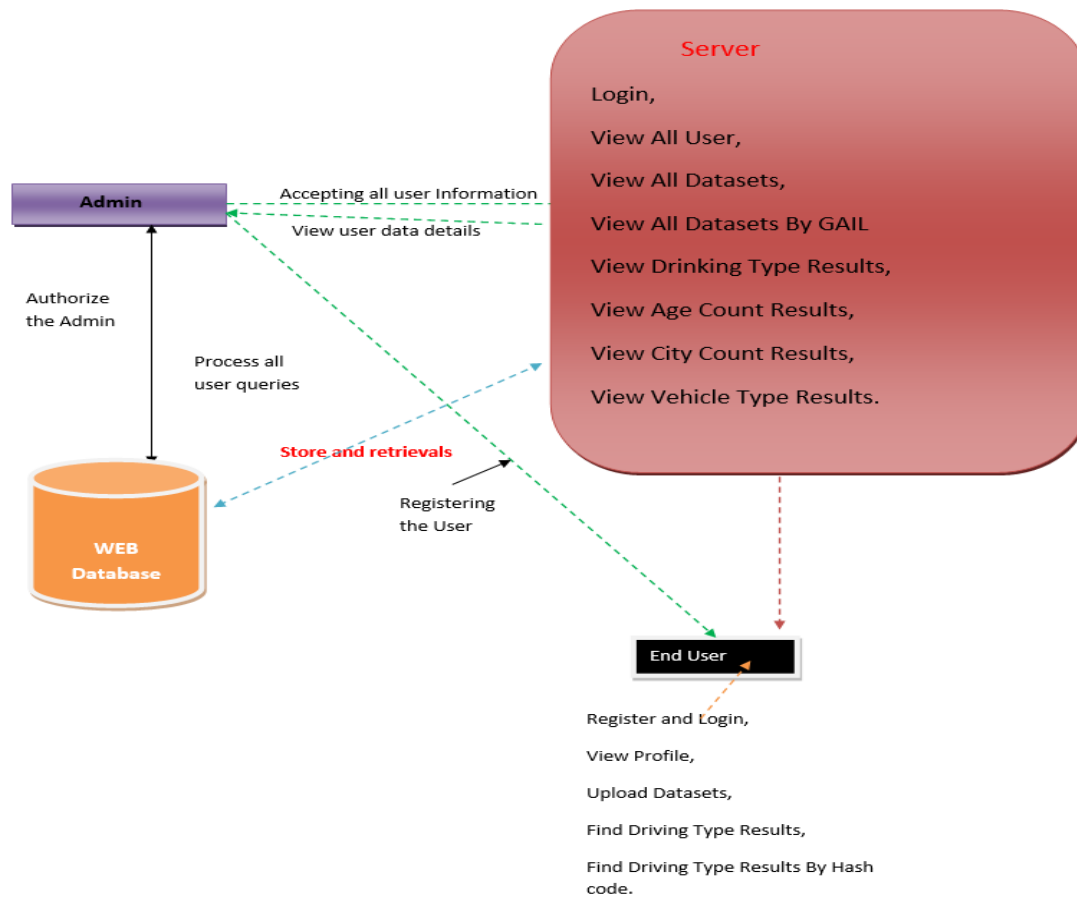


Figure 5.1: Architecture diagram

5.2. INTRODUCTION TO UML DIAGRAMS

As the strategic contribution of software continues to grow, the industry pursues ways of simplifying development, improving quality, reducing costs, and minimizing release cycles. Technologies such as reusable components, graphical programming, design templates, and software architecture are some of these strategies. As organizations get larger, they look for means of controlling the scope and magnitude of their programs, reducing their complexity. They are aware of issues related to traffic distribution, error resilience, concurrent processing, data replication, and distributed deployment. The web has enriched many architectural challenges while making some things simpler. Unified Modelling Language (UML) emerged to meet these requirements. System architecture planning entails in essence describing an application's structure, components, modules, interactions, and data organization in order to

realize certain goals. This may be accelerated through the application of UML visualizations. Throughout the project, eight core UML diagrams were elucidated.



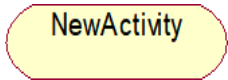
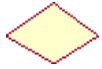

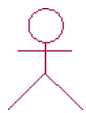
- Use Case Diagram.
- Class Diagram.
- Activity Diagram.
- Sequence Diagram.
- Collaboration Diagram.
- State chart Diagram.
- Component Diagram.
- Deployment Diagram.

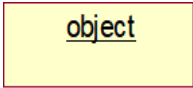

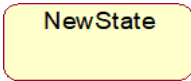


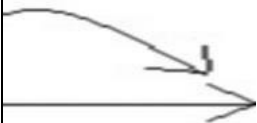
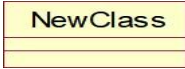

5.2.1 GOALS


The objectives of UML are multi-faceted, seeking to offer a general-purpose visual modelling language for software systems. UML emphasizes extensibility and specialization, making it possible to extend fundamental concepts to accommodate different application domains. In addition, UML is intended to be programming language and development process independent, making it applicable in various contexts. By establishing the foundation for a formal understanding, UML facilitates consistency and allows the creation of strong modelling tools.

- Provide users with an initiative and expressive visual language for constructing and trading transparent models.
- Provide for customization and extension options to enable the fundamental concepts to be used in a greater number of circumstances.
- Be neutral with respect to programming languages and software development paradigms.
- Provide a robust, formal foundation for the modelling language.
- These are the principal goals of the UML design:
- Accelerate the expansion of the market for object-oriented software tools.
- Advocate the application of sophisticated development concepts such as team collaborations, software architecture, design patterns, and modularity.
- Implement and encourage the best development practices.

5.3 UML NOTATIONS

S.NO	SYMBOL NAME	NOTATION	DESCRIPTION
1.	Initial Activity		This diagram depicts the flows initial point or activity.
2.	Final Activity		A bull's eye icon marks the conclusion of the activity graphic.
3.	Activity		Represented by a rectangle with a rounded edge.
4.	Decision		One that requires decision-making.
5.	Use Case		Explain how a user and a system communicate.
6.	Actor		A function a user has in relation to the system.

7.	Object		A Real -Time entity.
8.	Message		To communicate between the lives of object.
9.	State		It depicts events that occur during an objects lifetime.
10.	Initial State		Represents the objects initial state.
11.	Final State		Represents the objects final state.
12.	Transition		Label the transition with the event that triggered it and the action that result from it .
13.	Class		A group of items with similar structures and behaviours.
14.	Association		Relationship between classes.

15.	Generalization		Relationship between more general class and a more specific class.
-----	----------------	---	--

5.4. UML DIAGRAMS

5.4.1. USE CASE DIAGRAM

A use case diagram is one kind of behavior diagram, which is generated from a use-case study and is one method of applying the Unified Modelling Language (UML) in software engineering. A use case diagram essentially explains the actors, goals (represented as use cases), and relationships among those use cases within a system. The primary purpose of a use case diagram is to show which functions of a system are executed by which actors. It is immediately apparent which actor roles belong to the system. During the requirement elicitation and analysis stage, use cases describe what the system can perform. Use scenarios refer to how the system performs while idle. Use cases are inside the system, and the actors exist outside the system. A device boundary demarcates a collection of use cases within the use case diagram, and an actors' diagram. The application A diagram should be understood to comprehend the behavior of the element.

1. Sequences direct attention to the link to external ambience.
2. This outlines the performer's role and the system.
3. Actors can portray people or a building.

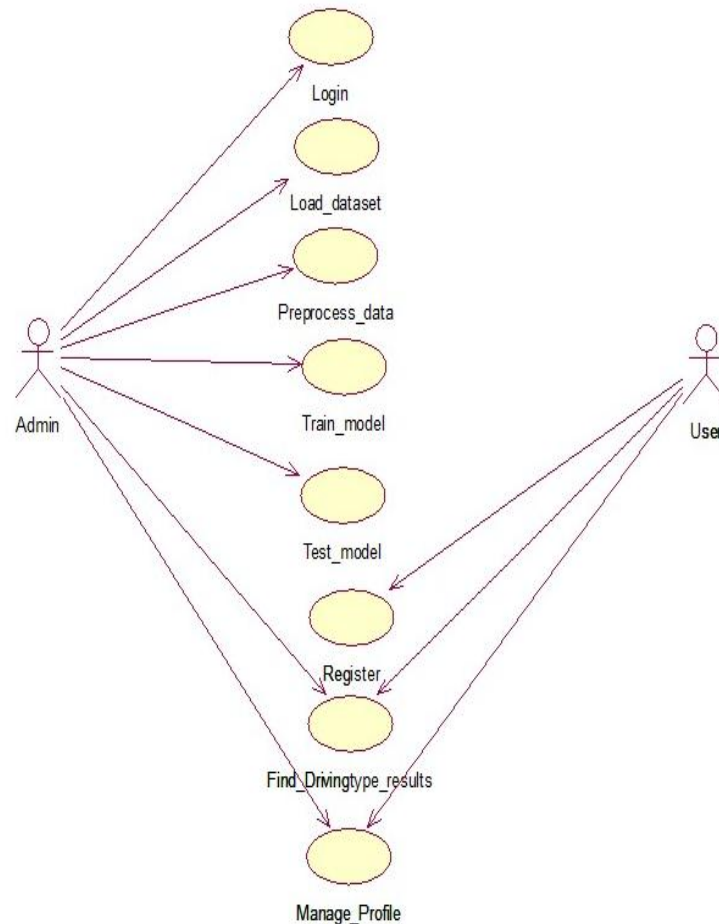


Figure 5.2: Use Case Diagram

5.4.2. CLASS DIAGRAM

A Common Language for Modelling (UML) class diagram is a static structural diagram in software engineering employed to illustrate the classes, properties, and relationships between the classes that constitute a system. It is used in analysis to display the system's details. Architecture scrutinizes the class diagram to decide which classes have too many functions and, if any, whether they should be split. The relationships between the classes are established. The Class Diagram is a method employed by the developer to produce classes. A class diagram is a collection of connected objects that are all related and share the same characteristics, operations, associations, and interlinks and rules known as semantics. A class is a large collection of things in a manufacturing.

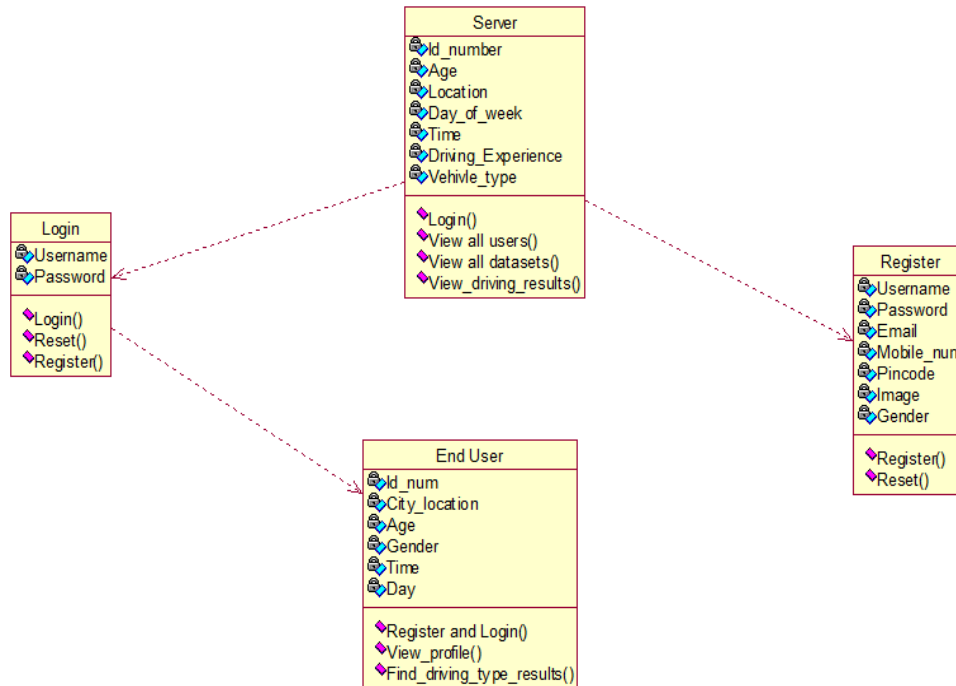


Figure 5.3: Class Diagram

In the Unified Modeling language, a class diagram is a type of static structural diagram that displays the functions, interactions, and relationships between objects to depict a system's structure. The cornerstone of object-oriented modeling is the class diagram. Image, build dataset, pre-processing, segmentation, and classification are the classes represented in together with the corresponding properties, processes, and relationships between those classes.

5.4.3. SEQUENCE DIAGRAM

A sequence diagram in the Unified Modeling Language (UML) is an interaction diagram that describes how activities operate over time and interact with one another. Sometimes it is considered synonymous with a message sequence chart. Sequence diagrams encompass timing diagrams, event-trace diagrams, and event context representations. In certain cases, one may refer to the sequence diagram as an event diagram or an event scenario.

Sequence diagrams demonstrate the communication between components of a system. Such diagrams are often used to describe requirements to entrepreneurs and software engineers for newer or existing systems. An interaction diagram that highlights the timing of message sending. Objects participating in an interaction are illustrated in a sequence diagram to show how long they exist and what messages they send or schedule during that time.

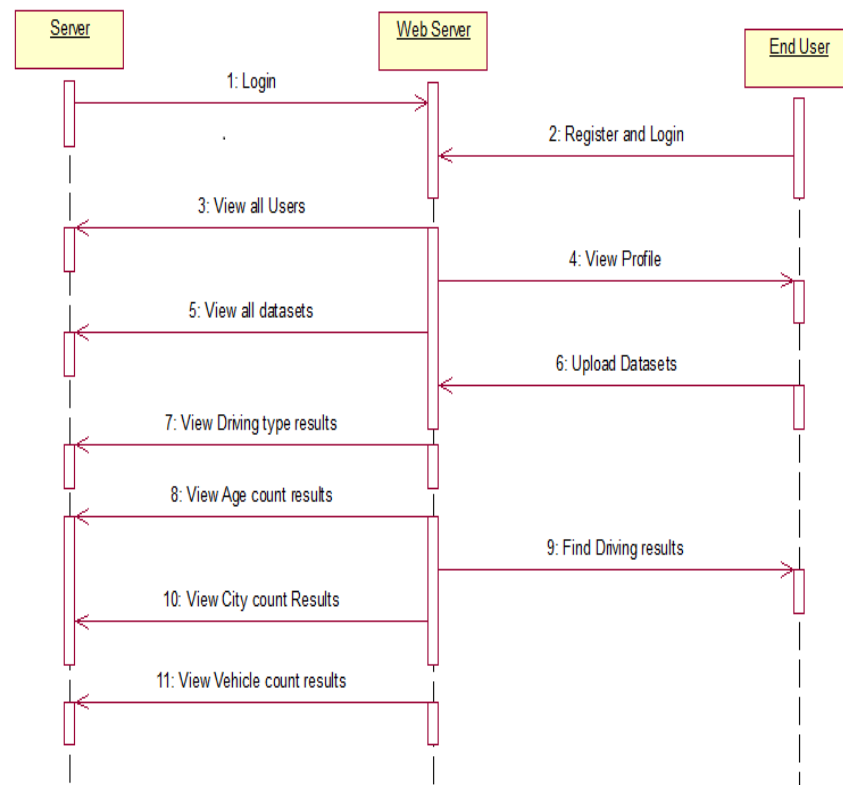


Figure 5.4: Sequence Diagram

5.4.4. COLLABORATION DIAGRAM

A collaboration diagram features a numbering scheme that indicates the sequence of method calls, as illustrated below. The number indicates the order of the method calls. The collaboration diagram continues with the same order management scheme. The messages in methods work just like that in a sequence diagram. The only difference is that while the sequence diagram does not specify an organization of object, the collaboration diagram does.

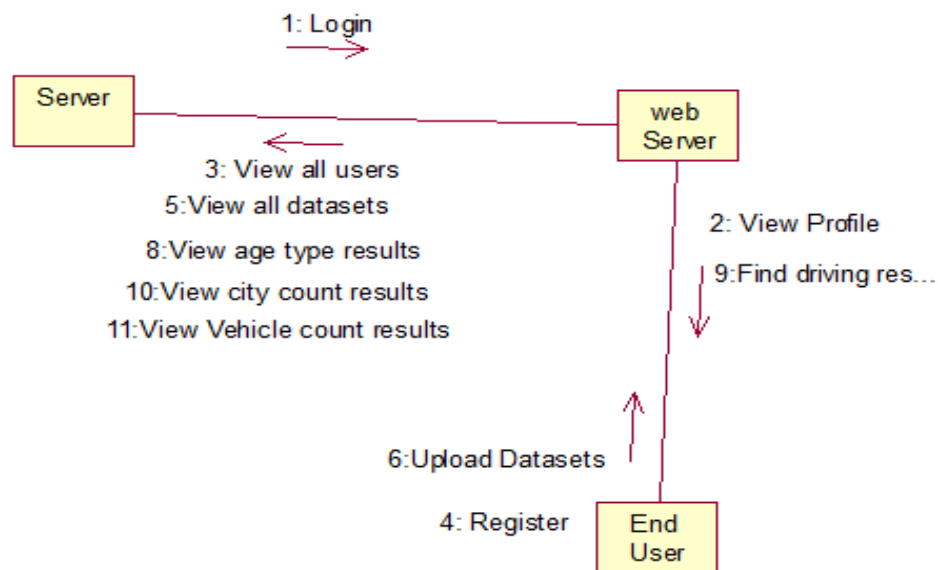


Figure 5.5: Collaboration Diagram

5.4.5. DEPLOYMENT DIAGRAM

The hardware and software elements that constitute a deployment are defined through deployment diagrams. Component and deployment diagrams share much in common. Deployment diagrams display diagrams of the deployment of components in hardware, which are utilized to define the components. The artefacts of software in a system are the focal point of UML. But these two specific diagrams are intended to emphasize the software and hardware components. Unlike deployment diagrams, which aim to focus on the hardware topology of a system, most UML diagrams are applied to handle logical entities. Diagrams are employed by the system engineers to perform deployment. You can describe the role played by deployment diagrams as:

1. Consider how hardware in a system is structured.
2. Describe the hardware components that are utilized in order to execute software components.
3. Describe the runtime processing nodes.

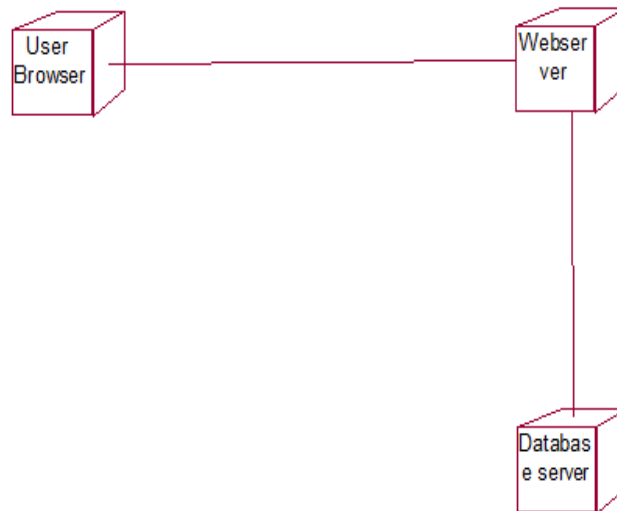


Figure 5.6: Deployment Diagram

5.4.6. ACTIVITY DIAGRAM

It is the activity diagram that depicts the condition for the flow of activities in the whole system. A flowchart with definite states is almost like an activity diagram. Following the order of activities taking place in your system can be done by using this activity diagram. Activities resemble states; however, they are slightly more rounded. As a result, they do not bear states because they occur and then go straight into any other state. The "diamond" conditional branch specifies which activity to change to based on a property and is stateless as well. Activity Diagram includes:

1. Action states.
2. Transition.
3. Objects.
4. Contains Fork, Join and branching relations along with flow Chart symbols.

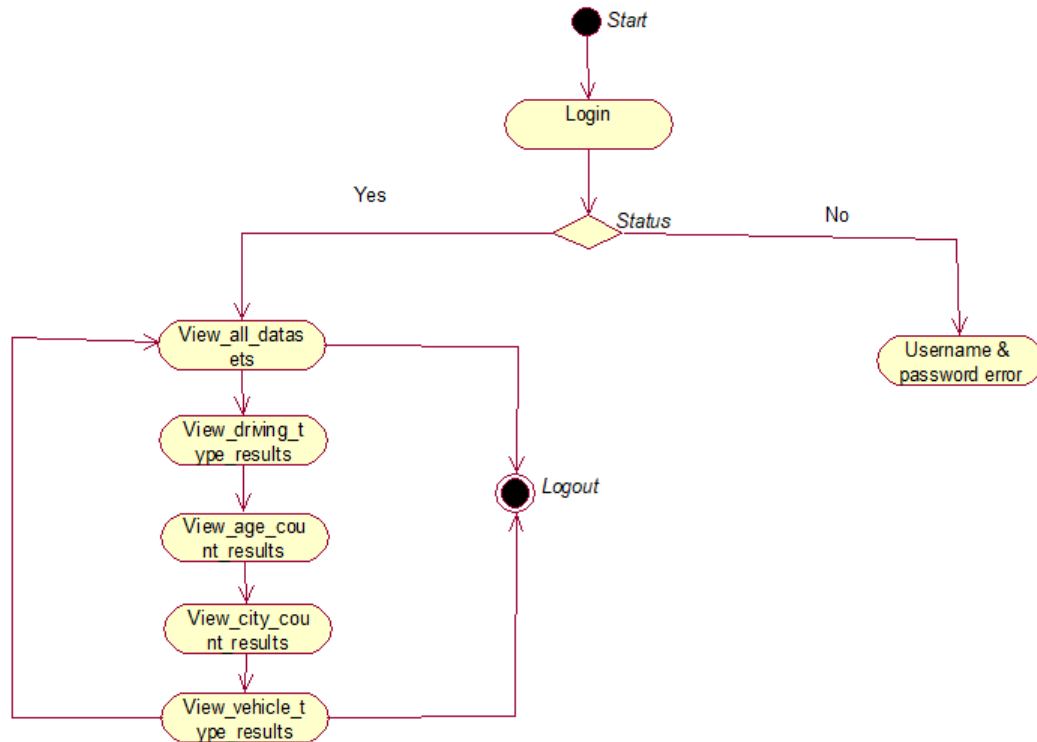


Figure 5.7: Activity Diagram

5.4.7. COMPONENT DIAGRAM

A special UML diagram is called a component diagram. The purpose of component diagrams varies from what we have already discussed. Although it pinpoints components as delivering some functionalities, it does not speak about the system functionally. From this perspective, component diagrams are being used to illustrate the entire physical picture of the components constituting the system. These components include files and libraries. Component diagrams can be described as a static implementation view of a system. Static implementation prescribes the composition of the components at some specific time. One single component diagram cannot represent the whole system; rather, it is a set of diagrams showing a view of the same system in various perspectives. The purpose of the component diagram can be summarized as follows:

1. Visually identify parts of a system
2. Create executables by forward and reverse engineering
3. Describe how the parts interconnect and laid out

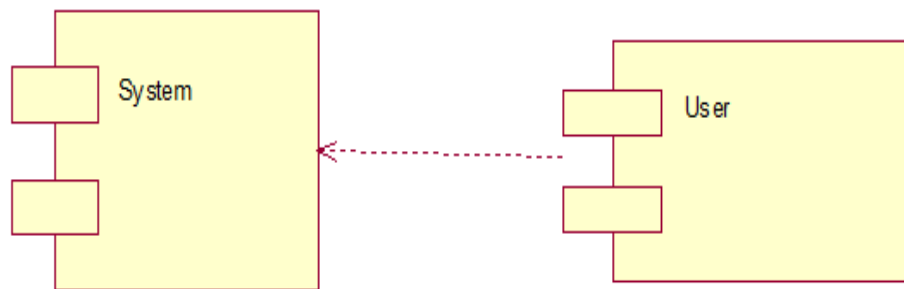


Figure 5.8: Component Diagram

5.4.8. STATE CHART DIAGRAM

A diagram of a state chart is a simple illustration of states as transfer of control among states. A state is a condition of an object existing and changing due to an event. The main purpose of a state chart diagram is to model the lifetime of such an object from birth to death. Forward and reverse engineering of a system also uses state chart diagrams. The goal of the model is, however, to model the reactive system.

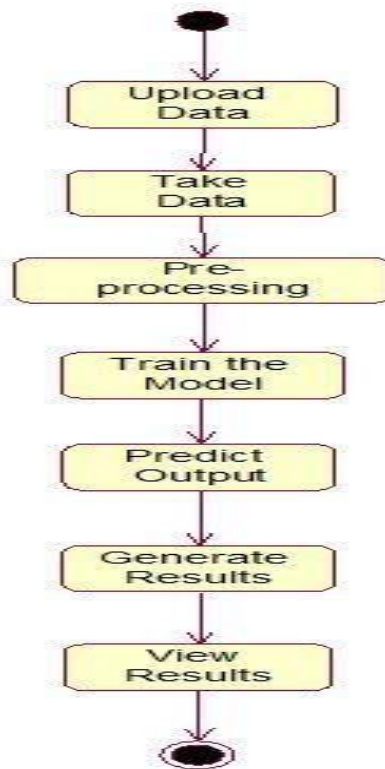


Figure 5.9: State Chart Diagram

CHAPTER-6

SYSTEM CODING AND IMPLEMENTATION

6. SYSTEM CODING AND IMPLEMENTATION

6.1. Introduction to Java Programming Language:

The first name of the language was to be "oak," but it was later renamed "Java" in 1995. The main motivation behind this language was to create a platform-independent (that is, architecture-neutral) language to be used in developing software for embedding in consumer electronic devices.

- Java is an object-oriented language.
- Java gives utmost control to the programmer irrespective of all these restrictions posed by the Internet environment.
- Finally, when it came to Internet programming, Java was C's junior sibling in its day.

6.1.1.Importance of Java :

It has greatly influenced the internet. The reason for this lies in the fact that Java widens the set of objects that can pass freely in cyberspace. In a network, a server and a personal computer transfer two classes of objects: passive information and dynamic active programs. The programs that are dynamic and self-executing pose serious threats to security and probability. Java stops those threats and in the process, it has opened up a novel and exhilarating class of programming called an Applet.

Characteristics:

- Simple
- Secured and Robust
- Portable
- Object oriented
- High Performance
- Scalable
- Multithreading

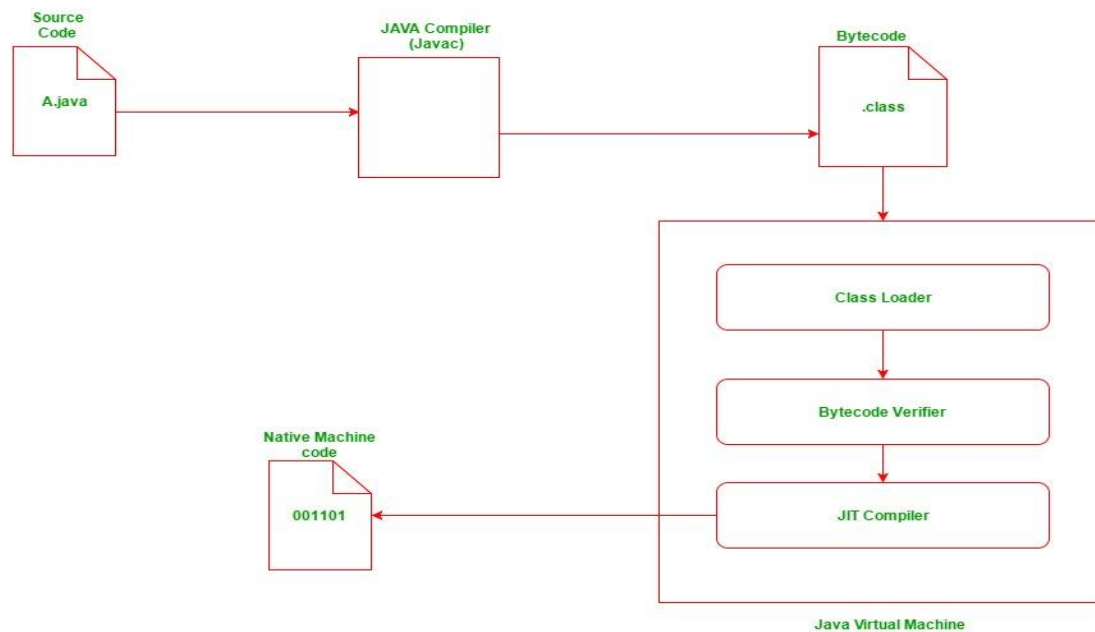


figure 6.1.1: working of java program

6.2. Java Virtual Machine (JVM)

Java Virtual Machine (JVM) Java, apart from being a language, has a virtual machine. The virtual machine is actually an important part of Java technology. This virtual machine can be integrated into a web browser or run in the operating system. After loading small pieces of Java code on that particular machine, authentication for the piece of code is required. Part of that loading involves calling a class loader which conducts byte code verification so that the code generated by the compiler does not cause the machine on which it is loaded to break down. Byte code verification is done once compilation is over to prove that everything is 'right,' and hence byte code verification forms part of Java code compilation and execution.

Java programming is used to generate and run byte codes. The first box explains that the Java source code exists within a .java file, compiled via a Java compiler called 'javac.' This compiler generates, as output, a .class file containing the byte code. The class file is loaded either over the network or loaded locally on your machine into the execution environment, which is called the Java virtual machine, which then interprets and executes it in byte code form. When you will run your code, the Java compiler generates machine code (which is referred to as byte code) meant for an imaginary machine called the Java Virtual Machine (JVM). This JVM then runs that byte code. The JVM was developed for the resolution of portability issues. The code

is compiled for one machine and interpreted on all other machines. This machine is called the Java Virtual Machine.

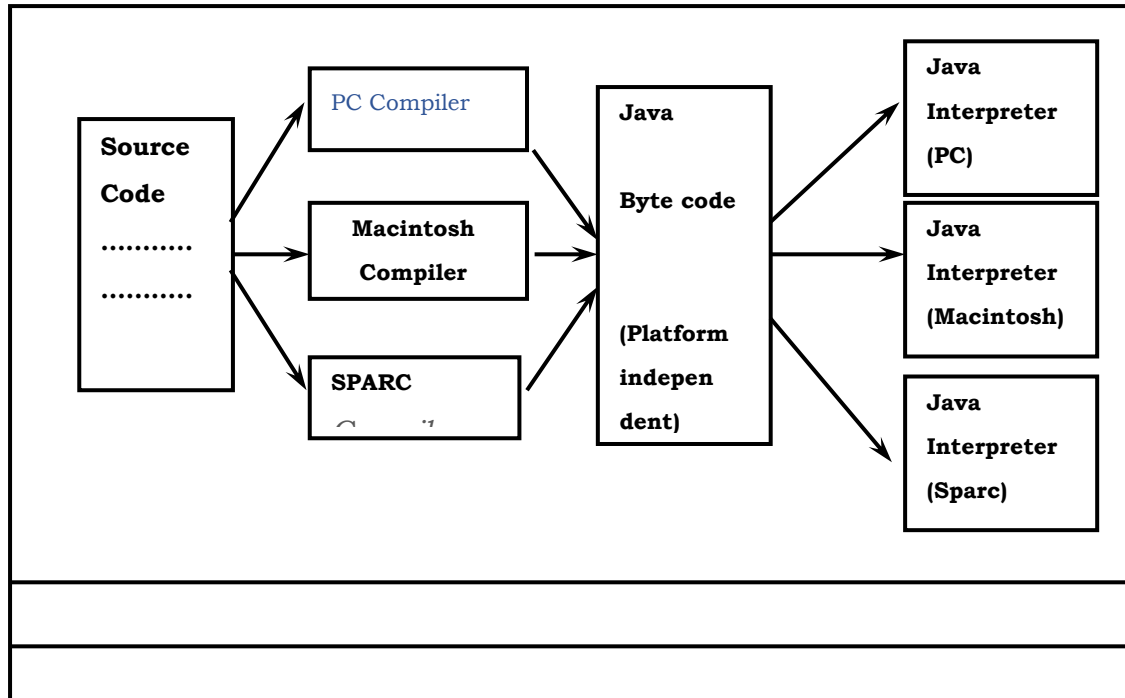


figure 6.2.1: Java Virtual Machine

6.2.1. JAVASCRIPT

A script type of programming language created by the Netscape Communication Corporation is known as JavaScript. Live Script was the original name for JavaScript, and the name was changed to JavaScript to suggest its relationship to Java. JavaScript supports both the client-side and server-side development of web-based applications. On the client-side, it can be used to write programs that are executed by Web browsers in the context of a Web page. On the server-side, it can write Web server programs that will process information given by the Web browser and update what is displayed on the browser accordingly.

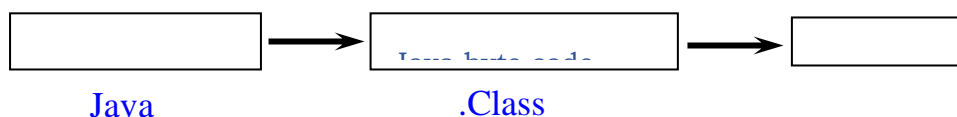


figure 6.2.2. process of JAVA Program

Between client-side and server-side web programming, however, we prefer using JavaScript for client-side programming as most of the browsers support it. JavaScript is nearly as simple to learn as HTML, and JavaScript statements are embedded in HTML documents enclosed

between a pair of scripting tags.

```
<SCRIPTS>..</SCRIPT>.
```

```
<SCRIPT LANGUAGE = "JavaScript">
```

JavaScript statements

```
</SCRIPT>
```

JavaScript enables us to accomplish the following:

- Check the contents of a form and perform calculations.
- Cause scrolling or varying messages in the browser's status line.
- Make images rotate or animate so that they vary as the mouse passes over them.
- Detect the browser in use and display different content for different browsers.
- Detect installed plug-ins and notify the user if a plug-in is required.

We can do much more with JavaScript, including creating entire application.

JavaScript Vs Java

JavaScript and Java are two distinct languages, and some of the clear differences between them are:

- Java applets are typically contained in a box in a web document, but JavaScript affects any part of the web document.
- When compared, JavaScript is best known for simple applications and can be used to add interactive functionalities to the web. However, Java can be used for complex applications that may need more advanced languages.

There are multiple distinctions between the two, and the essential point to keep in mind is that JavaScript is not Java. Although they are not the same thing, the two languages can be used for different tasks; in fact, they can be used in conjunction to amplify their respective advantages.

ADVANTAGES

- JavaScript can be used for Sever-side and Client-side scripting.
- It is more flexible than VBScript.

- JavaScript is the default scripting languages at client side since all the browsers support it.

6.2.2. Hyper Text Markup Language

The World Wide Web (WWW) uses Hypertext Markup Language (HTML), which enables users to create Web pages with text, images, and hyperlinks to other Web pages. Although it is an application of International Standard 8879, SGML (Standard Generalized Markup Language), HTML is not a programming language; rather, it is customized for hypertext and tailored for the Web. The idea behind hypertext is that we can simply go from one point to another rather than reading content in a strict sequential arrangement. The information can be sorted according to our preferences and areas of interest. Simply put, a markup language is a collection of elements that are separated by special characters that specify how text or other items contained within the elements look. Work that loads to other papers or specific sections of a single document is called a hyperlink.

Any kind of document can be shown using HTML on the machine that hosts it, which may be located in a different geographic location. Because of its versatility, it can be utilized on any desktop or platform. HTML offers tags, or special codes, to enhance the document's appearance. HTML tags don't care about case. The display of the document can be improved by utilizing fonts, graphics, various sizes, color, etc. The document itself contains everything that isn't a tag.

Basic HTML Tags :

<code><!-- --></code>	Specifies comments
<code><A>.....</code>	Creates hypertext links
<code>.....</code>	Formats text as bold
<code><BIG>.....</BIG></code>	Formats text in large font.
<code><BODY>...</BODY></code>	Contains all tags and text in the HTML document
<code><CENTER>...</CENTER></code>	Creates text
<code><DD>...</DD></code>	Definition of a term
<code><DL>...</DL></code>	Creates definition list

...	Formats text with a particular font
<FORM>...</FORM>	Encloses a fill-out form
<FRAME>...</FRAME>	Defines a particular frame in a set of frames
<H#>...</H#>	Creates headings of different levels
<HEAD>...</HEAD>	Contains tags that specify information about a document
<HR>...</HR>	Creates a horizontal rule
<HTML>...</HTML>	Contains all other HTML tags
<META>...</META>	Provides meta-information about a document
<SCRIPT>...</SCRIPT>	Contains client-side or server-side script
<TABLE>...</TABLE>	Creates a table
<TD>...</TD>	Indicates table data in a table
<TR>...</TR>	Designates a table row
<TH>...</TH>	Creates a heading in a table

ADVANTAGES

- A HTML document is small and hence easy to send over the net. It is small because it does not include formatted information.
- HTML is platform independent.
- HTML tags are not case-sensitive.

6.2.3. Java Database Connectivity

What Is JDBC?

A Java API for running SQL statements is called JDBC. It should be noted that JDBC is a trademarked word rather than an abbreviation, yet it is sometimes assumed to represent Java Database Connectivity. It is composed of a collection of Java programming language classes and interfaces. Writing database applications with a pure Java API is made possible by JDBC, which offers tool/database developers a standard API. Sending SQL commands to almost any relational database is simple with JDBC. With the JDBC API, a single program may be written

that can transmit SQL queries to the relevant database. A programmer may write code once and have it run anywhere thanks to the combination of Java and JDBC.

What Does JDBC Do?

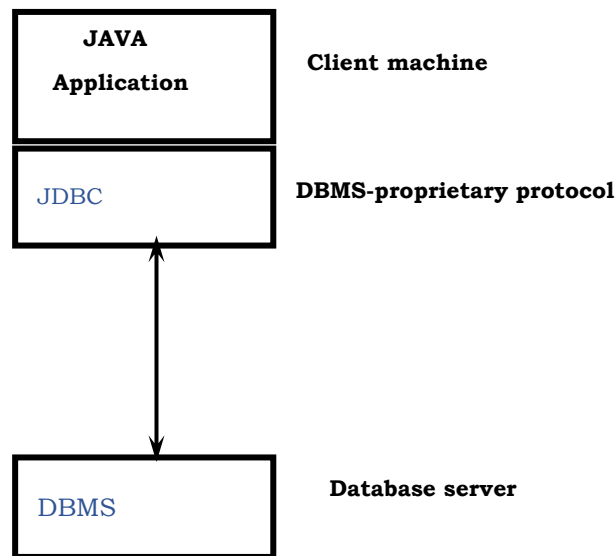
Simply put, JDBC makes it possible to do three things:

- Establish a connection with a database
- Send SQL statements
- Process the results.

Two-tier and Three-tier Models

Both two-tier or three-tier database access models are supported by the JDBC API. A Java program or applet communicates directly with the database in the two-tier paradigm. A JDBC driver capable of interacting with the specific database management system that is being used is necessary for this. The database receives a user's SQL queries, and the user receives the outcomes of those queries. It's possible that the database is on a different computer that the user has a network connection to. With the user's computer acting as the client and the database-hosting computer acting as the server, this setup is known as a client/server setup. The network may be the Internet or an intranet, which links workers within a company, for instance. A "bottom tier" of services receives commands in the three-tier paradigm, and they subsequently submit SQL queries to the database. The intermediate tier receives the results from the database's processing of the SQL commands and forwards it to the user. Since the middle tier allows for control over accessibility and the kind of adjustments that can be applied to company data, MIS directors find this three-tier approach to be particularly appealing. Another benefit of having a middle layer is that it allows the user to use an intuitive higher-level API, which the middle tier then translates into the proper low-level calls. Lastly, the three-tier architecture frequently offers performance benefits.

Up until now, the intermediate tier has usually been written in fast-performing languages like C or C++. However, implementing an intermediate tier in Java is becoming feasible with the advent of efficient compilers that convert Jvm byte code into effective machine-specific code. This is a significant benefit since it enables the use of Java's security, multithreading, and resilience capabilities. To provide access to databases from a Jvm middle tier, JDBC is necessary.



JDBC Driver Types

The JDBC drivers that we are aware of at this time fit into one of four categories:

- JDBC-ODBC bridge plus ODBC driver
- Native-API partly-Java driver
- JDBC-Net pure Java driver
- Native-protocol pure Java driver

Java Server Pages (JSP)

A straightforward yet effective technique for building and managing dynamic-content websites is Java Server Pages. Java Server Pages, which are based on the programming language Java, provide open standards, a mature reusable component model, and shown portability. Content creation and presentation can be separated thanks for the Pages for Java Server framework. In addition to reducing maintenance hassles, this division enables web staff to concentrate on their specialties. With less worry about affecting one another's work, web page designers can now focus on layout and app designers on code.

Features of JSP

- Separation of Concerns
- Embedded Java Code
- Tag Libraries
- Cross-Platform
- Automatic Compilation
- Integration with JavaBeans
- Support for MVC Architecture
- Expression Language (EL)
- Custom Tags
- Session Management
- Error Handling
- Security
- Dynamic Content Generation
- Support for HTTP Requests and Responses
- Support for Directives
- Database Integration

Steps in the execution of a JSP Application:

1. By including the JSP file name in a form tag of an HTML page, the client requests a JSP file from the web server.
2. The Java Web Server receives this request. If the request is for a JSP file, Java Web Server receives it at the server side and forwards it to the JSP engine.
3. A JSP engine is a software that can comprehend JSP tags, transforms them into Servlet programs, and stores the results on the server side. Once this servlet has been loaded into memory, it is executed, and the output is returned to the Java Web Server before being sent back to the client.

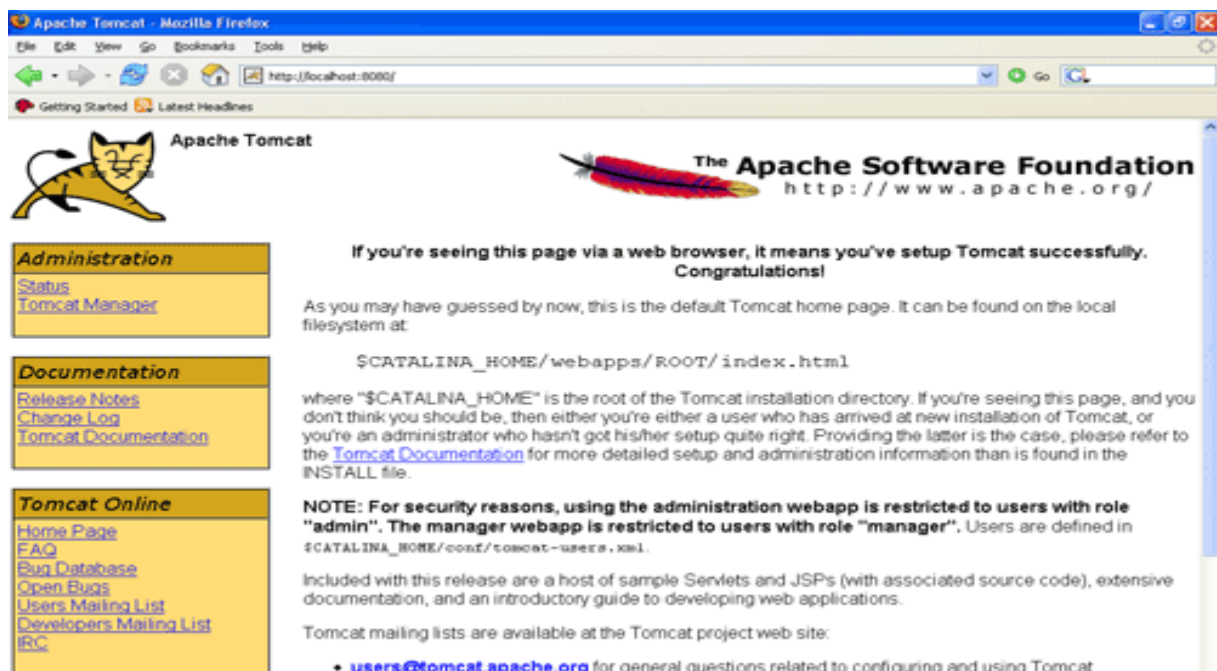
JDBC connectivity

The JDBC facilitates database-independent communication between a variety of tabular data sources and the J2EE platform. Application Component Providers can use JDBC technology to:

- Connect and authenticate to a database server;
- Manage transactions;
- Transfer SQL queries to an engine for databases for preprocessing and execution;
- Run stored procedures; and
- Examine and alter the output of select statements.

Tomcat 6.0 web server

The Apache Group created the open access web server Tomcat. The recognized Standard Configuration for the Java Serving and Java Server Pages, among other technologies uses Apache Tomcat as its servlet container. Sun uses the Java Community Process to design the Java Services and the Java Server Pages specifications. While application servers (such as BEAs Web logic, one of the most well-known application servers) handle both web and business components, web servers, such as Apache Tomcat, solely support web components. Install any web server, such as JRun, Tomcat, etc., to run your web application when you construct it using jsp /servlet.



6.3. Sample Code:

```
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<title>admin login page</title>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<link href="css/style.css" rel="stylesheet" type="text/css" />

<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />

<script type="text/javascript" src="js/cufon-yui.js"></script>

<script type="text/javascript" src="js/cufon-times.js"></script>

<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>

<script type="text/javascript" src="js/script.js"></script>

<script type="text/javascript" src="js/coin-slider.min.js"></script>

<style type="text/css">

<!--

.style1 {color: #FF3300}

.style2 {color: #009966}

-->

</style>

</head>

<body>

<div class="main">

<div class="header">

<div class="header_resize">

<div class="logo">
```

<h1>Modeling Human Driving Behavior through Generative Adversarial Imitation Learning</h1>

</div>

<div class="menu_nav">

Home

User

<li class="active">Server

</div>

<div class="clr"></div>

<div class="slider">

<div id="coin-slider"> </div>

<div class="clr"></div>

</div>

<div class="clr"></div>

</div>

</div>

<div class="content">

<div class="content_resize">

<div class="mainbar">

<div class="article">

<h2>Welcome to Server Login </h2>

<div class="clr"></div>

```
<div class="img"></div>
```

```
<div class="post_content">
```

```
<p align="justify">
```

```
<strong>Note :</strong> <span class="style1">*</span> = <span
class="style1">Required </span></p>
```

```
<form action="authentication.jsp?type=<%= "admin"%>" method="post">
```

```
<table width="88%">
```

```
<tr>
```

```
<th width="115" height="52"><span class="style4 style2">Username</span>
<span class="style1">*</span> </th>
```

```
<td width="260"><input id="name" name="userid" type="text" /></td>
```

```
</tr>
```

```
<tr>
```

```
<th height="44"><span class="style4 style2">Password</span> <span class="
style1">*</span> </th>
```

```
<td><input type="password" id="password" name="pass" class="text" /></td>
```

```
</tr>
```

```
<tr>
```

```
<td height="51"></td>
```

```
<td><input name="imageField" type="submit" id="imageField" value="Login"
/>
```

```
<input name="Reset" type="reset" value="Reset" />
```

```
</td>
```

```
</tr>
```

```
</table>
```

```
</form>
```

```
</p>
```

```
</div>

<div class="clr"></div>

</div>

<p class="pages"><small></small></p>

</div>

<div class="sidebar">

  <div class="searchform">

    <form id="formsearch" name="formsearch" method="post" action="#">

      <span>

        <input name="editbox_search" class="editbox_search" id="editbox_search"
maxlength="80" value="Search our ste:" type="text" />

      </span>

      <input name="button_search" src="images/search.gif" class="button_search"
type="image" />

    </form>

  </div>

  <div class="clr"></div>

  <div class="gadget">

    <h2 class="star"><span>Sidebar</span> Menu</h2>

    <div class="clr"></div>

    <ul class="sb_menu">

      <li><a href="index.html">Home</a></li>

      <li><a href="userlogin.jsp">User</a></li>

      <li><a href="adminlogin.jsp">Server</a></li>

    </ul>

  </div>
```

```
<div class="gadget">

    <h2 class="star">&nbsp;</h2>

</div>

</div>

<div class="clr"></div>

</div>

</div>

<div class="fbg"></div>

<div class="footer"></div>

</div>

<div align=center></div></body>

</html>

<title>users and admin authentication</title>

<% @page
import="com.oreilly.servlet.*,java.sql.*,java.lang.*,java.text.SimpleDateFormat,java.util.*,ja
va.io.*,javax.servlet.*,javax.servlet.http.*" %>

<% @ page import="java.sql.*"%>

<% @ include file="connect.jsp" %>

<% @ page import="java.util.Date" %>

<%

    String type=request.getParameter("type");

    application.setAttribute("type",type);

    try{

        if(type.equalsIgnoreCase("user"))

        {

            String username=request.getParameter("userid");
```

```
String Password=request.getParameter("pass");

String sql3="select * from user where username='"+username+"' and
password='"+Password+"'";

Statement stmt3 = connection.createStatement();

ResultSet rs3 =stmt3.executeQuery(sql3);

if(rs3.next())

{

    application.setAttribute("user",username);

    String sql="SELECT * FROM user where username='"+username+"'
and password='"+Password+"' and status='authorized';

    Statement stmt = connection.createStatement();

    ResultSet rs =stmt.executeQuery(sql);

    if(rs.next())

    {

        response.sendRedirect("usermain.jsp");

    }

    else

    {

        String sql1="SELECT * FROM user where
username='"+username+"' and password='"+Password+"' and status='waiting';

        Statement stmt1 = connection.createStatement();

        ResultSet rs1 =stmt1.executeQuery(sql1);

        if(rs1.next())

        {

            out.print("Sorry You are not authorized user");

            %><br /><a href="index.html">Click Here to
Home</a><%
```



```
        }

    }

}

else

{

    response.sendRedirect("wronglogin.jsp");

}

}

else if(type.equalsIgnoreCase("admin"))

{

    String username=request.getParameter("userid");

    String Password=request.getParameter("pass");

    String sql="SELECT * FROM admin where username='"+username+"'
and password='"+Password+"'";

    Statement stmt = connection.createStatement();

    ResultSet rs =stmt.executeQuery(sql);

    if(rs.next())

    {

        application.setAttribute("admin",username);

        response.sendRedirect("adminmain.jsp");

    }

    else

    {

        response.sendRedirect("wronglogin.jsp");

    }

}

}
```

```
    }  
    catch(Exception e)  
    {  
out.print(e);  
    }
```

CHAPTER-7

SYSTEM TESTING

7. SYSTEM TESTING

7.1. SOFTWARE TESTING TECHNIQUES

Software testing is a method for assess the quality of software products and detect faults so that they can be corrected. Software testing strives to achieve its objectives, but there are considerable limitations. Conversely, in order for testing to be efficient, commitment to the established goals must be shown.

7.1.1. Testing Objectives

1. User stories, design, specifications, and code constitute the work products
2. In order to ensure that all the conditions hold.
3. That the test object is thorough and satisfies expectations of users as well as stakeholders

7.1.2. Test Case Design

Every engineering product can be tested in one of these.

7.1.3. White Box Testing

Software testing techniques include white box and black box testing. This article discusses white box testing, also known as testing for structural integrity, white box evaluation, open box evaluation, and translucent box testing. Its focus is on evaluating software's foundational code and infrastructure in relation to current inputs and anticipated and intended results. It focuses on the internal workings of a program and is concerned with the analysis of internal structure. White box testing's primary goal is to focus on the software's inputs and outputs while also guaranteeing its security. The phrases "transparent package," "white box," and "clear box" all describe the ability to see through the software's outside. Designers use a white testing box.

7.1.4. Black Box Testing

Verification of software programs' functions without access to their internal code architecture, implementation, or internal paths is called "black box testing" in the field of software testing. "Black box testing" is an example of a type of software testing which is only interested in input and output of software programs along with software requirements and specifications.

You may use whichever package of software you prefer as a Black-Box. Examples are an

Oracle database, a Google website, the Windows operating system, or even your own personal program. You can conduct black box testing on these programs by concentrating on their inputs and outputs and having no knowledge about how their code underneath is actually implemented.

This approach looks for faults in the following categories:

1. Deficient or missing capacities
2. Interface faults
3. Structure of information errors
4. Behavioral or execution faults
5. Start and finish faults

7.2. STRATEGIES FOR SOFTWARE TESTING

- A unit test
- Integrity Checks
- Validation Examination
- System Evaluation
- Security Checks
- Performance Evaluation

7.2.1. Unit Testing

The module is the most minute component of software architecture that is being tested under unit testing. Under the limits of the module, important control channels are examined using the procedural description of design as a reference. The most minute units of a program that can be tested independently, referred to as units, are examined independently and individually during unit testing in order to ensure correct functioning. This testing process is employed by software developers and, on occasion, QA personnel during the development cycle. The primary purpose of unit testing is to test and verify written code in isolation to make sure it works as planned. If properly done, unit testing can assist in finding coding defects that otherwise would be hard to find. TDD is an effective method that continuously tests and improves the product development process holistically. Unit testing is one of the components of TDD.

This test technique is the first step in software testing and contains tests that precede integration testing and other forms of testing. Unit testing confirms a unit's independence from external

code or functionalities. Manual testing remains an alternative even when automation testing is more widely used.

7.2.2. Integration Testing

Integration testing is the act of building a program's structure while executing tests to identify interface issues. For building a design-based program structure, unit-tested methods are to be employed. Integration testing is a testing process that theoretically binds and tests software modules together. Multiple software modules developed by different programmers constitute a typical software project. Identifying faults with the way different software modules interact when integrated is the purpose of this level of testing. The interaction among these modules is tested during integration testing. It is referred to as "String Testing," and the end product is "Thread Testing."

Top-Down Integration:

The second step in the test process is top-down integrations, a technique to develop and test a program structure incrementally. Various modules of a software, product, or application are integrated by descending through systematic control hierarchy between modules from the top, beginning with the chief control or home control or index program. The project framework consists of a mix of breadth- or depth-first activities or modules associated with the master program.

Bottom-up Integration:

The building and testing of some atomic modules, or the product's most elementary features, is the initial step in the following testing approach. As all processes or modules are integrated bottom-up, residual is not required, and processing for modules associated with a specific level is always accessible.

7.2.3. Validation testing

Validation testing ensures that the software written and tested meets the requirements of the client or user. Logic or business scenarios must be adequately tested. In this case, you need to test each important aspect of the application. You should always be able to validate the business logic or scenarios presented to you as a tester. One such process that promotes careful analysis of working is the validation process.

Validation testing verifies that the program has been tested and developed to satisfy user or

customer specifications. The justifications or situations for business needs should be comprehensively tested. Each major part of the application should be tested in this scenario. You will always be given scenarios or business logic as a tester, which can be verified independently. One such process that assists in a detailed performance analysis is the validation process.

7.2.4. System Testing

The primary aim of system testing is to strictly test computer-based systems. Although every test has a specific aim, they all verify that each part of the system is appropriately integrated in an effort to achieve the objectives. Testing a fully integrated software system is part of system testing. A computer system is generally built by combining software (any Software is the only part of a computer system. The program consists of modules which, when combined with other elements of software and hardware, create a full computer system. That is, a computer system consists of many software programs that do different jobs. Software itself cannot perform these tasks without help. System Testing involves the right hardware should be utilized to assist.

System testing is a collection of processes employed to confirm the general operation of a computer system utilizing combined software. System testing practice entails analyzing an application's or software's end-to-end process from the perspective of a user. Every module necessary for an application is thoroughly analyzed, and system product testing is conducted to make sure that the ultimate features and functionality work as intended. Because the testing surroundings are similar to the production environment, the test is referred to as end-to-end testing.

7.2.5. Security testing

Security Testing is a key aspect of testing software because it helps us discover Software risks, hazards, and vulnerabilities programs and safeguard our program from evil intruders. The primary aim of security examinations is to detect potential ambiguities and weaknesses of a program that keeps application running. By doing security testing, we may find out any possible security threats and help the programmer fix any problem. It is a method for ensuring data security while preserving software usability.

7.2.6. Performance Testing

Performance Testing is a method for assessing system responsiveness and stability under changing workloads. Performance testing assesses the dependability, scalability, and resource use of the system.

Performance Evaluation Method:

Load testing is the easiest method of testing how effectively a system will run under a specific load. The results of a load test will indicate how much workload is placed on the application server, database, and other systems and the relative importance of major business transactions. Stress testing is conducted to determine the maximum capacity of the system and how it will function if the current load is more than the forecasted maximum.

Soak tests, often called endurance tests, are used to evaluate a system's performance under a steady load. During soak testing, memory usage is monitored to identify performance issues like memory leaks. Monitoring the system's performance over time is the main objective. When testing during a "spike," the user base is rapidly expanded and the system's performance is swiftly examined. The main objective is to assess the system's workload management capabilities.

CHAPTER-8

RESULTS

8. RESULTS

OUTPUT SCREEN SHOTS WITH DESCRIPTION:

HOME PAGE:

Here user view the home page of Modelling Human Driving Behaviour Through GAIL Using Machine Learning Approach



figure 8.1: Home Page

SERVER:

Here we can connect the server



figure 8.2: server

SERVER MENU:

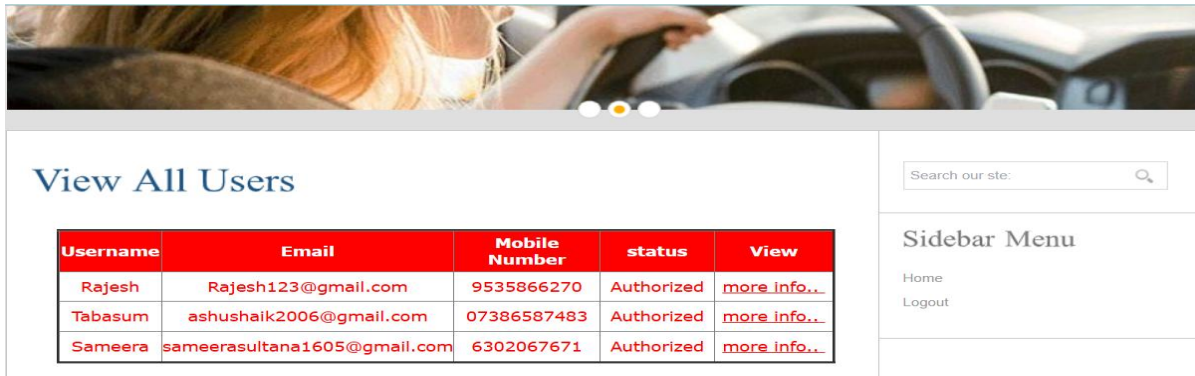
In the page, we can see the contents of Server Menu



figure 8.3: Server Menu

VIEW ALL USERS:

Here we can see all the users who are logged into this application



Username	Email	Mobile Number	status	View
Rajesh	Rajesh123@gmail.com	9535866270	Authorized	more info...
Tabasum	ashushaik2006@gmail.com	07386587483	Authorized	more info...
Sameera	sameerasultana1605@gmail.com	6302067671	Authorized	more info...

figure 8.4: All Users

VIEW ALL DATASETS:

Here we can see all the databases used in this project.

idnumber	City Location	day	Sex	Age	Time	Day of week	Educational level	Vehicle driver relation	Driving experience	Type of vehicle	Owner of vehicle	St v w
ti0avcqy	Delhi	08-Feb-2020	M	34	31-Dec-1899	Saturday	Elementary school	Employee	Below 1yr	Other	Owner	Un
dd98qb6d	New Delhi	09-Feb-2020	F	47	31-Dec-1899	Saturday	Elementary school	Employee	5-10yr	Lorry	Owner	Un
xfwojl20	Mumbai	10-Feb-2020	F	52	31-Dec-1899	Saturday	Elementary school	Owner	5-10yr	Public	Owner	Un
b9kf2s7w	Gurugram	11-Feb-2020	M	46	31-Dec-1899	Wednesday	Elementary school	Employee	2-5yr	Automobile	Governmental	2
i36fuou9	Nagpur	12-Feb-2020	F	58	31-Dec-1899	Wednesday	Elementary school	Employee	Above 10yr	Pick up upto 10Q	Governmental	
rxhygr8x	Indore	13-Feb-2020	M	58	31-Dec-1899	Friday	Elementary school	Employee	2-5yr	Lorry	Owner	Un

figure 8.5: View All DataSets

DIFFERENT DRIVING STYLES:

This page shows the count of people following different driving styles

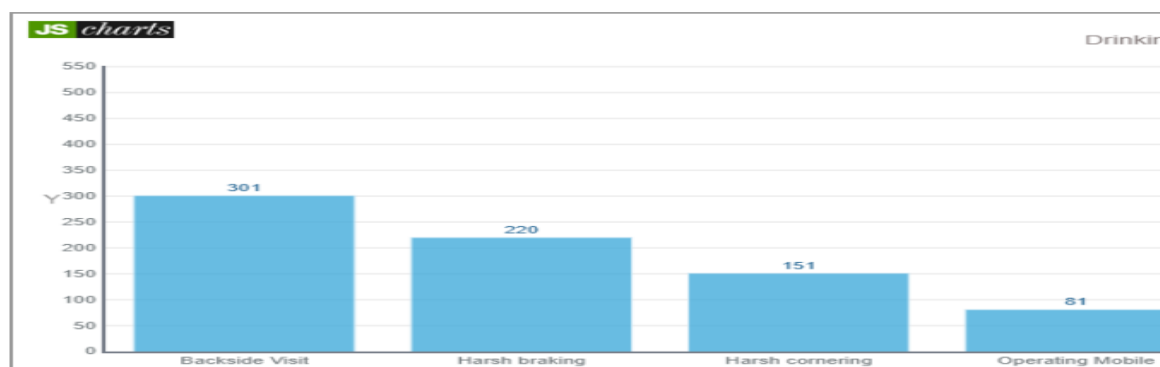


figure 8.6: Different driving styles

AGE COUNT RESULT:

This page show the age of persons who drive different vehicles.

[View All Age Count Results !!!](#)

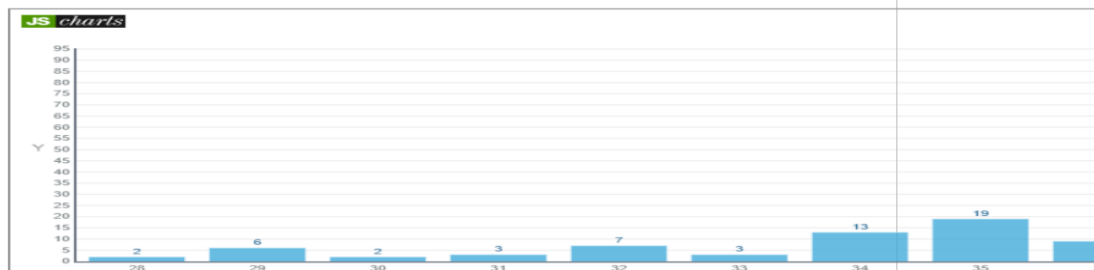


figure 8.7: Age Count Result

VEHICLE COUNT RESULT:

This page show the count of different vehicles that runs on the highway. Here the bar with no label describes unknown vehicles.

[View Vehicle Type Results](#)

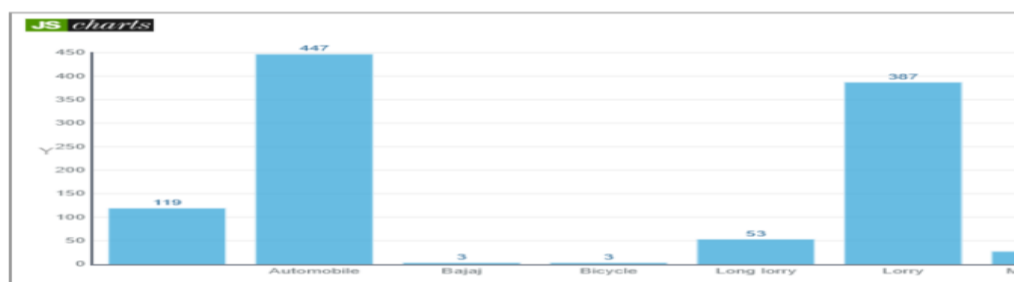



figure 8.8: Vehicle count Result

USER LOGIN PAGE:

This is the user login page

Welcome to User Login

Note : * = Required



Username *

Password *

[New user? Register](#)

Sidebar Menu


- Home
- User
- Server

figure 8.9: User Login Page


USER PROFILE:

This page shows the user details such as name , email, mobile number etc

User Profile Details



Name	Tabasum
E-Mail	ashushaik2006@gmail.com
Mobile	07386587483
Date Of Birth	07-06-2004
Status	Authorized
Address	Lakshmi hostel near annamacharya arch boyinapalli



Sidebar Menu

- Home
- Logout

[Back](#)

figure 8.10: User Profile


UPLOAD DATASETS:

In this page we can upload the dataset

Upload Datasets !!!

Datasets.xls

[Back](#)



Sidebar Menu

- Home
- Logout

[Back](#)

figure 8.11: Upload datasets

FINDING DRIVING TYPE STYLES:

This page takes an entire record of an individual user from database and gives the driving type result

Find Driving Type Results...!!!

Enter RI Number	00avcdy
Enter City Location	Delhi
Enter Day	08-02-2020
Select Sex	M
Enter Age	34
Enter Time	01:15:00
Enter Day Of Week	Saturday
Enter Education Level	Elementary school
Enter Vehicle class category	Owner
Enter Driving experience	Below 1yr
Enter Type of vehicle	Other

Find Driving Type

Search our site:

User Menu

- Home
- Log Out

figure 8.12: Find Driving Type Results

DRIVING TYPE RESULTS:

This page shows the driving style of the user. Here the user driving style is driving by operating mobile.



Driving Type Results !!!

Driving Type Results Type
Operating Mobile

Search our site:

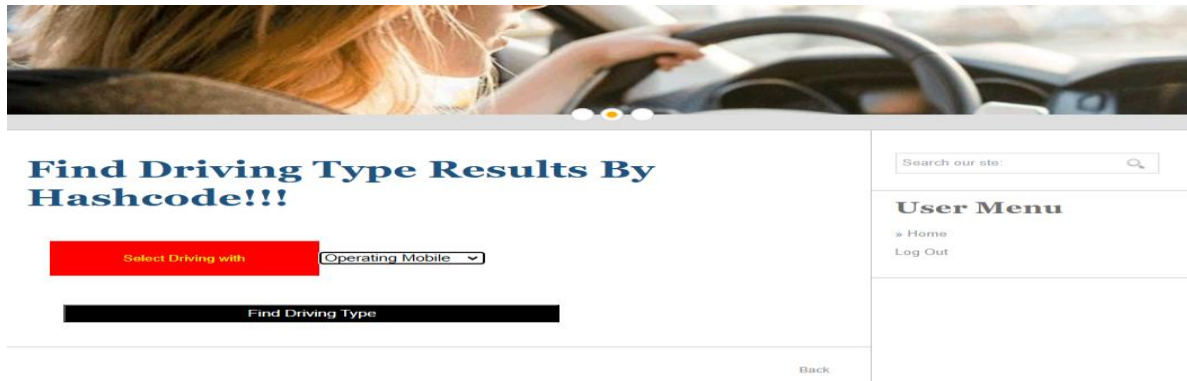
User Menu

- Home
- Log Out

figure 8.13: Driving Type Results

FINDING DRIVING TYPE RESULTS BY HASHCODE:

This page shows the results of driving by using hashcodes.



The screenshot shows a web application interface. At the top, there is a header image of a person driving. Below the image, the title 'Find Driving Type Results By Hashcode!!!' is displayed in blue. There is a red button labeled 'Select Driving with' and a dropdown menu currently showing 'Operating Mobile'. Below these is a black button labeled 'Find Driving Type'. On the right side, there is a search bar with the placeholder text 'Search our site:' and a magnifying glass icon. Below the search bar is a 'User Menu' section with links for 'Home' and 'Log Out'. At the bottom right, there is a 'Back' link.

figure 8.14: Finding Driving Type Results By Hashcode

HASHCODERESULTS:

Here all the driving results that are determined using hash codes are displayed.

Driving Type Results By HashCode!!!

Idnumber	Driving Type
ti0avcqy	Operating Mobile
sl4u8e6e	Operating Mobile
r4ht266w	Operating Mobile
hlsvu1cg	Operating Mobile
42zbu6hv	Operating Mobile
se7l2mki	Operating Mobile

figure 8.15: Hashcode results

CHAPTER 9

CONCLUSION AND FUTURE ENHANCEMENTS

9.CONCLUSION AND FUTURE ENHANCEMENT

To validate the safety of autonomous driving algorithms, consistent human driving models are needed. Human driving is modelled in this work as a sequential decision-making issue under uncertainty. It is a non-linear, stochastic, continuously state and action space issue with an uncertain cost function. When the price of a function is unclear or difficult to specify, learning from examples is a compelling approach to solve MDPs. GAIL was presented with the promise of (at least in theory) precise imitation even for situations involving huge (perhaps continuous) state & action spaces, following a series of studies on inverse reinforcement learning. With indefinite state and activity areas non-linear and stochastic policies, and a cost function that is difficult to precisely characterize, driver modelling is a challenging task. These characteristics make it suitable for development.

However, human demonstrations depend on latent causes of variability that GAIL cannot discover on its own. Additionally, GAIL cannot directly solve driver modelling because it is a multi-agent problem. Three GAIL extensions that addressed these constraints were described in this study. It began by outlining PS-GAIL, which considered the multi-agent nature of the issue that results from interactions amongst traffic participants. Second, it introduced RAIL, which uses incentive augmentation to provide the driver modelling agent domain knowledge about traffic laws.

Third, it introduced Burn-Info GAIL, which tackles the problem of latent variability disentanglement in demonstrations. Experiments with driver modelling, such as training driver behaviour models from actual driving demonstration data, were used to demonstrate all three extensions. Future research will focus on using learned driver models and improving model performance. The performance of models can be improved in a number of ways, such as (i) employing coordinating graphs [68], [69], and graph neural network models [67] to centrally make agent interactions explicit.(ii) examining recently suggested substitute imitation learning techniques [70], and (iii) expanding tests to more extensive driving data and domains. We expect to employ these models for the future, and the learning process of human driver models will eventually be used to validate autonomous vehicles in simulation.

BIBLIOGRAPHY

- [1] J. Colyar and J. Halkias, “US highway 101 dataset,” Federal Highway Administration (FHWA), Tech. Rep. FHWA-HRT-07-030, 2007.
- [2] F. Altché and A. de La Fortelle, “An lstm network for highway trajectory prediction,” in IEEE International Conference on Intelligent Transportation Systems (ITSC), 2017.
- [3] N. Deo, A. Rangesh, and M. Trivedi, “How would surround vehicles move? a unified framework for maneuver classification and motion prediction,” IEEE Transactions on Intelligent Vehicles, vol. 3, no. 2, pp. 129–140, 2018.
- [4] B. Kim, C. M. Kang, J. Kim, S. H. Lee, C. C. Chung, and J. W. Choi, “Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network,” in IEEE International Conference on Intelligent Transportation Systems (ITSC), 2017.
- [5] R. Krajewski, T. Moers, A. Meister, and L. Eckstein, “Béziervae: Improved trajectory modeling using variational autoencoders for the safety validation of highly automated vehicles,” in IEEE Intelligent Transportation Systems Conference (ITSC), 2019.
- [6] S. Yoon and D. Kum, “The multilayer perceptron approach to lateral motion prediction of surrounding vehicles for autonomous vehicles,” in IEEE Intelligent Vehicles Symposium (IV), 2016.
- [7] A. Armand, D. Filliat, and J. Ibanez-Guzman, “Modelling stop intersection approaches using gaussian processes,” in IEEE International Conference on Intelligent Transportation Systems (ITSC), 2013.
- [8] J. Wiest, M. Höffken, U. Kreßel, and K. Dietmayer, “Probabilistic trajectory prediction with gaussian mixture models,” in IEEE Intelligent Vehicles Symposium (IV), 2012.
- [9] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, “Social lstm: Human trajectory prediction in crowded spaces,” in IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
- [10] N. Rhinehart, R. McAllister, K. Kitani, and S. Levine, “Precog: Prediction conditioned on goals in visual multi-agent settings,” in IEEE/CVF International Conference on Computer Vision (ICCV), 2019.

- [11] S. Suo, S. Regalado, S. Casas, and R. Urtasun, “TrafficSim: Learning to simulate realistic multi-agent behaviors,” in IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2021.
- [12] C. Tang and R. R. Salakhutdinov, “Multiple futures prediction,” in Advances in Neural Information Processing Systems (NeurIPS), 2019.
- [13] S. Pruekprasert, X. Zhang, J. Dubut, C. Huang, and M. Kishida, “Decision making for autonomous vehicles at unsignalized intersection in presence of malicious vehicles,” in IEEE Intelligent Transportation Systems Conference (ITSC), 2019.
- [14] H. Okuda, K. Harada, T. Suzuki, S. Saigo, and S. Inoue, “Design of automated merging control by minimizing decision entropy of drivers on main lane,” in IEEE Intelligent Vehicles Symposium (IV), 2017.
- [15] D. Isele, “Interactive decision making for autonomous vehicles in dense traffic,” in IEEE Intelligent Transportation Systems Conference (ITSC), 2019.
- [16] J. Fisac, E. Bronstein, E. Stefansson, D. Sadigh, S. S. Sastry, and A. Dragan, “Hierarchical game-theoretic planning for autonomous vehicles,” in International Conference on Robotics and Automation (ICRA), 2019.
- [17] D. S. González, M. Garzón, J. S. Dibangoye, and C. Laugier, “Human-like decision-making for automated driving in highways,” in IEEE Intelligent Transportation Systems Conference (ITSC), 2019.
- [18] P. Abbeel and A. Y. Ng, “Apprenticeship learning via inverse reinforcement learning,” in International Conference on Machine Learning (ICML), 2004.
- [19] P. G. Gipps, “A behavioural car-following model for computer simulation,” *Transportation Research Part B: Methodological*, vol. 15, no. 2, pp. 105–111, 1981.
- [20] A. Kuefler, J. Morton, T. Wheeler, and M. Kochenderfer, “Imitating driver behavior with generative adversarial networks,” in Intelligent Vehicles Symposium (IV), 2017.

ORIGINALITY REPORT

21 %
SIMILARITY INDEX

14 %
INTERNET SOURCES

7 %
PUBLICATIONS

11 %
STUDENT PAPERS

PRIMARY SOURCES

1 www.slideshare.net 3 %
Internet Source

2 Rudra Kumar Madapuri, P. C. Senthil Mahesh.
"HBS-CRA: scaling impact of change request
towards fault proneness: defining a heuristic
and biases scale (HBS) of change request
artifacts (CRA)", Cluster Computing, 2017 2 %
Publication

3 arxiv.org 2 %
Internet Source

4 Submitted to University of Hertfordshire 1 %
Student Paper

5 Raunak Bhattacharyya, Blake Wulfe, Derek J.
Phillips, Alex Kuefler, Jeremy Morton, Ransalu
Senanayake, Mykel J. Kochenderfer.
"Modeling Human Driving Behavior Through
Generative Adversarial Imitation Learning",
IEEE Transactions on Intelligent
Transportation Systems, 2023 1 %
Publication

6 www.coursehero.com 1 %
Internet Source

7 docshare.tips 1 %
Internet Source

8 Submitted to Birla Institute of Technology and
Science Pilani 1 %
Student Paper

Submitted to Chicago State University

1st INTERNATIONAL CONFERENCE
ON
**ADVANCEMENT IN VLSI AND
SIGNAL PROCESSING**
(VMTW - ICAVSP 2K24)

25th & 26th November, 2024



VIGNAN'S

Institute of Management and Technology for Women

An Autonomous Institution

(Sponsored by Laxmi Educational Society, Affiliated to JNTU & Approved by AICTE, New Delhi)
Kondapur(V), Ghattieswar (M), Medchal Mulbagal (DL), Telangana - 501301

Ph: 96529 10002/3 | www.vmtw.in



Certificate

This is to Certify that, Dr. / Mr. / Ms. S. Tabasum
of Annamacharya University, Rajampet
has presented/participated a paper in the **1st International Conference on Advance-
ment in VLSI and Signal Processing** organized by the **Department of Electronics
and Communication Engineering** at Vignan's Institute of Management and Tech-
nology for Women, Kondapur (V), Hyderabad, Medchal Dist-501301, Telangana, India.,
held on 25th & 26th November, 2024.

Title of the Paper : Gail For Modeling Human
Driving Behaviour

Dr. VIJAYKUMAR R. URKUDE
CONVENER-2

Dr. SK. MASTHAN BASHA
CONVENER-1

Mr. P. HARIKRISHNA
HOD-ECE

Dr. G. APPARAO NAIDU
PRINCIPAL



1st INTERNATIONAL CONFERENCE
ON
**ADVANCEMENT IN VLSI AND
SIGNAL PROCESSING**
(VMTW - ICAVSP 2K24)

25th & 26th November, 2024



VIGNAN'S

Institute of Management and Technology for Women

An Autonomous Institution

(Sponsored by Laxmi Educational Society, Affiliated to JNTU-H & Approved by AICTE, New Delhi)
Kondapur(V), Chubbasa (M), Medchal Malkajgiri (DL), Telangana - 501301

Ph: 96529 10002/3 | www.vmtw.in

Certificate

This is to Certify that, Dr. / Mr. / Ms. S. Sameera Sultana
of Annamacharya University, Rajampet
has presented/participated a paper in the **1st International Conference on Advance-
ment in VLSI and Signal Processing** organized by the **Department of Electronics
and Communication Engineering** at Vignan's Institute of Management and Tech-
nology for Women, Kondapur (V), Hyderabad, Medchal Dist-501301, Telangana, India.,
held on 25th & 26th November, 2024.

Title of the Paper : Yail For Modelling Human
Driving Behaviour

Dr. VIJAYKUMAR R. URKUDE
CONVENER-2

Dr. SK. MASTHAN BASHA
CONVENER-1

Mr. P. HARIKRISHNA
HOD-ECE

Dr. G. APPARAO NAIDU
PRINCIPAL

1st INTERNATIONAL CONFERENCE
ON
**ADVANCEMENT IN VLSI AND
SIGNAL PROCESSING**
(VMTW - ICAVSP 2K24)

25th & 26th November, 2024



VIGNAN'S

Institute of Management and Technology for Women

An Autonomous Institution

(Sponsored by Laxmi Educational Society, Affiliated to JNTUH & Approved by AICTE, New Delhi)
Kondapur(V), Ghatkesar (M), Medchal-Mulka(giri) (D), Telangana - 501301

Ph: 96529 10002/3 | www.vmtw.in



Certificate

This is to Certify that, Dr. / Mr. / Ms. M. Sunil Kumar Goud
of Annamacharya University, Rajampet
has presented/participated a paper in the 1st International Conference on Advance-
ment in VLSI and Signal Processing organized by the Department of Electronics
and Communication Engineering at Vignan's Institute of Management and Tech-
nology for Women, Kondapur (V), Hyderabad, Medchal Dist-501301, Telangana, India.,
held on 25th & 26th November, 2024.

Title of the Paper : Yail For Modelling Human
Driving Behaviour

Dr. VIJAYKUMAR R. URKUDE

CONVENER-2

Dr. SK. MASTHAN BASHA

CONVENER-1

Mr. P. HARIKRISHNA

HOD-ECE

Dr. G. APPARAO NAIDU

PRINCIPAL

1st INTERNATIONAL CONFERENCE
ON
**ADVANCEMENT IN VLSI AND
SIGNAL PROCESSING**
(VMTW - ICAVSP 2K24)

25th & 26th November, 2024



VIGNAN'S

Institute of Management and Technology for Women

An Autonomous Institution

(Sponsored by Laxmi Educational Society, Affiliated to JNTUH & Approved by AICTE, New Delhi)
Kondapur(V), Ghatkesar (M), Medchal-Malkajgiri (D), Telangana - 501301

Ph: 96529 10002/3 | www.vmtw.in



Certificate

This is to Certify that, Dr. / Mr. / Ms. C. Venkata Bharadwaja
of Annamacharya University, Rajampet
has presented/participated a paper in the 1st International Conference on Advance-
ment in VLSI and Signal Processing organized by the Department of Electronics
and Communication Engineering at Vignan's Institute of Management and Tech-
nology for Women, Kondapur (V), Hyderabad, Medchal Dist-501301, Telangana, India.,
held on 25th & 26th November, 2024.

Title of the Paper : Yail For Modelling Human
Driving Behaviour

Dr. VIJAYKUMAR R. URKUDE
CONVENER-2

Dr. SK. MASTHAN BASHA
CONVENER-1

Mr. P. HARIKRISHNA
HOD-ECE

Dr. G. APPARAO NAIDU
PRINCIPAL