

MOTO-MAKEOVER

Capstone Project Report

Submitted by:

102116028 Diwi Malik

102116102 Raghav Sharma

102116112 Sanchit Raj

102116097 Kushal Srivastava

102116060 Samridhi Wadhwa

BE Fourth Year, CSE

CPG No: 148

Under the Mentorship of

Dr. Deep Mann

Dr. Simran Setia



Computer Science and Engineering Department
Thapar Institute of Engineering and Technology, Patiala

December 2024

ABSTRACT

The MOTO-MAKEOVER project aims to innovate within the car customization industry by developing an interactive application using Unity, a leading game development platform. This app empowers users by providing detailed 3D models of various vehicles that can be customized with a wide range of options, including paint jobs, decals, rims, spoilers, and more.

Unlike traditional car customization methods, which can be expensive and time-consuming, MOTO-MAKEOVER offers a virtual platform where users can visualize modifications in real-time, ensuring that the final appearance aligns perfectly with their expectations before any physical changes are made. The app features an extensive catalog of customizable products, each meticulously modeled for seamless integration with the vehicle models provided within the app.

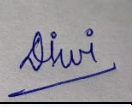



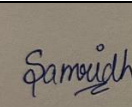
In addition to its practical customization tools, MOTO-MAKEOVER serves as an educational resource by linking each customization option to detailed product information, installation guides, and educational content. This approach not only enhances user experience but also fosters a deeper understanding of automotive customization processes and product applications.

By combining cutting-edge technology with an intuitive user interface, MOTO-MAKEOVER is positioned to revolutionize the way car enthusiasts engage with automotive customization, making it accessible.

DECLARATION

We hereby declare that the design principles and working prototype model of the project entitled MOTO-MAKEOVER is an authentic record of our own work carried out in the Computer Science and Engineering Department, TIET, Patiala, under the guidance of Dr Deep Mann and Dr Simran Setia during 6th semester (2024)

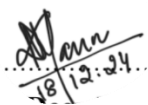
Date:

Roll No.	Name	Signature
102116028	Diwi Malik	
102116102	Raghav Sharma	
102116112	Sanchit Raj	
102116097	Kushal Srivastava	
102116060	Samridhi Wadhwa	

Counter Signed By:

Faculty Mentor:

Co-Mentor:



Dr. Deep Mann



Dr. Simran Setia

ACKNOWLEDGEMENT

We wish to acknowledge the support that has been received from our mentors Dr. Deep Mann and Dr. Simran Setia. They have been invaluable to our project, keeping us an almost endless supply of technical know-how. They have been truly great mentors throughout this entire process.

Last but not least, we also owe thanks to Dr. Shalini Batra, Head, Computer Science and Engineering, and the entire faculty and staff in the department of Computer Science and Engineering. Also, to our friends who sacrificed so much of their valuable time to help us in all possible ways towards successful completion of this project. Thank you to all the people who have been involved in one or the other manner in this project.

In fine, we would also like to thank the families of ours for their unwavering love and support. They have always wanted the best for us and we admire their determination and sacrifice.

Date:

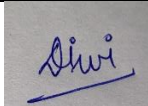

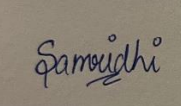


Roll No.	Name	Signature
102116028	Diwi Malik	
102116102	Raghav Sharma	
102116060	Samridhi Wadhwa	
102116112	Sanchit Raj	
102116097	Kushal Srivastava	

TABLE OF CONTENT

TABLE OF CONTENTS	PAGE NO.
ABSTRACT	2
DECLARATION	3
ACKNOWLEDGEMENT	4
TABLE OF CONTENTS	5
LIST OF TABLES	7
LIST OF FIGURES	8
1. Introduction	9
1.1 Project Overview	9
1.2 Need Analysis	9
1.3 Research Gaps	10
1.4 Problem Definition and Scope	11
1.5 Assumptions and Constraints	12
1.6 Standards	13
1.7 Approved Objectives	14
1.8 Methodology	14
1.9 Project Outcomes and Deliverables	15
1.10 Novelty of Work	16
2. Requirement Analysis	17
2.1 Literature Survey	17
2.1.A Enhanced User Experience	18
2.1.B Improved Design and prototyping	19
2.1.C Efficient Manufacturing	19
2.1.D Training and Education	19
2.1.E Marketing and sales	20
2.1F key Learning	20
2.1G Future Research Direction	21
2.1H Existing Project	21
2.2 Software Requirement Specification	21
2.2.A Introduction	21
2.2.B Overall Description	22
2.2.C External Interface Requirements	23
2.2.D Other Non-functional Requirements	24
2.3 Cost Analysis	24
2.4 Risk Analysis	24
3. Methodology Adopted	25

3.1 Investigate Technique	25
3.1.A Research Techniques	25
3.1.B Design and Implementation	25
3.1.C Backend and Real Time Features	25
3.2 Proposed Solution	26
3.3 Work Breakdown Structure	27
3.4 Tools and Technology	27
4. Design Specifications	28
4.1 System Architecture	28
4.1.A Block Diagram	28
4.1.B Component Diagram	29
4.1.C MVC Architecture	30
4.2 Design Level Diagrams	31
4.2.A Use Case Diagram	31
4.2.B Use Case Template	32
4.2.C Swimlane Diagram	34
4.2.D Class Diagram	35
4.2.E ER Diagram	36
4.2.F Data Flow Diagram	39
5. Implementation and Experimental Results	42
5.1 Procedural Flow	42
5.2 Working Project	43
5.3 Testing Process	47
5.3.1 Black Box Testing: Specific Car UI	47
5.3.2 White Box Testing	49
5.4 Algorithmic Approaches Used	50
6. Conclusions and Future Scope	59
6.1 Work Accomplished	59
6.2 Conclusion	59
6.3 Benefits	61
6.4 Future Scope	61
7. Project Metrics	62
7.1 Challenges Faced	62
7.2 Relevant Subjects	62
7.3 Peer Assessment Matrix	62
7.4 Role Playing and Work Schedule	63
7.5 Students Outcome Description and Performance Indicators	64
7.6 Brief Analytical Assessment	64
Appendix A: References	65
Appendix B: Plagiarism Report	66

LIST OF TABLES

Table	Page No.
Sample Assumptions	12
Sample Constraints	12
Literature Survey	17
Use case Template	32
Black Box Testing	48
White Box Testing	49
Peer Assessment Matrix	54
Role Playing and Work Schedule	55
Student Outcomes Description and Performance Indicators	56

LIST OF FIGURES

Figure	Page No.
Figure 4.1(a): Block Diagram	28
Figure 4.1(b): Component Diagram	29
Figure 4.1(c): MVC Architecture	30
Figure 4.2(a): Use Case Diagram	31
Figure 4.2(b): Swimlane Diagram	34
Figure 4.2(c): Class Diagram	35
Figure 4.2(d): E-R Diagram	36
Figure 4.2(e): DFD	39
Figure 5.1 (a): Working Flow	42
Figure 5.2(a): Opening screen	43
Figure 5.2(b): Tyre customization	44
Figure 5.2(c): Color customization	44
Figure 5.2(d): Headlight customization	45
Figure 5.2(e): Spoiler, taillight, indicator customization	46
Figure 5.2(f): Interior customization	46
Figure 5.2(g): Seat cover customization	47

INTRODUCTION

1.1 Project Overview

The project has an entirely new approach to the car customization segment through the game development platform Unity. Users can alter and customize any of the detailed 3D models the app offers. With this new way, alterations come as real-time virtual yet real-life detail that make the users totally at ease that the finished work will come out as their imagination.

The application is going to have in its kit of tools an endless array of customization accessories, spoilers, and all. Each of which will include its separate but uniquely made-from-scratch 3D configurations that will attach themselves seamlessly with the 3D models available within the application.

Every customization product would be linked with photographs, specification analysis, and educational content. The scope of this has majorly raised the possibilities of providing the users the education that the users are interested in learning the fun aspects of the main subjects concerned with car customization, products, and applications with the input from this budding application.

1.2 Need Analysis

This project revolutionizes car customization through unity by modifying a real-time detailed 3D image model. It provides tools for modification, with specific accessories made in 3D for integration in the application. Providing pictures, specifications, and educational content on this app will go a long way in ensuring that users know and love the art of customizing a car. The app merges the teaching of creativity with learning.

A. Product Suitability:

- The fact that it has a very much extensive catalog of customization options means that users can try out many different styles and changes before making the physical change. This will also determine which products really suit their car considering such aspects as appearance, functionality, and compatibility.

B. Educational Value

- The app will work as an educational tool, including the details of customizing products like specifications, installation guide, and learning materials concerning the scope of application. This will not only inform the user on the product but educate the process of customizing and its elements, thus making a fuller understanding and appreciation of the craft of car customization.

1.3 Research Gap

A. Integration of Educational Content: It was purposeful to aim with educative content attached to the customization products but it is a gap to find out the usefulness of such integration. Research should find out how users consume and relate to such materials in that customization app and whether or not that approach works in enhancing their knowledge or thereby in the decision-making process.

B. Real-time Visualization Accuracy: The project realizes its dependability on real-time visualization through modifications that are perfect and according to user expectations. There is, however, a gap in regard to validating the accuracy as well as reliability that these visualizations represent modifications as accurate and all times. Future research would thus compare virtual customizations with real world outcomes to ensure fidelity.

C. Scalability and Performance: The application may be just fine with scaling and performance when the number of models and selection options becomes larger. This will, however, provide good coverage against possible problems such as loading time, rendering quality, and system requirement by research optimizing unity for such massive applications.

D. Market Adoption and Commercial Viability: The project augurs well in revolutionizing the customization industry for cars; however, little has been done toward the market acceptance and commercial viability of such applications. Understand the market needs and key barriers to adoption and further innovate development and marketing strategy.

1.4 Problem Definition and scope

The following outlines the specific problems addressed and the scope of MOTO-Makeover.

A. Problem Definition

Since the inception of custom car designing, the whole process has involved literally changing parts of the car and trying to alter by flipping options in the head of the user before making the ultimate change to the car such that there is always a big gap between the expectation it raises and what actually ends up being the case. Thus, it is highly disappointing and costs very much. And the other custom animation tools are either quite simple or very complicated making them useless without having proper training. So, there should an online platform which is user-friendly enough for everyone to experiment, visualize and put the customizing of an object, such as a car model, in a virtual space before actualizing the final product accordingly to vision of how it is supposed to look.

B. Scope

It is this shortcoming that the current project seeks redress through sponsoring, development of a next-generation car customization application through Unity, a world-leading gaming platform. The extent of the project includes:

- i.**3D Car Models:** The app would allow more 3D models of different cars which can be selected and customized with various options.
- ii.**Real-time Customization:** The developed app would allow users to visualize the modifications made on the real-time basis and hence reflecting the changes within the 3D model immediately. The user would have an experience on how a particular option affects everything before finalizing it.
- iii.**Extensive Customization Catalog:** The app would provide an exhaustive catalog for customization type along with educational material. This will not help users make a proper decision but also create a useful data source for the automotive.
- iv.**User-Friendly Interface:** The application is designed with aforementioned features primarily for the accessibility such that it grants easy access to both novices and experienced users through the process of achieving customizability.
- v.**Educational Content:** Educational content will help the users learn the customization process with the different available products, thereby improving their knowledge and engagement with the app.

vi.**Admin Management:** The app will ensure that the management of admins will monitor the catalog of car models and their related customization products so that the content kept there would be updated and relevant. This is mainly meant to provide a wholesome interactive platform for car customization, which would cater to the difference between the user expectations and final outcome.

Bridging the gap towards making the industry wait revolutionary as well as touchier, educational customization is.

1.5 Assumptions and Constraints

A. Assumptions

These are the assumptions underlying the development and deployment of MOTO-MAKEOVER:

S. No.	Sample Assumptions
1	The development and deployment of the car customization application, along with support for 3D modeling, is feasible with the Unity engine.
2	The project assumes that the catalog of customization options will be exhaustive and according to the user's preferences for the better user experience.
3	It is assumed that all educational contents that are being put in the application (for example, installation guides, product information) would be helpful for the users.
4	All necessary models will probably be available as 3D models, whether free or not.

Table 1: Sample Assumption

B. Constraints

The following are the constraints that must be considered during the implementation of MOTO-Makeover:

S. No.	Sample Constraints
1	The application can run on various laptops according to the processing power and graphics functions they have, which restrain that pre-made 3D models should run hitch-free and provide high quality.
2	Yet another complicated task is deriving a design interface which is as simple and user-friendly as possible to manage all that complexity in customizations without making it difficult for the users.
3	Use of prefabricated 3D models and different types of customizations should be in accordance with licensing agreement and laws on intellectual property. This is one very significant constraint for ensuring

	that all the assets are legally acquired and used in the application.
4	The regular update and maintenance of the extensive catalog of customization options within the app is yet another challenge. The system should accommodate updates on a very frequent basis to ensure that the most recent models and customization features are available for the users.

Table 2: Sample Constraints

1.6 Standards

3D Modelling and Rendering Standards:

A. FBX/OBJ File Formats:

The application will rely on the common standardized 3D file format of FBX and OBJ for the import/export of car models and customization products between platforms, most importantly, to ensure the fidelity of creating these very detailed 3D models in the application.

B. User Interface (UI) Standards:

The user-friendly responsive interface allows easy navigation visibility of 3D model and customization options into the app as per unity best practices, whereby this is allowed as per the best practices in developing interface in Unity.

C. Educational Content Standards:

Every piece of educational content related to every customization product will meet standards on accuracy and relevance such that users will get the real information maximizing their understanding of customizing and available products.

D. Cross-Platform Development Standards:

Unity has been used for cross-platform optimization, so the application should work as perfectly among the different platforms, enabling everyone to enjoy a high-quality experience from a device of choice.

This, thus, guarantees a seamless experience through its extremely accurate, highly-defined, top-quality 3D models, coupled with informative and educative content in the MOTO-MAKEOVER application.

1.7 Approved Objectives

- A. Develop and incorporate educational materials with every customized product in order to increase the user's knowledge and enhance decision-making.
- B. Show that the system of real-time visualizations does not deviate from the actual dwelling results through correct rendering algorithms.
- C. Design application architecture to optimize handling large-scale 3D models or customizations necessary for meeting user demand, in particular improving Unity performance to address key aspects like rendering quality, load times, and system requirements.
- D. To include educational content that explains to users about the whole process of customizing and importance of each modification as well as care related to customized parts.

1.8 Methodology

- Mastery and Learning of Unity: This is quite encompassing, take your time to know the ins and outs of the 3D modeling features of Unity to create user-friendly interfaces, as it is a greater part for application.
- Customization Catalog Development: Generate or curate a variety of models, detailed, scalable, and optimized for use in the Unity environment by the application, that meet all of the above criteria.
- Testing and Interactive Tools: develop flexible tools for stringing custom products onto car models; followed and rigorous testing procedure for functionality, performance and user-friendly across devices.
- Iterative Improvement: Feedback from the tests will be used to improve the application, fix bugs, enhance performance and finally perfect the design of a top-notch user-ready product for the car customization market.

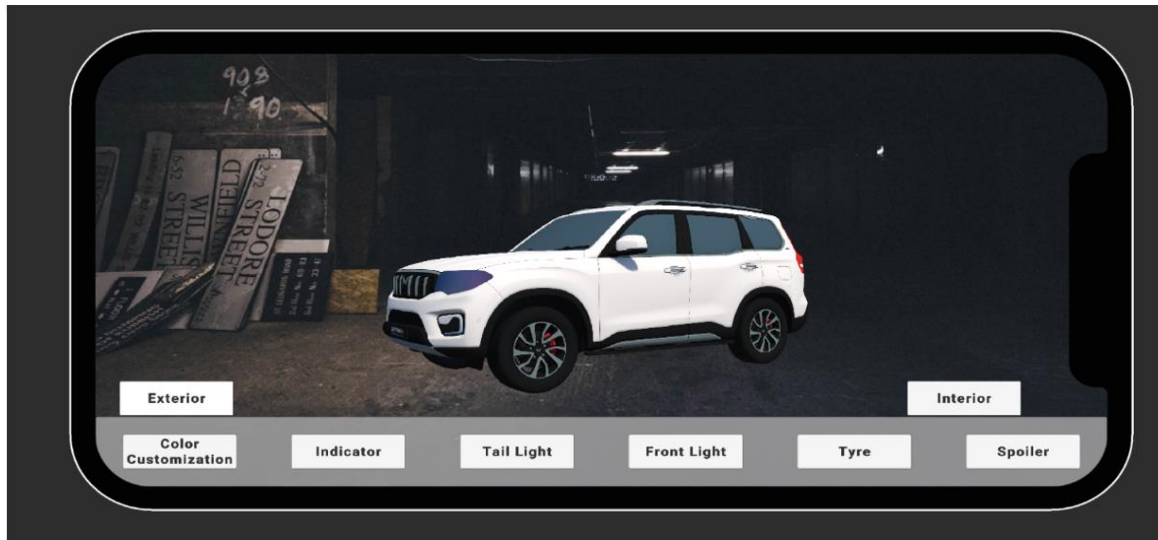


Figure 1.8(a)

1.9 Project Outcomes and Deliverables

- **Enhanced User Experience:** It immerses the users in real-time visualization of car modifications with a more interactive and satisfying process of customization.
- **Accurate Customization:** The application provides the user with detailed 3D models of vehicles to provide that all alterations are done towards all the precise visions the users have, resulting in perfect and personalized customizations.
- **Extensive Customization Options:** Users can customize it in large ways and such a wide range of customization enables customizing users with adding in tons of different spoilers, among other things, so it all could be a little bit more personalized in the actual modification of their vehicles.
- **Educational Resource:** Features a thorough description, specification, and installation instruction for every custom product that gives a source of knowledge to the users regarding the customization and the use of these products.
- **Promotion of Automotive Education:** Application would infuse high-tech with interactive learning experience as it would walk the users deeply into the fascinating world of car customization to develop a greater sense of appreciation of the craft.
- **Industry Revolutionization:** This approach of the application regarding car customizing can

be a revolution in the automotive world where new records will be set into user engagement, customization, and resource education.

- **Increased User Engagement:** The immersive and interactive experience that this feature would give will definitely expect to see higher retention rates once established among the enthusiast and professional car users alike.

1.10 Novelty of Work

- The MOTO-MAKEOVER project introduces a completely new concept in car customization by enriching it with advanced 3D modeling and real-time visualization within an application, which is powered by the Unity game development platform. Unlike other similar tools, very few make it possible for a person to experience realistic personalization while creating aspects of a Car's Design virtually down to the smallest details. These often-bound changes remain either superficial or require specialized change programs.

- The novelty of this project lies in its joining of exquisitely detailed, customizable 3-D cars with broad-access interactivity of an extensive catalog of modifications to one convenient, user-friendly interface on desktop device. This app would not only make it easy to customize but also democratize it to high-concept automotive design as applicable to broader audiences, including enthusiasts at the casual end and professionals at the high end.

- By introducing high technology in first-of-its-kind educational methodology, MOTO-MAKEOVER creates a fresh avenue for its future opening to the automotive industry. It will be a groundbreaking step to set a new standard in digital auto makeover.

REQUIREMENT ANALYSIS

2.1 Literature Survey

By reviewing the literature, spurs increased interest in the dimension of car customization to middle-income earners: in-depth and detailed demand for personalized experiences. This survey of the literature explores a feasibility study for a Unity-based application that enables consumers to customize existing pre-created 3D cars from economy-friendly manufacturers. This report, which includes reviews giving existing research on 3D development in cars, anticipates mapping out the major functionalities, possible and future directions of an accessible vehicle customization application.

S. No	Roll No.	Name	Paper Title	Tools/ Technology	Finding	Citation
1	102116028	Diwi	Using Augmented Reality technology to Support the Automobile Development	AR Toolkit, Head-Mounted Displays, 3D Modelling Software	Integration of technology for high interaction in car customization.	Jürgen F, Jürgen G, Carsten M, and Rafael R., MAY 2004
2			Cooperative Design Support within Automobile Advance Development Using Augmented Reality Technology	AR Technology	Expansion of 3D car models to affordable, middle-class consumers.	Friind J, Gausemeier J, Matysczok C, Radkowski R., May 2004
3	102116060	Samridhi Wadhwa	Vehicle Customization using Augmented Reality	AR SDK (Software Development Kit), Unity or Unreal Engine	AR emphasizes easy manufacturability of customization options.	Deep M, Tejas P, Bhavesh P, Abhishek S, Quazi TZ, Chetan T., Volume-04, Issue-11,

						November-2022
4			The Exploration of Customization in Augmented Reality in Automotive	Unity, AR Hardware	Enables practical and aesthetic customization in consumer apps	Xiaoyu A, Qingdan J, Syed M., SEP 2019
5	102116097	Kushal Srivastava	Application of Virtual and Augmented Reality in Automotive	VR Headsets, Autodesk MAYA, Blender	Project focuses on augmenting reality for complex technical products.	Zdeněk C, Gabriel F, Nikoleta M., NOV 2019
6			Application of Augmented Reality in Automotive Industry	Augmented Reality Technology	3D technology enhances educational content on vehicle parts.	Denis G, Adrián A, Javier G, FEB 2024
7	102116102	Raghav Sharma	The Application of Augmented Reality in the Automotive Industry	AR Development Platforms, 3D Modelling Software	Focus on effective learning for users unfamiliar with car customization.	Răzvan G, Florin G, Eugen V., MAY 2020
8			The Intelligent Welding Gun: Augmented Reality for Experimental Vehicle Construction	Augmented Reality Tools and Frameworks	3D technology enhances user visualization of vehicle modifications.	Florian E, Fabian S, Kay K, Gudrun K, Joachim S, Jöm T, Hesam N., JUNE 2004
9	102116112	Sanchit Raj	Context-Aware 3D Visualization and Collaboration Services for Ubiquitous Cars Using Augmented Reality	Augmented Reality Tools, Context-Aware System	3D customization technology enhances personalization in vehicle experiences.	Jae Y, Guewon R., MAR 2007
10			ARVIKA- Augmented Reality for Development. Production and Service.	AR Visualization Tools, 3D Modelling and CAD Integration	Human-centered design and ergonomic principles aim to boost productivity.	Friedrich, W FEB 2002

Table 3: Literature Survey

A. Enhanced User Experience and Customization

Users who avail themselves of existing studies, such as Jürgen et al. and Friind et al., place

emphasis on the integration of technology for high interaction with car customization options. However, these previous approaches were entirely for luxury cars or a specific brand; our app will extend this experience by offering a myriad of 3D car models at much more affordable prices. Then, changes like color, wheels, or accessories can be added directly to these models, thus making a more egalitarian platform with the needs of middle-class consumers. [1], [2]

B. Improved Design and Prototyping

Research by Florian et al. discusses the benefits of using 3D technology in vehicle design and prototyping, which, while traditionally geared towards manufacturers, can also be leveraged in consumer applications. In the context of our app, this technology allows users to visualize their modifications with high accuracy, ensuring that their customizations are both practical and aesthetically pleasing. Although the focus is on the user experience, the principles of efficient design and prototyping can guide future app enhancements, possibly including collaboration features with professional designers. [8]

C. Efficient Manufacturing and Assembly

While AR's role in manufacturing is not directly applicable to the current scope of the app, the broader industry trends highlight the importance of customization options that can be easily manufactured and assembled. As our app focuses on pre-made 3D models, it inherently simplifies the customization process, allowing users to experiment with designs that can be realistically applied to budget-friendly vehicles. This approach ensures that the customizations are both feasible and affordable.[3]

D. Training and Education

It can be prepared in a way-not through customization, but through knowledge-hence making it really effective learning for all the users. Răzvan et al. [7] and Denis et al. [6] argue that this will definitely include 3D technology and education in the application. Users would not be able to learn where the spare parts are fitted in cars but they would have acquired knowledge to know how the parts create the car. Most likely, these would also benefit those who do not know much about cars

in terms of customization but need to know what they do beforehand.[11]

E. Marketing and Sales

The fundamental aspects of 3D customization technology are quickly becoming a deal in marketing, as Xiaoyu et al. [4] say, and it's all because of the much-increased personalization in experiences by vehicles-making. The app brings into reality the imagination of such by actually giving a place where an individual can check and customize cars on a budget, an action that can ultimately reach others. The application builds on equity and affordability-both assets bringing bright prospects of occupying a niche in a market segment that is generally absent in resources to reach a title of middle class.

F. Existing Projects

- **ARVIKA Project:** Sponsored by the German Federal Ministry of Education and Research, this program is dedicated to research on augmenting realities in the design, production, and maintenance operations of complex technical products, including automobiles and aircraft. It stresses a human-centered design and ergonomic principles in order to increase productivity, effectiveness, and satisfaction of workers by implementing AR.[10]
- **BMW, Mercedes, and Volkswagen AR Showrooms:** Car manufacturers BMW, Mercedes, and Volkswagen have also adapted AR in their showrooms, allowing customers to view virtual models of cars, explore the features of a car, and customize options with AR. These AR showrooms provide customers with an extra dimension in their car-buying experience.
- **AR-Assisted Maintenance and Repair Services:** Automotive companies are increasingly utilizing AR technology to provide maintenance and repair services. AR-based applications offer real-time guidance and instructions to technicians, streamlining tasks such as diagnostics, troubleshooting, and repair procedures. By leveraging AR, automotive service providers enhance service quality, reduce downtime, and improve overall customer satisfaction. While many car manufacturers are incorporating augmented reality (AR) technology into their vehicles, there is currently no app that allows individuals to virtually customize their cars using high-quality, premade 3D models. As a result, car owners still need to visit dealerships or repair shops to explore customization options.

- By leveraging the insights from these research papers and focusing on the integration of premade 3D models, the proposed car customization app has the potential to be a valuable resource for car enthusiasts and a significant contributor to the growing adoption of digital customization technologies within the automotive industry.

2.2 Software Requirement Specification

This section describes the software requirements that will be essential to develop and implement the Moto-makeover:

A. Introduction

This app is set to change the entire paradigm of the industry. The Car Customization App will soon allow everyone to have their own custom-made 3D models created with Unity technology. What remains is this app now indulging users in the interactive processes of modifying and personalizing car models, live visualization of modification, heavy educational material associated with the customization goods, and so forth. Middle income clients who are after car manufacturers can avail themselves of cheaper costs for online services because, unlike the expensive or posh car brands, this ranges a greater number of different car manufactures in comparison.

Objective: To seamlessly induce the realistic modeling in an interactive scenario such features-as customization of the cars in many different ways, instantaneously viewable changes, and informative features at cheaper rates.

Scope: The software requirement specification document will specify software requirements for the Car Customization Application. This will cover the overall description of the software, the external interface requirements as well as non-functional requirements. The application shall embed an end-to-end solution comprising of a detailed 3D model viewer, customization tools for every car feature, a catalog with hundreds of customization options, and a good user interface for education content access.

B. Overall Description

i.Product Perspective: Car Customization App will be an application that delivers interaction through Unity 3D. Real-time interaction and rendering will be the two primary functions of the application regarding the automobiles. This application would most probably be integrated with Local Storage in the retrieval management of its 3D models and customization data. This will be a platform established for individuals to choose and apply visualizations on cars virtually. It will have learning materials on the customization process as well as the specifications of the products.

ii.Product Functions:

- **Model Selection:** Users will be capable of going through an extensive catalogue of car models through which they will choose a model.
- **Customization Tools:** It is the part where users modify the features of the car in terms of color, engaging accessories, and decals.
- **Real-Time Visualization:** Any changes that the user wants to make on the car model will be reflected instantly as he makes a choice.
- **Educational Content:** The app will explain customization products in terms of specifications, installation guides, and lots more.
- **User Authentication:** Secure login and account management for saving user preferences and customization progress.
- **Download and View:** Download and view 3D model of the customized car.

iii.User Classes and Characteristics:

- **General Users:** These are individual persons with the intent of customizing their cars. They would use the application to adopt new features and modifications of the cars, real-time change viewing, and access to educational content.
- **Admin Users:** These are the administrators who will manage the car models, customization options, and user accounts. An admin can then access necessary backend tools for managing the data as well as updating content.

iv.Operating Environment:

- **Frontend:** The Application would be available on the web. Unity will be used to develop the application for 3D modeling purposes and real-time interaction.
- **Backend:** An app that uses Local Storage to store 3D models and customization data, has a backend server to help with user authentication and also helps manage data for the user.

C. External Interface Requirements:

i. User Interfaces:

- **Model Selection Interface:** Model Selection Interface: Allows users to browse and select among car models.
- **Customization Interface:** Uses tools for modifying car characteristics and reflects changes in real time.
- **Educational Content Interface:** Provide information about the customization products and effective applications.
- **Login Page:** Allows users to authenticate and access their accounts.
- **Download Page:** Provides options to download and view the 3D model of the customized car.

ii. Hardware Interfaces:

- **Devices:** Laptops are gadgets where sufficient processing capacity exists to handle 3D rendering.

iii. Software Interfaces:

- **Unity Engine:** For 3D modeling and real time interaction.
- **Databases:** Playfab and dropbox.
- **Local Storage:** For managing and retrieving 3D models and customization data.
- **Backend Server:** For user authentication and data management.

D. Non-Functional Requirements:

- i. Performance:** Real-time updating of models with minimal latency. Support for multiple concurrent users without degradation in performance.
- ii. Security:** Secure data storage integrated with data protection law. Strongest possible authentication to protect user accounts.
- iii. Usability:** Very easy intuitive interface to navigate and customize experience. Clear instructions and feedback throughout the process.
- iv. Reliability:** High reliability with minimal downtime and error handling. Data backup and recovery mechanisms.
- v. Compatibility:** Compatibility with major web browsers. Support for various screen sizes and resolutions.
- vi. Maintainability:** Modular design for easy updates and feature additions. Comprehensive documentation for users and developers.

2.3 Cost Analysis for Purchasing Predefined 3D Models

- **Cost per Model:** Depending on the quality, detail, and licensing terms, purchasing predefined 3D models can have a range of different prices for different models. The exact cost will depend on the level of detail and the licensing agreements (e.g., royalty-free or one-time purchase).
- **Bulk Purchase Discounts:** Some platforms or vendors may offer discounts for bulk purchases if multiple models are required

2.4 Risk Analysis

Customization Limitations

- **Limited Modifiability:** Some purchased models may have restrictions or technical limitations on how much they can be customized, potentially hindering the ability to implement certain user-defined modifications within the app.

METHODOLOGY ADOPTED

3.1 Investigative Technique:

(A) Research Techniques: Conduct in-depth research on current 3D customization technologies, user needs, and industry trends. This research will guide the selection of appropriate tools and technologies for the application and ensure the app meets market demands.

(B) Design and Implementation: Design a user-friendly interface that allows for intuitive navigation and real-time 3D customization. The implementation will focus on creating a modular architecture that supports easy updates and the integration of various car models from different manufacturers.

(C) Backend and Real-Time Features: The backend will be built to manage data securely, ensuring fast, real-time updates to 3D models. Integration of Local Storage will allow users to access and manipulate high-quality 3D models, with seamless support for simultaneous multi-user interactions.

3.2 Proposed Solution

The proposed solution is to create an inexpensive user-centered application that will allow users, especially from the middle class, to use high-quality 3D visual effects over a range of car models. Unlike other existing solutions that are focused on only showing expensive automobiles or narrow brands, such an application will be open to a larger audience with models from various manufacturers.

- **Extensive 3D Model Library:** This app will present car models from a multitude of manufacturers to help users choose and customize cars regarding a diverse range of brands.
- **Real-Time Customization:** Users will view changes such as color, wheels, and accessories, and view those changes made in real-time via 3D models. Minimum latency is ensured by the application to provide as much smoothness and interaction as possible.
- **User-Friendly Interface:** The application will have a user-friendly and easy interface, making it easy even to users who have little technical knowledge. The customization process will also be guided by clear instructions and feedback into that process.
- **Educational Content:** The app will also have content beyond the customization-for example, specifications for the cars being customized, installation guides, maintenance tips, etc., which

would enhance one's experience towards an added value aside from the customization.

- **Backend Support:** The application will have a well-built backend system: Local Storage will be used for security purposes for data management and user authentication, and it will also store the 3D models. Thus, keeping the data private and also providing support for real-time updates across multiple users.

- **Cross-Platform Compatibility:** Accessing the app will be possible through almost all web browsers and, thereby achieving cross-platform compatibility that guarantees access to a wider audience on all devices.

- **Future Scalability:** The application has been designed as a modular growing application such that the addition of new functionalities, car models, and content will be done easily as the application matures.

Finally, the application targets the area of accessible and diverse creative designs and user participation, thereby transforming personalization for the masses.

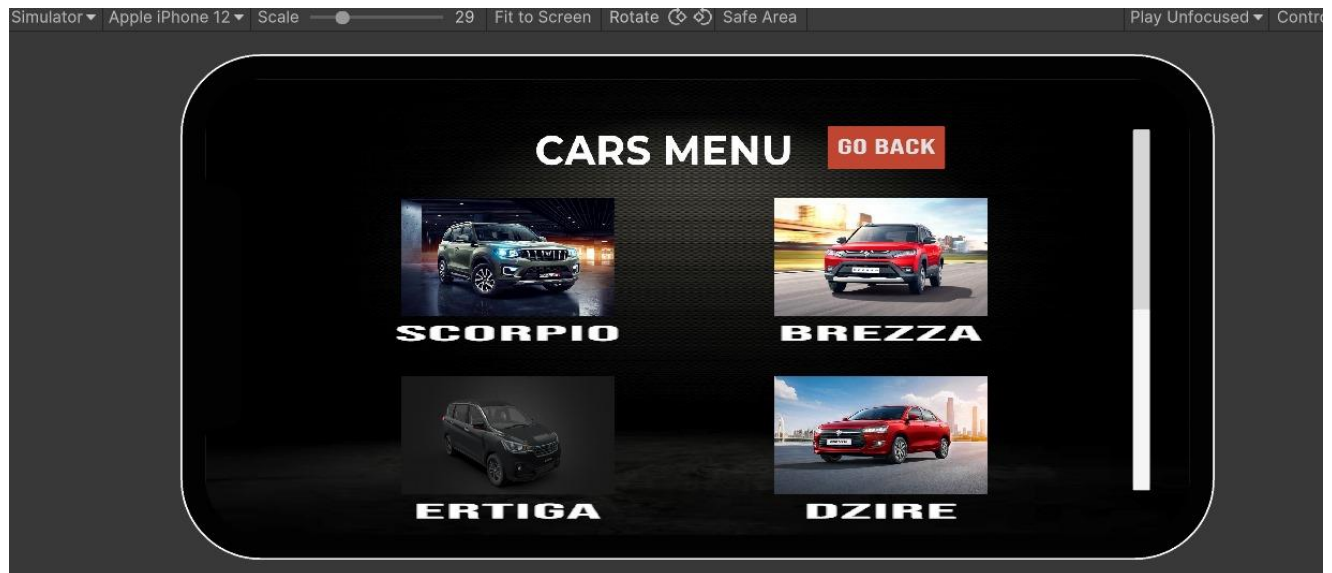


Figure 3.2(a)

3.3 Work Breakdown Structure

The following is the Work Breakdown Structure assigned to this project:

Phase 1: 3D Model Creation

Designing custom optimized 3D models with detail for application usage.

Phase 2: Customization Catalog Development

Creation of a different range of outdoor and indoor customizations that can inspire the models.

Phase 3: Interface Development

A user-interface will be created in Unity to manage intuitive customization and visualization of changes.

Phase 4: Testing and Feedback Loop

Intense testing of the models and customization capabilities, iteration based on user feedback.

Phase 5: Deployment and Launch

Making the app production-ready so that it meets all condition for deployment and actually operates as intended.

3.4 Tools & Technology

Here are the current tools and technology used in developing the Moto-Makeover:

- **Unity:** This is the one central platform for 3D modeling, the user interface design, and AR integrations.
- **3D Modeling Tools:** Software within Unity or 3D modeling tools outside Unity for creating models from scratch, it being one focus area that emphasizes customization of 3D model creations so that they provide completely different and personalized experiences for customers.
- **Customization Options Integration:** Unity tools allow for the seamless integration of customizable elements, enabling users to modify the 3D models interactively.

This approach emphasizes the custom creation of 3D models, offering users a unique and tailored customization experience.

4.1 System Architecture

The following components and diagrams depict the system architecture of the Moto-makeover system:

A. Block Diagram

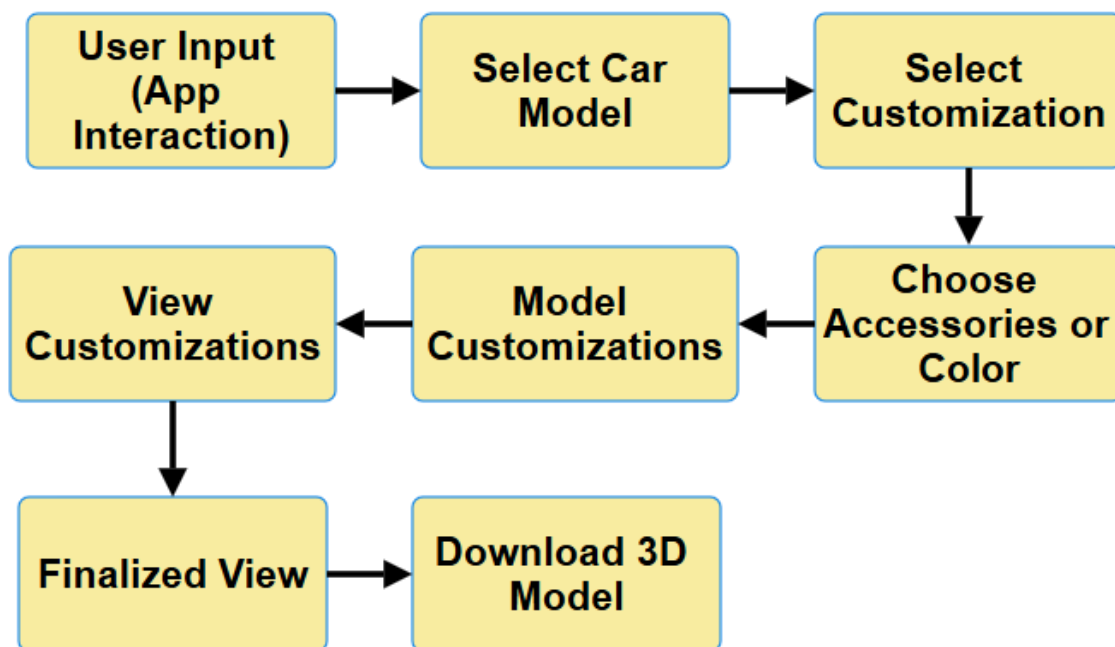


Figure 4.1(a): Block Diagram of MOTO Makeover

The block diagram of the car customization application with the flow and main components involved in customizing a car is illustrated in figure 4.1(a). The process begins when the user selects a car model from the set of options given in the application. Customization options managed by the model customizations component would largely appear to the user after the selection. Some of these options may include color adjustments, addition of accessories, or any other modifications that may personalize a model to the user's liking. The user would instantly see the changes done once the user has done applying customizations. This would enable them to see how the modifications affect the overall appearance of the car. Besides that, there is an option for

them to download their modified model of the car in 3D format, which they could save and use outside that environment. It would present a very brief overview of the entire process from selection to visualization and downloading of the personal car model.

B. Component Diagram

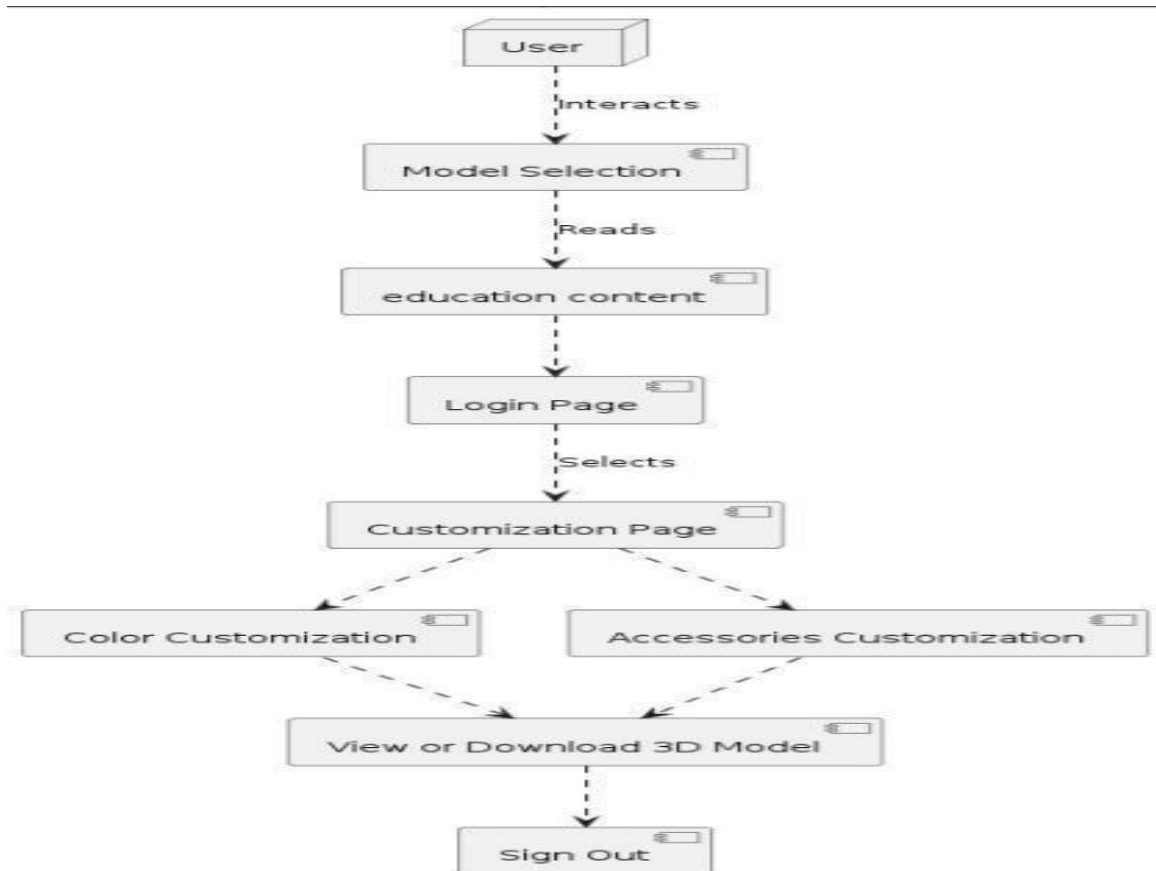


Figure 4.1(b): Component Diagram of MOTO Makeover

The component diagram of MOTO-MAKEOVER application is shown in figure 4.1(b). It specifies the sequential flow of user interaction with different components of the system. The process starts with Model Selection, where the user selects from various models that are available by choosing a model from the set of options. Once a user has chosen a model, they are interfacing with the Educational Content component - much of which is focused on clarifying what customization is, explaining the options provided through the process, and clarifying how the user would go through it. The next page for users will be the Login Page as the application demands that a user log into the full functionality of the application. After successful login, the user is redirected to the Customization page where they are able to modify a single selection by changing the color and

selecting accessories for the chosen car. There is full garage clearance for exterior car contents and possibly some interiors available for modification. Once the modifications are completed, the user can use 3D Model Viewer/Downloader to see the final appearance or be able to download the 3D version of the custom-built car. This step allows the user to download and use the customized model outside the application.

C. MVC Architecture

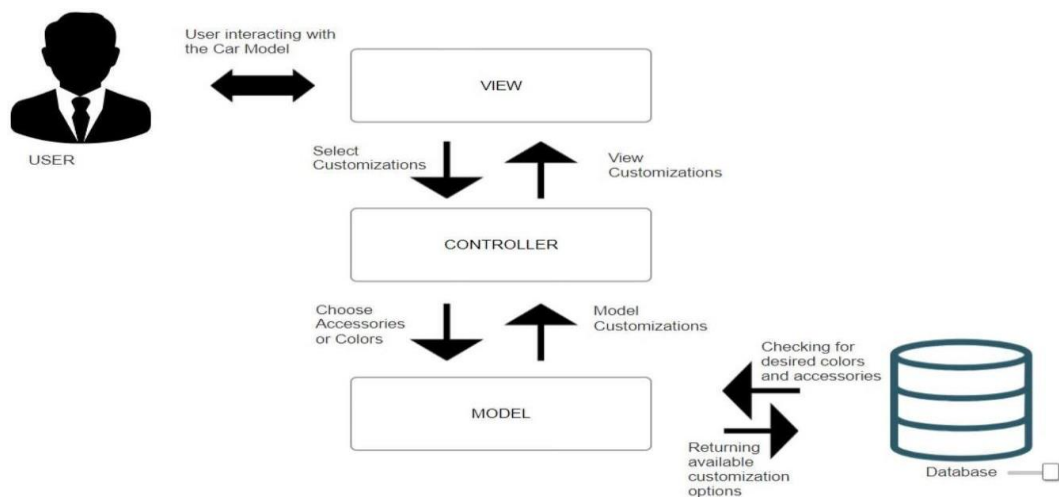


Figure 4.1(c): MVC Architecture of MOTO Makeover

The figure 4.1(c) represents the interaction flow between the different components of the MOTO-MAKEOVER application, specifically focusing on how the user interacts with the system to customize a car. User interacts with the View at first. For instance, he/she will select from the accessories and color change for the car. Raising events is proportionate to the option chosen by the user in the view. The Controller accepts event inputs from the user and calls the related customization from the Model layer to process it. This model layer is expected to fetch such information from the Database, where available accessories, color options, and all other customization items are stored. The model will then build a car model out of the instructions selected by the user. Once the constituent parts of a model object are assembled, the model layer will send it to the controller and finally it will be routed to the view. The view will then update the 3D rendering of a car such that a user can visualize what has been changed. Thus, the user would appreciate real-time changes of the selected modifications on the car model. Henceforth, whenever further customizations are needed, the user will continue through the view, and the whole process

is repeated. This allows the view and controller class with this model to create a highly informative and flexible custom configuration experience that is dynamic and responsive, allowing users to modify their car model iteratively and fine-tuning them.

4.2 Design Level Diagrams

A Use case Diagram

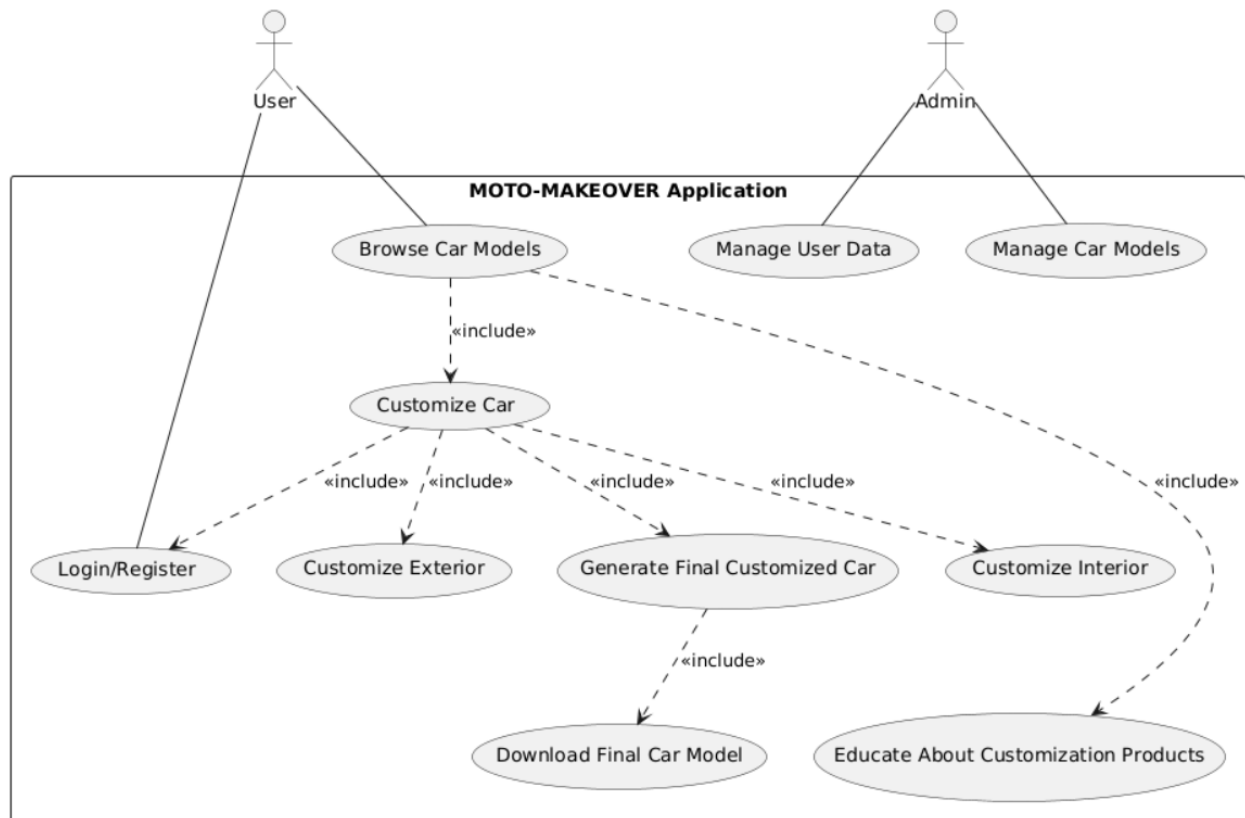


Figure 4.2(a): Use Case Diagram of MOTO Makeover

The figure 4.2(a) gives the use case for the Car Customization process under the Moto-Makeover application. Following log in, the user is presented with more than one three-dimensional model of a car, which are representative of different makes and models, and allows the user to explore and modify the model either externally or internally. The user decides on a model, modifies it with either color or accessories, and visualizes all changes being affected. He downloads the model after he is satisfied with all aspects of the customization. The system ensures that the user is authenticated, that the 3D

models are loaded properly, and that the entire process of customizing is stable. The system provides for cases wherein users cannot execute an action, require more information, or are collaborating with one another on the same car and sets up the conditions for easy customization.

The customization process of cars in the Moto-Makeover application is depicted in figure 4.2 (a). After logging in to the application, a user is presented with several 3D models of cars from various makes and models together with accessing and modifying. Customization can be done on the outside and inside of the car. Further, users select their needed car model and decide on color or accessory enhancement to visualize modification changes that would be real time. Downloading the model after he is pleased with the customization is therefore done by the user. Authentication of the user, proper loading of the 3D models, and the entire customization process being completely stable is also ensured by the system. The system would also address cases where users could not perform the function, would like more detail, or are collaborating with one another on the same car and set up the conditions under which these events can occur easily.

B Use case Template

Use Case Id	UC_1
Use Case Name	Car Customization
Date Created	01-05-2024
Description	<p>Upon logging into our platform, users are presented with a variety of 3D car</p> <p>models to choose from, each representing different makes and models. They can then explore these models in detail, visualizing modifications to get a feel</p> <p>for how their customizations will look.</p> <p>The customization process covers both the exterior and interior of the car.</p>
Primary Actor	User
Secondary Actor	None

Pre-Conditions	<ol style="list-style-type: none"> 1. User must have authenticated and logged in 2. System must have a variety of 3D car models available 3. System must be stable and operational 4. 3D model of the car must be properly initialized and loaded
Post-Conditions	<ol style="list-style-type: none"> 1. Car Customization is successfully completed 2. Changes are applied to the car model 3. Availability of Downloadable file.
Main Flow	<ol style="list-style-type: none"> 1. User selects a car model. 2. User then register's an if not registered. 3. User logs in if already registered. 4. User selects color customization or accessory customization. 5. System displays the customized car. 6. User can then download the model.
Alternate Flow	<ol style="list-style-type: none"> 1. User encounters navigation difficulties within the customization catalog and requires assistance. 2. User requests more detailed information about a customization product but finds the information insufficient or unclear. 3. Multiple users collaborate on customizing the same car, possibly leading to conflicts

Table 4: Use Case template of MOTO Makeover

C. Swimlane Diagram

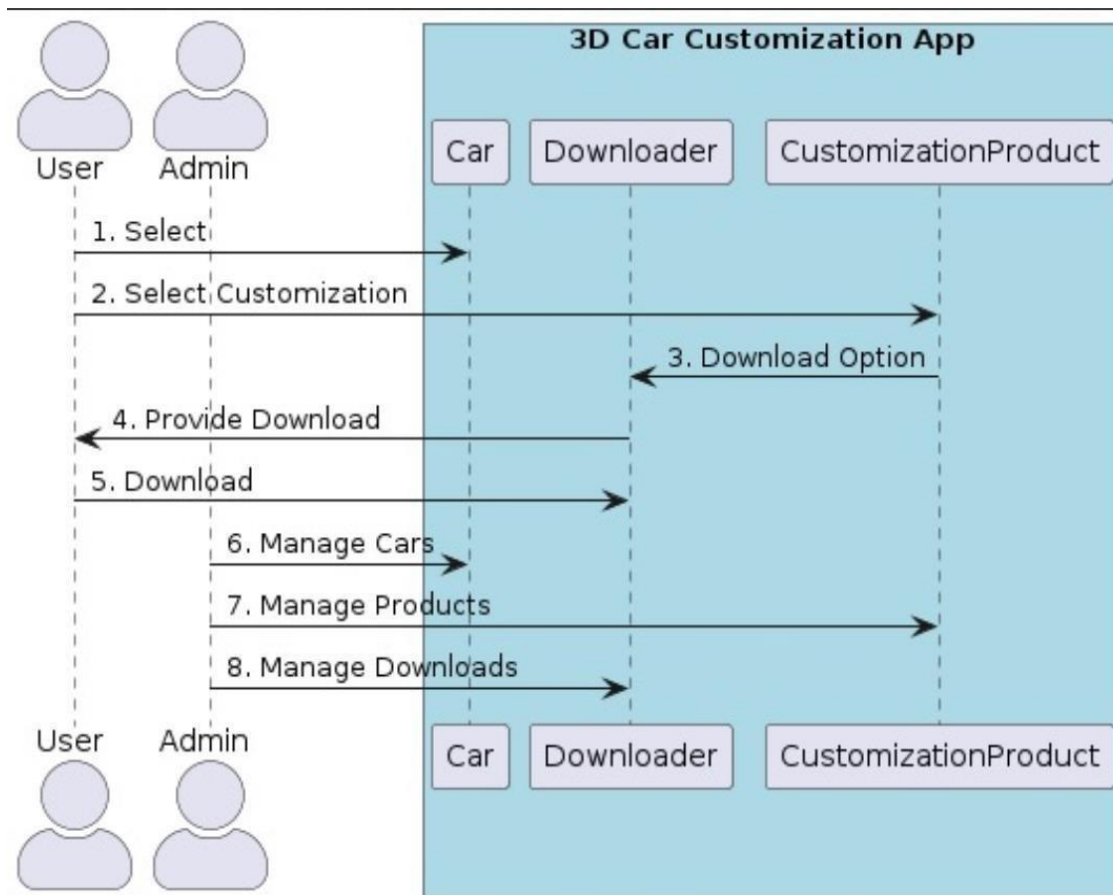


Figure 4.2(b): Swimlane Diagram of MOTO Makeover

The figure 4.2(b) illustrates the swimlane diagram that defines the distinct roles and responsibilities of the user and admin within the Moto-Makeover application. The user's role is centered on selecting a car model and customizing it with various available options. Once the customization is completed, the user can submit a download request via the Downloader component, which then allows them to download the personalized car model. Meanwhile, the admin plays a crucial role in managing the backend aspects of the application. This includes overseeing the available car models, managing the customization products or options that users can choose from, and handling the download processes of customized car models.

D. Class Diagram

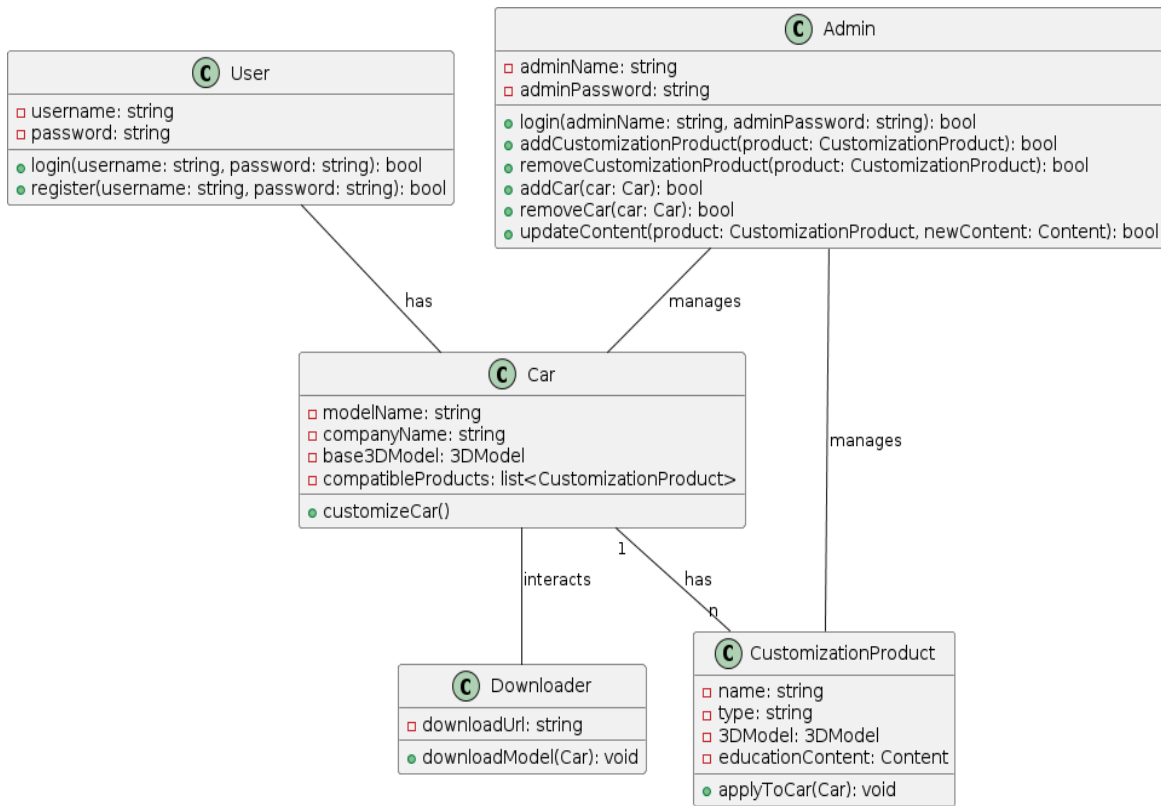


Figure 4.2(c): Class Diagram of MOTO Makeover

The class diagram for the MOTO-MAKEOVER application is shown in figure 4.2(c) and indicates briefly the relationship of some important components in the system. An example is the User class that contains the attributes username and password and methods such as login (username, password) for authenticating the user and an alternative register (username, password) for new users. The admin class, however, uses attributes like adminName and adminPassword. It also offers admin managing system methods login (adminName, adminPassword), addCustomizationProduct(product), removeCustomizationProduct(product), addCar(car) or removeCar(car) to manage the car models. The Admin class also offers the method updateContent(product, newContent) to update educational-related context. Further classes are Car-modelName, company name, base3DModel, and compatibleProduct-with a method customizeCar() that users can customize into a selected model. The CustomizationProduct class has attributes name, type, 3DModel, and educationContent, applying the method applyToCar(Car), whereby customization products can be applied to selected cars. The

Downloader class which has downloadUrl attribute offers the method downloadModel(Car) which downloads the customized car model. The relationships described in this diagram are as follows: Users are able to customize the cars hence making use of User class to the Car class. The Admin class, however, is connected to both the Car and CustomizationProduct classes because it is the one managing both types. The Car class maintains a relationship with the CustomizationProduct class, reflecting the list of compatible customization products for each car. Finally, the Downloader class interacts with the Car class to handle the downloading process of the customized car model. This class diagram provides a modular framework that defines how users, admins, car models, and customization options interact within the MOTO-MAKEOVER application.

E. ER DIAGRAM

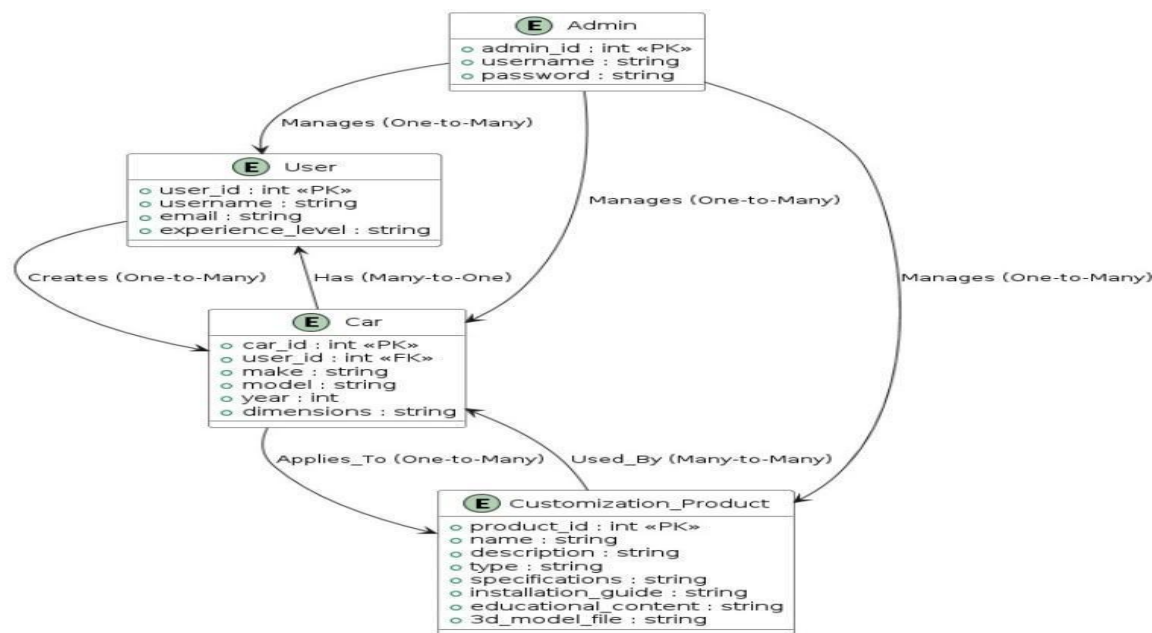


Figure 4.2(d): ER Diagram of MOTO Makeover

The figure 4.2(d) illustrates the Entity-Relationship (ER) diagram for the MOTO-MAKEOVER application, outlining the primary entities, their attributes, and the relationships among them. The

User entity represents individuals utilizing the app for car customization, with attributes including a unique userID, username, email, and an optional experienceLevel. The Car entity corresponds to the cars that users own and customize, featuring attributes such as carID, make, model, year, and dimensions, along with a foreign key linking it to the User entity, signifying ownership. The Customization Product entity encompasses the various customization options available for cars, characterized by attributes like productID, name, description, type, specifications, installationGuide, educationalContent, and 3DModelFilePath, which stores the path to the 3D model used for customization. All Admin attributes are the administrator themselves, such as admin ID, username, and password, governing their duty to manage the application. The relationship evidenced in the diagram is highly consequential to understanding the system data flow. As creates, it implies a relationship between Users and Cars which one user can own or create of multiple cars; it is one to many. For example, the Has relationship between Car and User is one-way indicating the many-to-one aspect. This means that one or rather each car is owned by one person, implying that a single customization product may apply on various cars where it is termed as Applies To or it states that it is a one-to-many relationship. The Used By relationship would, therefore, be many-to-many between Car and Customization Product as an implication that each car can use many customization products and, inversely, each product be applied to many cars. Besides these, there is a 'Manages' relationship among Admin and other entities, that is, with User, Car, and Customization Product, where it holds a one-to-many relationship. This shows the admin role concerning multiple users, cars, and customization products. Thus, they understand how this entity is engaged with the operation of overhead within the MOTO-MAKEOVER application. This ER diagram gives a whole view of how users interact with cars, administrators, and customization products in the application under the MOTO-MAKEOVER brand regarding the structure and data relationships from which the system functionalities are derived. All Admin attributes are just the administrator itself, such as admin ID, username, and password, mooring his duty to manage the application. That relationship which is evidenced in the diagram is very consequential to understanding the system data flow. Where creates-with that relational aspect between Users-to-Cars, it means one user can own or create multiple cars; but it is one to many. Has relationship between cars to user is one-indicating the aspect of many to one, which means the one car could be owned by one user. In other terms, each car is owned by one person, implying that a single customization product may apply on various cars where it is termed as Applies To or

it states that it is a one-to-many relationship. The Used by Relationship will, therefore, be many-to-many between Car and Customization Product as an implication that each car can use many of their customization products and, inversely, each product be applied to many cars. However, there is a "Manages" relationship among Admin and other entities relating to User or Administrator. For example, such an entity may be Car or Customization Product, where it holds a one-to-many relationship. Thus, they understand how this entity is engaged with the operation of overhead within the MOTO-MAKEOVER application. This ER diagram gives a whole view of how users interact with cars, administrators, and customization products in the application under the MOTO-MAKEOVER brand concerning the structure and data relationships from which the system functionalities are derived. This ER diagram provides a complete picture of how users, cars, customization products, and admins interact within the MOTO-MAKEOVER application, establishing the structure and data relationships upon which the system operates. All Admin attributes are just the administrator himself, such as admin ID, username, and password, mooring his duty to manage the application. The relationship evidenced in the diagram is very consequential for understanding the system data flow. Where creates-with that relational aspect between Users-to-Cars, it means one user can own or create multiple cars; but it is one to many. Has relationship between car to user is one-indicating aspect of many to one which shows that one car could be owned by one user.

F. Data Flow Diagram

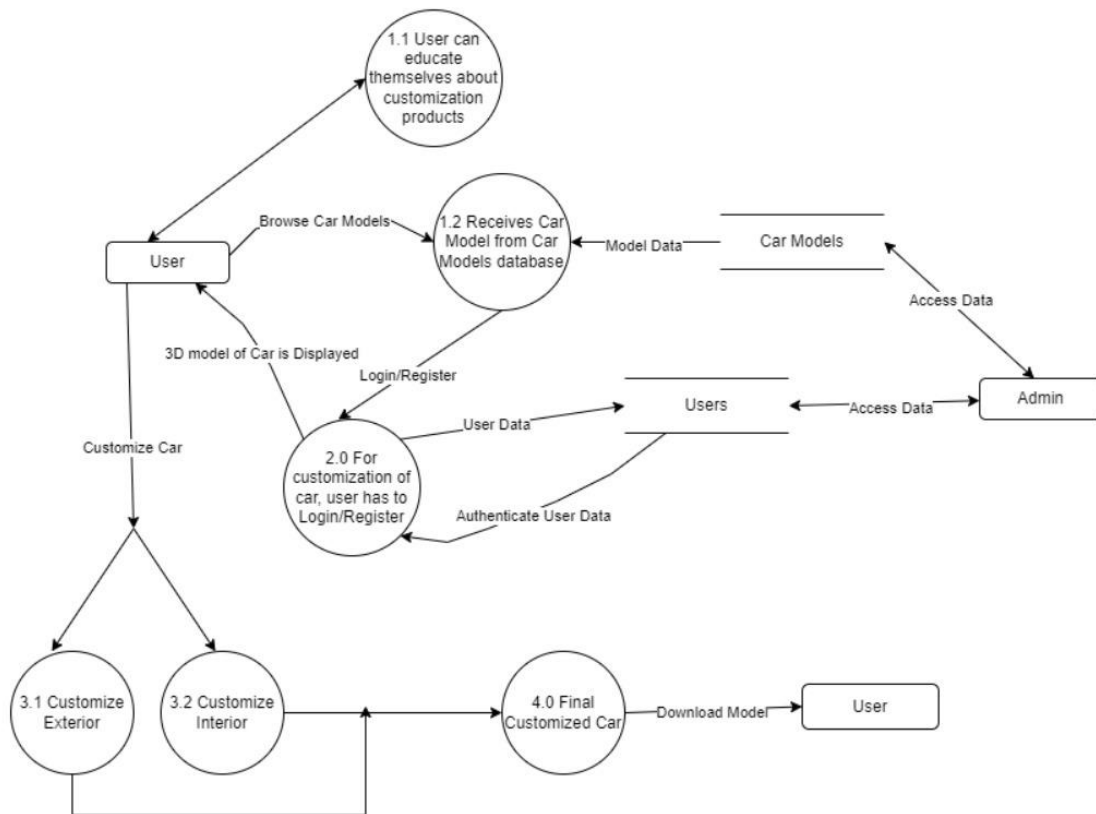


Figure 4.2(e): Data Flow Diagram of MOTO Makeover

The figure 4.2(e) is the data-flow diagram for the MOTO-MAKEOVER application, showing information and processes flowing in and out of the system. The diagram focuses on the user's operation of browsing through available car models, selecting one for customization and then downloading the finalized customized car model. This all starts once the user switches on the system to obtain the available car models from the Car Models Database. After viewing the model characteristics, he might decide on what customization products to explore in the application and then select one car model. The system takes the particular model from the Car Models Database and presents it to the user as a 3D model for viewing and customizing. Before he can start customizing his car, the user is required to log in or register, which involves an identity check to authenticate and validate the data using access to Users Database. A confirmation of the identity of the user, as well as validation of permissions, allow him to proceed with the process of

customization. External and interior customization areas comprise two sections. Exterior components include the following: color changes, wheels, and spoilers. On the other hand, interior components comprise seats, dashboards, and other internal components. The customizations would be submitted to the application, which would then produce the final customized car model with all the changes that the user has made on the car. The downloaded final customized car model is ready at this stage ties are derived. All Admin attributes are just the administrator itself, such as admin ID, username, and password, mooring his duty to manage the application. That relationship which is evidenced in the diagram is very consequential to understanding the system data flow. Where creates-with that relational aspect between Users-to-Cars, it means one user can own or create multiple cars; but it is one to many. Has relationship between cars to user is one-indicating the aspect of many to one, which means the one car could be owned by one user. In other terms, each car is owned by one person, implying that a single customization product may apply on various cars where it is termed as Applies To or it states that it is a one-to-many relationship. The Used By Relationship will, therefore, be many-to-many between Car And Customization Product as an implication that each car can use many of their customization products and, inversely, each product be applied to many cars. However, there is a "Manages" relationship among Admin and other entities relating to User or Administrator. For example, such an entity may be Car or Customization Product, where it holds a one-to-many relationship. Thus, they understand how this entity is engaged with the operation of overhead within the MOTO-MAKEOVER application. This ER diagram gives a whole view of how users interact with cars, administrators, and customization products in the application under the MOTO-MAKEOVER brand concerning the structure and data relationships from which the system functionalities are derived. This ER diagram provides a complete picture of how users, cars, customization products, and admins interact within the MOTO-MAKEOVER application, establishing the structure and data relationships upon which the system operates. All Admin attributes are just the administrator himself, such as admin ID, username, and password, mooring his duty to manage the application. The relationship evidenced in the diagram is very consequential for understanding the system data flow. Where creates-with that relational aspect between Users-to-Cars, it means one user can own or create multiple cars; but it is one to many. Has relationship between car to user is one-indicating aspect of many to one which shows that one car could be owned by one user. Here, the administrator controls the application at a very macro level, where he manages the database of car

models as well as user databases for the smooth functioning of the whole system. The admin's role includes overseeing the available car models, managing user data, and maintaining the overall operation of the application.

The data-flow diagram outlines a clear process flow: the user begins by browsing and selecting a car model, followed by logging in or registering. After completing both exterior and interior customizations, the system generates the final customized car model, which the user can download. Throughout this process, the admin monitors and manages the databases to ensure seamless functionality.

IMPLEMENTATION AND EXPERIMENTAL RESULTS

5.1 Procedural Flow:

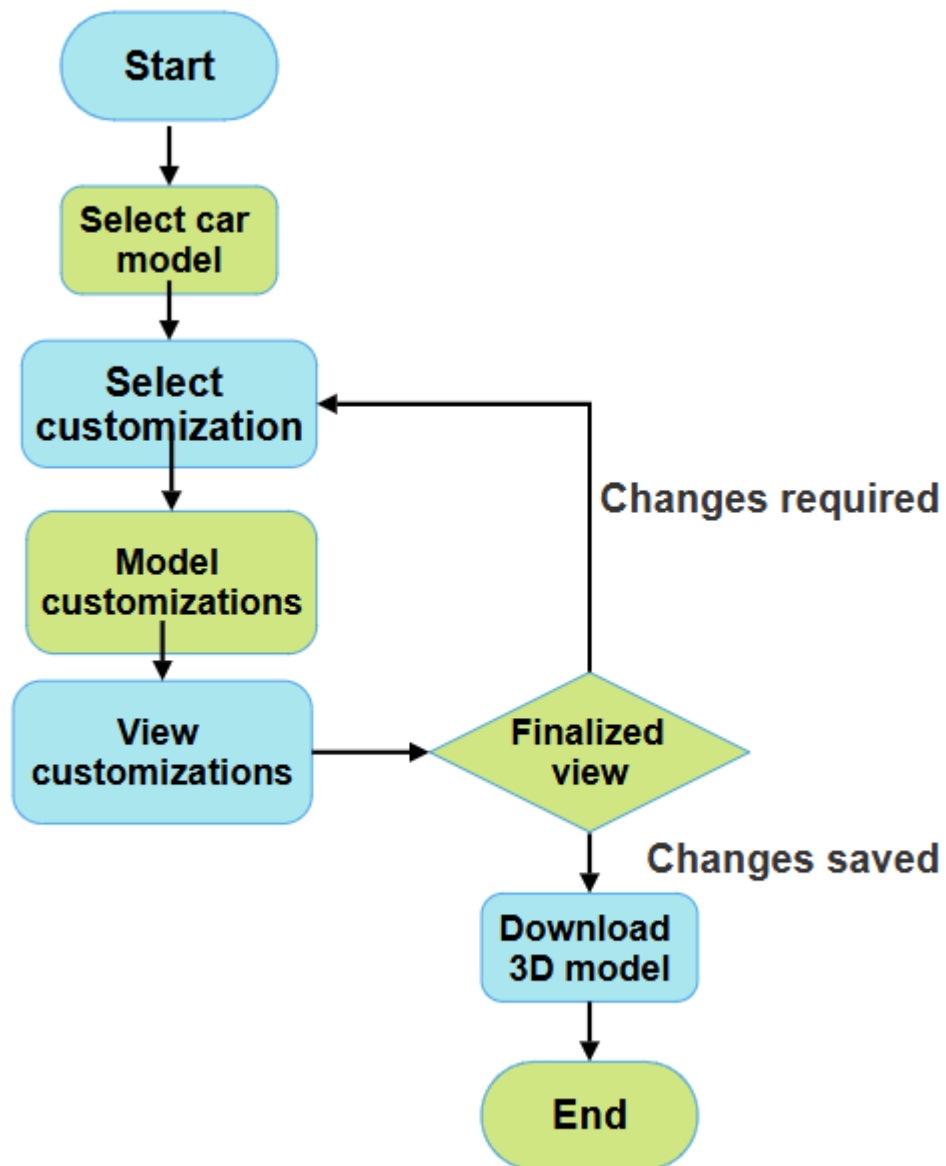


Figure 5.1(a) Working Flow

The user begins the customization process and then selects a 3D car model from the library to work on. To modify the car visually, he starts browsing the available customizations such as accessories, spoilers, paint, etc. After this, the main 3D model customization happens in real-time with Unity's detailed virtual rendering engine where he applies the selected tools and accessories to the chosen car model. Now the user can preview and analyze the applied changes on the car model from multiple angles to assess the look and details. The app provides an AR (Augmented Reality) view of the finalized model to help users visualize the model in a real-world scenario. If the user approves the modified design of the car model, he will download the final 3D model. If unsatisfied with the modification, he can return to the "Select Customization" option and repeat the process. Once the car model is finalized user can download the file saving the customizations for future use, printing, and showcasing.

5.2 Working Project:

Opening screen of our prototype.

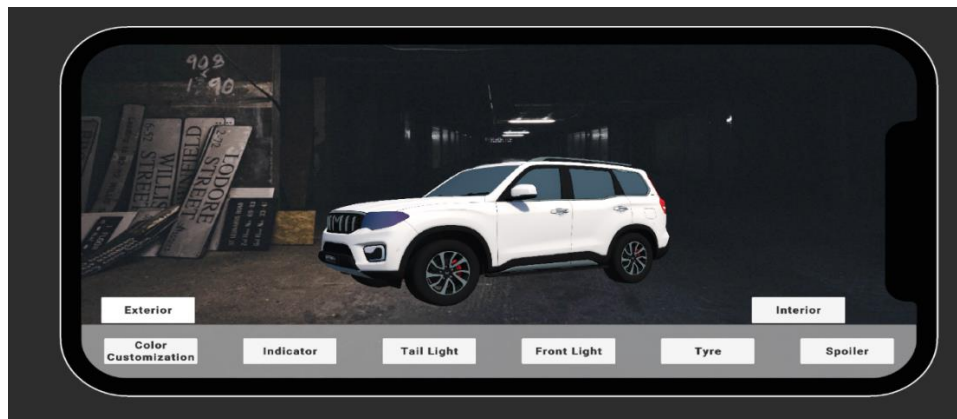


Figure 5.2(a): Opening screen

The figure 5.2(a) depicts the main interface of the Unity-based car customization app, Moto-Makeover. The prototype currently focuses on customizing the Mahindra Scorpio-N, offering both exterior and interior modifications. Exterior customization options include color changes, tail light, headlight, indicator, tire, and spoiler variations. Interior modifications encompass seat covers,

doors, dashboard, steering wheel, interior roof, and armrest selections. This app provides a visually engaging platform for users to experiment with different configurations and personalize their virtual Scorpio-N.

Created buttons in Unity that allow users to customize the car-tyre.



Figure 5.2(b): Tyre customization

The figure 5.2(b) demonstrates the tyre customization feature of the Moto-Makeover app. Users can select different tyre styles (1 for now) to personalize the appearance of their virtual Mahindra Scorpio-N. Additionally, the app allows for 360-degree rotation of the vehicle, providing a comprehensive view of the customized tires from all angles. This feature enhances the user's ability to visualize the final look of the vehicle with the chosen tires.

Created buttons in Unity that allow users to customize the car color.

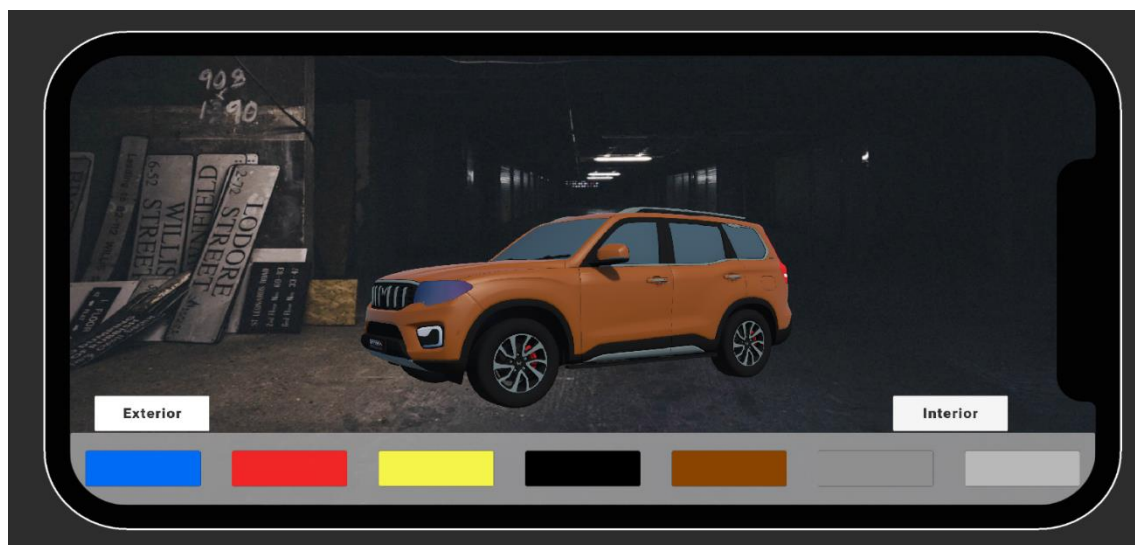


Figure 5.2(c): Color customization

The figure 5.2(c) demonstrates the color customization feature of the Moto-Makeover app for the Mahindra Scorpio-N. The available color options include blue, red, yellow, black, brown, grey and silver. Users can visually see how these different colors would look on the vehicle before making their final selection. This feature enhances the user experience by providing a realistic preview of the customized vehicle.

The figure demonstrates the headlight customization feature.

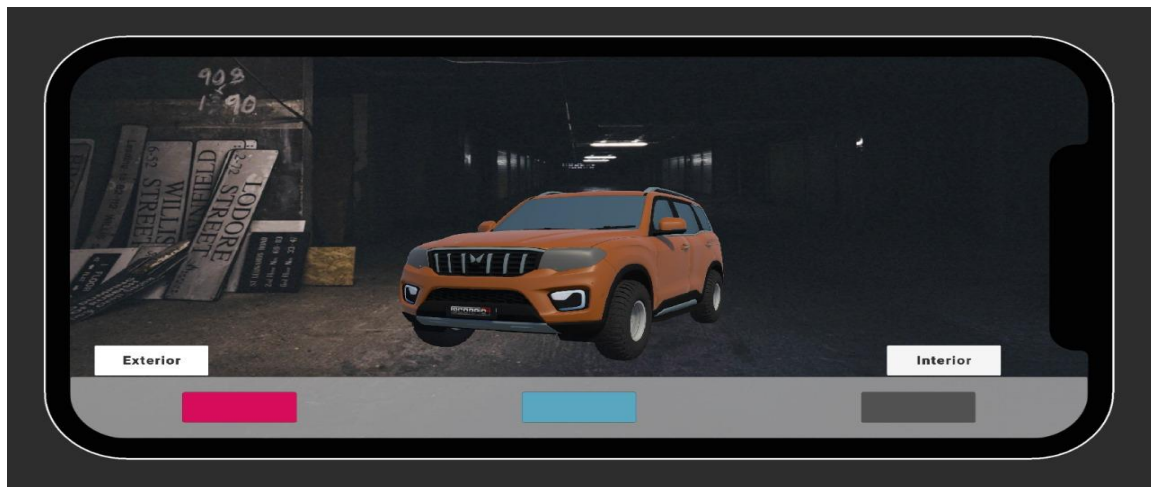


Figure 5.2(d): Headlight customization

The figure 5.2(d) demonstrates the headlight customization feature of the Moto-Makeover app. Users can choose from three available headlight colors: crystal red, light blue, and light black. This limited selection is likely due to government regulations regarding headlight colors. The application enables the users to see a visual representation of how the colors would look on their vehicles in the final selection process.

Created buttons on Unity for spoiler modification.

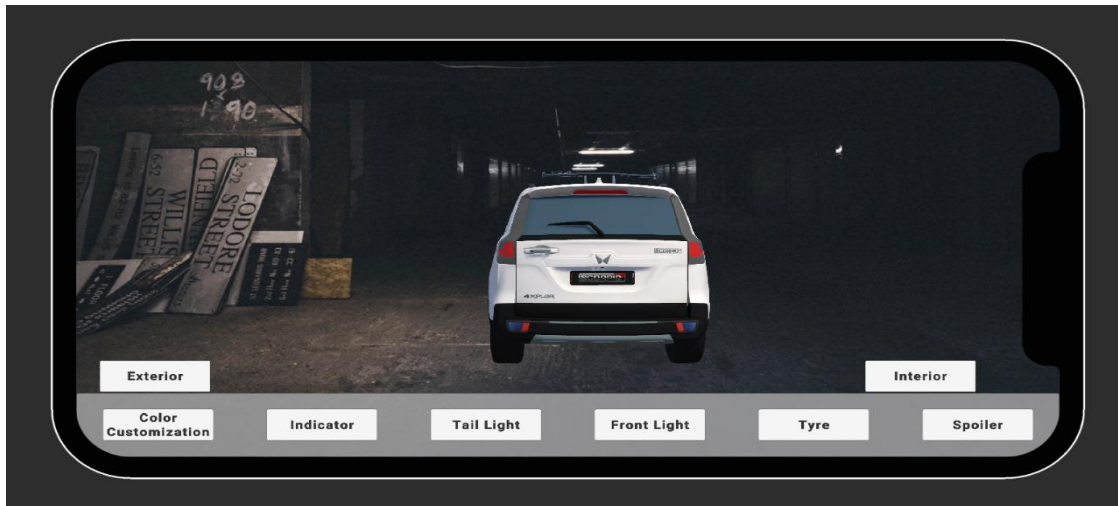


Figure 5.2(e): Spoiler, taillight, indicator customization

The figure 5.2(e) illustrates the functionalities of the Moto-Makeover application in customizing spoilers, tail lights, and indicators. The users are allowed to opt for their favored styles of spoilers on the body shell and customize their tail lights and indicators as per their choice. It includes a visual representation of the customized vehicle with the modifications.

Showcasing the interior camera.

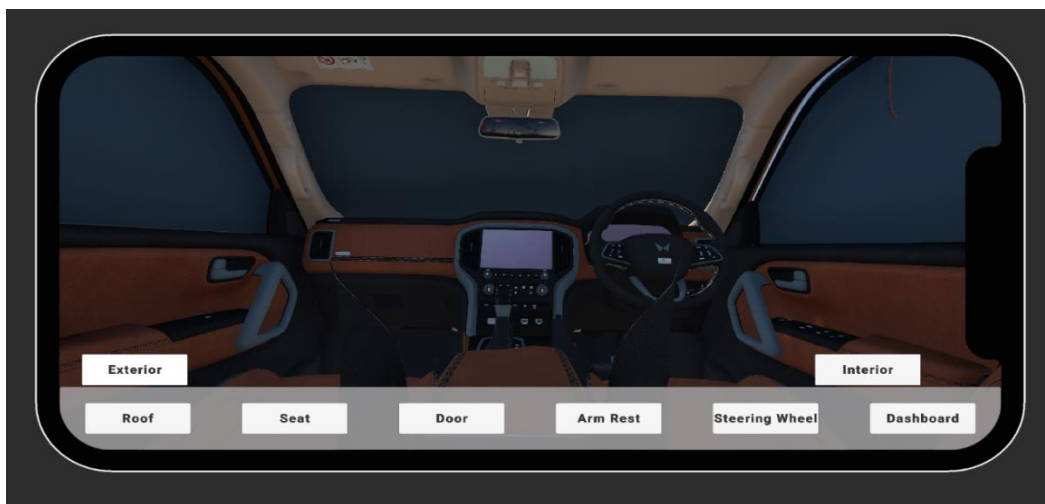


Figure 5.2(f): Interior customization

The figure 5.2(f) shows interior camera feature of Moto-Makeover App which enables the user to experience the interiors in Mahindra Scorpio-N in greater detail and also find out what effect the different customizations would have. Different types of interiors, such as roof, seats, doors, armrests, steering wheel, and dashboard can be seen with changing materials, colors, or designs.

Button for customizing seat covers

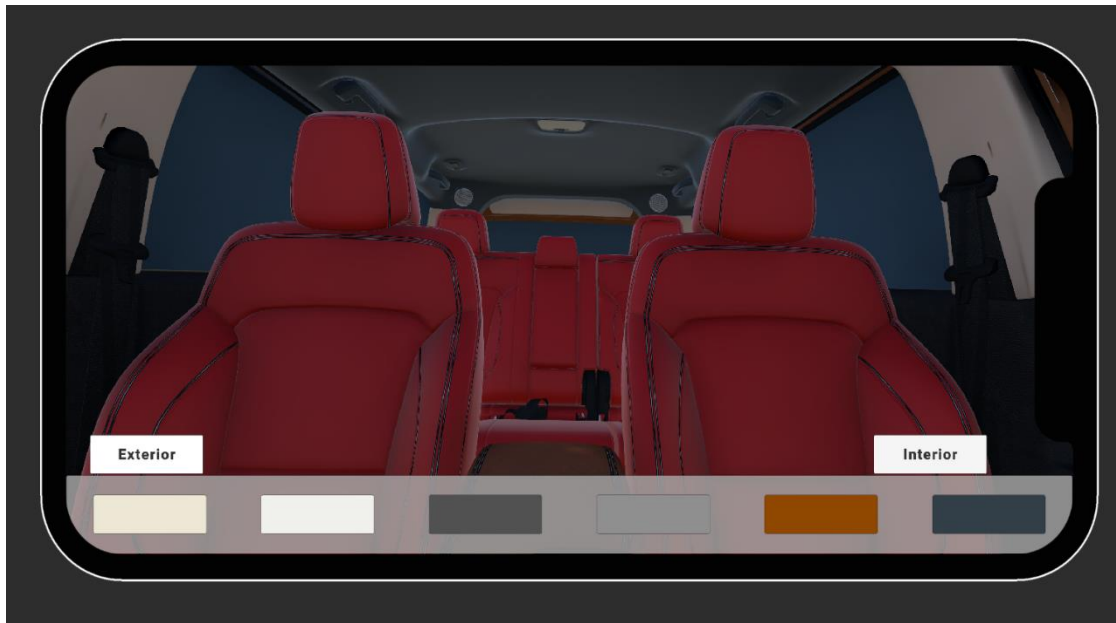


Figure 5.2(g): Seat customization

The figure 5.2(g) depicts customizations in seat covers and interior roofs using the Moto-Makeover app. Users can personalize their interior as these seat covers can be customized from a range of textures and colors. More, various colors are available for the interior roof. This makes it easy for a user to design a personalized and attractive interior for his or her virtual Mahindra Scorpio-N.

5.3 Testing Process:

5.3.1 Black Box Testing: Specific Car UI

S. No.	Action/ Description	Input	Expected Result	Actual Result	Status	Comments
1	Launch the specific car page (Interior view).	Open car page	The interior car UI loads correctly with all buttons.	UI loaded successfully.	PASS	Verify all buttons are visible and aligned.
2	Verify the presence of Exterior and Interior toggle buttons.	Check UI buttons	Both buttons are visible and clickable.	Both buttons are visible and clickable.	PASS	Toggle buttons are functional.
3	Click on the Exterior button.	Click Exterior button	Exterior car view is displayed.	Exterior view is displayed.	PASS	Correct UI transition to Exterior view.
4	Verify all customization buttons in the Exterior view: Color Customization, Indicator, etc.	Check UI buttons	All buttons are visible and aligned in the Exterior view.	Buttons aligned correctly.	PASS	Buttons are displayed and functional.
5	Click on each Exterior customization button (Color Customization, Tail Light, etc.).	Click Buttons	Corresponding customization options or content loads.	Content loaded successfully	PASS	No lag or errors observed
6	Click on the Interior button.	Click Interior button	Interior car view is displayed.	Interior view is displayed.	PASS	Correct UI transition to Interior view.
7	Verify all customization buttons in the Interior view: Seat, Roof, Door, etc.	Check UI buttons	All buttons are visible and aligned in the Interior view.	Buttons aligned correctly.	PASS	Buttons respond correctly to user input.
8	(Negative Case) Click on each Interior customization button (Seat, Roof, Door, etc.).	Click "Seat" button	Corresponding educational content loads correctly.	Null content exception.	FAIL	Issue Fixed: See below for resolution.
9	Verify the Download button functionality.	Click Download	Download process initiates or confirmation appears.	Download started successfully.	PASS	Ensure no UI overlap or unexpected errors.
10	Test responsiveness	Change screen	UI	UI adjusted	PASS	Alignment

	by resizing the screen.	resolution	components adjust properly without clipping.	correctly.		and scaling are responsive.
11	Verify no overlapping of buttons or UI elements when toggling views.	Switch between Interior/Exterior	UI elements remain aligned and responsive.	UI elements responsive.	PASS	UI flow remains smooth and functional.
12	Verify scrolling behaviour in case the list of customization options exceeds the screen size.	Scroll UI	Scrollbar works smoothly and all options are accessible.	Scrollbar worked correctly.	PASS	All options visible and accessible.

Table 5: Black Box Testing

5.3.2 White Box Testing:

Test Case ID	Function Name	Test Scenario	Code Path/ Function	Expected Behavior	Actual Behavior	Status	Resolution/ Comments
WTC_01	LoadingManager.LoadScreen()	Verify progress bar updates to 100% on app load.	LoadingManager.LoadScreen()	Progress bar reaches 100% and transitions to Main Menu.	Progress bar reaches 100%.	PASS	Loading completes smoothly.
WTC_02	UIManager.OnButtonClick()	Verify "Car Menu" button navigates to Car Page.	UIManager.OnButtonClick("CarMenu")	Car Page scene loads correctly.	Car Page loaded successfully	PASS	Smooth transition to Car Menu
WTC_03	ContentManager.LoadContent()	Verify educational content loads for "Seat".	ContentManager.LoadContent("Seat")	Educational content for Seat is displayed.	Educational content loaded.	PASS	Content displayed without issues.
WTC_04	ScrollView.Update()	Verify vertical scroll functionality in Car	ScrollView.Update()	Scrollbar works smoothly, showing	Scrollbar works smoothly.	PASS	Scrolling behaves as expected.

		Menu.		g all items.			
WTC_05	UIManager.OnButtonClick()	Verify "Back" button takes user to Main Menu.	UIManager.OnButtonClick("Back")	Returns to the Main Menu screen.	Main Menu loaded successfully.	PASS	Verified button navigation.
WTC_06	CustomizationHandler.LoadOptions()	Handle missing customization options gracefully.	CustomizationHandler.LoadOptions()	Display error message when options are null/missing.	App crashes to desktop	FAIL	Fixed: Added a null check with error message : "No customization options available."

Table 6: White Box Testing

5.4 Algorithmic Approaches Used:

1. Color Customization-

using UnityEngine;

using System.Collections.Generic;

```
public class CarColorCustomization : MonoBehaviour
```

```
{
```

```
    [System.Serializable]
```

```
    public class RendererInfo
```

```
    {
```

```
        public Renderer renderer;
```

```
        public int materialIndex = 0;
```

```
        [HideInInspector] public Material originalMaterial; // Store the original material
```

```
    }
```

```
    public List<RendererInfo> carParts = new List<RendererInfo>();
```

```
    public Material[] colorMaterials;
```

```
    private int currentIndex = -1; // Track the current material index (start at -1 for default)
```

```
    void Start()
```

```
    {
```

```
        // Store the original materials for each car part
```

```
        foreach (var part in carParts)
```

```

    {
        if (part.renderer != null && part.materialIndex < part.renderer.materials.Length)
        {
            part.originalMaterial = part.renderer.materials[part.materialIndex];
        }
    }
}

public void SetOrToggleCarColor(int colorIndex)
{
    if (colorIndex < 0 || colorIndex >= colorMaterials.Length)
    {
        Debug.LogError("Invalid color index or color materials array is empty!");
        return;
    }

    if (currentIndex == colorIndex)
    {
        // If the current index is already the same as the clicked index, revert to the original
        materials for all car parts
        foreach (var part in carParts)
        {
            if (part.renderer != null)
            {
                Material[] materials = part.renderer.materials;
                materials[part.materialIndex] = part.originalMaterial; // Revert to the stored original
            }
        }
        currentIndex = -1; // Reset the index to indicate the default materials are active
    }
    else
    {
        // Create new material arrays for each car part renderer and assign the selected custom
        material
        foreach (var part in carParts)
        {
            if (part.renderer != null)
            {
                Material[] materials = part.renderer.materials;
                materials[part.materialIndex] = colorMaterials[colorIndex]; // Assign the selected
                custom material
                part.renderer.materials = materials;
            }
        }
    }
}

```

```

        currentIndex = colorIndex; // Update the current index
    }
}

```

2. Spoiler-

using UnityEngine;

```

public class SpoilerToggleManager : MonoBehaviour
{
    public GameObject spoilerParent; // Parent object containing all spoilers
    public GameObject[] spoilers; // Array of spoilers to be activated or deactivated

    private int currentIndex = -1; // Track the current active spoiler index (-1 means none active)

    // Method to activate or deactivate a spoiler by index
    public void SetOrToggleSpoiler(int index)
    {
        if (spoilers.Length == 0 || index < 0 || index >= spoilers.Length)
            return;

        // If the current index is already the same as the clicked index, deactivate all spoilers
        if (currentIndex == index)
        {
            spoilers[index].SetActive(false);
            currentIndex = -1; // Reset the index to indicate no spoiler is active
        }
        else
        {
            // Deactivate all spoilers first
            foreach (var spoiler in spoilers)
            {
                spoiler.SetActive(false);
            }

            // Activate the selected spoiler
            spoilers[index].SetActive(true);
            currentIndex = index; // Update the current index
        }
    }
}

```

3 Seat Cover- using UnityEngine;

```
public class SeatCoverCustomizer : MonoBehaviour
{
    public Renderer[] seatCoverRenderers; // Array of Renderer components for multiple
    seat covers
    public Camera mainCamera;           // Attach the main camera here
    public Camera interiorCamera;       // Attach the interior camera here
    public Material[] customMaterials;   // Array of custom materials for different colors

    private Material[][] defaultMaterials; // Array of arrays to store default materials for
    each seat cover
    private int currentIndex = -1;        // Track the current material index (start at -1 for
    default)

    void Start()
    {
        if (seatCoverRenderers.Length == 0)
        {
            Debug.LogError("Seat Cover Renderers are not assigned!");
            return;
        }

        // Initialize and store the default materials for each seat cover renderer
        defaultMaterials = new Material[seatCoverRenderers.Length][];
        for (int i = 0; i < seatCoverRenderers.Length; i++)
        {
            defaultMaterials[i] = seatCoverRenderers[i].materials;
        }
    }

    public void SetOrToggleSeatCoverColor(int index)
    {
        if (customMaterials.Length == 0 || index < 0 || index >= customMaterials.Length)
            return;

        if (currentIndex == index)
        {
            // If the current index is already the same as the clicked index, revert to the default
            materials for all seat covers
            for (int i = 0; i < seatCoverRenderers.Length; i++)
            {
                seatCoverRenderers[i].materials = defaultMaterials[i];
            }
        }
    }
}
```

```

        // Toggle cameras
        interiorCamera.enabled = false;
        mainCamera.enabled = true;
        currentIndex = -1; // Reset the index to indicate the default materials are active
    }
    else
    {
        // Create new material arrays for each seat cover renderer and assign the selected
        custom material
        for (int i = 0; i < seatCoverRenderers.Length; i++)
        {
            Material[] newMaterials = new
            Material[seatCoverRenderers[i].materials.Length];
            for (int j = 0; j < seatCoverRenderers[i].materials.Length; j++)
            {
                newMaterials[j] = customMaterials[index]; // Assign the selected custom
                material
            }
            seatCoverRenderers[i].materials = newMaterials;
        }

        // Toggle cameras
        interiorCamera.enabled = true;
        mainCamera.enabled = false;
        currentIndex = index; // Update the current index
    }
}
}

```

4 Steering Wheel- using UnityEngine;

```

public class SteeringWheelMaterialChanger : MonoBehaviour
{
    public Renderer[] steeringWheelRenderers; // Array of Renderer components for
    multiple steering wheels
    public Material[] defaultBlackMaterials; // Array to store the default black materials
    public Material[] customMaterials; // Array to store custom materials for the
    steering wheels
    private Material[][] originalMaterials; // Array of arrays to store original materials
    for each steering wheel
    private int currentMaterialIndex = -1; // Track the current material index (-1 indicates
    default materials are active)

    void Start()
    {

```

```

    if (steeringWheelRenderers.Length == 0)
    {
        Debug.LogError("Steering Wheel Renderers are not assigned!");
        return;
    }

    // Initialize and store the original materials for each steering wheel
    originalMaterials = new Material[steeringWheelRenderers.Length][];
    for (int i = 0; i < steeringWheelRenderers.Length; i++)
    {
        originalMaterials[i] = steeringWheelRenderers[i].materials;
    }
}

public void ToggleMaterials(int index)
{
    // Validate the material index and ensure custom materials are set
    if (defaultBlackMaterials == null || customMaterials == null || index < 0 || index >=
customMaterials.Length)
    {
        Debug.LogError("Invalid material index or custom materials are not set!");
        return;
    }

    // Toggle between the original and custom materials based on the current index
    if (currentMaterialIndex == index)
    {
        Debug.Log("Switching to original materials");
        ApplyOriginalMaterials();
        currentMaterialIndex = -1; // Reset the index to indicate the default materials are
active
    }
    else
    {
        Debug.Log("Switching to custom material at index " + index);
        ApplyCustomMaterial(index);
        currentMaterialIndex = index; // Update the current index
    }
}

private void ApplyCustomMaterial(int index)
{
    // Loop through each steering wheel renderer and replace default materials with the
custom material at the given index
    for (int i = 0; i < steeringWheelRenderers.Length; i++)
    {

```

```

        Material[] newMaterials = new Material[originalMaterials[i].Length]; // Create a
new array to hold modified materials

```

```

        for (int j = 0; j < originalMaterials[i].Length; j++)
        {
            // Replace the specific black material with the selected custom material
            bool materialReplaced = false;

            for (int k = 0; k < defaultBlackMaterials.Length; k++)
            {
                if (originalMaterials[i][j].name == defaultBlackMaterials[k].name + "
(Instance)")
                {
                    newMaterials[j] = customMaterials[index]; // Replace with the custom
material
                    materialReplaced = true;
                    break;
                }
            }

            // If not replaced, keep the original material
            if (!materialReplaced)
            {
                newMaterials[j] = originalMaterials[i][j];
            }
        }

        steeringWheelRenderers[i].materials = newMaterials; // Apply the modified
materials to the renderer
    }
}

private void ApplyOriginalMaterials()
{
    // Revert each steering wheel renderer to its original materials
    for (int i = 0; i < steeringWheelRenderers.Length; i++)
    {
        steeringWheelRenderers[i].materials = originalMaterials[i];
    }
}
}

```

5 Arm Rest- using UnityEngine;

```

public class ArmrestCustomizer : MonoBehaviour

```



```

{
    public Renderer armrestRenderer;    // Attach the armrest's Renderer here
    public Material[] customMaterials;  // Array of custom materials to apply
    public Material brownMaterial;      // The brown material to be replaced
    private Material[] originalMaterials; // Array to store original materials
    private int currentIndex = -1;       // Track the current material index (start at -1 for
default)

    void Start()
    {
        if (armrestRenderer == null)
        {
            Debug.LogError("Armrest Renderer is not assigned!");
            return;
        }

        // Store the original materials of the armrest
        originalMaterials = armrestRenderer.materials;
    }

    public void ToggleMaterial(int index)
    {
        if (customMaterials.Length == 0 || index < 0 || index >= customMaterials.Length)
        {
            Debug.LogError("Invalid custom materials or index out of range!");
            return;
        }

        if (currentIndex == index)
        {
            // If the current index is already the same as the clicked index, revert to the original
materials
            Debug.Log("Switching to original materials");
            ApplyOriginalMaterial();
            currentIndex = -1; // Reset the index to indicate the default materials are active
        }
        else
        {
            // Apply the selected custom material
            Debug.Log("Switching to custom material");
            ApplyCustomMaterial(index);
            currentIndex = index; // Update the current index
        }
    }

    private void ApplyCustomMaterial(int index)

```

```

{
    // Replace the specific brown material with the selected custom material
    Material[] newMaterials = new Material[originalMaterials.Length]; // Create a new
array to hold the modified materials

    for (int i = 0; i < originalMaterials.Length; i++)
    {
        if (originalMaterials[i].name == brownMaterial.name + " (Instance)")
        {
            newMaterials[i] = customMaterials[index]; // Replace with the selected custom
material
        }
        else
        {
            newMaterials[i] = originalMaterials[i]; // Keep the other materials unchanged
        }
    }

    armrestRenderer.materials = newMaterials; // Apply the modified materials to the
renderer
}

private void ApplyOriginalMaterial()
{
    // Revert to the original materials
    armrestRenderer.materials = originalMaterials;
}
}

```

CONCLUSIONS AND FUTURE SCOPE

6.1 Work Accomplished

This complete development is focused on application handling for car customization using Unity. Handled the following aspects in their actual implementation:

- **Product Customization:** This enables user to customize almost every car line from its interior to exterior components. The user can now alter certain elements such as the tail lights, spoilers, roofs, tires, and the entire color of the vehicle in this application for a detailed personal experience.
- **User Authentication:** All login/logout mechanisms for the users have been built in, secure access is now possible for the users to this application. When users log into this app, their details will also be safe under this user name so as to easily retrieve those preferences in customization, thus ensuring a personalized experience each time.
- **Real-Time Database Integration:** This application is now completely integrated into Local Storage i.e. Playfab and DropBox for real-time customization storage and retrieval. This makes it possible for the users to see the modification made to his vehicle instantly in the app, which provides a very smooth and dynamic user experience.
- **User Interface (UI):** The final design UI is very refined, user-friendly, and optimized for little hassle through a complete auto customization process. The layout is intuitive enough for the user to navigate through all of the customization options conveniently.

Now these features produce the best performance above all in automotive customization solutions. In fact, it is an absolutely safe, real-time, and engaging phenomenon.

6.2 Conclusions

Integrating modifications on our Unity-based app for car customization takes a big step towards rendering a wholesome experience on the platform to the users. The appendage of comprehensive details on the interior and exterior parts for the car, secured authentication of the user, and local storage bring the app fully into possible remarkable development of base. These achievements show a commitment to providing a well-rounded experience for users in modification-aesthetic immersion, accessibility, and dynamism. Now that the entire project has come alive, we will again lay the foundation by adding features continuously and improving the user experience for a broader

customization scope. We are committed to changing the world of car customization and ensuring that our users always have a top-notch platform that evolves along with.

6.3 Benefits

Personalized Experience: The application also allows its users to personalize everything in the car, right from the inside features to the outside modifications, thereby giving them an experience very tailor-made for them.

User-Friendly Interface: Intuitive Design and very easy to use - even an illiterate person can work out how to do things. Suffice to say, the navigation through all the customizations becomes really.

Real-Time Updates: With real-time updates through Local Storage, the user can see the current realization of most by his modification entry for seamless interaction.

Cost-Effective Customization: Our application is rather cheaper as compared to the already available costly specific car-applications, unlike the existing solutions aimed at luxury car or brand-specific features.

Educational Content: The app allows a person to customize a car but also provides knowledge related to a few of the many car parts and their functions; therefore, adding learning to the word value.

Secure and Reliable: By implementing a secure login and data management structure, its users can enjoy 100% safety of their user information while other customization data is secured in this application-based platform.

Future Expansion: Modular in design, the application can allow direct interoperability through updating and inclusions. Its usage shapes its effective and efficient growth concerning demand and the user-technological evolution.

6.4 Future Scope:

1. Paint damage detection in all-inclusive assessments of an automotive repair program.
2. Extension of overall capacity and usage of the system for application in automotive repair into education and insurance.

3. Further development of object detection models, texture detection algorithms, and depth estimation algorithms for improving the real-world applicability of their accuracy and adaptability.
4. Exploring scalable solutions with customizable product details for tackling the diverse needs of users.

7.1 Challenges Faced:

1. Bringing the system onboard in such a way that users can adjust everything from being fully internal to external becomes a real challenge in trying to develop well-working system towards the ultimate goal.
2. It required tuning to avoid lagging in real-time updates between the database (PlayFab and DropBox) and local storage system.
3. To perform several rounds of test design iterations to provide users a user-friendly and easy to access application.

7.2 Relevant Subjects:

1. Software Development and Programming
2. Automotive Engineering
3. Augmented Reality and 3D Visualization

7.3 Peer Assessment Matrix:

Evaluation By:	Evaluation of:					
		Diwi	Samridhi Wadhwa	Raghav Sharma	Sanchit Raj	Kushal Srivastava

Diwi	4	5	5	4	4
Samridhi Wadhwa	4	4	5	5	4
Raghav Sharma	4	4	4	4	5
Sanchit Raj	5	5	4	4	5
Kushal Srivastava	5	4	4	5	4

Table 7: Peer Assessment Matrix

7.4 Role Playing and Work Schedule:

Members	Sanchit Raj	Raghav Sharma	Kushal Srivastava	Samridhi Wadhwa	Diwi Malik
Modules					
Initial Research	✓	✓	✓	✓	✓
Unity & App Development	✓	✓	✓	✓	
Integrating different frameworks		✓	✓		✓
Integrating customized products	✓	✓		✓	
Testing		✓	✓		✓
Monitoring and Modifications	✓	✓	✓	✓	✓
Documentation	✓	✓	✓	✓	✓
Final Report	✓	✓	✓	✓	✓

Table 8: Role Playing and Work Schedule

7.5 Student Outcomes Description and Performance Indicators:

SO	SO Description	Outcome
1.6	Development of algorithms for customization features.	Designed the logic for car modifications, including interior and exterior components.
1.6	Identification of secure login needs and real-time data retrieval.	Implemented dynamic real-time updates to display seamless customization effect.
3.4	Creation of a user-friendly, intuitive interface.	Designed the UI for ease of navigation for customization.
3.4	Collaboration in modular system design.	Worked collaboratively to integrate product customization, UI, and database.
6.3	Modular design to accommodate future features.	Implemented scalable database systems for future expansion.
6.4	Focus on affordability and accessibility.	Developed cost-effective solutions for wider-audience reach.

Table 9: Student Outcomes Description and Performance Indicators

7.6 Brief Analytical Assessment:

The car customization application is built on Unity and offers a very powerful platform for personal and experiential user access. These include superior detailed product customizations, a secure user authentication system, real-time database integration, as well as a very intuitive and accessible UI. It allows a user to change the interior and exterior parts of a car, get real-time updates, and store their modifications securely.

References

- [1]Jürgen F, Jürgen G, Carsten M, and Rafael R. “Using Augmented Reality Technology to Support the Automobile Development”, MAY 2004
- [2]Friind J, Gausemeier J, Matysczok C, Radkowski R. “Cooperative Design Support within Automobile Advance Development Using Augmented Reality Technology” MAY 2004
- [3]Deep M, Tejas P, Bhavesh P, Abhishek S, Quazi TZ, Chetan T. “VEHICLE CUSTOMIZATION USING AUGMENTED REALITY (AR): A LITERATURE REVIEW”, Volume-04, Issue-11, November-2022
- [4]Xiaoyu A, Qingdan J, Syed M. “The exploration of customization in augmented reality from the affordance lens: A three-stage hybrid approach”, SEP 2019
- [5]Zdeněk C, Gabriel F, Nikoleta M. “Application of virtual and augmented reality in automotive” NOV 2019
- [6]Denis G, Adrián A, Javier G, “Application of augmented reality in automotive industry” FEB 2024
- [7]Răzvan G, Florin G, Eugen V. “The Application of Augmented Reality in the Automotive Industry”, MAY 2020
- [8]Florian E, Fabian S, Kay K, Gudrun K, Joachim S, Jöm T, Hesam N. “The Intelligent Welding Gun: Augmented Reality for Experimental Vehicle Construction”, JUNE 2004
- [9]Jae Y, Guewon R. “Context-aware 3D visualization and collaboration services for ubiquitous cars using augmented reality”, MAR 2007
- [10]Friedrich, W. “ARVIKA - Augmented Reality for Development, Production and Service. Siemens AG, Automation and Drives Advanced Technologies and Standards”, FEB 2002

Plagiarism Report

Motomakeover final			
ORIGINALITY REPORT			
4%	4%	2%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS
PRIMARY SOURCES			
1	www.coursehero.com Internet Source	1	%
2	Mateus, João Manuel Andrade. "Guia de Boas Práticas e Protótipo para Desenvolvimento Educacional de Crianças em Idade Escolar", ISCTE - Instituto Universitario de Lisboa (Portugal), 2023 Publication	<1	%
3	su-plus.strathmore.edu Internet Source	<1	%
4	Deepthi Aggarwal, Kakoli Banerjee, Vikram Bali, Mansi Gupta, Pranav Kumar, Prabha Kumari. "A Study for the Development of a Photogrammetry Based AR and GR Projection System and Examining its Further Application", 2022 1st International Conference on Informatics (ICI), 2022 Publication	<1	%
5	www.tandfonline.com Internet Source	<1	%

6	Kirim Yagmur, Okechukwu Okorie. "Hybrid process improvement framework to reduce the eutrophication potential of in-situ leaching for uranium extraction: A qualitative study", Journal of Cleaner Production, 2024 Publication	<1 %
7	link.springer.com Internet Source	<1 %
8	www.designsociety.org Internet Source	<1 %
9	dokumen.pub Internet Source	<1 %
10	eduprojecttopics.com Internet Source	<1 %
11	api.openalex.org Internet Source	<1 %
12	articardio.de Internet Source	<1 %
13	campar.in.tum.de Internet Source	<1 %
14	doczz.net Internet Source	<1 %
15	Noble Anumbe, Clint Saidy, Ramy Harik. "A Primer on the Factories of the Future", Sensors, 2022	<1 %

16	muuktest.com Internet Source	<1 %
17	Dallmann, Nicholas A.. "Limitations of Classical Tomographic Reconstructions from Restricted Measurements and Enhancing with Physically Constrained Machine Learning.", Arizona State University, 2020 Publication	<1 %
18	digitalcommons.unf.edu Internet Source	<1 %
19	repository.tudelft.nl Internet Source	<1 %
20	sjcit.ac.in Internet Source	<1 %
21	www.hni.uni-paderborn.de Internet Source	<1 %
22	Zhe Chuan Feng. "Handbook of Zinc Oxide and Related Materials - Materials", CRC Press, 2012 Publication	<1 %
23	elibrary.tucl.edu.np Internet Source	<1 %
24	vdoc.pub Internet Source	<1 %

25

Lecture Notes in Computer Science, 2016.

Publication

<1%

26

Ton Duc Thang University

Publication

<1%

Exclude quotes Off

Exclude matches Off

Exclude bibliography Off