BLM4531 Final Proje Ödevi

Alp Ertunga Elgün

19290238

Rent-A-Car Sitesi

Github:

https://github.com/Sayuris1/rent_a_car_asp

SQL DATABASE:

MS SQL kullanıyoruz. Her table'da id, otomatik olarak artan identity column'u. 3 Adet table'imiz var:

Car_types → Her row ayrı bir araba modelini ifade ediyor. Columnlar ise:

id: Otomatik artan identity column'u. (int)

daily_cost: Arabanın günlük kiralanma ücreti. (int)

door: Arabanın kapı sayısı. (int)

passenger: Arabanın kaç yolcu alabileceği. (int)

transmission: Arabadaki vites turu. Herhangi kısa bir string olabilir ama programda sadece b elirli türlerle sınırladık. (string)

ac: Arabada klima var mı. (binary)

fuel: Arabada kullanılan yakıt turu. Herhangi kısa bir string olabilir ama programda sadece belirli türlerle sınırladık. (string)

type_name: Türün ismi. Resimlerin doğru gösterilmesi için /jpgs de bu isimde bir .jpg bulunmalı. (string)

Cars → Her row ayrı bir arabayı ifade ediyor. Columnlar ise:

id: Otomatik olarak artan identity column'u. (int)

type_id: Car_types table'ının id'sine bağlı one-to-many bağlantısının ucu. Car_types'da gelecerli bir id olmak zorunda. (int)

Bookings → Her row ayrı bir rezervasyonu ifade ediyor. Columnlar ise:

id: Otomatik olarak artan identity column'u. (int)

car_id: cars table'ının id'sine bağlı one-to-many bağlantısının ucu. Cars'da geçerli bir ad olmak zorunda. (int)

days_to_pick: Rezervasyonun kaç gün sonra başlayacağı. Eğer rezervasyon aktif ise negative olabilir. days_to_drop'dan büyük olamaz. (int)

days_to_drop: Rezervasyonun kaç gün sonra biteceği. Negatif olamaz. -1'olunca bu row silinmelidir. Days_to_pick'den küçük olamaz. (int) : Arabanın kapı sayısı. (int)

Bağlantılar EXISTS kullanılarak rahatça kullanılabilir.

Örnek olarak

```
WHERE NOT EXIST ( SELECT * FROM bookings WHERE cars.id = bookings.car_id AND NOT
(
        (
                bookings.days_to_pick >= {Session["days_to_drop"]}
                AND days_to_pick >= {Session["days_to_pick"]}
        )
        OR (
                bookings.days_to_drop <= {Session["days_to_drop"]}
                AND days_to_drop <= {Session["days_to_pick"]}
        )
)
)
```

Query'siyle bir arabanın seçilen aralıkta rezervesi olup olmadığını öğrenebiliriz.

Eğer pick_date, drop_date aralığı days_to_pick, days_to_drop aralığıyla kesişiyorsa bu aralıkta rezervasyon yapmamalıyız.

Ya da eğer iki değer de diğer iki değerden büyük ya da küçük ise rezervasyon yapabiliriz.

Master Page:

Navigation bar'ı ve Footer'ı her sayfada otamatik olarak göstermek için master page kullanıyoruz.

Site.Master'da aşağıdaki gibi layout'u ayarlıyoruz:

```
<!-- Navigation Bar -->
<div class="w3-bar w3-white w3-large w3-top" style="position: fixed; z-index: 1;">
    <a href="/Pages/Default.aspx" class="w3-bar-item w3-button w3-red w3-mobile"><i
            class="fa fa-car w3-margin-right"></i>Main Page</a>
    <a href="/Pages/Cars.aspx" class="w3-bar-item w3-button w3-mobile">Cars</a>
    <a href="/Pages/Admin.aspx" class="w3-bar-item w3-button w3-right w3-light-grey w3-mobile">Admin
        Login</a>
</div>

<!-- Main Content -->
<div class="container body-content">
    <asp:ContentPlaceHolder ID="MainContent" runat="server">
    </asp:ContentPlaceHolder>
</div>

<!-- Footer -->
<footer class="w3-padding-32 w3-black w3-center w3-margin-top" style="z-index: -2; position: relative;">
    <h5>Find Us On</h5>
    <div class="w3-xlarge w3-padding-16">
        <i class="fa fa-facebook-official w3-hover-opacity"></i>
        <i class="fa fa-instagram w3-hover-opacity"></i>
        <i class="fa fa-snapchat w3-hover-opacity"></i>
        <i class="fa fa-pinterest-p w3-hover-opacity"></i>
        <i class="fa fa-twitter w3-hover-opacity"></i>
        <i class="fa fa-linkedin w3-hover-opacity"></i>
    </div>
    <p>Powered by <a href="https://www.w3schools.com/w3css/default.asp" target="_blank"
            class="w3-hover-text-green">w3.css</a></p>
</footer>
```

Diğer sayfaların .aspx'lerine de "" MasterPageFile="~/Pages/Site.Master" "" kodunu ekliyerek bu masterPage'e bağlı olduklarını söylüyoruz.

Database_helper Class:

database_helper class'ını MS SQL'e bağlanma işlemlerini kullanması dağa rahat fonksiyonlara wraplamak için kullanıyorum. Lib.cs dosyasının içinde bu class'a ulaşabilirsiniz.

public class database_helper : Page olarak tanımlı. Page classi VS tarafından otomatik oluşturulmuş bir class ve .aspx.cs file'larının içinde, sayfaları düzenlerken kullandığımız class'ın parent'ti. database_helper'i da yine sayfaları düzenleyen bu class'ların içinde kullanmak istediğimiz için Page yerine database_helper kullanıyoruz. Örnek olarak:

Lib.cs →

public class database_helper : Page {

… }

Cars.aspx.cs →

public partial class Cars : database-helper{

… }

Böylece database_helper'i Cars'ın içinde kullanabiliyoruz. Ve Page, Cars'ın parenti olarak kalıyor.

```
public database_helper(){
    // Init
    db_connection = new SqlConnection(WebConfigurationManager.ConnectionStrings["db"].ConnectionString);
    sql_command = new SqlCommand("", db_connection);
    sql_adapter = new SqlDataAdapter(this.sql_command);

    // Open
    db_connection.Open();
}
```

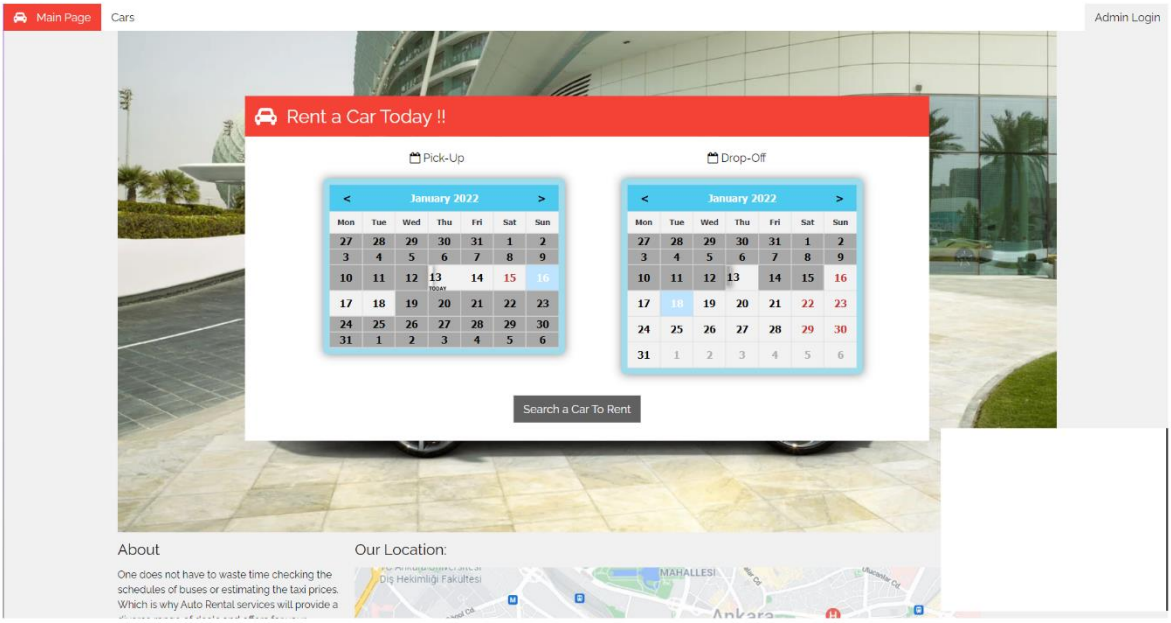
bind_new_…_command fonksiyonları (… → select, delete, update SQL queryleri) sql_command'ı yeni command yapmak ve sql_adapter'a bu command'ı bağlamak için kullanılıyor.

execute_new_..._command fonksiyonları yine yeni bir bind oluşturuyor ve bu bind'ı executeliyor.

…_bind fonksiyonları (… → datalist, repeater) Bunlara DataSet'i bağlar.

fill_new_dataset fonksiyonu yeni bir DataSet oluşturup bunu sql_adapter'daki seçilmiş table ile doldurur.

Ana Sayfa:




```
bool is_dates_selected(DateTime pick_date, DateTime drop_date){
    // Is both dates are selected
    if (!pick_date_time.Equals(DateTime.MinValue) && !drop_date_time.Equals(DateTime.MinValue))
        return true;
    else
        return false;
}
```

```
protected void Page_LoadComplete(object sender, EventArgs e){
    // Disable search button if dates are not selected on post back
    if (!is_dates_selected(pick_date_time, drop_date_time)){
        Button search_button = (Button)Master.FindControl("MainContent").FindControl("search_button");
        search_button.Enabled = false;
        search_button.Text = " Plase Select Dates Before Searching !!! ";
    }
}
```

Eğer pick-up date ve drop-off date seçili değilse yukarıdaki fonksiyon ile “Search A Car To Rent” buttonunu devre dışı bırakıyoruz.

```
protected void selection_changed(object sender, EventArgs e){
    // Assign selected dates
    pick_date_time = calendar_pick.SelectedDate;
    drop_date_time = calendar_drop.SelectedDate;

    // If dates are selected, enable search button
    if (is_dates_selected(pick_date_time, drop_date_time)){
        Button search_button = (Button)Master.FindControl("MainContent").FindControl("search_button");
        search_button.Enabled = true;
        search_button.Text = " Search a Car To Rent ";
    }
}
```

Date seçtiğimizde ise yukarıdaki fonksiyon ile geri aktıve ediyoruz.

Date seçerken pick-off tarihi drop-off' dan sonra olamayacağı için aşağıdaki fonksiyon ile yanlış günleri seçilemez hale getiriyoruz.

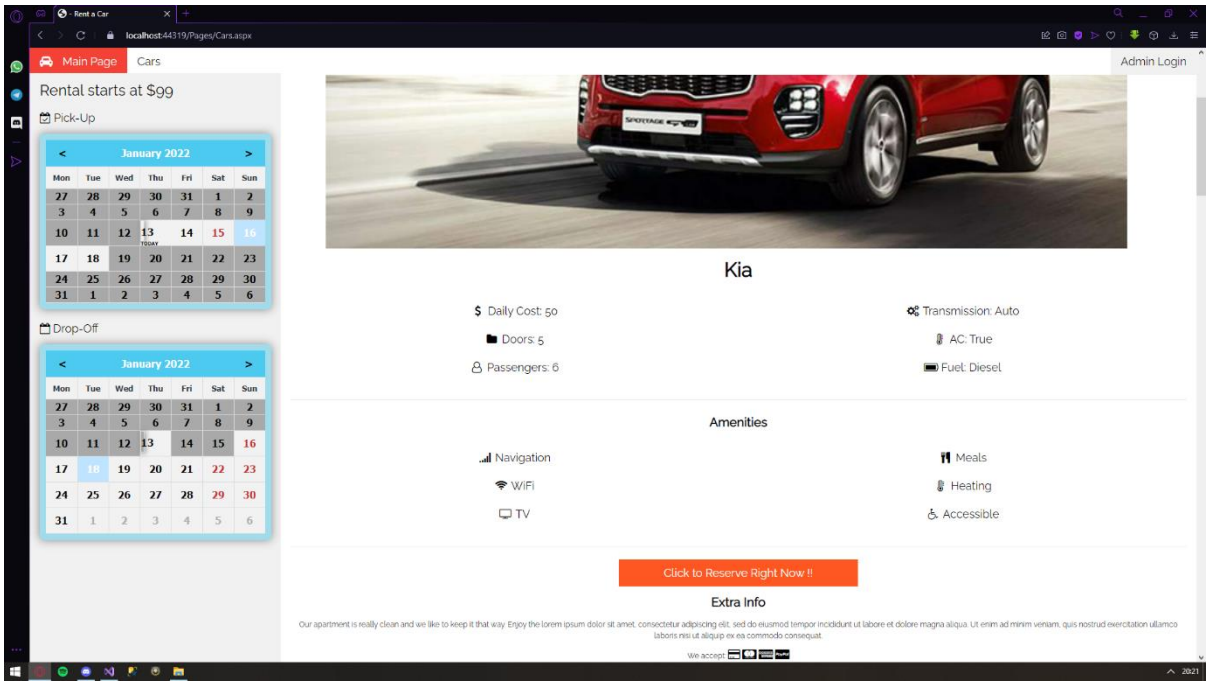
```
void make_unselectable(DayRenderEventArgs e){
    // Makes dates on calendar unselectable
    e.Day.IsSelectable = false;
    e.Cell.BackColor = System.Drawing.Color.DarkGray;
}
```

```
protected void pick_render(object sender, DayRenderEventArgs e)
{
    // If before today or after drop date, make dates unselectable
    if (e.Day.Date < (DateTime.Today)
        || (!drop_date_time.Equals(DateTime.MinValue) && e.Day.Date > drop_date_time)){

        make_unselectable(e);
    }
}

protected void drop_render(object sender, DayRenderEventArgs e){
    // If before today or before pick date, make dates unselectable
    if (e.Day.Date < (DateTime.Today)
        || (!pick_date_time.Equals(DateTime.MinValue) && e.Day.Date < pick_date_time)){

        make_unselectable(e);
    }
}
```

Search buttonuna basınca'da:

```
protected void search_clicked(object sender, EventArgs e){
    // If dates are selected, assign days to session and redirect
    if (is_dates_selected(pick_date_time, drop_date_time)){
        Session["days_to_pick"] = (pick_date_time - DateTime.Today).Days;
        Session["days_to_drop"] = (drop_date_time - DateTime.Today).Days;

        Response.Redirect("Cars.aspx");
    }
```

Cars Sayfasi:



Soldaki side bar'a ana sayfada kullandığımız calendar'ları ekliyoruz.

Main content olarak arabaları göstermek için repeater kullanıyoruz. Böylece istediğimiz arabayı dinamik olarak siteye kolayca ekleyebiliyoruz. Ayrıca eğer o tür arabadan seçili tarihlerde hiç bulunmuyorsa kolayca onu kullanıcıya göstermiyoruz.

Eğer ana sayfada date seçmeden yukarıdaki bar'dan Cars buttonuna basarak bu sayfaya gelmişsek, "Click to Reserve" buttonlarını aşağıdaki cod ile devre dışı bırakıyoruz ve tüm arabaları repeater'a ekliyerek gösteriyoruz.

```
protected void Page_Load(object sender, EventArgs e) {
    if (!IsPostBack) {
        // If dates selected, assign
        if (Convert.ToInt32(Session["days_to_pick"]) != -1 && Convert.ToInt32(Session["days_to_drop"]) != -1) {
            pick_date_time = DateTime.Today.AddDays(Convert.ToInt32(Session["days_to_pick"]));
            drop_date_time = DateTime.Today.AddDays(Convert.ToInt32(Session["days_to_drop"]));
        } else {
            pick_date_time = DateTime.MinValue;
            drop_date_time = DateTime.MinValue;
        }

        // Show pick dates
        calendar_pick.SelectedDate = pick_date_time;
        calendar_drop.SelectedDate = drop_date_time;
    }
}
```

Burada DateTime.MinValue repeater'a arabaları bağlarken tüm arabaları seçmemizi sağlıyor. Session değerleri -1 iken ise aşağında bind yaptıktan sonra buttonları devre dışı bırakıyoruz.

Repeater'a server'dan yeni table bağlarken, kullandığımız table'ı başka bir DataSet'de saklıyoruz. Dağa sonra buttonları kullanırken ve gösterilecek uygun img'yi seçerken buradaki dataları kullanıyoruz.

Select query'sinde, istediğimiz tarih aralığında boş olan arabaları buluyoruz ve bu aralabarın car_type'larını istiyoruz.

```
void bind_repeater(){
    // Bind to repeater
    database_helper db_helper = new database_helper();
    // New dates both must be before booked pick date or after booked drop date
    db_helper.bind_new_select_command($"SELECT * FROM car_types WHERE EXISTS ( SELECT * FROM cars WHERE
    db_helper.fill_new_dataset("car_types");

    // Store data table
    if (dataset_store.Tables["car_types"] != null)
        dataset_store.Tables.Remove("car_types");
    dataset_store.Tables.Add(db_helper.get_dataset().Tables["car_types"].Copy());

    db_helper.repeater_bind("car_repeater", "car_types", Master.FindControl("MainContent"));
    db_helper.dispose_all_and_close();
}
```

Page Load tamamlaninca baglama islemini yapiyoruz.

```
protected void Page_LoadComplete(object sender, EventArgs e) {
    bind_repeater();

    // Disable book button if dates are not selected
    if (Convert.ToInt32(Session["days_to_pick"]) == -1 || Convert.ToInt32(Session["days_to_drop"]) == -1) {
        foreach(RepeaterItem item in car_repeater.Items) {
            Button reserve_button = (Button) item.FindControl("reserve_button");
            reserve_button.Enabled = false;
            reserve_button.Text = " Please Search Before Booking !!! ";
        }
    }

    // Set img urls
    int i = 0;
    foreach(RepeaterItem item in car_repeater.Items) {
        Image car_img = (Image) item.FindControl("car_img");
        // Row[] --> row. Row[][] --> colum data
        car_img.ImageUrl = $"/jpgs/{dataset_store.Tables["car_types"].Rows[i++][7].ToString().Trim()}.jpg";
    }
}
```

Burada depoladığımız DataSet'de olan her araba için 7. Stundaki değer "car_type_name" ve arabaların img'leri ise /jpg klasöründe arabanın_ismi.jpg olarak tutuluyor. Yeni araba ekledikten sonra bu klasöre doğru isimlendirme yaparak arabanın resmini eklememiz arabanın resmini göstermek için yeterli.

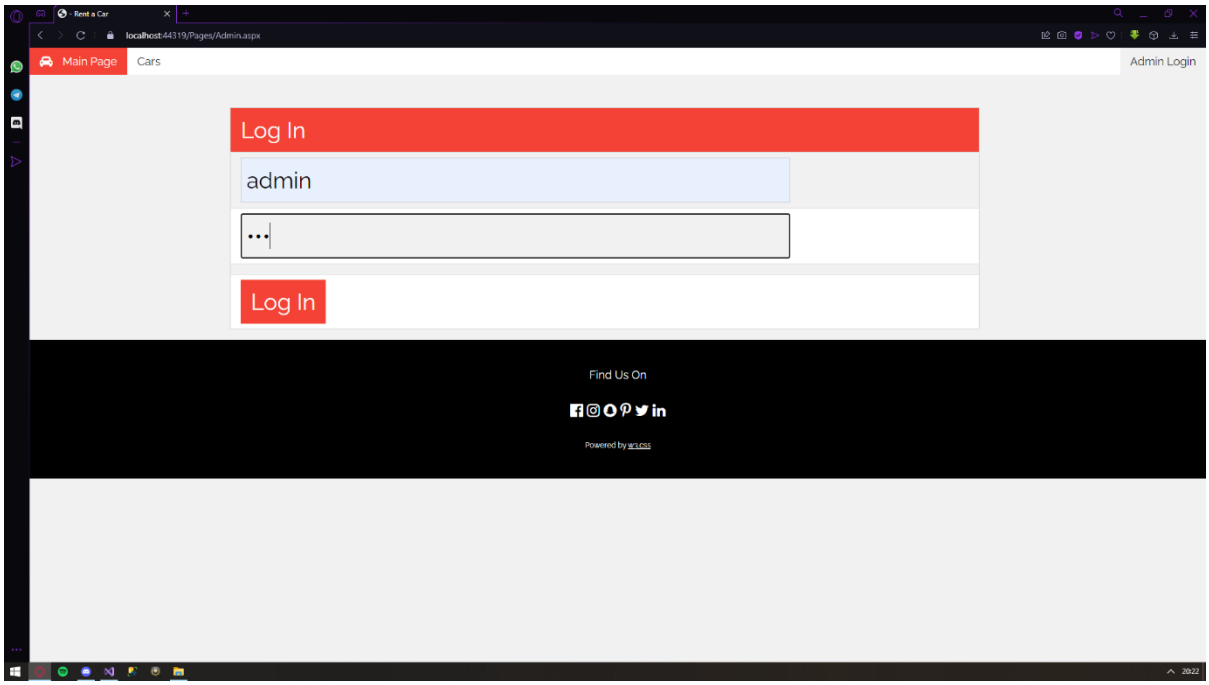
Reserve buttonuna basınca aşağıdaki cod çalışıyor.

Burada storelediğimiz DataSet'in tıkladığımız row'undaki ilk satır, car_type_id oluyor. Sonra bu car_type_id'ye sahip uygun car'ı car table'dan seçiyoruz ve bookings table'ına bu car_id ile yeni bir booking giriyoruz.

```
protected void reserve_clicked(object sender, CommandEventArgs e) {
    // Item index starts from 1 so use index - 1
    int id = (Int32)dataset_store.Tables["car_types"].Rows[Int32.Parse(e.CommandArgument.ToString()) - 1][0];
    database_helper db_helper = new database_helper();
    db_helper.bind_new_select_command($"SELECT id, type_id FROM cars WHERE type_id = {id} AND NOT EXISTS (SELECT * FROM bookin
    db_helper.fill_new_dataset("cars");

    // Store data table
    if (dataset_store.Tables["cars"] != null)
        dataset_store.Tables.Remove("cars");
    dataset_store.Tables.Add(db_helper.get_dataset().Tables["cars"].Copy());

    db_helper.execute_new_insert_command($"INSERT INTO bookings (car_id, days_to_pick, days_to_drop) VALUES ({dataset_store.Ta
    db_helper.dispose_all_and_close();

    // Rebind
    bind_repeater();

    ScriptManager.RegisterClientScriptBlock(this, this.GetType(), "alertMessage", "alert('Booking successful !!!')", true);
}
```

Booking işleminden sonra, repeater'i yeni bookinglere göre güncelleyip tekrar bindliyoruz. Ve kullanıcıya booking işleminin başarıyla yapıldığını söylüyoruz.

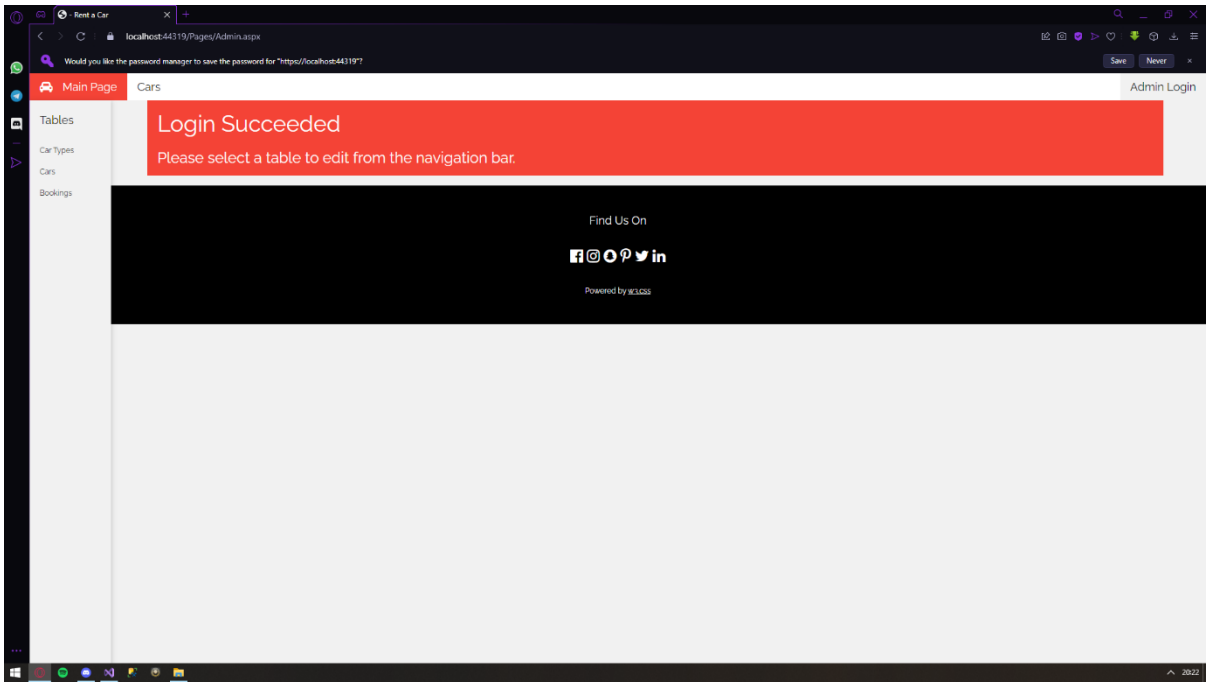
Admin Page:



Login ekranı için ASP:Login kullandık. Login yaptıktan sonra açılması için Logged in kısımlarını ASP:Panel içine ekledik.

```
protected void Page_Load(object sender, EventArgs e){
    // If logged in
    if (is_logged_in){
        // Disable login panel
        admin_login.Visible = false;
    }
    else{
        // Disable logged in panel
        logged_in_panel.Visible = false;
    }
}

protected void on_authenticate(object sender, AuthenticateEventArgs e){
    // If username and password are true, call on_logged_in
    if (admin_login.UserName == "admin" && admin_login.Password == "123")
        e.Authenticated = true;
    else
        e.Authenticated = false;
}

protected void on_logged_in(object sender, EventArgs e){
    is_logged_in = true;
}
```

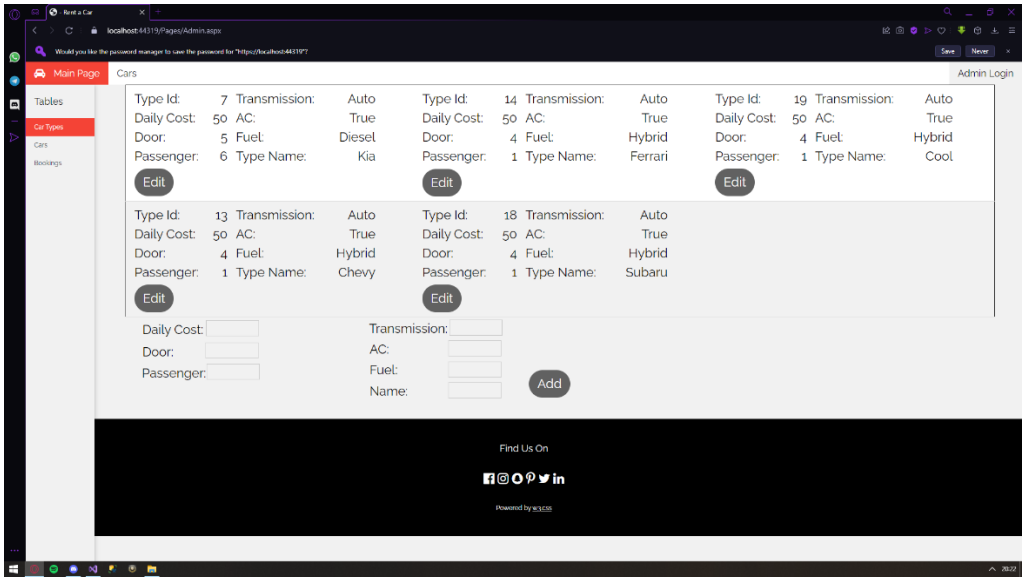
Logged in panelinde ilk olarak Login success ve soldaki tables paneli görünebilir. Panalden seçim yaptıktan sonra Login success devre dışı bırakılır ve seçilen kısım aktıve edilir.



active_tab, tab değiştirilirken kapatılmak üzere tutulur. Aynı zamanda clicked button red'den hover-red'e geçirlmek için tutulur.

```
void change_tab(object sender, string current_tab){
    Panel last_panel = (Panel) Master.FindControl("MainContent").FindControl(active_tab);
    last_panel.Visible = false;

    Panel active_panel = (Panel) Master.FindControl("MainContent").FindControl(current_tab);
    active_panel.Visible = true;

    Button clicked = (Button) sender;
    clicked.CssClass = clicked.CssClass.Replace("w3-hover-red", "w3-red");

    if (last_clicked != null){
        Button last_button = (Button) Master.FindControl("MainContent").FindControl(last_clicked);
        last_button.CssClass = last_button.CssClass.Replace("w3-red", "w3-hover-red");
    }
```

```
protected void car_types_tab_clicked(object sender, EventArgs e){
    // Configure db operations
    fill_car_types_datalist();
    change_tab(sender, "car_types_panel");
}

protected void cars_tab_clicked(object sender, EventArgs e){
    // Configure db operations
    fill_car_datalist();
    change_tab(sender, "cars_panel");
}

protected void bookings_tab_clicked(object sender, EventArgs e){
    fill_bookings_datalist();
    change_tab(sender, "bookings_panel");
}
```



Server'daki dataların kolayca gösterilebilmesi ve editlenebilmesi için DataList kullandık.

```
void fill_car_datalist(){
    // Configure db operations
    database_helper db_helper = new database_helper();
    db_helper.bind_new_select_command("SELECT cars.id, cars.type_id, car_types.name FROM cars JOIN car_types ON c
    db_helper.fill_new_dataset("cars");
    db_helper.data_list_bind("car_list", "cars", Master.FindControl("MainContent"));
    db_helper.dispose_all_and_close();
}

void fill_car_types_datalist(){
    // Configure db operations
    database_helper db_helper = new database_helper();
    db_helper.bind_new_select_command("SELECT * FROM car_types");
    db_helper.fill_new_dataset("car_types");
    db_helper.data_list_bind("car_types_list", "car_types", Master.FindControl("MainContent"));
    db_helper.dispose_all_and_close();
}

void fill_bookings_datalist(){
    // Configure db operations
    database_helper db_helper = new database_helper();
    db_helper.bind_new_select_command("SELECT * FROM bookings");
    db_helper.fill_new_dataset("bookings");
    db_helper.data_list_bind("bookings_list", "bookings", Master.FindControl("MainContent"));
    db_helper.dispose_all_and_close();
}
protected void car_edit(Object sender, DataListCommandEventArgs e) {
    // Make edit view visible
    car_list.EditItemIndex = e.Item.ItemIndex;

    fill_car_datalist();
}

protected void car_cancel(Object sender, DataListCommandEventArgs e) {
    // Make edit view not visible
    car_list.EditItemIndex = -1;

    fill_car_datalist();
}

protected void car_delete(Object sender, DataListCommandEventArgs e) {
    // Make edit view not visible
    car_list.EditItemIndex = -1;

    // Delete
    string delete_id = ((Label)e.Item.FindControl("edit_car_id")).Text;

    database_helper db_helper = new database_helper();
    db_helper.execute_new_delete_command("DELETE FROM cars WHERE id=" + delete_id);
    db_helper.dispose_all_and_close();

    // Refill
    fill_car_datalist();
}
```

Burada dikkat edilmesi gereken işlemlerden sonra, datanın değişeceği için dataLıstın, tekrar doldurulması gerektiğidir.

```
bool is_correct_for_car_type(string daily_cost, string door, string passenger, string transmission, string ac, string fuel, string name){
    int _;
    string[] transmission_targets = {"Auto", "Manuel"};
    string[] fuel_targets = {"Diesel", "Hybrid", "Gas", "LPG"};
    string[] ac_targets = {"True", "False"};
    if(daily_cost == "" || door == "" || passenger == "" || transmission == "" || fuel == "" || name == ""){
        ScriptManager.RegisterClientScriptBlock(this, this.GetType(), "alertMessage", "alert('Cant insert empty data !!!')", true);
        return false;
    }
    else if (!Int32.TryParse(daily_cost, out _) || !Int32.TryParse(door, out _) || !Int32.TryParse(passenger, out _)){
        ScriptManager.RegisterClientScriptBlock(this, this.GetType(), "alertMessage", "alert('Cant insert string for int !!!')", true);
        return false;
    }
    else if (!transmission_targets.Contains(transmission)){
        ScriptManager.RegisterClientScriptBlock(this, this.GetType(), "alertMessage", "alert('Cant insert wrong transmission !!!')", true);
        return false;
    }
    else if(!fuel_targets.Contains(fuel)){
        ScriptManager.RegisterClientScriptBlock(this, this.GetType(), "alertMessage", "alert('Cant insert wrong fuel !!!')", true);
        return false;
    }
    else if (!ac_targets.Contains(ac)){
        ScriptManager.RegisterClientScriptBlock(this, this.GetType(), "alertMessage", "alert('Cant insert wrong ac !!!')", true);
        return false;
    }
    else
        return true;
}
```

```
protected void car_update(Object sender, DataListCommandEventArgs e) {
    // Make edit view not visible
    car_list.EditItemIndex = -1;

    // Get strings
    string update_id = ((Label)e.Item.FindControl("edit_car_id")).Text;
    string update_type_id = ((TextBox)e.Item.FindControl("edit_car_type_id")).Text;

    if (is_correct_for_car(update_type_id)){
        // If exist, update
        database_helper db_helper = new database_helper();
        db_helper.execute_new_update_command($"Update cars SET type_id={update_type_id}WHERE id={update_id}");
        db_helper.dispose_all_and_close();
    }

    // Refill
    fill_car_datalist();
}

protected void add_car(object sender, EventArgs e){
    // Get strings
    string add_type_id = ((TextBox)Master.FindControl("MainContent").FindControl("add_car_type_id")).Text;

    if (is_correct_for_car(add_type_id)){
        // If exist, add
        database_helper db_helper = new database_helper();
        db_helper.execute_new_insert_command($"INSERT INTO cars (type_id) VALUES ({add_type_id})");
        db_helper.dispose_all_and_close();
    }

    // Refill
    fill_car_datalist();
}
```

Yeni bir veri ekleyecekken ya da veri güncellenecekken verinin doğru olduğundan emin olunmalıdır. Bunu iş_correct_for_... fonksiyonu (... → car, car_type, booking) control eder. Verilerin boş olmaması, doğru tipte olması, database relation kurallarına uyması verilen seçeneklerden biri olması gerekir (Transmission = CokGuzelAraba olamaz. Auto ya da Manuel dışında başka bir şey olamaz.).

Kaynakca

Template: W3.CSS
https://www.w3schools.com/w3css/w3css_templates.asp

Mans: Microsoft

https://docs.microsoft.com/en-us/dotnet/api/?view=netframework-4.8

Ekampus Slaytlar