HUMAN-COMPUTER INTERACTION SOLUTION

FOR

PASS IT DRIVING SCHOOL DESKTOP APPLICATION

Student Name : Warnakula Adithiya Sembukkuttige Dilasha Mischell Jaksa

Student No : WAR22195728

Subject : Human-Computer Interaction (2218)

University College Birmingham

Table of Contents

1. Introduction 5

2. System Design 6

2.1 Mood board 6

2.2 Requirements defining 7

2.2.1 Client Requirements 7
2.2.2 Hardware requirements 9
2.2.3 Software requirements 9

2.3 Flowchart 10

2.4 Storyboard 11

2.5 Technical Design - Code 14

2.5.1 Application Termination 14
When the exit button is clicked in the dashboard, this code will trigger a warning message in order to get the confirmation for the application termination from the user. This eliminated the potential effects that can happen to the user experience by user's slips and mistakes. 14
2.5.2 Error provider 14
An error provider is implemented to check null values before the form submission. The code will also generate error messages and successful messages. This also reduces the impact of slips and mistakes of the user. 15
2.5.3 Dashboard Division according to the user 15
..... 15
2.5.4 Forms triggered by another form under sub-modules 16
The forms triggered by other forms in the sub-modules are also opened in the same panel inside the dashboard using this code. This helps to maintain the consistency throughout the design. 16

2.6 Technical Specifications 17

2.7 Consideration of design rules 17

2.8 Justification of the design 18

3. Implementation 21

3.1 Login User Interface 21

3.2 Student module 22

3.2.1 Lesson materials sub-module 23
3.2.2 Schedules sub-module 26
3.2.3 Progress module 27
3.2.4 Book a lesson 28
3.2.5 Contact Instructor 29

3.3 Admin module 30

3.3.1 Student details module 31
3.3.2 Instructor details module 33
3.3.3 Schedule details module 34
3.3.4 Lesson upload and progress upload 35
3.3.5 Request view module 37

4. Evaluation 38

4.1 Suitability for the target audience 38
4.2 Usability 38
4.3 Maintainability 39
4.4 Portability.....39
4.5 Robustness 39
4.6 Constraints 39
4.7 Legal and Ethical Considerations.....40
4.8 Strengths and weaknesses of the solution 40
5. Conclusion.....41

Table of Figures

Figure 1 - Mood Board 6
Figure 2 - Flowchart for the scenario 10
Figure 3 - Storyboard Part 1 11
Figure 4 - Storyboard Part 2 11
Figure 5 - Storyboard Part 3 12
Figure 6 - Storyboard part 4 12
Figure 7 - Storyboard Part 5 13
Figure 8 - Storyboard Part 6 13
Figure 9 - Application termination code 14
Figure 10 – Error provider raising error indications 1 14
Figure 11 - Error provider and error and success messages 15
Figure 12 - Opening the dashboard depending on the user login 15
Figure 13 - Form triggered by another form in a sub-module 16
Figure 14 - Login UI 21
Figure 15 - Validation message box for successful login 21
Figure 16 - Validation message box for failed login 22
Figure 17 - Student user dashboard 22
Figure 18 - Lesson materials UI 23
Figure 19 - Lesson materials in Introductory lesson type UI 24
Figure 20 - Confirmation message for successful download of the lesson 24
Figure 21 - Lesson locked message box 25
Figure 22 - Booking successful confirmation 25
Figure 23 - Schedule UI 26
Figure 24 - Request schedule change confirmation message 26
Figure 25 - Progress UI 27
Figure 26 - Booking a lesson UI 28
Figure 27 - Contact Instructor UI 29
Figure 28 - Successful Message sent 29
Figure 29 - Admin Dashboard UI 30
Figure 30 - Student Details UI 31
Figure 31 - New student form submission 31
Figure 32 - Edit existing student details 32
Figure 33 - Instructor details UI 33
Figure 34 - Edit instructor details UI 33
Figure 35 - Schedule details UI 34
Figure 36 - Schedule edit UI 34
Figure 37 - Lesson upload UI 1 35
Figure 38 - Lesson upload UI 2 35
Figure 39 - Progress upload 36
Figure 40 - Request View UI 1 37
Figure 41 - Review view UI 2 37

1. Introduction

This assignment proposes a design solution for the Pass IT Driving school and implements and evaluates it to cater to its requirement of seeking an efficient HCI solution for a desktop application to manage their student records, instructor records, lesson schedules, lesson materials, etc. The process of producing the above-mentioned solution will address HCI principles together with mood boards, details requirement defining, storyboards, flowcharts, and technical specifications in the proposal stage, developing design documentation. The design documentation will be utilized in the implementation phase to develop a user-centered interface showcasing adherence to the discussed HCI principles in the proposal stage. In the end, the implemented design solution will be evaluated, assessing its suitability for the target audience, usability, maintainability, portability, and robustness. It will also discuss the time constraints and legal and ethical considerations of the solution while coming to a conclusion with a discussion of overall strengths and weaknesses. Recommendations for improvements will be discussed for each of the above based on the findings of the evaluation. Overall, the assignment intends to demonstrate a process of proposing, developing and accessing an HCI solution to cater carefully processed client requirements tailored for the efficient use of a driving school.

2. System Design

2.1 Mood board

Figure 1 - Mood Board

2.2 Requirements defining

2.2.1 Client Requirements

2.2.1.1 Functional Requirements

- Students and Admins must have user accounts that can be used to access the software.
- Students need to view their profile and an overview.
- Students must be able to edit their profile details.
- Students need to book and cancel their lessons and exams.
- Students need to see their instructor's details and contact them
- Students must have access to their lesson and exam schedules
- Students need to access their lesson materials
- Students need to see their progress in lessons and exams
- Admins want to create, edit, and delete student and instructor details and schedules.
- Admins want to upload lesson progress and lesson materials for students.
- Admins want to view requests sent by the students.

2.2.1.2 Non-functional Requirements

- Performance

  Software should respond at a fair speed

- Scalability

  Functionalities of the software can adapt to scaling to cater to adding requirements.

- Usability

  Software must be easy to use and understand

- Accessibility

  Software is accessible to all intended users

- Responsiveness

  Software allows users to terminate actions and provides appropriate feedback

- Efficiency

  Software needs minimum movements to interact with and has an anticipatory design

- Digestibility

  Information should be easy to ready and capture. Visual elements must be in a guiding manner.

- Simplicity

  Software should have a less crowded design with more familiar yet fewer number of elements.

2.2.2 Hardware requirements

- CPU - ARM64 or x64 processor or equivalent
- RAM - 8GB (minimum 4GB)
- Hard disk - SSD (least capacity -128 GB)

2.2.3 Software requirements

- Operating System - Windows 10 or later
- Design software - Visual Studio 2022, Adobe Photoshop CC 2019

2.3 Flowchart

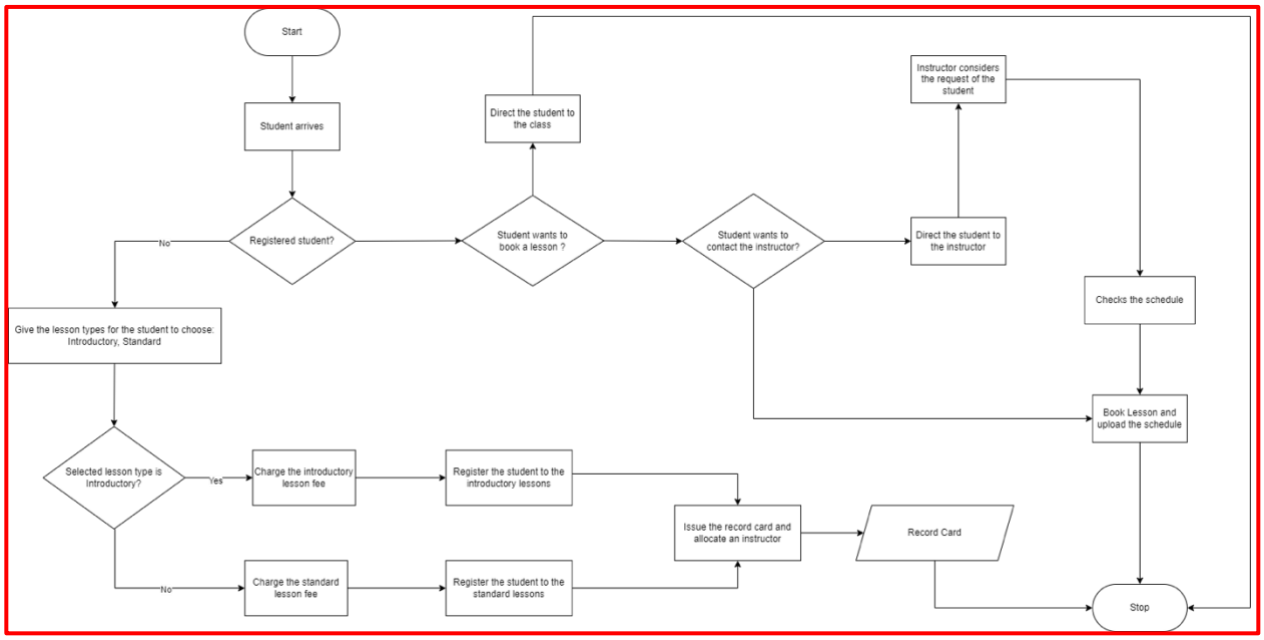

Figure 2 - Flowchart for the scenario

2.4 Storyboard

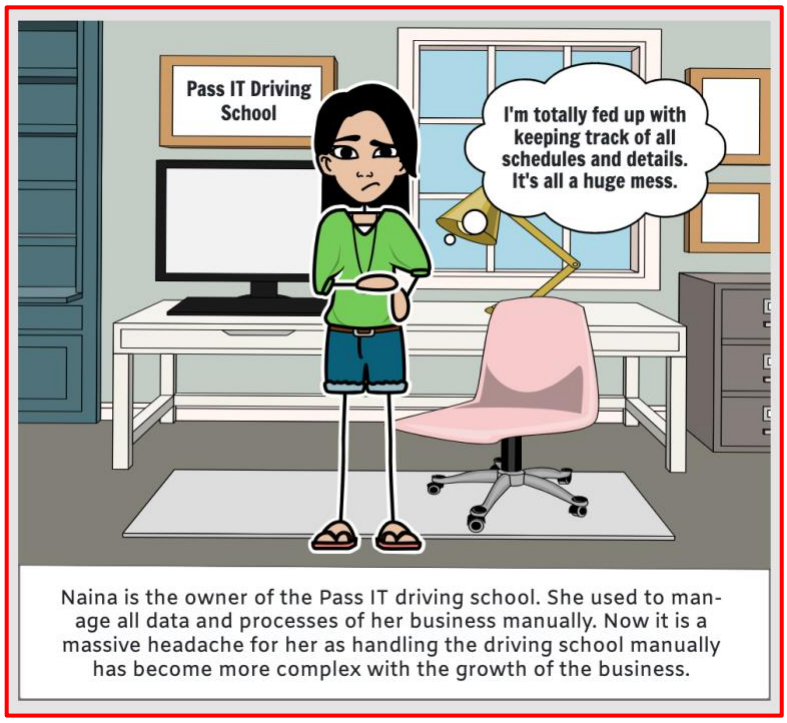

Figure 3 - Storyboard Part 1

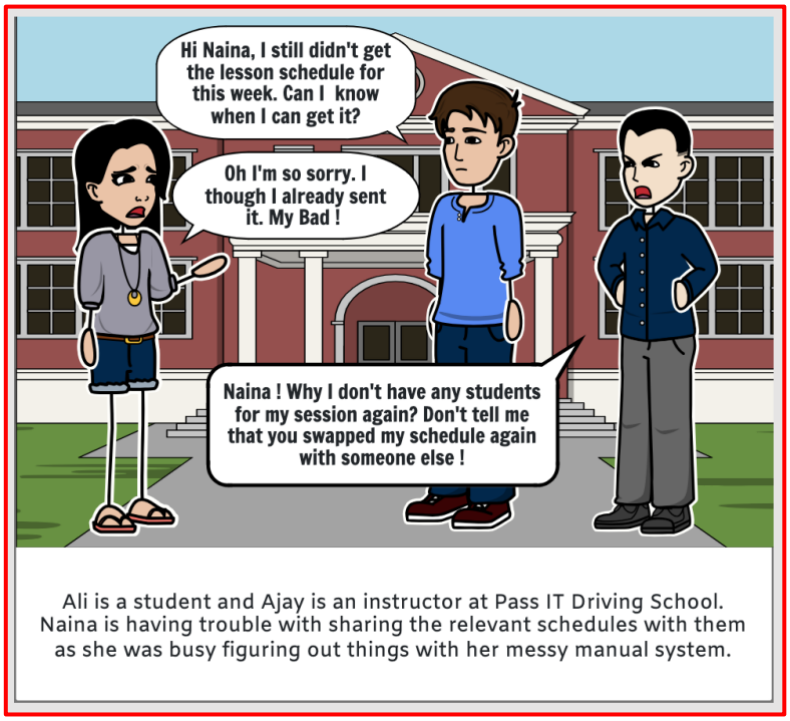

Figure 4 - Storyboard Part 2

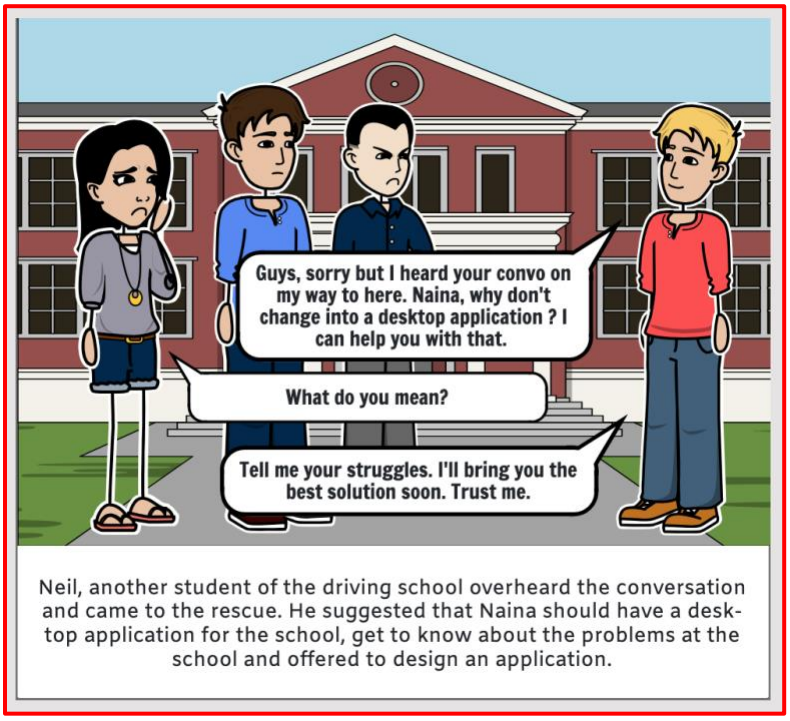

Figure 5 - Storyboard Part 3

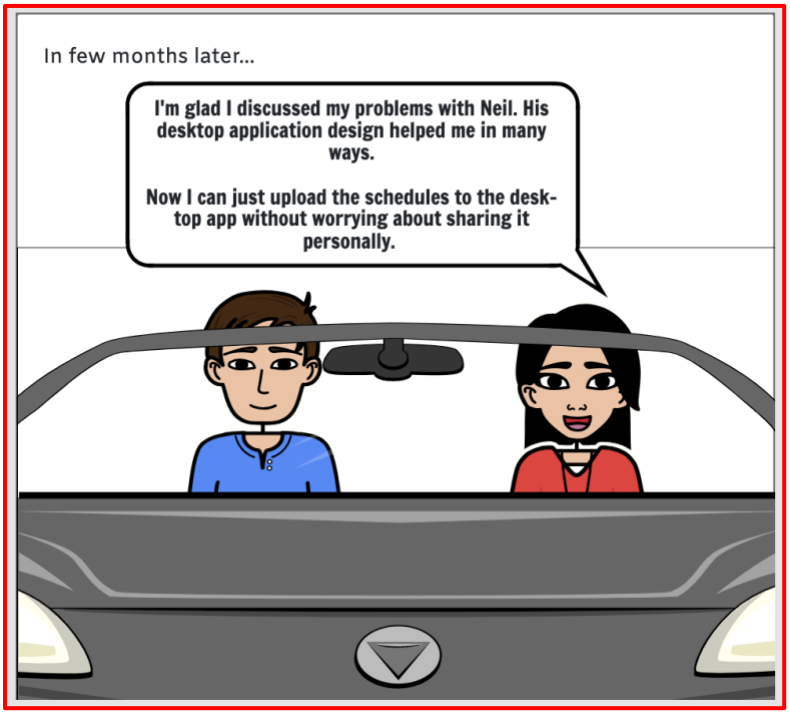

Figure 6 - Storyboard part 4

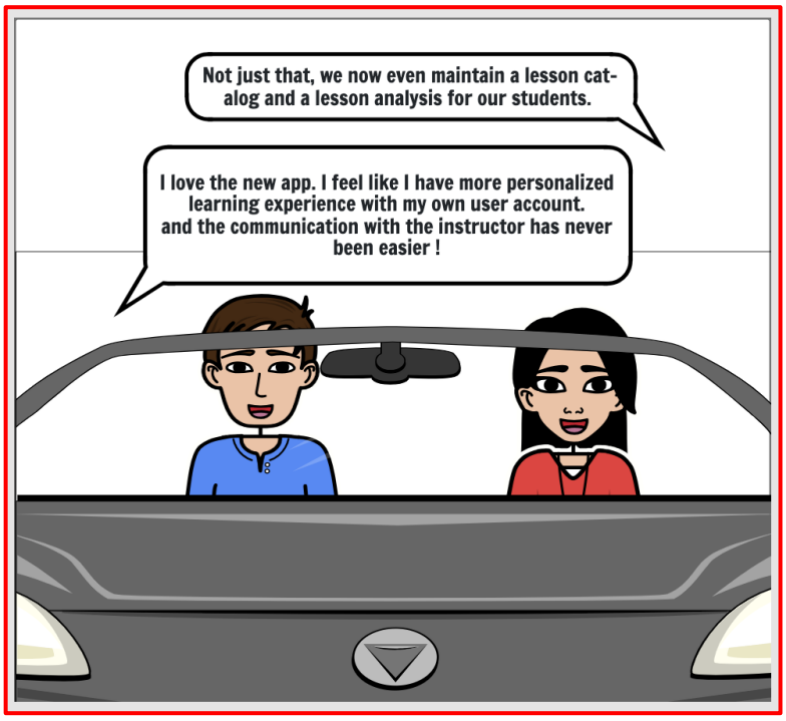

Figure 7 - Storyboard Part 5

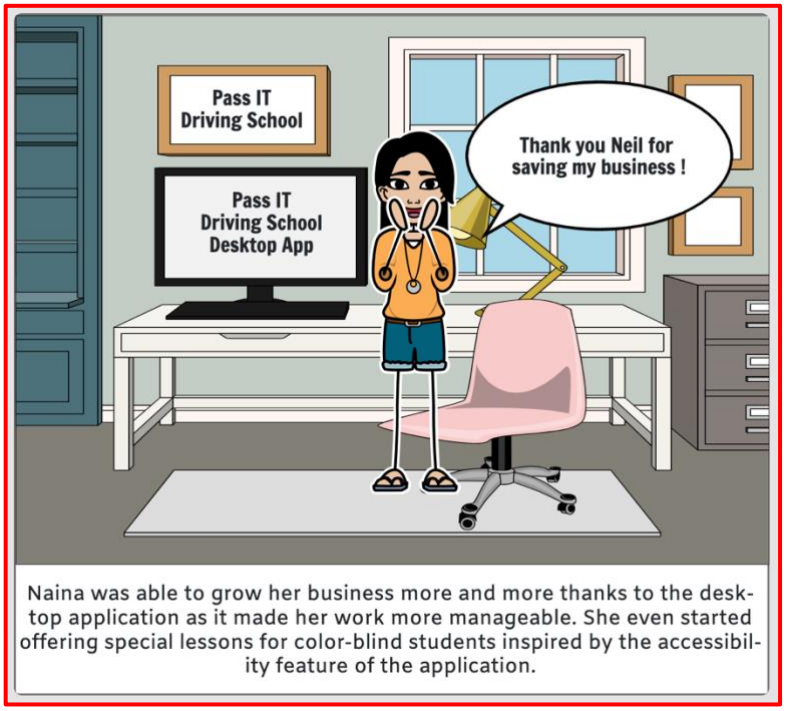

Figure 8 - Storyboard Part 6

2.5 Technical Design - Code

2.5.1 Application Termination

```
Private Sub PictureBox2_Click(sender As Object, e As EventArgs) Handles PictureBox2.Click
    MessageBox2.Show()
    MessageBox2.MessageName.Text = "Close application"
    MessageBox2.Message.Text = "Do you want to exit ?"
    MessageBox2.Button1.Text = "Cancel"
    MessageBox2.Button2.Text = "Yes"
End Sub
```

Figure 9 - Application termination code

When the exit button is clicked in the dashboard, this code will trigger a warning message in order to get the confirmation for the application termination from the user. This eliminated the potential effects that can happen to the user experience by user's slips and mistakes.

2.5.2 Error provider

```
2 references
Public Class FormEdit

    Private errorProvider As New ErrorProvider()
    0 references
    Private Sub IconButton1_Click(sender As Object, e As EventArgs) Handles IconButton1.Click
        errorProvider.Clear()
        If String.IsNullOrEmpty(TextBox1.Text.Trim) Then
            errorProvider.SetError(TextBox1, "Please enter a value")
            TextBox1.Focus()
        End If

        If String.IsNullOrEmpty(TextBox2.Text.Trim) Then
            errorProvider.SetError(TextBox2, "Please enter a value")
            TextBox2.Focus()
        End If

        If String.IsNullOrEmpty(TextBox3.Text.Trim) Then
            errorProvider.SetError(TextBox3, "Please enter a value")
            TextBox3.Focus()
        End If

        If String.IsNullOrEmpty(TextBox4.Text.Trim) Then
            errorProvider.SetError(TextBox4, "Please enter a value")
            TextBox4.Focus()
        End If

        If errorProvider.GetError(TextBox1) <> "" OrElse
                errorProvider.GetError(TextBox2) <> "" OrElse
                errorProvider.GetError(TextBox3) <> "" OrElse
                errorProvider.GetError(TextBox4) <> "" Then
```

Figure 10 – Error provider raising error indications 1

```
        If errorProvider.GetError(TextBox1) <> "" OrElse
                errorProvider.GetError(TextBox2) <> "" OrElse
                errorProvider.GetError(TextBox3) <> "" OrElse
                errorProvider.GetError(TextBox4) <> "" Then

            MessageBox1.Show()
            MessageBox1.MessageName.Text = "Error Null Values"
            MessageBox1.Message.Text = "Form has null values !"
            MessageBox1.MessageButton.Text = "Ok"
        Else
            Me.Hide()
            MessageBox1.Show()
            MessageBox1.MessageName.Text = "Form Submitted"
            MessageBox1.Message.Text = "Successfully Submitted !"
            MessageBox1.MessageButton.Text = "Ok"
            DashboardAdmin.Show()

        End If

    End Sub
End Class
```

Figure 11 - Error provider and error and success messages

An error provider is implemented to check null values before the form submission. The code will also generate error messages and successful messages. This also reduces the impact of slips and mistakes of the user.

2.5.3 Dashboard Division according to the user

```
2 references
Public Class MessageBox1

    4 references
    Public Property DashboardType As String

    0 references
    Private Sub MessageButton_Click(sender As Object, e As EventArgs) Handles MessageButton.Click
        If MessageName.Text = "Successful Login" Then
            Me.Hide()
            If DashboardType = "Admin" Then
                DashboardAdmin.Show()
            ElseIf DashboardType = "Student" Then
                Dashboard.Show()
            End If
        Else
            Me.Hide()
        End If
    End Sub

    0 references
    Private Sub Timer1_Tick(sender As Object, e As EventArgs) Handles Timer1.Tick
        Me.BringToFront()

    End Sub
End Class
```

Figure 12 - Opening the dashboard depending on the user login

After logging in to the dashboard must appear for each user based on their user type. Here in this code, we have simplified the code and implemented it in the message form itself to maintain reusability, simplicity and performance. The code triggers the dashboard form depending on a variable passed through the login form.

2.5.4 Forms triggered by another form under sub-modules

```
0 references
Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
    Me.Hide()
    Dim parentForm As DashboardAdmin = TryCast(Me.ParentForm, DashboardAdmin)

    ' Check if the parent form reference is not null and if the panel exists
    If parentForm IsNot Nothing AndAlso parentForm.PanelDesktop IsNot Nothing Then
        StudentReq.TopLevel = False
        StudentReq.Dock = DockStyle.Fill
        parentForm.PanelDesktop.Controls.Clear()
        parentForm.PanelDesktop.Controls.Add(StudentReq)
        StudentReq.Show()
    End If
End Sub
```

Figure 13 - Form triggered by another form in a sub-module

The forms triggered by other forms in the sub-modules are also opened in the same panel inside the dashboard using this code. This helps to maintain the consistency throughout the design.

2.6 Technical Specifications

File formats: To ensure the application's performance and considering the loading time, the design uses JPEG file format for most of its elements.

Using consistent design patterns: The design has used consistency throughout its elements, including buttons, tables, icons, etc., with a fixed color palette to simplify the learnability of the design for future developers.

Maintainability: As the design uses a minimum amount of file formats and consistent design patterns for its elements, the maintenance of the application is more comprehensible.

2.7 Consideration of design rules

Modularity: The design has two main modules separated depending on the user types: student and admin. The module student contains all the functionalities needed for the student user, which is divided into five sub-modules: lesson materials, schedules, progress, bookings, and contacting the instructor. The module admin contains every functionality the admin user requires and is divided into six sub-modules: student details, instructor details, schedule details, progress upload, lesson material upload, and request viewing. The sub-modules are identified depending on the functional requirements of the application.

Reusability: The design is implemented using a fixed color palette, which has improved the reusability of the colors, and only five button styles are repeated throughout the design. Almost all the elements are created using Rounded rectangles and rectangles except the icons. Different validation, confirmation, and message boxes are generated using the same form through a code that implements consistency throughout the message boxes. Forms are triggered inside the respective dashboards, minimizing the use of buttons for navigation.

2.8 Justification of the design

The design development was completed under the five stages of the design process under user-centered design. This allowed the solution to be built based on the user's requirements and objectives, addressing the user experience (Anon., n.d.). During the first stage of empathizing, the two personas were created to understand the users and their requirements better.

With the above Figure and Figure personas, the above-mentioned requirements were defined. After ideating the design through brainstorming, the design prototype was created in the visual studio. It was tested with the support of a driving school administrator and a student.

The HCI principles were considered under three main categories: Learnability, Flexibility, and Robustness (Dix A, Finlay J, n.d.), for the design. Concerning the novice users, learnability was implemented using familiarity and consistency. The overall design of the solution supports user's existing knowledge as it uses familiar designs for buttons, validation message boxes and login form etc. Consistency is implemented throughout design elements such as buttons, message box forms, Headings and subheadings, tables, etc. Every form submission has been implemented to look and work the same throughout the application, supporting users in consistently behaving. This can be mainly observed in the admin user module, where the user controls the details of three modules: student, instructor, and schedules.

Considering the exchange of information between the user and the application (Anon., 2004)flexibility is implemented using task migratability. Almost all the buttons generate validation, confirmation, or messages depending on their functionality, including the login user interface with credential validations. This allows the app to decrease the effect of human mistakes and slips on the overall user experience.

Robustness is implemented using responsiveness to enhance the communication between the user and the application (Dix A, Finlay J, n.d.).Every button that performs CRUD operations indicates the success or failure status of the process.

One primary consideration of HCI principals in this design was accessibility. As the empathizing stage indicated the possibility of users with color blindness, the design was developed using a fixed color palette that is color-blind friendly.

The sans serif fonts of "Neue Haas Unica" and "Titillium Web" are used throughout the design with a minimalist theme using fewer number of elements to maintain efficiency and digestibility. Maintaining a minimalist theme over the design has also given the ability to implement simplicity in the application.

- **Git Hub Repository: https://github.com/Mishy34/Driving-School-Desktop-App.git**

Instruction to run the application :

- Change the recommended display scale and layout to 200% by clicking on the Advanced scaling settings inside Display properties of windows. (You will need to logout and login for the settings to apply)
- After cloning the project from the repository kindly change the branch to "Version-9" from "masters"
- In the application properties change the Target framework and Target OS to .net 7.0
- Rebuild the app and you are ready to run.

Credentials to login :

Students User

Username : student@gmail.com

Password : Newpw1@123

Admin User

Username : admin@gmail.com

Password : Newpw2@123

3. Implementation

3.1 Login User Interface

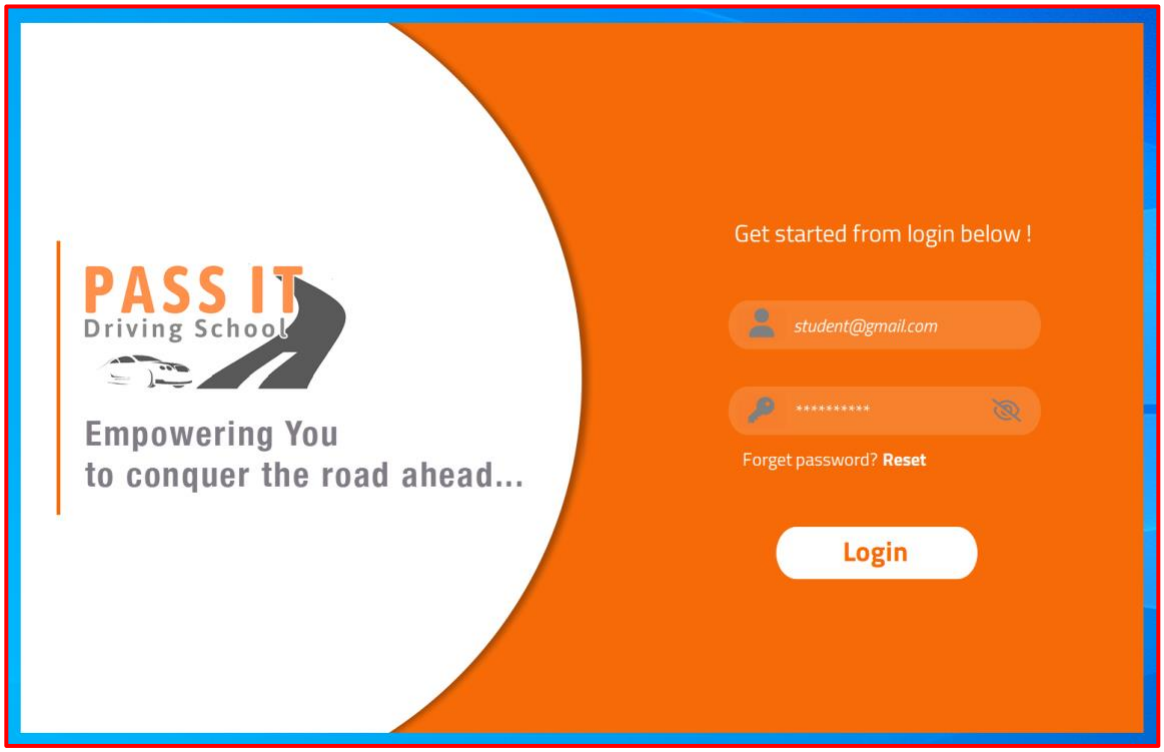

Figure 14 - Login UI

The login user interface is the opening interface of the application. The user needs to enter the credentials and login to use the application. The password is hidden behind "*" characters, but if needed, the user can unhide and view the password using the "Eye" icon. When the login button is clicked the credentials are validated and a successful or failure message box will appear.

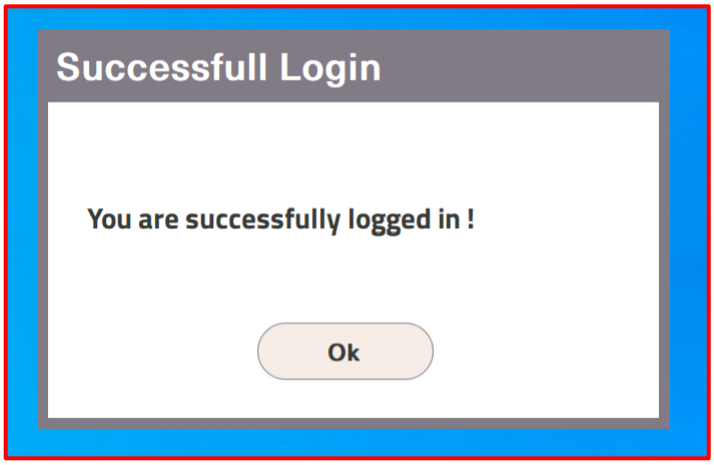

Figure 15 - Validation message box for successful login

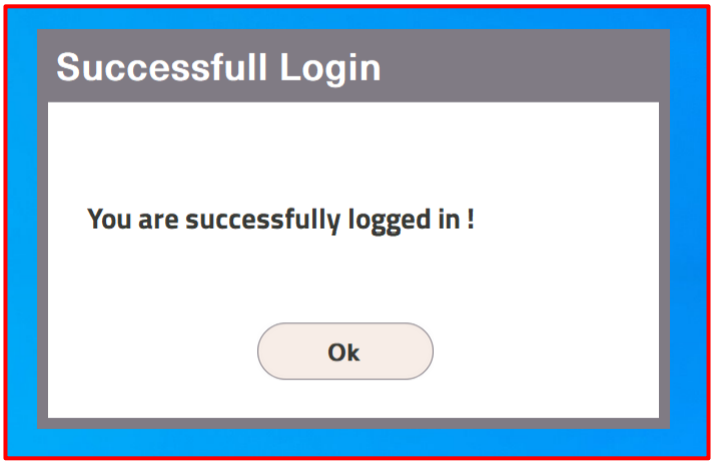

Figure 16 - Validation message box for failed login

3.2 Student module

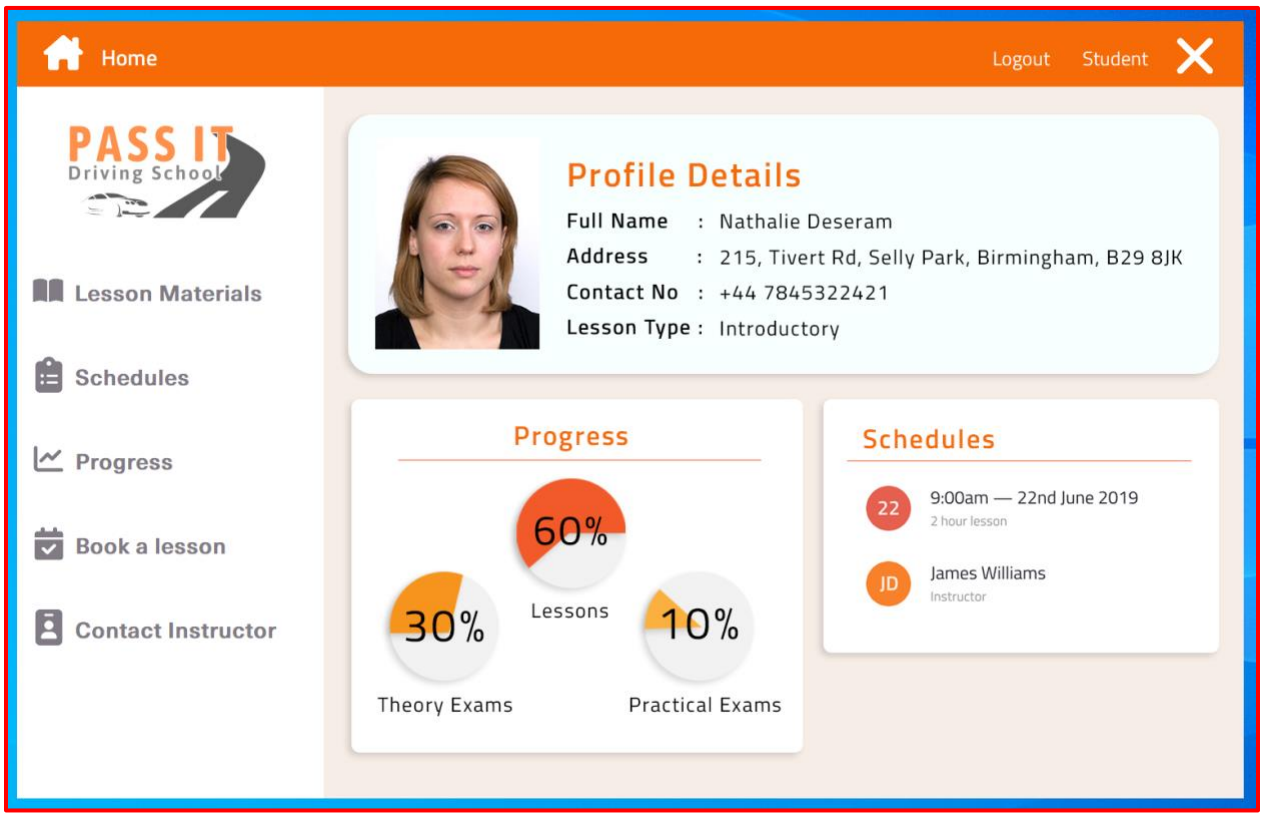

Figure 17 - Student user dashboard

Student user login credentials will redirect the user to the student dashboard. The student dashboard gives an overview of the student profile, the progress, and the upcoming schedules. It also includes a logout button in the upper ribbon, including a close button for the application

termination. The side menu has five buttons giving access to the five sub-modules of the student module: Lesson Materials, Schedules, Progress, Book a Lesson, and Contact Instructor.

3.2.1 Lesson materials sub-module

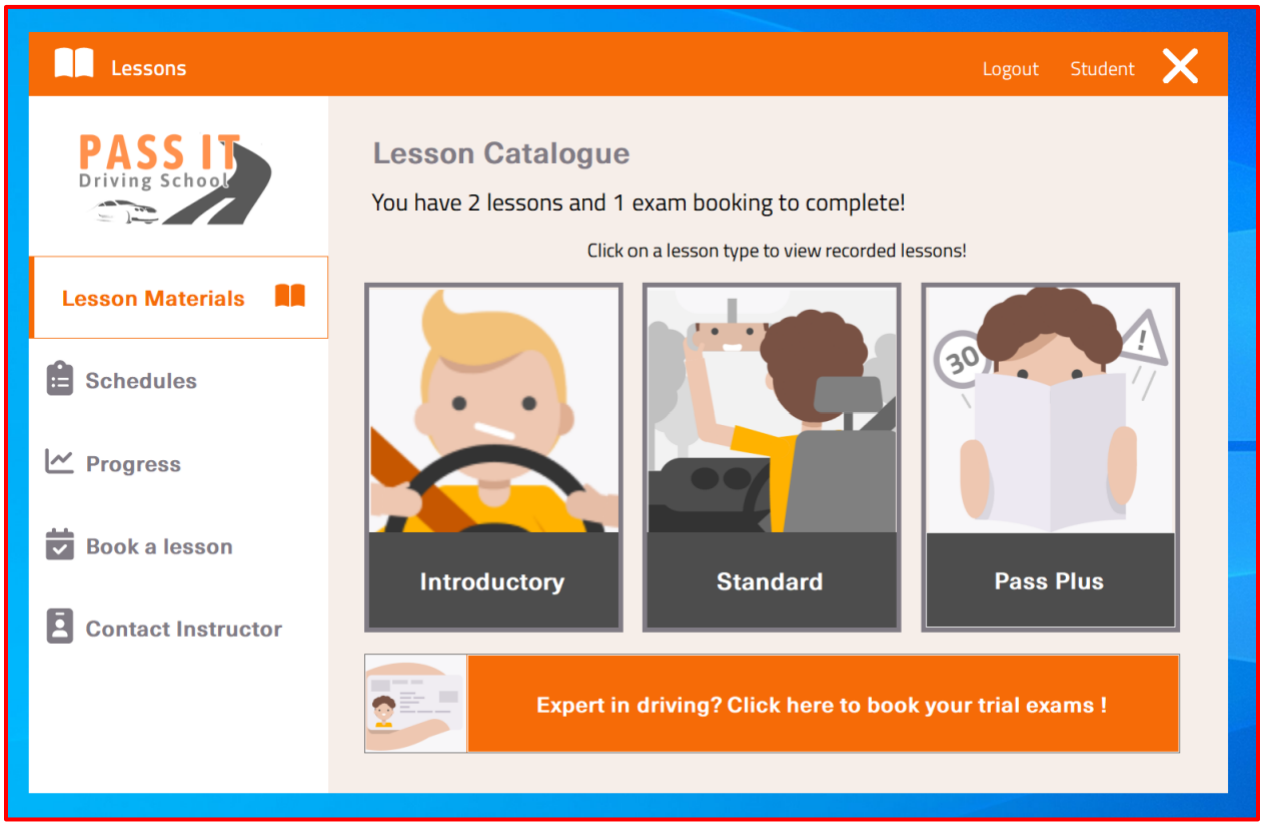

Figure 18 - Lesson materials UI

The lesson materials user interface allows users to navigate through different lesson types available. If the user has booked only introductory level lessons, the user is only allowed to view those lesson materials.

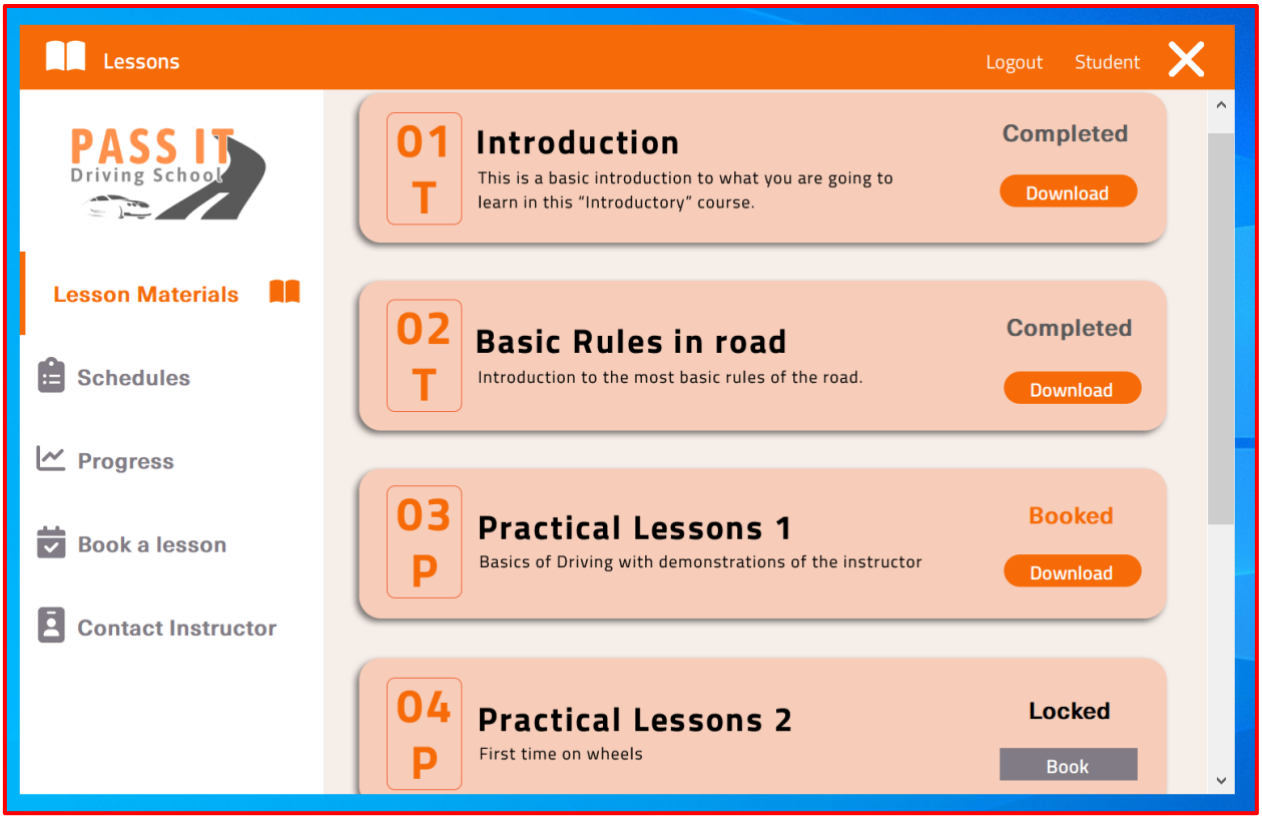

Figure 19 - Lesson materials in Introductory lesson type UI

The lesson materials in the introductory lesson type user interface allow users to download lessons and view lesson status. When users click the download button, a confirmation message indicates the successful download of the lesson. When users click the book button, a message pops up allowing the user to book the lesson or to undo the click by clicking cancel.

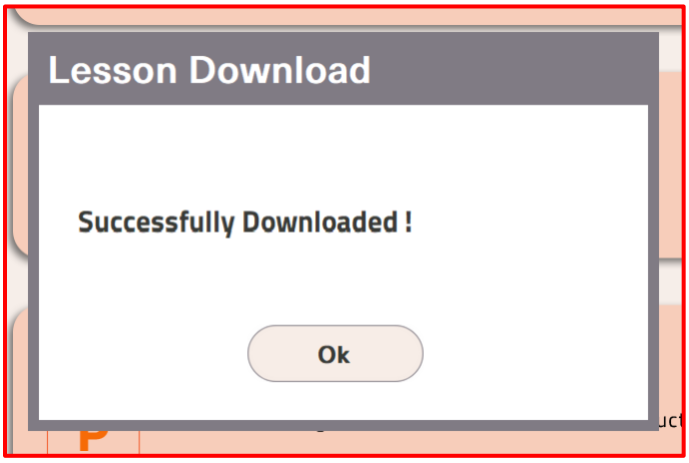

Figure 20 - Confirmation message for successful download of the lesson

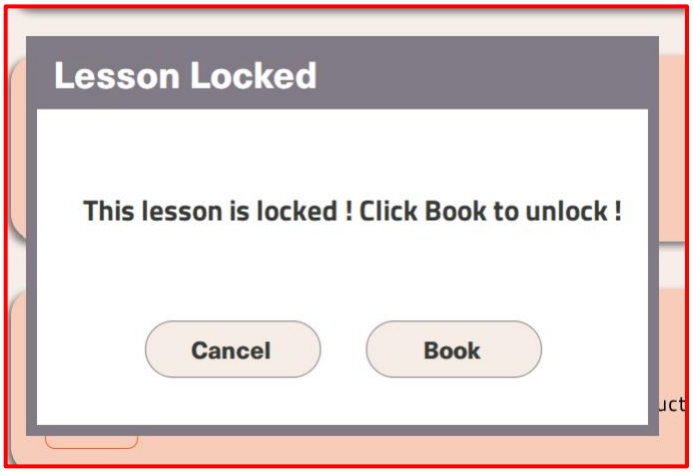

Figure 21 - Lesson locked message box

If users click on the button book, a confirmation message indicates the request has been made to the administration.

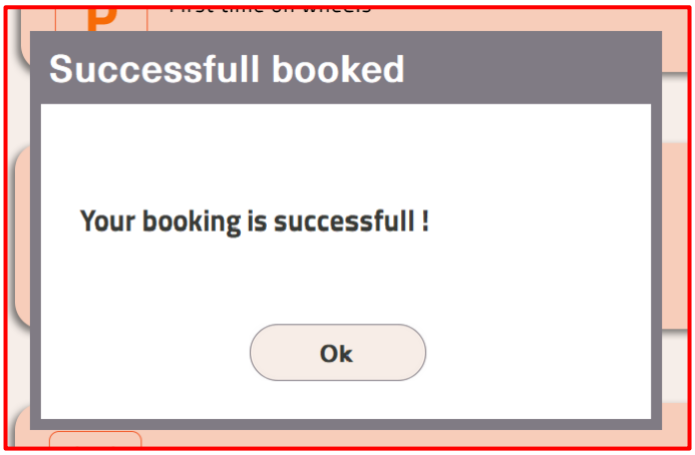

Figure 22 - Booking successful confirmation

3.2.2 Schedules sub-module

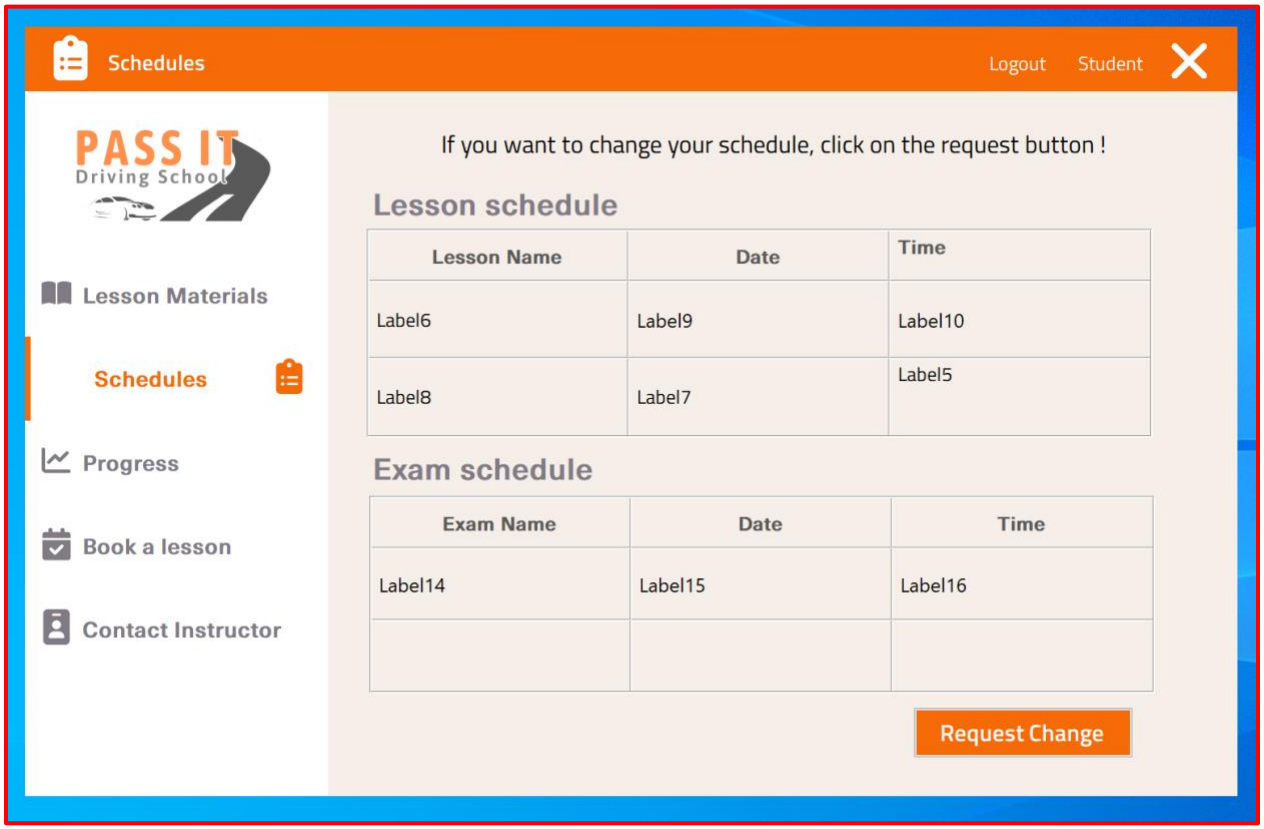

Figure 23 - Schedule UI

The schedules module gives an overview of the lesson and exam schedules booked for the user. If the users want any changes, they can click on the request change button.

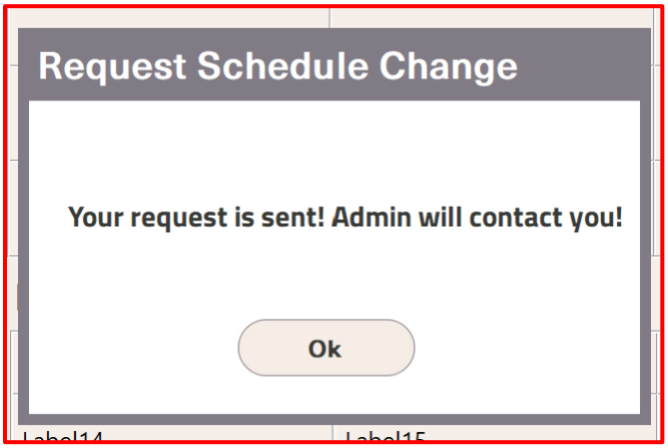

Figure 24 - Request schedule change confirmation message

3.2.3 Progress module

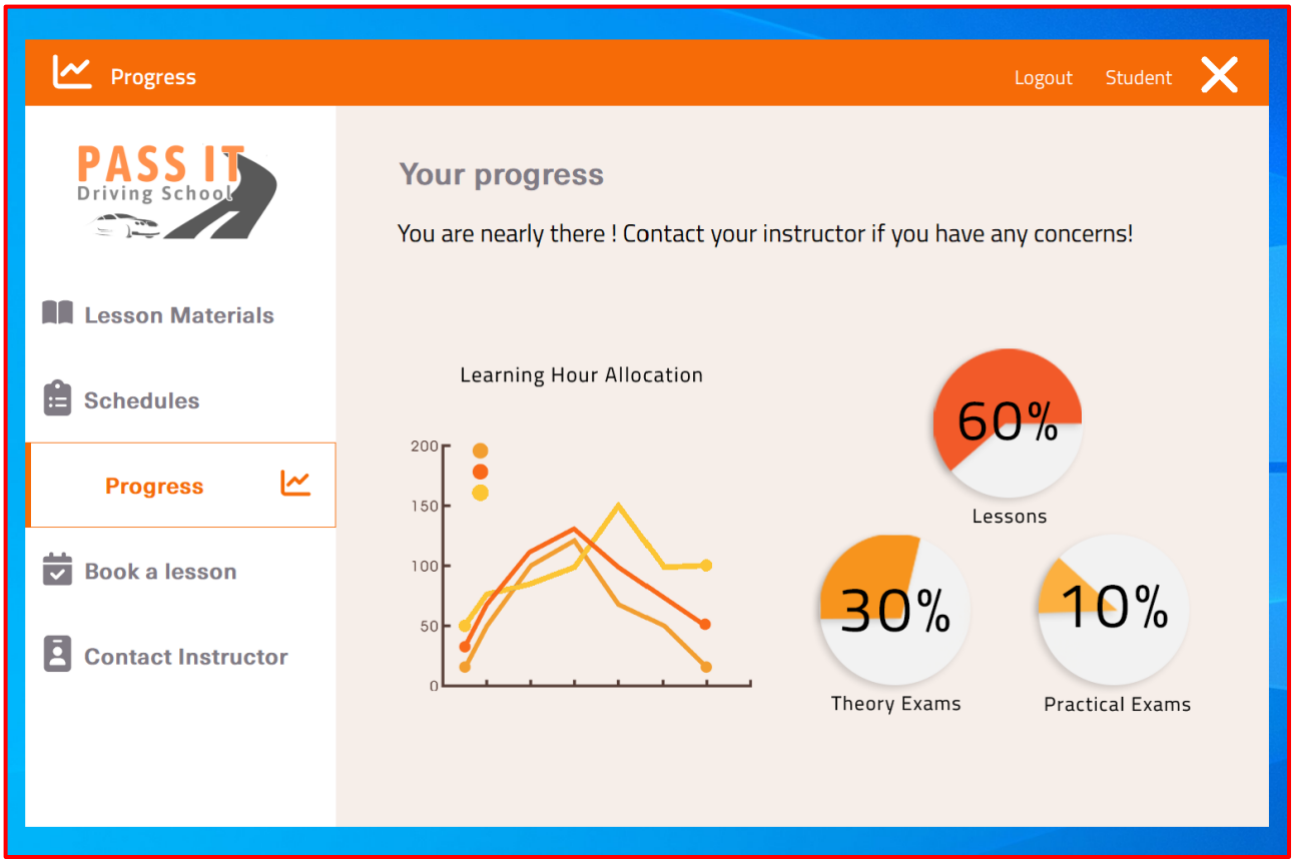

Figure 25 - Progress UI

The progress user interface shows an overview of lessons and exams the user attempts.

3.2.4 Book a lesson

Figure 26 - Booking a lesson UI

Book a lesson user interface allows users to book lessons by clicking on the book button.

3.2.5 Contact Instructor

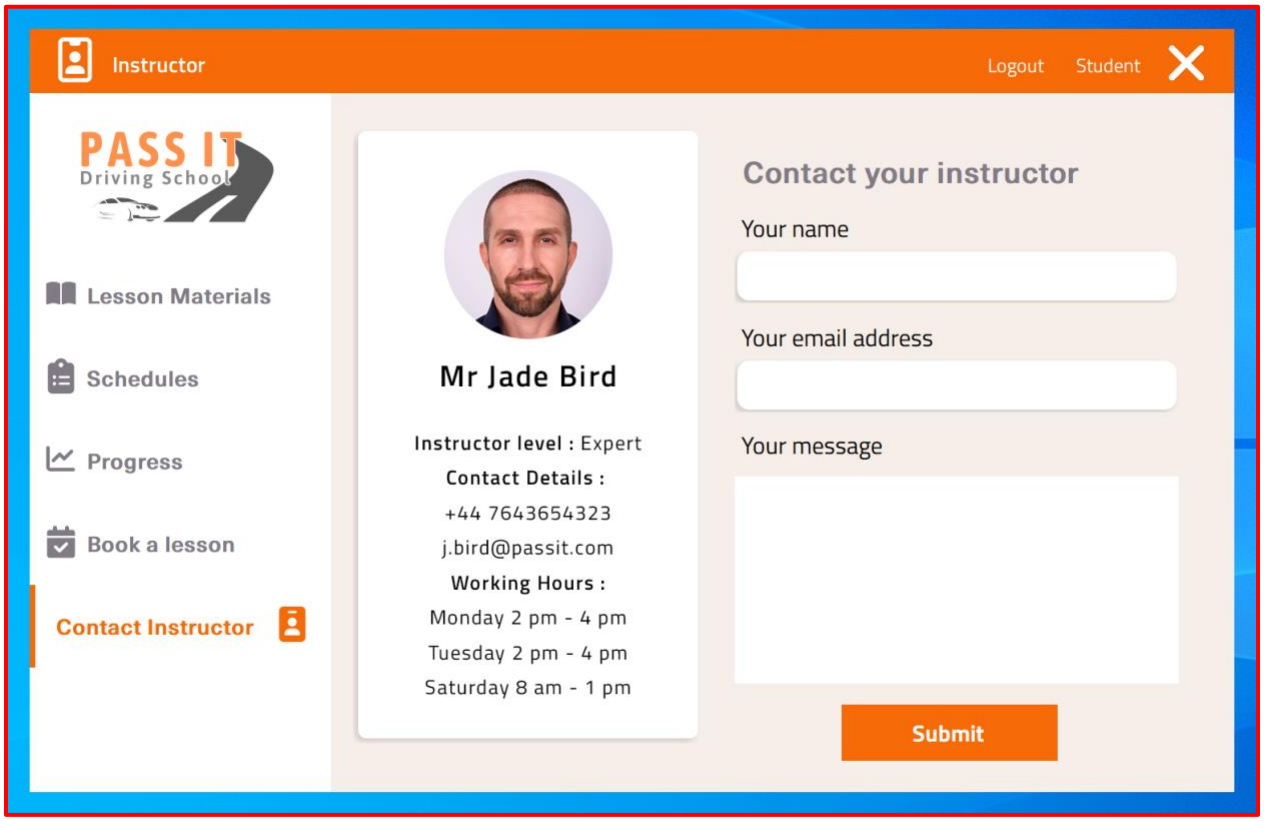

Figure 27 - Contact Instructor UI

In the contact instructor module, the users can view their instructor's details and contact them directly using the contact form.

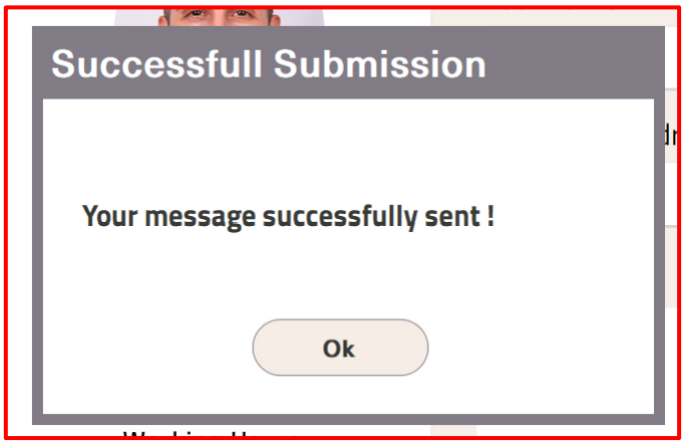

Figure 28 - Successful Message sent

3.3 Admin module

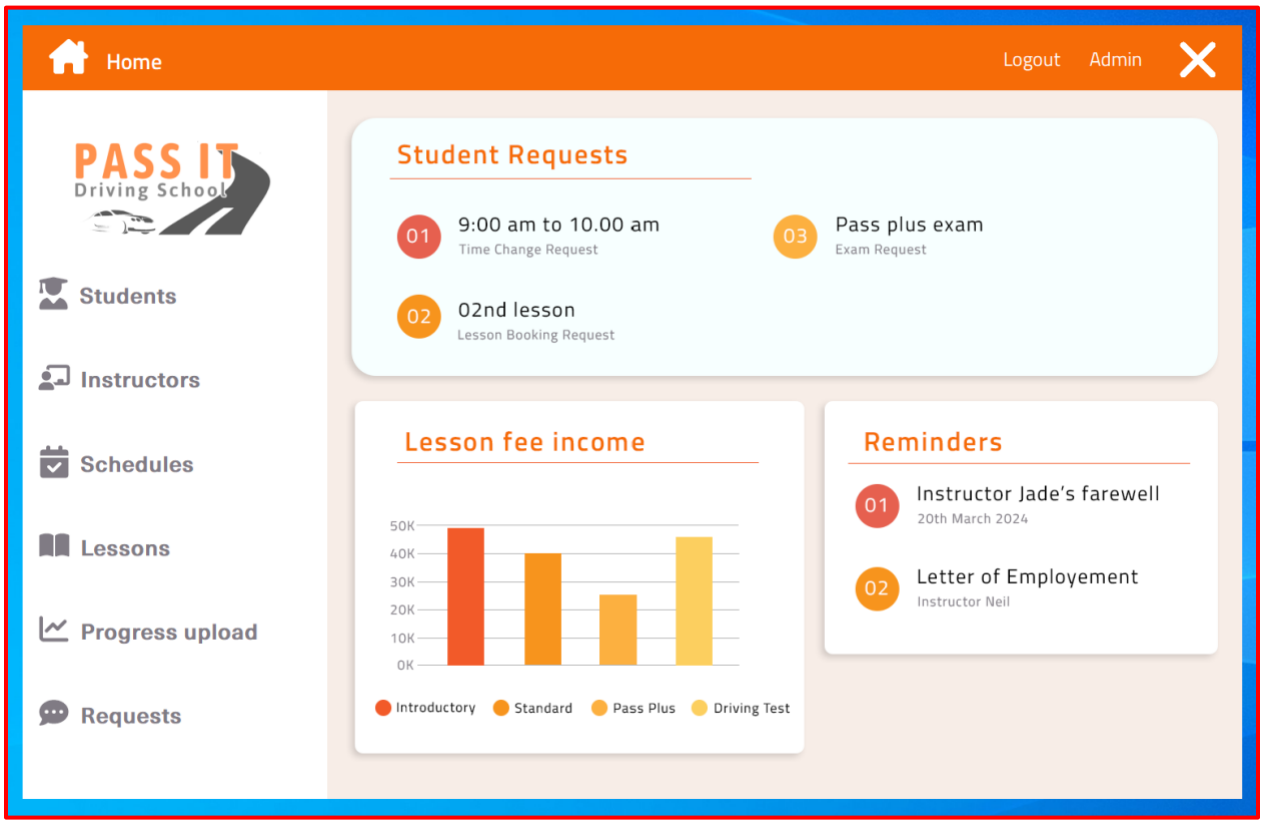

Figure 29 - Admin Dashboard UI

After successfully logging in using the admin credentials, the users will be directed to the admin dashboard. This includes an overview indicating the latest student requests, reminders, and the income chart of the lessons. It allows the users to log out using the logout button and terminate the application using the close button. The side menu gives access to the six sub-modules: Student details, Instructor details, Schedule details, Lesson materials, Progress upload, and request view.

3.3.1 Student details module

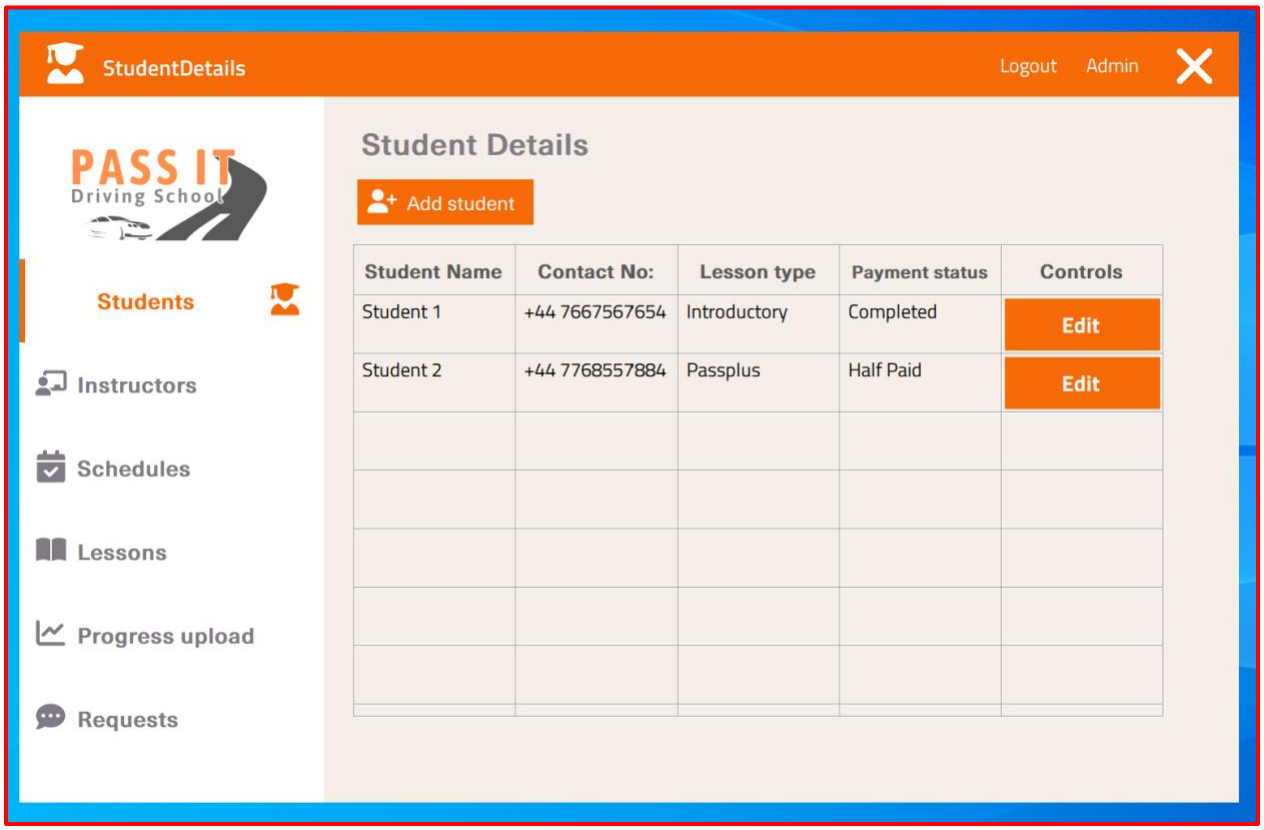

Figure 30 - Student Details UI

The student details user interface allows users to add new students and edit existing student details.

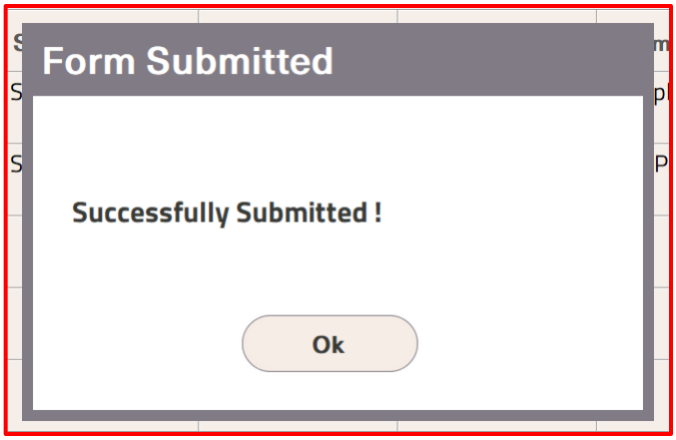

Figure 31 - New student form submission

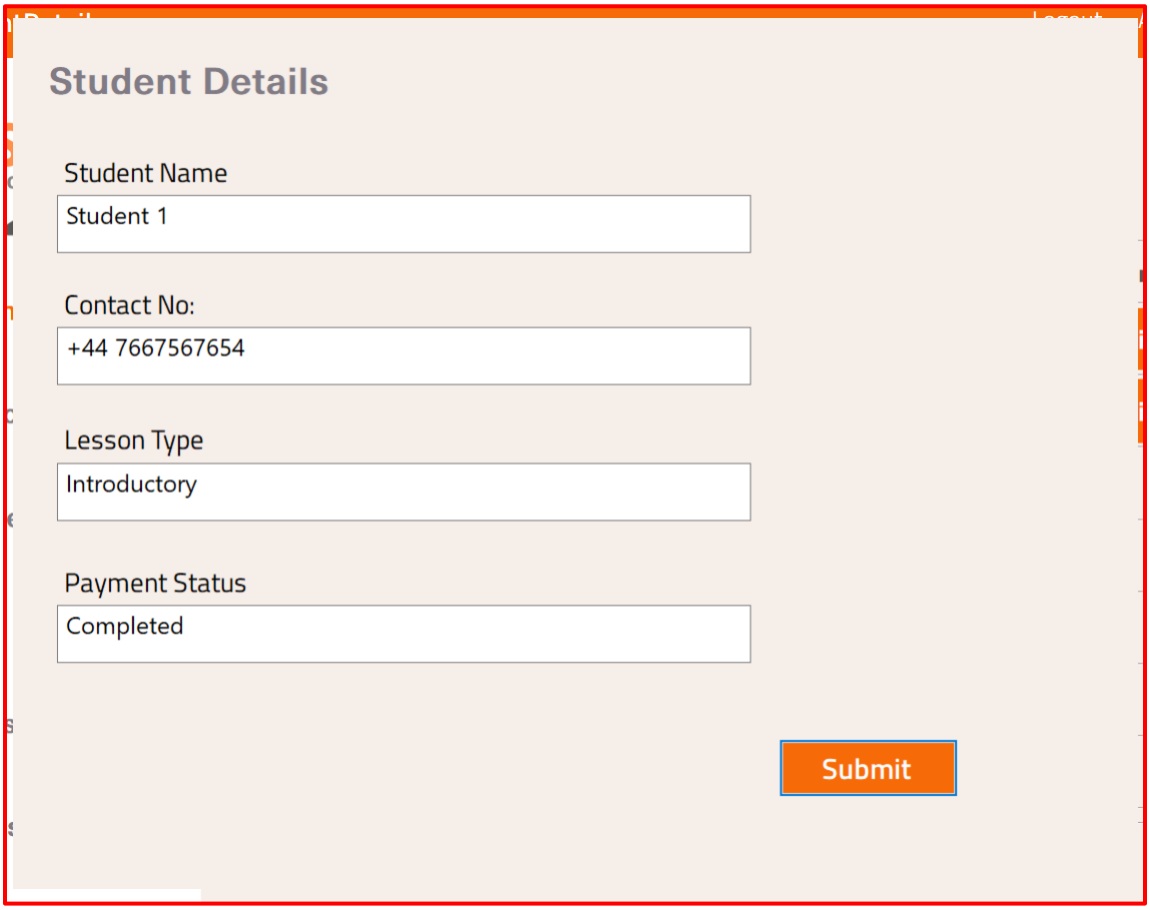

Figure 32 - Edit existing student details

3.3.2 Instructor details module

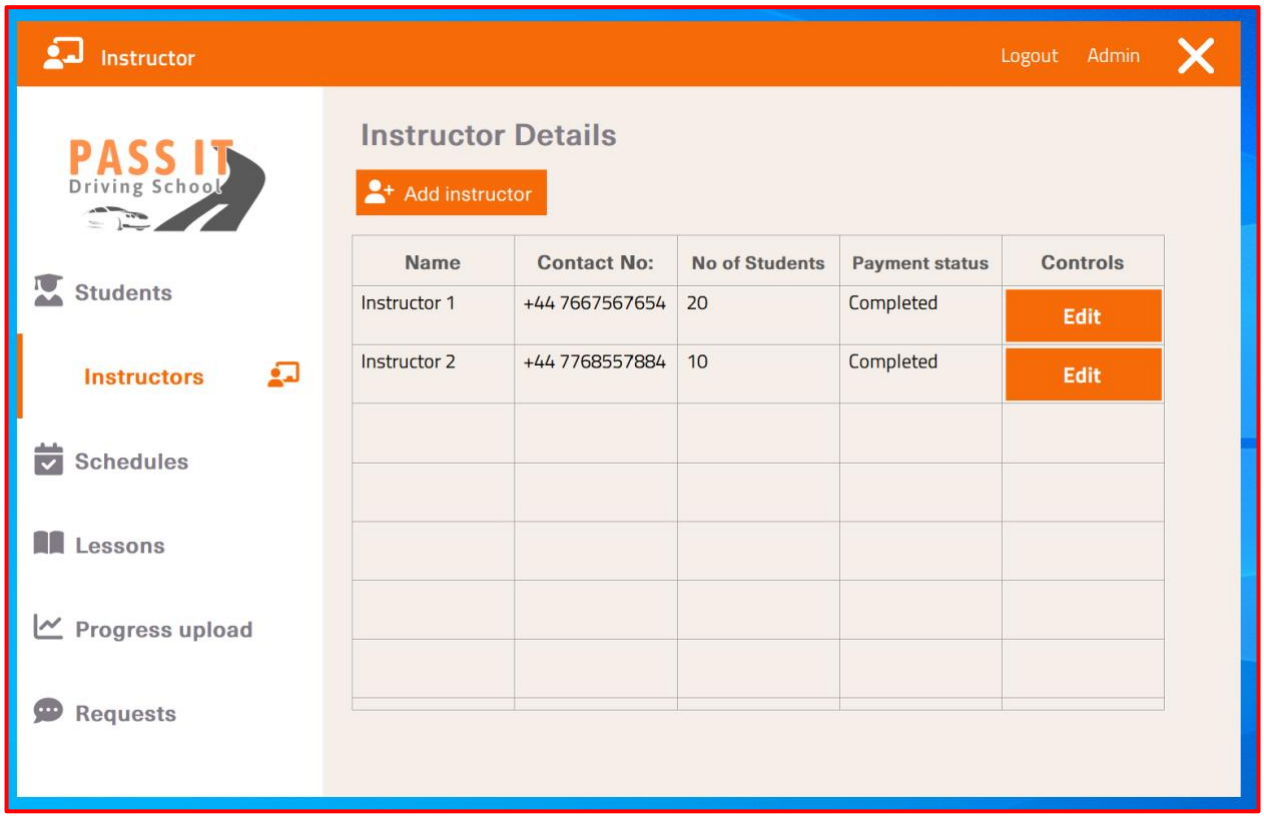

Figure 33 - Instructor details UI

Instructor details UI allows the user to add new instructors and edit the existing instructors.

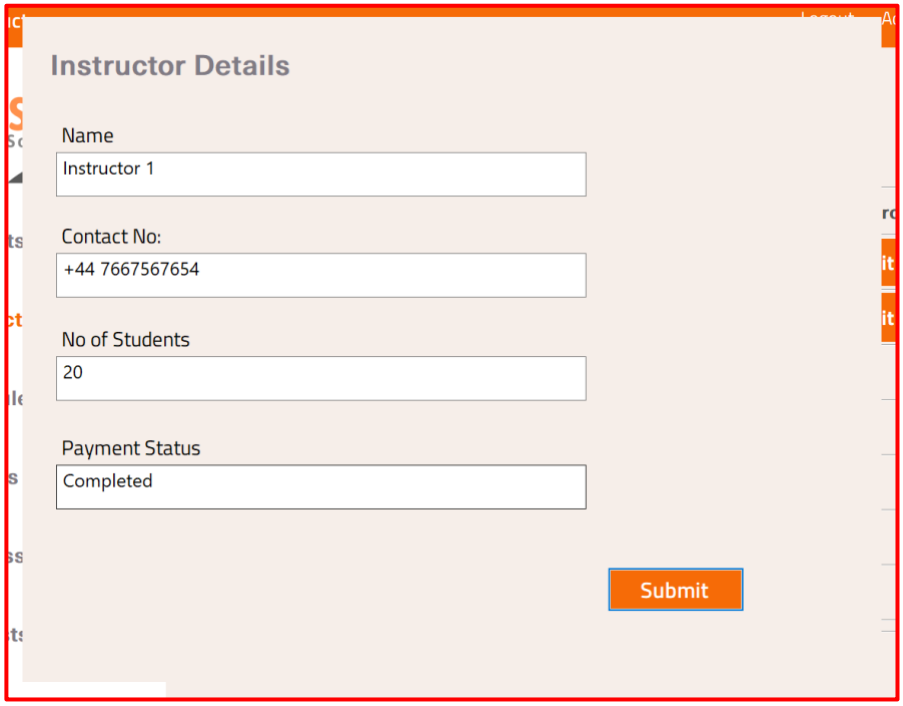

Figure 34 - Edit instructor details UI

3.3.3 Schedule details module

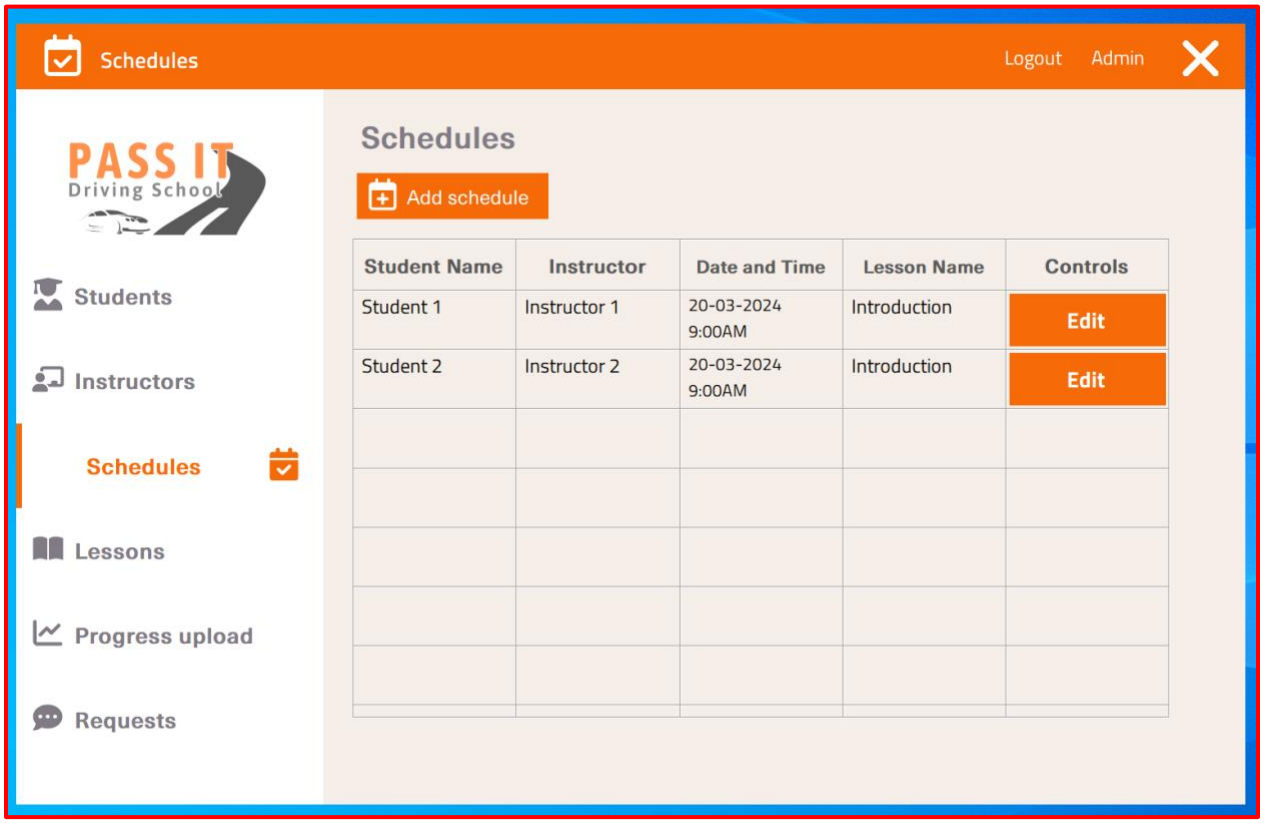

Figure 35 - Schedule details UI

Schedule details UI allows users to add new schedules and edit existing schedules.

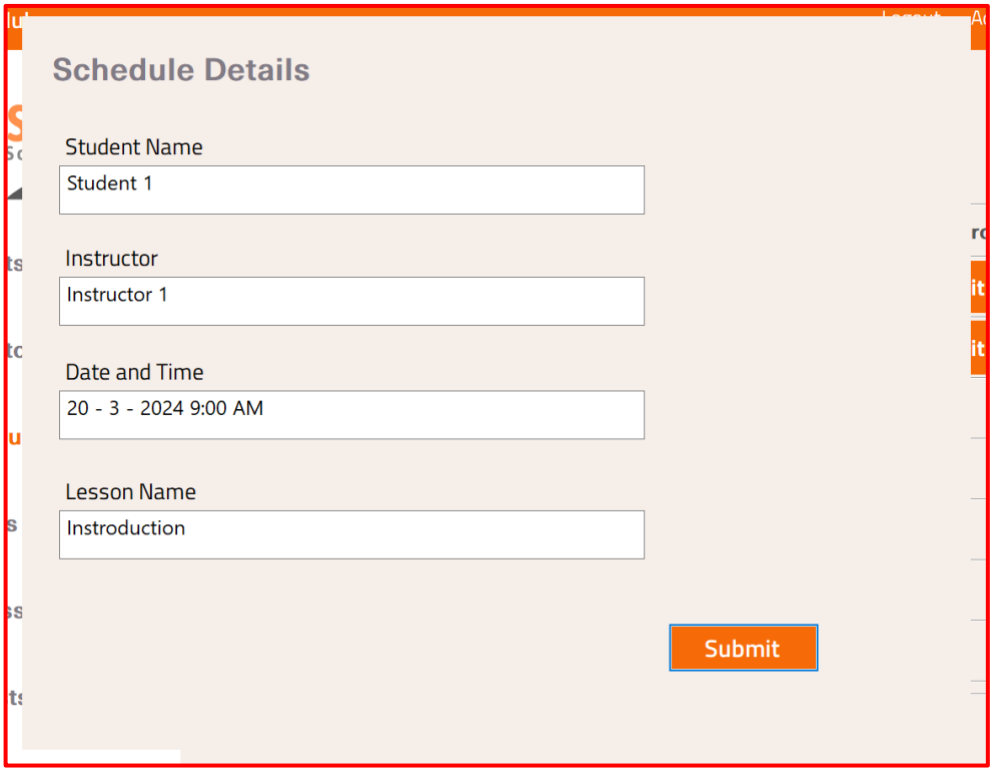

Figure 36 - Schedule edit UI

3.3.4 Lesson upload and progress upload

Lesson upload and progress upload UIs allows user to upload lessons and progresses of each students.

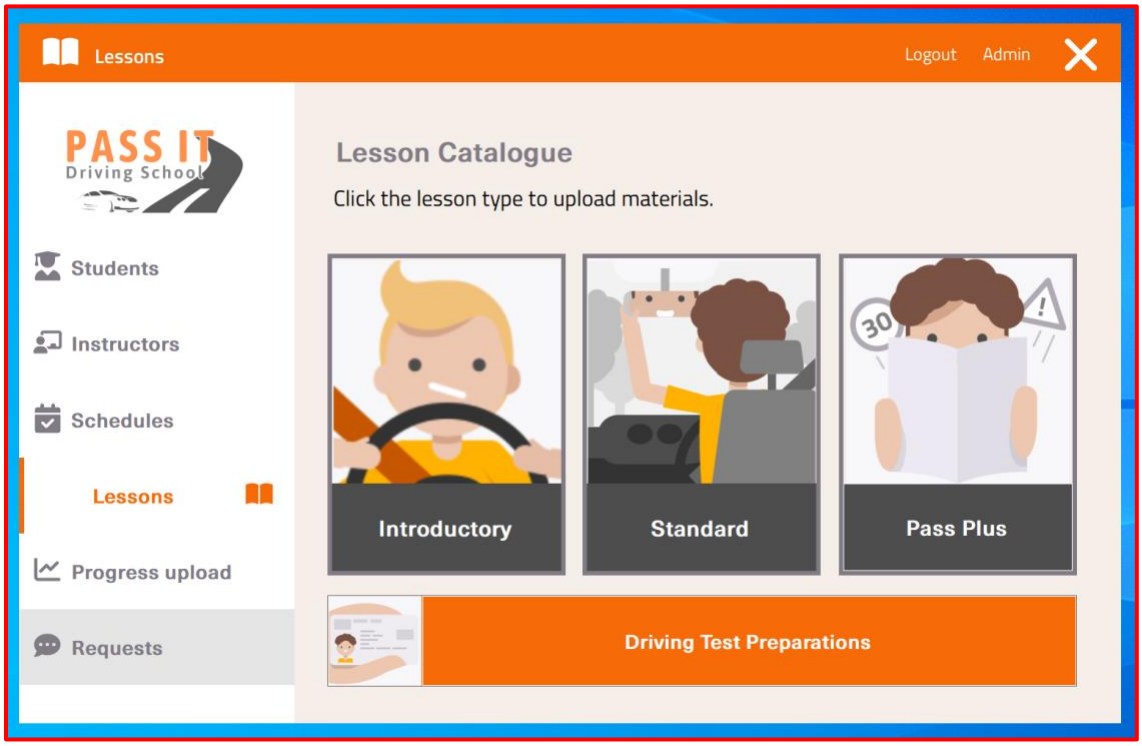

Figure 37 - Lesson upload UI 1

Figure 38 - Lesson upload UI 2

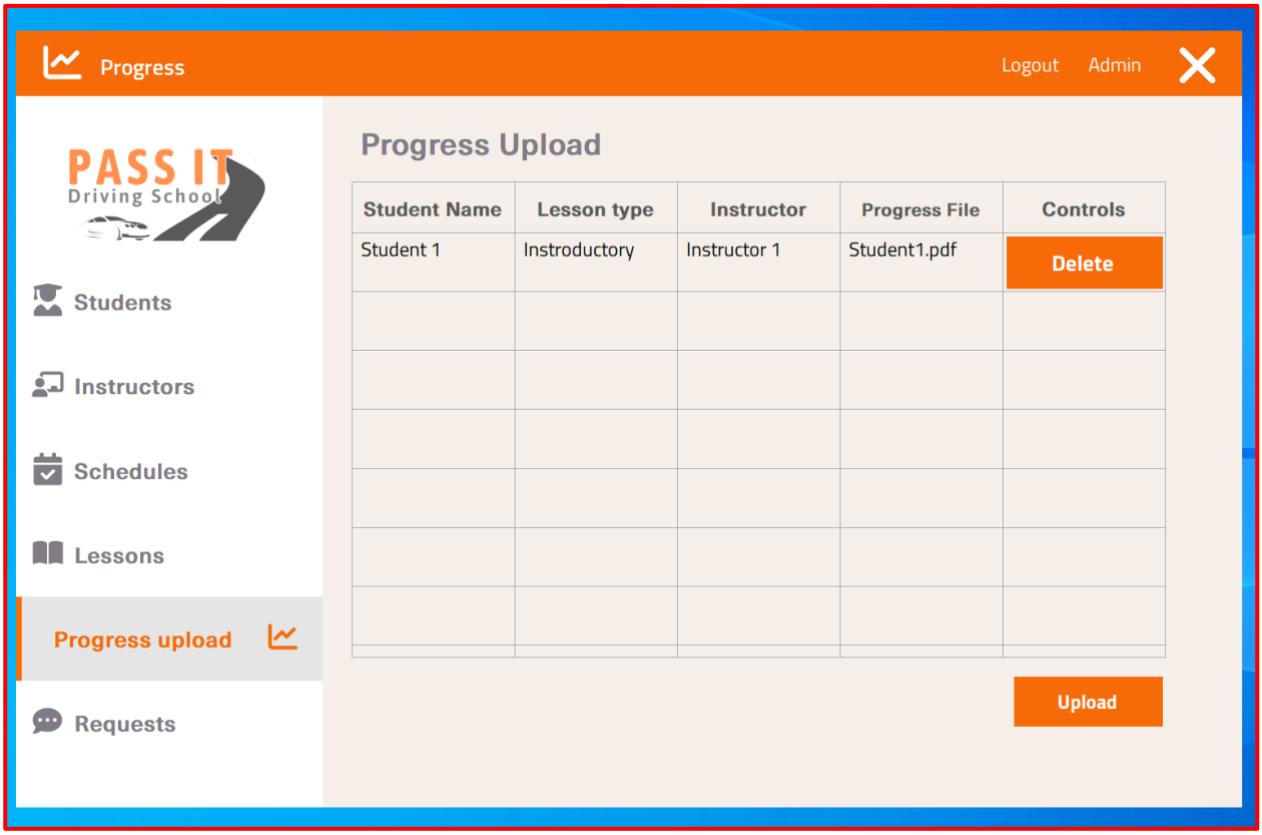

Figure 39 - Progress upload

3.3.5 Request view module

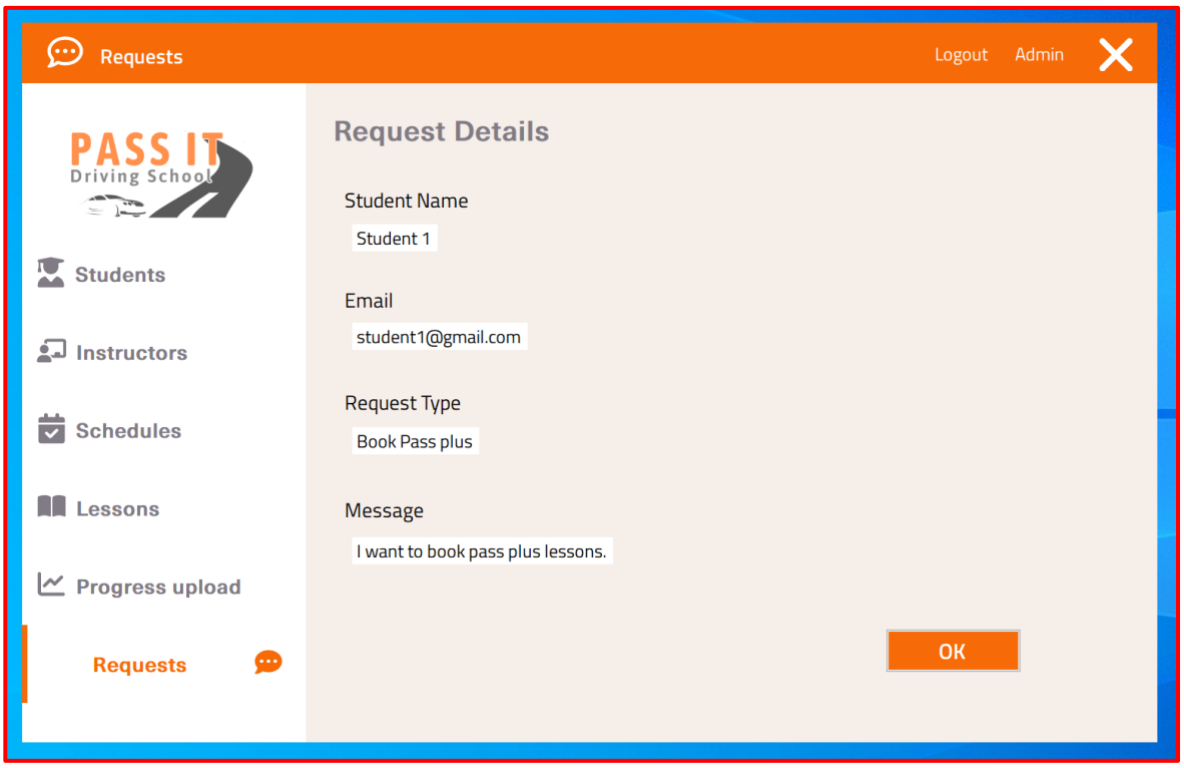

Figure 40 - Request View UI 1

Review user interface allows users to view students' requests.

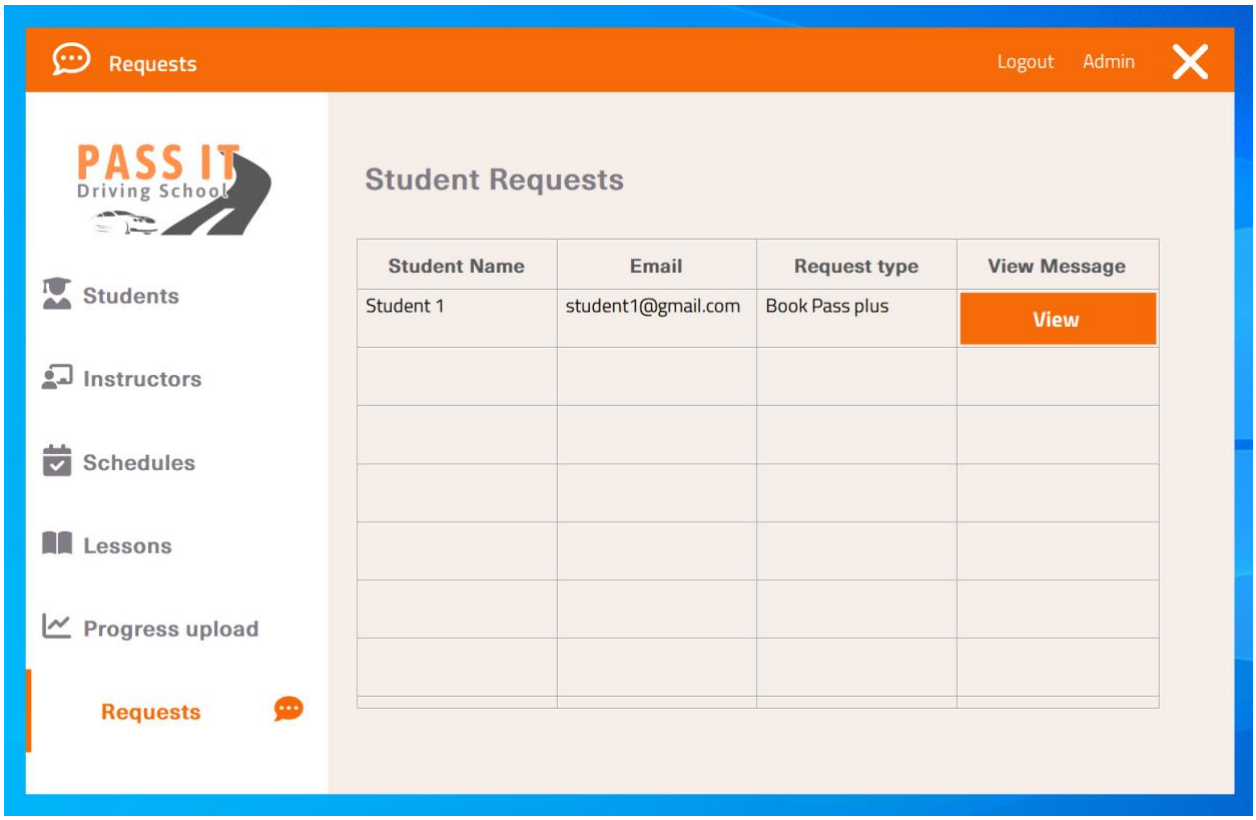

Figure 41 - Review view UI 2

4. Evaluation

4.1 Suitability for the target audience

The solution is implemented and designed with a user-centered approach, which focuses on different types of users and their requirements. The design had a development process of empathizing, defining, ideating, prototyping, and testing stages, allowing the design to build its base on the personas created in the first stage. Due to this, users have a very direct and robust interaction with the application.

Though the design does have accessibility for color-blind people, it doesn't address any other popular accessibility features such as theme customization, light, and dark toggle buttons, zoom options, etc. Another weakness of the design is not having the capacity for customization.

The design can be improved more by properly studying different users with different accessibility requirements. Conducting accessibility testing with the aid of the study would impact identifying the potential challenges in improving accessibility.

4.2 Usability

The user receives feedback at every milestone they arrive at during the user journey. These feedbacks consist of validations, confirmations and other messages depending on their purpose. The overall design maintains consistency throughout all the elements.

Though the minimalist design does provide simplicity, it can also be confusing to navigate as it does not provide much direction through elements.

The design elements need to be more optimized to provide more guidance toward intuitive navigation.

4.3 Maintainability

Modularity is maintained throughout the whole solution as it has two main modules depending on the two user types of the system. The two main modules are then divided into sub-modules depending on the functional requirements of each.

The solution doesn't have proper version control implemented at the moment, which makes it vulnerable and challenging when updating.

Implementing a proper version control using a version control system like Git will allow the solution to easily upgrade without potential damage to the existing version.

4.4 Portability

The solution is not portable at this level. It needs to be implemented with responsiveness and cross-platform compatibility. Reducing the dependencies and implementing code for responsiveness needs to be done.

4.5 Robustness

Users receive validation messages in login as well as in other interactions where it is needed. It also allows users to recover from mistakes and slips, making the experience nearly error-less.

4.6 Constraints

The design proposal phase was challenging due to defining the scope of the solution. It was time-consuming, which limited the time allocated for the actual implementation. Limiting the project scope to prioritize the most crucial features allowed to handle the time in an effective manner.

4.7 Legal and Ethical Considerations

Legal and ethical considerations are essential for the overall user experience as they build up the user's trust in the solution. This solution needs to implement privacy compliance to communicate with the users about how their data will be used throughout the solution, enhancing more accessibility features, maintaining information transparency, and ensuring security measures are taken.

4.8 Strengths and weaknesses of the solution

The solution is a very strongly user-centered design developed under the five stages of the design process. The design exhibits consistency, which is an essential aspect of providing a better user experience. Familiarity with the design has implemented alignment with users' mental models and expectations. The design provides an intuitive user experience as it contains modularity, making the solution available for flexibility and scalability.

The solution has taken an approach to accessibility by addressing color-blind users' needs. However, it still lacks more accessibility features to cater to a wider audience with different needs. The solutions need to have more opportunities for the users to personalize their experience based on their preferences. It also needs to enhance the hierarchy of visual elements in order to provide more guidance and intuitive navigation inside the design. The design also needs to have more contextual help and documentation for user support. The design also needs to address its dependencies ensuring that the design is compatible through different platforms. The data visualization of the design needs to be more visually appealing as it represents lots of data while scaling. It's better if the design has a novice user onboarding feature, such as highlighting the key functionalities to them as they first log in. The progressive disclosure of the design needs to be enhanced as it can be used to keep information in control and interfaces uncluttered. The solution can also implement micro-interactions as an enhanced feature to provide visual feedback to the user's actions without causing distractions.

5. Conclusion

In conclusion, this assignment has navigated the process of producing a design document and proposing a design, implementing the proposed design, and evaluating and assessing the design for the Desktop application of the Pass IT driving school. The design phase of this assignment methodically addressed the requirements, producing a well-documented design. The documented design was implemented, turning it into a user-centered solution with intuitive interfaces, practically demonstrating the HCI principles. Finally, in the evaluation the implemented solution was analyzed to identify strengths, shortcomings and limitations while providing recommendations to enhance user experience in the future versions. The overall assignment offers a paradigm for driving schools who are seeking to deploy an HCI solution which will prioritize the user while meeting the requirements of the organization. The process through design to evaluation emphasizes the role of HCI principals and design rules in producing a comprehensive solution.

6. References

- Dix A, Finlay J, n.d. *Human Computer Interaction.* s.l.:2004.

- Anon., 2004. [Online]
  Available at: What is User-centered design?
  [Accessed 01 2024].

- Anon., n.d. *User Centered Design.* [Online]
  Available at: https://www.interaction-design.org/literature/topics/user-centered-design
  [Accessed 10 01 2024].