**ClearDrive**

**dokumentáció**

Készítette: Krisztin Benedek László,

Kovácsházi Ádám, Surányi István

2025.

Szegedi SZC Vasvári Pál Gazdasági és Informatikai Technikum

**ClearDrive dokumentáció**

# **Az ötlet**

*a. A probléma*

Nagy tapasztalattal rendelkező, szegedi gépjárművezető oktató hívta fel a figyelmünket arra, hogy a mindennapi (szegedi) közlekedés során fellépő rendellenességek (veszélyek: kidőlt fa az úton, fa letört ága veszélyezteti a parkoló autókat; hibák: jelzőlámpa meghibásodott, úthibák, nem látszik az útfelfestés, nem látszik a tábla/lámpa; illetve illegális szemétlerakatok) bejelentése kizárólag telefonon keresztül lehetséges. Ezen felül valamennyi ezen problémák közül más-más fenntartóhoz tartozik, amelyek között magánszemélyek számára elérhető központi bejelentési lehetőség nincs, így nehezítve a hiba bejelentését, elhárítását. Ezt a problémát lenne hivatott az általunk tervezett szoftver áthidalni.

# **A kivitelezés nagyvonalakban**

1. *Mit csináltunk?*

A végső rendszer három részből áll: egy asztali programból, egy webes felületből és egy mobilalkalmazásból. Ezek közül az asztali program és a webes felület teszi lehetővé az adminisztrátorhoz tartozó feladatkörök elvégzését. Ha felhasználóként jelentkezünk be a web-re, akkor a hibajegyek prioritásának növelése és a kommenteléseké a főszerep, az adminisztrátori felületeket nem érik el. A mobil teljesmértékben a felhasználóknak készült amelynek a fű funkciója a hiba bejelentés.

1. *Tech Stack*

A projektben felhasznált technológiák igen sokszínűek. Igyekeztünk a legmodernebb eszközöket használni, hogy megbízható és időtálló legyen a munkánk.

--szöveg—

1. *Mostani funkciók*

Jelen pillanatban az alábbi műveletek lehetségesek az alkalmazásokban:

* Regisztráció / Bejelentkezés (2FA is)
* Jelszóváltoztatás
* Profilok készítése, szerkesztése és törlése
* Hibák bejelentése, szerkesztése és törlése
* Adminisztrátorok szerkesztése és törlése
* Szervezetek létrehozása, szerkesztése és törlése
* Komment írása, hozzáfűzése és törlése
* Barátok bejelölése

1. *Mi lesz még?*

A jövőben tervezünk a jelenlegiek mellé további funkciókat is bevezetni, növelve a felhasználói élményt. Ezek közé tartozik, hogy a felhasználók tudjanak egymással üzeneteket váltani, mint bármilyen más közösségimédiás alkalmazásban, képfelismerés, a rendszer a kép készítésekor felismerje, hogy milyen hiba létható a képen és bekategorizálja, AI asszisztens.

# **A Backend**

1. *Tech Stack*

--szöveg—

1. *Projektstruktúra*

--szöveg—

1. *Az adatbázis*

Az adatbázis tervezéskor a fő szempont az volt, hogy minden olyan adatot, amit a programnak tárolnia kell, el is tudja tárolni.

--szöveg—

Az adatbázis egyed-kapcsolat diagramja:

--kép—

Az adatbázis bachmann-ábrája:

--kép—

1. *DbConnector és DataController*

--szöveg—

1. *A DataControllerFactory*

--szöveg—

1. *Hitelesítés, JWT, biztonság*

A backend nagyon sok érzékeny adat feltöltését és elolvasását teszi lehetővé, emiatt szükség volt valamilyen hitelesítésre. Ehhez mi a JWT-t (JSON Web Token) választottuk, mert egy modern megoldás, egyszerű az implementálása és az iskolában is ajánlották.

--szöveg—

1. *2FA*

--szöveg--

1. *Tesztelés*

--szöveg—

1. *CI/CD, Github Actions*

--szöveg—

1. *Nehézségek*

--szöveg—

**A frontend**

# **A mobil**

1. *Design tervek*

A design-nál próbáltunk arra törekedni hogy a felhasználók szemszögéből letisztult és egyszerű legyen minél felhasználóbarátabb.

1. *Tech Stack*

--szöveg—

1. *Projectstruktúra*

--szöveg—

1. *View-ek, komponensek*

--szöveg—

1. *HomePage*

--szöveg—

1. *Styles*

--szöveg—

1. *Services*

--szöveg—

1. *Nehézségek*

--szöveg—

# **A web**

1. *Design tervek*

A design-nál próbáltunk arra törekedni hogy a felhasználók szemszögéből letisztult és egyszerű legyen minél felhasználóbarátabb.

1. *Tech Stack*

--szöveg—

1. *Projectstruktúra*

--szöveg—

1. *View-ek, komponensek*

--szöveg—

1. *HomePage*

--szöveg—

1. *Styles*

--szöveg—

1. *Services*

--szöveg—

1. *Nehézségek*

--szöveg—

# **A desktop**

1. *Design tervek*

A design-nál próbáltunk arra törekedni hogy a felhasználók szemszögéből letisztult és egyszerű legyen minél felhasználóbarátabb.

1. *Tech Stack*

A frontend alapját a WPF (Windows Presentation Foundation) adja, a többi használt technológia ehhez igazodik. Jó példa erre az MVVM (Model-View-ViewModel) architektúra, a **RelayCommand** és a **INotifyPropertyChanged**. Formázásra a XAML és a ResourceDictionary, valamint a ControlTemplates és Styles használata jellemző. HTTP kérésekhez az **HttpClient**-et alkalmazzuk.

--szöveg—

1. *Projectstruktúra*

A gyökérkönyvtárban található a **README.md** fájl, amely a projekt dokumentációját tartalmazza, valamint a **csomagokat kezelő fájlok** (pl. .csproj), a fő XAML fájl (MainWindow.xaml) és a **styles** konfigurációs fájlok (ResourceDictionary és App.xaml). A kódfájlok a **Views**, **ViewModels** és **Services** mappákban találhatók, amelyek a projekt szerkezetét alkotják. A **Views** mappában az alkalmazás felhasználói felületét tartalmazó XAML fájlok találhatóak, míg a **ViewModels** mappában az MVVM logikát implementáló osztályok találhatók. A **Services** mappa tartalmazza a különböző szolgáltatásokat, mint pl. adatkezelés vagy API hívások.

A projekt fő fájlja az **App.xaml**, amely az alkalmazás indításakor betöltődik, és meghatározza az alkalmazás globális erőforrásait, stílusait. A fő ablak a **MainWindow.xaml**, ahol a felhasználói felület (UI) komponensek kerülnek elrendezésre.

Az oldalhoz szükséges egyéb fájlokat a **Resources** könyvtárba helyeztük el. Ide tartoznak például a betűtípusok, az alkalmazás alap stílusfájljai, és egyéb vizuális elemek, mint az ikonok és képek.

--szöveg—

1. *View-ek, komponensek*

Az desktop fő részei a view-ok és a komponensek. A view-ok tartalmazzák a program legfontosabb részeit például a DragAndDropTablePage.xaml, AdminChangePage.xaml és a LoginView.xaml.

A komponeneseket arra használtuk az oldalon hogy a view-kat kiegészítsük.

--szöveg—

1. *HomePage*

Az alkalmazásba való belépés után ez az oldal üdvözli a felhasználót. Emelet megjelenik az oldalsó menü, ahol kilehet választani a megtekinteni kívánt oldalt. Mint például Feladatok Tábla, Térkép, Adminisztráció, Rendszergazda adminisztráció, Statisztika és Profil oldal. Ezek alatt a kijelentkezés gomb található.

1. *Styles*

A WPF-ben a **Style** egy **ResourceDictionary**-ben kerül definiálásra, amelyben különböző vezérlőelemekhez tartozó vizuális jellemzők vannak megadva. A **Style** segít abban, hogy az alkalmazásban ugyanazt a formázást mindenhol egységesen alkalmazzuk anélkül, hogy minden egyes vezérlőhöz újra és újra be kellene írni a formázást.

--szöveg—

1. *Services*

A **Services** (Szolgáltatások) a WPF alkalmazásokban a háttérben futó logikát és az alkalmazás funkcionális részeit tartalmazzák, amelyeket a ViewModel-ek és más komponensek használnak a felhasználói felület működtetéséhez. A **Services** célja, hogy az alkalmazás üzleti logikáját, adatkezelését, hálózati kommunikációt vagy más külső szolgáltatásokat különálló, újrafelhasználható egységekbe szervezzék, így az alkalmazás karbantartása és tesztelése egyszerűbbé válik.

A WPF alkalmazásokban a **Services** a Model és a ViewModel rétegek közötti interfészként működnek, és általában nem tartalmaznak UI-val kapcsolatos logikát. A fő céljuk az, hogy biztosítsák a szükséges adatokat, erőforrásokat és szolgáltatásokat, amelyeket a ViewModel-ek és a felhasználói felület használnak.

**1. EnumToBoolConverter.cs**

Ez egy **értékátalakító (Value Converter)**, amelyet az adatbinding során használhatunk. A fő célja, hogy egy **Enum érték és egy string összehasonlítása alapján visszaadjon egy boolean értéket**.

* **Használati példa**: Ha van egy RadioButton, amely egy Enum értékhez kötött, ez az átalakító segít meghatározni, hogy a gomb ki van-e választva.
* **Convert**: Összehasonlítja az aktuális értéket a paraméterként kapott Enum értékkel, és visszaad egy true/false értéket.
* **ConvertBack**: Ha a visszaalakítás történik (pl. egy RadioButton kiválasztása), az adott Enum értéket állítja be.

**2. ITokenService.cs & TokenService.cs**

Ez a service egy **felhasználói token és felhasználónév tárolásáért és visszaolvasásáért felelős**.

* Az **ITokenService** az interfész, amely definiálja a funkciókat.
* A **TokenService** az interfész implementációja, amely a tokeneket és a felhasználónevet az **IsolatedStorage**-ban tárolja.

Funkciók:

* **SaveToken()** és **GetToken():** JWT token mentése és visszaolvasása.
* **SaveUserName()** és **GetUserName()**: Felhasználónév mentése és visszaolvasása.
* **RemoveToken()** és **RemoveUserName():** Az adott adatok törlése.

**3. TextBoxHelper.cs**

Ez egy **WPF Attached Property Helper**, amely lehetővé teszi, hogy egy **helyőrző szöveget (Placeholder)** jelenítsünk meg TextBox és PasswordBox vezérlőkben.

Főbb funkciói:

* **GetPlaceholder() és SetPlaceholder()**: Beállítja vagy lekéri a helyőrző szöveget.
* **PlaceholderProperty**: Ez egy DependencyProperty, amely figyeli a változásokat.
* **Adorner használata**: Egyedi PlaceholderAdorner osztály segítségével jeleníti meg a helyőrzőt, és automatikusan frissíti a megjelenését.

1. *Nehézségek*

--szöveg—

# **A csapatmunka**

1. *A csapat*

A projekten hárman dolgoztunk: Kovácsházi Ádám, Surányi István és Krisztin Benedek László.

Ugyan mindhárman egyenlő mértékben dolgoztunk, közülünk Ádám nőtt ki, mint projektvezető. Övé volt az ötlet és ő csinálta a teljes mobilt és a backend nagy részét. István a web-et programozta és a backend alapját. Benedek pedig a desktop-ot programozta. Természetesen mindenki belekóstolt a fejlesztés összes részébe, hogy full-stack tapasztalatunk is legyen.

1. *Munkaelosztás*

Ádám fő feladata a mobil és backend fejlesztése volt, ezen kívül hozzá fűződik a mobil és a web platformra portolása és a csapat általános koordinálása.

István első feladata a backend alapjának elkészítése volt. Ezután tért át a web fejlesztésére.

Benedek feladata a desktop alkalmazása fejlesztése volt és ez idő alatt elkészítette a web alapját. Emelet a backenden ő készítette el az illegalParking API végpontot és hozzátartozó részeket.

1. *Kommunikáció*

A csapat a kapcsolatot leginkább személyesen tartotta az iskolában és vagy a TIK-ben, hiszen ez a kommunikáció legegyszerűbb és leghatékonyabb módja. Beszélgetés közben látjuk, hogy mit csinál a másik és mi is tudunk neki helyben segíteni.

Amikor távol voltunk egymástól, kénytelenek voltunk más módszerekhez fordulni. Ebben segített nekünk a GitHub, de még jobban a Discord. Itt tudtunk beszélgetni és akár azt is megnézni, hogy mi történik a csapattársunk számítógépén, ha segítségre volt szüksége.

# **API dokumentáció**

1. *Backend*

--szöveg—

1. *Frontend*

--szöveg—

# **Az adatok kezelése (mobil)**

**Komponens adatai:**

--felsorolás, magyarázattal—

**Függvények:**

--felsorolás, magyarázattal—

**API meghívások:**

--felsorolás, magyarázattal—

**Eseményvezérlés:**

--felsorolás, magyarázattal—

# **Az adatok megjelenítése (mobil)**

**Komponens adatai:**

--felsorolás, magyarázattal—

**Függvények:**

--felsorolás, magyarázattal—

**API meghívások:**

--felsorolás, magyarázattal—

**Eseményvezérlés:**

--felsorolás, magyarázattal—

**Stíluskészítése:**

--felsorolás, magyarázattal—

# **Az oldalsó menü (mobil)**

**Komponens adatai:**

--felsorolás, magyarázattal—

**Függvények:**

--felsorolás, magyarázattal—

**Stíluskészítése:**

--felsorolás, magyarázattal—

**Használt komponensek:**

--felsorolás, magyarázattal—

**Kommunikáció:**

--felsorolás, magyarázattal—

# **Az adatok kezelése (web)**

**Komponens adatai:**

--felsorolás, magyarázattal—

**Függvények:**

--felsorolás, magyarázattal—

**API meghívások:**

--felsorolás, magyarázattal—

**Eseményvezérlés:**

--felsorolás, magyarázattal—

# **Az adatok megjelenítése (web)**

**Komponens adatai:**

--felsorolás, magyarázattal—

**Függvények:**

--felsorolás, magyarázattal—

**API meghívások:**

--felsorolás, magyarázattal—

**Eseményvezérlés:**

--felsorolás, magyarázattal—

**Stíluskészítése:**

--felsorolás, magyarázattal—

# **Az oldalsó menü (web)**

**Komponens adatai:**

--felsorolás, magyarázattal—

**Függvények:**

--felsorolás, magyarázattal—

**Stíluskészítése:**

--felsorolás, magyarázattal—

**Használt komponensek:**

--felsorolás, magyarázattal—

**Kommunikáció:**

--felsorolás, magyarázattal—

# **Az adatok kezelése (desktop)**

**Komponens adatai:**

* ***\_viewModel****: A nézetmodell példánya, amely összekapcsolja a felhasználói felületet az adatokkal és logikával.*
* ***WindowState****: Az ablak aktuális állapota (normál, maximalizált vagy minimalizált).*
* ***TgBtnIsChecked****: A nézetmodellből érkező állapotváltozó, amely a navigációs menü megjelenítéséért felelős.*
* ***tt\_toDoTable, tt\_map, tt\_profile, tt\_administration, tt\_statistic, tt\_godadmin****: A különböző menüpontokhoz tartozó tooltipek láthatóságát szabályozó elemek.*
* ***\_pageInstances****: Az oldalnézetek példányainak tárolására használt szótár, amely biztosítja, hogy egy oldal csak egyszer jöjjön létre.*
* ***\_serviceProvider****: Dependency Injection konténer, amely az alkalmazás különböző szolgáltatásait biztosítja.*
* ***\_webSocketService****: WebSocket kapcsolat kezelő szolgáltatás.*
* ***\_tokenService****: Az autentikációhoz és a felhasználói tokenek kezeléséhez szükséges szolgáltatás.*
* ***\_adminService****: Az adminisztrációs szolgáltatás, amely az aktuális admin adatait kezeli.*
* ***CurrentAdmin****: A bejelentkezett adminisztrátor adatai.*
* ***CurrentView****: Az aktuálisan megjelenített nézet az alkalmazásban.*
* ***SuperAdminMenuVisibility****: A szuperadmin menü láthatóságát szabályozó változó.*
* ***InitializeAsync()****: Az alkalmazás inicializálását végző aszinkron metódus, amely beállítja az alapértelmezett nézetet és az admin jogosultságokat.*
* ***CloseClicked()****: A WebSocket kapcsolat lezárásáért és az alkalmazás bezárásáért felelős parancs.*
* ***LogoutClicked()****: A kijelentkezést végző parancs, amely törli a hitelesítési adatokat és visszairányít a bejelentkezési képernyőre.*
* ***OnPageSelected<TPage>()****: Az oldalváltásért felelős metódus, amely betölti a kiválasztott nézetet és frissíti az aktuális megjelenített oldalt.*

**LoginView/RegisterView***:*

* *\_****serviceProvider****: Az alkalmazás szolgáltatásainak elérhetőségét biztosító objektum.*
* **\_authService**: A hitelesítési szolgáltatás példánya.
* **\_tokenService**: A bejelentkezési tokeneket kezelő szolgáltatás példánya (csak a **LoginView.xaml.cs**-ben).
* **\_viewModel**: A regisztrációs nézethez tartozó nézetmodell (csak a **RegisterView.xaml.cs**-ben).
* **\_serviceService**: A szolgáltatások listáját kezelő háttérszolgáltatás (csak a **RegisterViewModel.cs**-ben).
* **\_serviceNames**: A szolgáltatások nevének listája (csak a **RegisterViewModel.cs**-ben).
* **\_services**: A szolgáltatásokat tartalmazó lista (csak a **RegisterViewModel.cs**-ben).

**Függvények:**

**MainView.xaml.cs:**

* ***btnMaximize\_Click()***

*Az ablak maximalizálásáért és visszaállításáért felelős.*

* ***btnMinimize\_Click()***

*Az ablak minimalizálását végzi el.*

* ***Grid\_MouseDown()***

*Az ablak mozgatását teszi lehetővé az egérrel húzva.*

* ***ListViewItem\_MouseEnter()***

*A navigációs menü tooltipeket kezeli a TgBtnIsChecked állapotától függően.*

* ***ToDoTable\_MouseLeftButtonDown()***

*A feladatkezelő oldal betöltését indítja el.*

* ***Map\_MouseLeftButtonDown()***

*A térkép oldal betöltéséért felelős.*

* ***Administration\_MouseLeftButtonDown()***

*Az adminisztrációs oldal megnyitását kezeli.*

* ***GodAdmin\_MouseLeftButtonDown()***

*A szuperadmin felületre navigál.*

* ***Statistic\_MouseLeftButtonDown()***

*A statisztikai oldal megjelenítését végzi.*

* ***Profile\_MouseLeftButtonDown()***

*A profiloldal betöltéséért felelős.*

**MainViewModel.cs (Logikai műveletek):**

* ***InitializeAsync()***

*Az alkalmazás inicializálását végző aszinkron metódus, amely:*

* + *Beállítja az alapértelmezett oldalt (HomePage).*
  + *Lekérdezi a bejelentkezett admin adatait.*
  + *Meghatározza a szuperadmin menü láthatóságát.*
* ***CloseClicked()***

*A WebSocket kapcsolat lezárásáért és az alkalmazás bezárásáért felelős.*

* ***LogoutClicked()***

*A kijelentkezést végző metódus:*

* + *Törli a felhasználói tokent és az azonosító információkat.*
  + *Megjeleníti a bejelentkezési felületet.*
  + *Elrejti a főablakot.*
* ***OnPageSelected<TPage>()***

*Az oldalváltásért felelős metódus, amely:*

* + *Ellenőrzi, hogy a kívánt oldal már létezik-e \_pageInstances belső szótárban.*
  + *Ha nem, akkor példányosítja a kívánt oldalt a \_serviceProvider segítségével.*
  + *Frissíti az aktuális nézetet (CurrentView).*
  + *Bezárja a navigációs menüt, ha az nyitva van.*

**LoginView.xaml.cs***:*

* ***Window\_KeyDown(e):*** *Az Enter billentyű lenyomására bejelentkezési műveletet indít.*
* ***txtUsername\_TextChanged(e):*** *Érvényesíti a bejelentkezési űrlapot, ha a felhasználónév változik.*
* ***txtPassword\_PasswordChanged(e):*** *Érvényesíti a bejelentkezési űrlapot, ha a jelszó változik.*
* ***SigninValidateForm():*** *Ellenőrzi a felhasználónév és jelszó meglétét, ennek megfelelően engedélyezi vagy tiltja a bejelentkezési gombot.*
* ***ButtonSignin\_Click(e):*** *Elindítja a bejelentkezést az \_authService használatával.*
* ***ButtonSignup\_Click(e):*** *Átvált a regisztrációs nézetre.*
* ***Border\_MouseDown(e):*** *Az ablakot mozgathatóvá teszi az egérrel.*
* ***btnClose\_Click(e):*** *Bezárja az alkalmazást.*
* ***btnMinimize\_Click(e):*** *Minimalizálja az ablakot.*
* ***forgottenPasswordBtn\_Click(e):*** *Átvált a jelszó-visszaállítási oldalra.*

**RegisterView.xaml.cs:**

* ***Window\_KeyDown(e):*** *Az Enter billentyű lenyomására regisztrációs műveletet indít.*
* ***txtSignupUsername\_TextChanged(e):*** *Érvényesíti a regisztrációs űrlapot, ha a felhasználónév változik.*
* ***txtSignupFirstname\_TextChanged(e):*** *Érvényesíti a regisztrációs űrlapot, ha a vezetéknév változik.*
* ***txtSignupLastname\_TextChanged(e):*** *Érvényesíti a regisztrációs űrlapot, ha a keresztnév változik.*
* ***txtSignupEmail\_TextChanged(e):*** *Érvényesíti a regisztrációs űrlapot, ha az e-mail cím változik.*
* ***comboSignupWork\_TextChanged(e):*** *Kezeli a munkakör kiválasztását és érvényesíti a regisztrációs űrlapot.*
* ***txtSignupPassword\_PasswordChanged(e):*** *Érvényesíti a regisztrációs űrlapot, ha a jelszó változik.*
* ***SignupValidateForm():*** *Ellenőrzi az űrlap helyességét, ennek megfelelően engedélyezi vagy tiltja a regisztrációs gombot.*
* ***ButtonSignup\_Click(e):*** *Elindítja a regisztrációt az \_authService segítségével.*
* ***ButtonSignin\_Click(e):*** *Átvált a bejelentkezési oldalra.*
* ***Border\_MouseDown(e):*** *Az ablakot mozgathatóvá teszi az egérrel.*
* ***btnClose\_Click(e):*** *Bezárja az alkalmazást.*
* ***btnMinimize\_Click(e):*** *Minimalizálja az ablakot.*
* ***Window\_Loaded(e):*** *Inicializálja a RegisterViewModel szolgáltatáslistáját.*

**RegisterViewModel.cs:**

* ***InitializeAsync():*** *Lekéri a szolgáltatásokat \_serviceService-en keresztül és feltölti a \_serviceNames listát.*
* ***IsValidEmail(email):*** *Reguláris kifejezéssel ellenőrzi az e-mail cím helyességét.*

**API meghívások:**

|  |  |  |
| --- | --- | --- |
| API hívás | Metódus | Leírás |
| \_adminService.GetByUserName(\_tokenService.GetUserName()) | InitializeAsync() | Lekéri a bejelentkezett admin adatait az adminisztrációs szolgáltatáson keresztül. |
| \_webSocketService.CloseWebSocketAsync() | CloseClicked() | Lezárja a WebSocket kapcsolatot. |
| \_tokenService.RemoveToken() | CloseClicked(), LogoutClicked() | Törli a hitelesítési adatokat. |
| \_tokenService.RemoveUserName() | LogoutClicked() | |  | | --- | |  |   Törli a felhasználónevet a helyi tárolóból. |
| handler.SetToken("") | LogoutClicked() | Kiüríti az autentikációs tokent a HTTP kliensben. |
| LoginView/RegisterView: | | |
| \_authService.LoginAdminAsync(username, password) | **POST** | Bejelentkezés API-hívás a szerverhez |
| \_tokenService.SaveUserName(username) | **LOCAL** | Felhasználónév mentése bejelentkezés után |
| \_authService.RegisterAdminAsync(firstname, lastname, username, password, email, serviceId) | **POST** | |  | | --- | |  |   Regisztrációs API-hívás |
| \_serviceService.SelectAllServices() | **GET** | Szolgáltatások lekérése a regisztrációs űrlaphoz |

**Eseményvezérlés:**

|  |  |  |
| --- | --- | --- |
| Esemény | Metódus | Leírás |
| Gombnyomás (Click event) | **btnMaximize\_Click(object sender, RoutedEventArgs e)** | Az ablak maximalizálásáért vagy visszaállításáért felelős. |
| **btnMinimize\_Click(object sender, RoutedEventArgs e)** | Az ablak minimalizálását végzi el. |
| **CloseClicked()** | Az alkalmazás bezárásáért felelős. |
| **LogoutClicked()** | Kijelentkezést és bejelentkezési képernyőre váltást kezel. |
| Egérmozgás (MouseEnter event) | **ListViewItem\_MouseEnter(object sender, MouseEventArgs e)** | A navigációs menü tooltipeket kezeli a **TgBtnIsChecked** állapotától függően. |
| Egérgomb lenyomás (MouseLeftButtonDown event) | **Grid\_MouseDown(object sender, MouseButtonEventArgs mouse)** | Az ablak mozgatását teszi lehetővé az egérrel húzva. |
| **ToDoTable\_MouseLeftButtonDown(object sender, MouseButtonEventArgs e)** | A feladatkezelő oldal betöltését indítja el. |
| **Map\_MouseLeftButtonDown(object sender, MouseButtonEventArgs e)** | A térkép oldal betöltéséért felelős. |
| **Administration\_MouseLeftButtonDown(object sender, MouseButtonEventArgs e)** | Az adminisztrációs oldal megnyitását kezeli. |
| **GodAdmin\_MouseLeftButtonDown(object sender, MouseButtonEventArgs e)** | A szuperadmin felületre navigál. |
| **Statistic\_MouseLeftButtonDown(object sender, MouseButtonEventArgs e)** | A statisztikai oldal megjelenítését végzi. |
| **Profile\_MouseLeftButtonDown(object sender, MouseButtonEventArgs e)** | A profiloldal betöltéséért felelős. |
| LoginView/RegisterView: | | |
| Ablakban billentyű lenyomása | **Window\_KeyDown** | Enter billentyű lenyomására a bejelentkezési vagy regisztrációs folyamat elindítása. |
| Felhasználónév mező változása | **txtUsername\_TextChanged** | Beviteli mező változására validáció futtatása. |
| Jelszó mező változása | **txtPassword\_PasswordChanged** | Jelszó beviteli mező változására validáció futtatása. |
| Bejelentkezési űrlap ellenőrzése | **SigninValidateForm** | Űrlapellenőrzés és bejelentkezés gomb aktiválása. |
| Bejelentkezés gomb kattintása | **ButtonSignin\_Click** | Bejelentkezési folyamat elindítása, animációk kezelése. |
| Regisztrációs oldalra navigálás | **ButtonSignup\_Click** | Regisztrációs oldalra váltás |
| Elfelejtett jelszó gomb kattintása | **forgottenPasswordBtn\_Click** | Jelszóváltoztató oldal megnyitása |
| Ablak egérrel húzása | **Border\_MouseDown** | Ablak húzásának engedélyezése egérkattintásra |
| Alkalmazás bezárása | **btnClose\_Click** | Alkalmazás bezárása |
| Ablak minimalizálása | **btnMinimize\_Click** | Ablak minimalizálása |
| Regisztrációs űrlap mezőinek változása | **txtSignup** | Regisztrációs űrlap validálása mezőváltozásra |
| Regisztrációs űrlap ellenőrzése | **SignupValidateForm** | Regisztrációs űrlap adatellenőrzése |
| Regisztráció indítása | **ButtonSignup\_Click** | |  | | --- | |  |  |  | | --- | | Regisztráció indítása, szerverhívás | |
| Regisztrációs űrlap betöltése | **Window\_Loaded** | Regisztrációs űrlap betöltéskor a szolgáltatások lekérése |

# **Az adatok megjelenítése (desktop)**

**Komponens adatai:**

**MainView.xaml.cs**

* ***\_viewModel****: A MainViewModel példánya, amely az adatok kezeléséért felelős.*
* ***btnMaximize\_Click****: Az ablak maximalizálásáért vagy normál méretűre állításáért felelős eseménykezelő.*
* ***btnMinimize\_Click****: Az ablak minimalizálásáért felelős eseménykezelő.*
* ***Grid\_MouseDown****: Az ablak mozgatásához szükséges eseménykezelő.*
* ***ListViewItem\_MouseEnter****: A navigációs elemek tooltipjeinek láthatóságát kezeli.*
* ***ToDoTable\_MouseLeftButtonDown****: A DragAndDropTablePage oldalra navigál.*
* ***Map\_MouseLeftButtonDown****: A MapPage oldalra navigál.*
* ***Administration\_MouseLeftButtonDown****: A UserChangePage oldalra navigál.*
* ***GodAdmin\_MouseLeftButtonDown****: A GodAdminPage oldalra navigál.*
* ***Statistic\_MouseLeftButtonDown****: A StatisticPage oldalra navigál.*
* ***Profile\_MouseLeftButtonDown****: A ProfilePage oldalra navigál.*

**MainViewModel.cs:**

* ***\_pageInstances****: Egy szótár (Dictionary<Type, object>), amely az egyes oldalak példányait tárolja.*
* ***\_serviceProvider****: Az IServiceProvider példánya, amely szolgáltatásokat biztosít az alkalmazás számára.*
* ***\_webSocketService****: Az IWebSocketService példánya, amely WebSocket kapcsolatok kezeléséért felel.*
* ***\_tokenService****: Az ITokenService példánya, amely a felhasználói token kezeléséért felel.*
* ***\_adminService****: Az IAdministratorService példánya, amely az adminisztrátorok adatait kezeli.*
* ***\_currentAdmin****: Az aktuálisan bejelentkezett adminisztrátor adatai (AdministratorDto).*
* ***\_currentView****: A jelenleg megjelenített oldal (object).*
* ***\_tgBtnIsChecked****: Egy boolean érték, amely jelzi, hogy a menü gomb be van-e kapcsolva.*
* ***\_superAdminMenuVisibility****: Az adminisztrátori menü láthatóságát meghatározó Visibility érték.*
* ***InitializeAsync****: Aszinkron inicializáló metódus, amely beállítja az alapértelmezett nézetet és adminisztrátori adatokat tölt be.*
* ***CloseClicked****: A WebSocket kapcsolat bezárásáért és az alkalmazás leállításáért felelős parancs.*
* ***LogoutClicked****: A kijelentkezési műveletet végrehajtó parancs.*
* ***OnPageSelected<TPage>****: Átváltja az aktuális nézetet a megadott oldalra, ha az még nincs betöltve, akkor létrehozza azt.*

**LoginView.xaml.cs:**

* ***\_serviceProvider****: Szolgáltatások injektálására szolgáló objektum.*
* *\_****authService****: Hitelesítési szolgáltatás az API-val való kommunikációhoz.*
* *\_****tokenService****: Felhasználói tokenek kezelésére szolgáló szolgáltatás.*

**RegisterView.xaml.cs:**

* *\_****serviceProvider****: Szolgáltatások injektálására szolgáló objektum.*
* *\_****authService****: Hitelesítési szolgáltatás az API-val való kommunikációhoz.*
* *\_****viewModel****: RegisterViewModel példány, amely a regisztrációs adatok kezelésére szolgál.*

**RegisterViewModel.cs:**

* **\_*serviceService:*** *Szolgáltatásokat kezelő API szolgáltatás.*
* ***\_serviceNames:*** *Szolgáltatásnevek listája a felhasználói interfészhez.*
* ***\_services:*** *Elérhető szolgáltatások adatai.*

**Függvények:**

**MainView.xaml.cs:**

* ***ListViewItem\_MouseEnter****:*

*A navigációs elemek tooltipjeinek (tt\_toDoTable, tt\_map, tt\_profile, stb.) láthatóságát kezeli a TgBtnIsChecked állapota alapján.*

* ***ToDoTable\_MouseLeftButtonDown****:*

*\_viewModel.OnPageSelected<DragAndDropTablePage>() → A DragAndDropTablePage oldalra navigál.*

* ***Map\_MouseLeftButtonDown****:*

*\_viewModel.OnPageSelected<MapPage>() → A MapPage oldalra navigál.*

* ***Administration\_MouseLeftButtonDown****:*

*\_viewModel.OnPageSelected<UserChangePage>() → A UserChangePage oldalra navigál.*

* ***GodAdmin\_MouseLeftButtonDown****:*

*\_viewModel.OnPageSelected<GodAdminPage>() → A GodAdminPage oldalra navigál.*

* ***Statistic\_MouseLeftButtonDown****:*

*\_viewModel.OnPageSelected<StatisticPage>() → A StatisticPage oldalra navigál.*

* ***Profile\_MouseLeftButtonDown****:*

*\_viewModel.OnPageSelected<ProfilePage>() → A ProfilePage oldalra navigál.*

**MainViewModel.cs:**

* ***InitializeAsync() (async)***
  + *currentView beállítása a HomePage példányára (\_serviceProvider.GetRequiredService<HomePage>()).*
  + *currentAdmin betöltése az IAdministratorService-ből a felhasználónév alapján.*
  + *\_superAdminMenuVisibility beállítása a CurrentAdmin.IsSuperAdmin értékének megfelelően (Visible vagy Collapsed).*
* ***OnPageSelected<TPage>()***
  + *Ellenőrzi, hogy az adott oldal (TPage) már létre van-e hozva. Ha nem, akkor példányosítja azt (\_serviceProvider.GetRequiredService<TPage>()).*
  + *Beállítja az \_currentView értékét az új oldalra, így az jelenik meg a felhasználó számára.*

**LoginVIew.xaml.cs:**

* ***Window\_KeyDown():*** *Enter billentyű lenyomására belépést kezdeményez.*
* ***txtUsername\_TextChanged(), txtPassword\_PasswordChanged():*** *Űrlapvalidáció indítása mezők módosításakor.*
* ***SigninValidateForm():*** *Ellenőrzi a beviteli mezőket, aktiválja a bejelentkezés gombot.*
* ***ButtonSignin\_Click():*** *Felhasználó hitelesítése és kétlépcsős azonosítás esetén megfelelő nézet megnyitása.*
* ***ButtonSignup\_Click():*** *Regisztrációs nézet megnyitása.*
* ***Border\_MouseDown():*** *Ablak mozgatását teszi lehetővé.*
* ***btnClose\_Click():*** *Az alkalmazás bezárása.*
* ***btnMinimize\_Click():*** *Az ablak minimalizálása.*
* ***forgottenPasswordBtn\_Click():*** *Elfelejtett jelszó ablak megnyitása.*

**RegisterView.xaml.cs:**

* ***Window\_KeyDown():*** *Enter lenyomásakor aktiválja a regisztrációs gombot.*
* ***txtSignupUsername\_TextChanged(), txtSignupFirstname\_TextChanged(), txtSignupLastname\_TextChanged(), txtSignupEmail\_TextChanged(), comboSignupWork\_TextChanged(), txtSignupPassword\_PasswordChanged():*** *Valídáció indítása mezők módosításakor.*
* ***SignupValidateForm():*** *Ellenőrzi az űrlapmezőket, engedélyezi vagy tiltja a regisztrációs gombot.*
* ***ButtonSignup\_Click():*** *Regisztráció végrehajtása API-n keresztül.*
* ***ButtonSignin\_Click():*** *Bejelentkezési nézet megnyitása.*
* ***Border\_MouseDown():*** *Ablak mozgatását teszi lehetővé.*
* ***btnClose\_Click():*** *Az alkalmazás bezárása.*
* ***btnMinimize\_Click():*** *Az ablak minimalizálása.*
* ***Window\_Loaded():*** *Regisztrációs nézet inicializálása.*

**RegisterViewModel.cs:**

* ***InitializeAsync():*** *Az elérhető szolgáltatásokat betölti az API-ból.*
* ***IsValidEmail():*** *E-mail cím érvényességét ellenőrzi reguláris kifejezéssel.*

**API meghívások:**

|  |  |  |
| --- | --- | --- |
| API hívás | Metódus | Leírás |
| GetByUserName() | IAdministratorService | Lekéri az aktuális adminisztrátor adatait a bejelentkezett felhasználó nevével. A felhasználónév az ITokenService-ből érkezik (\_tokenService.GetUserName()). |
| CloseWebSocketAsync() | IWebSocketService | Bezárja a WebSocket kapcsolatot. |
| RemoveToken() | ITokenService | Eltávolítja a hitelesítési tokent a rendszerből. |
| RemoveUserName() | ITokenService | Eltávolítja a bejelentkezett felhasználó nevét a rendszerből. |
| SetToken("") | AuthenticatedHttp  ClientHandler | Törli a hitelesítési tokent a hitelesített HTTP kliensből. |
| LoginView/RegisterView: | | |
| \_authService.LoginAdminAsync | **POST** | Admin bejelentkezés a megadott felhasználónévvel és jelszóval. |
| \_authService.RegisterAdminAsync | **POST** | Új admin regisztráció a megadott adatokkal. |
| \_tokenService.SaveUserName | **PUT** | A bejelentkezett felhasználónevet menti a token szolgáltatóba. |
| \_serviceService.SelectAllServices | **GET** | Az összes elérhető szolgáltatás lekérése a regisztrációhoz. |

**Eseményvezérlés:**

|  |  |  |
| --- | --- | --- |
| Esemény | Metódus | Leírás |
| btnMaximize\_Click | **private void btnMaximize\_Click**  **(object sender, RoutedEventArgs e)** | Az ablak maximalizálásáért vagy normál méretűre állításáért felelős. Ha az ablak maximálva van, akkor normál méretre állítja, különben maximalizálja. |
| btnMinimize\_Click | **private void btnMinimize\_Click**  **(object sender, RoutedEventArgs e)** | Az ablak minimalizálásáért felelős. |
| Grid\_MouseDown | **private void Grid\_MouseDown**  **(object sender, MouseButtonEventArgs mouse)** | Az ablak mozgatásáért felelős eseménykezelő. Ha a bal gombot lenyomják, akkor az ablakot a felhasználó mozgatni tudja. |
| ListViewItem\_MouseEnter | **private void ListViewItem\_MouseEnter**  **(object sender, MouseEventArgs e)** | A navigációs elemek tooltipjeinek láthatóságát kezeli. Ha a TgBtnIsChecked értéke igaz, akkor elrejti a tooltip-eket, különben láthatóvá teszi őket. |
| ToDoTable\_MouseLeft  ButtonDown | **private void ToDoTable\_MouseLeftButtonDown**  **(object sender, MouseButtonEventArgs e)** | A **DragAndDropTablePage** oldalra navigál. Az esemény akkor kerül végrehajtásra, amikor a felhasználó rákattint a ToDoTable elemre. |
| Map\_MouseLeftButtonDown | **private void Map\_MouseLeftButtonDown**  **(object sender, MouseButtonEventArgs e)** | A **MapPage** oldalra navigál. Az esemény akkor kerül végrehajtásra, amikor a felhasználó rákattint a Map elemre. |
| Administration\_MouseLeftButtonDown | **private void Administration\_MouseLeftButton**  **Down**  **(object sender, MouseButtonEventArgs e)** | A **UserChangePage** oldalra navigál. Az esemény akkor kerül végrehajtásra, amikor a felhasználó rákattint az Administration elemre. |
| GodAdmin\_MouseLeft  ButtonDown | **private void GodAdmin\_MouseLeftButtonDown**  **(object sender, MouseButtonEventArgs e)** | A **GodAdminPage** oldalra navigál. Az esemény akkor kerül végrehajtásra, amikor a felhasználó rákattint a GodAdmin elemre. |
| Statistic\_MouseLeftButtonDown | **private void Statistic\_MouseLeftButtonDown**  **(object sender, MouseButtonEventArgs e)** | A **StatisticPage** oldalra navigál. Az esemény akkor kerül végrehajtásra, amikor a felhasználó rákattint a Statistic elemre. |
| Profile\_MouseLeftButtonDown | **private void Profile\_MouseLeftButtonDown**  **(object sender, MouseButtonEventArgs e)** | A **ProfilePage** oldalra navigál. Az esemény akkor kerül végrehajtásra, amikor a felhasználó rákattint a Profile elemre. |
| LoginView/RegisterView: | | |
| KeyDown (Enter billentyű lenyomása) | **Window\_KeyDown** | Engedélyezi a bejelentkezési vagy regisztrációs gombot, majd meghívja a megfelelő metódust. |
| TextChanged (Felhasználónév) | **txtUsername\_TextChanged / txtSignupUsername\_TextChanged** | Ellenőrzi a felhasználónév mezőt és validálja az űrlapot. |
| TextChanged (Vezetéknév) | **txtSignupFirstname\_TextChanged** | Ellenőrzi a vezetéknév mezőt és validálja az űrlapot. |
| TextChanged (Keresztnév) | **txtSignupLastname\_TextChanged** | Ellenőrzi a keresztnév mezőt és validálja az űrlapot. |
| TextChanged (Email) | **txtSignupEmail\_TextChanged** | Ellenőrzi az e-mail mezőt és validálja az űrlapot. |
| PasswordChanged (Jelszó) | **txtPassword\_PasswordChanged / txtSignupPassword\_PasswordChanged** | Ellenőrzi a jelszó mezőt és validálja az űrlapot. |
| Click (Bejelentkezés) | **ButtonSignin\_Click** | Elküldi a bejelentkezési adatok API hívását és kezeli az eredményt. |
| Click (Regisztráció) | **ButtonSignup\_Click** | Elküldi a regisztrációs adatokat API hívásra és kezeli az eredményt. |
| Click (Ablak bezárása) | **btnClose\_Click** | Bezárja az alkalmazást. |
| Click (Ablak minimalizálása) | **btnMinimize\_Click** | Minimalizálja az ablakot. |
| Click (Elfelejtett jelszó) | **forgottenPasswordBtn\_Click** | Megnyitja a jelszóváltoztatási ablakot. |
| MouseDown (Ablak mozgatása) | **Border\_MouseDown** | Engedélyezi az ablak mozgatását egérrel. |
| Loaded (Ablak betöltődése) | **Window\_Loaded** | Inicializálja a szükséges adatokat a regisztrációs oldalon. |

**Stíluskészítése:**

* *A megjelenítést a ResourceDictionary segítségével alkalmazzuk.*
* **Oldalak fő felirata (mainText):**  
  A fő felirat szövegét szabályozza, amely a legnagyobb betűméretet és félkövér stílust alkalmazza. A szöveg középre van igazítva, és fehér színű.
* **Szöveg stílus (afterText):**  
  A TextBlock elemekhez rendelt stílus, amely kisebb betűmérettel és félkövér szöveggel szabályozza a menüpontok szövegét. A szöveg balra igazított, és a szövegek padolása és margója is meghatározott.
* **Attribútum szöveg stílus (afterAttributeText):**  
  Ez a stílus a kisebb, közepes súlyú szövegekhez van rendelve, ahol a betűtípus Courier New. A szöveg balra igazított és a margókat nem tartalmazza.

**ChangePage-k stílusai:**

* **CRUD szöveg stílus (crudText):**  
  A CRUD funkciókhoz használt szövegek stílusa. A betűméret 16, közepes súlyú és balra igazított. A szöveg fehér, és a font Verdana.
* **CRUD címke stílus (crudLabel):**  
  A CRUD címkékhez rendelt stílus, amely a szövegek megjelenését szabályozza a menüpontoknál. A betűméret és a betűtípus megegyezik a crudText-tel, de itt kifejezetten címkékhez van rendelve.
* **CRUD kiterjesztő stílus (crudExpander):**  
  Ez a stílus a Expander elemekhez rendelt beállításokat tartalmazza. A betűméret 16, és az interakciókhoz használt szövegek színét és elrendezését is meghatározza.
* **CRUD rádió gomb stílus (crudRadioBtn):**  
  A RadioButton elemekhez rendelt stílus, amely közepes betűméretet és fehér színt alkalmaz. A gombok háttérszíne nem módosul.
* **Új gomb stílus (crudNewBtn):**  
  A CRUD új gombokhoz rendelt stílus, amely egyedi háttérszínt és fekete szöveget alkalmaz. A betűméret 13 és a háttérszín YellowGreen.
* **Mentés gomb stílus (crudSaveBtn):**  
  A mentés gombokhoz alkalmazott stílus, amely narancssárga háttérszínt, fekete szöveget és 13-as betűméretet használ. A betűtípus Arial és a szöveg félkövér.
* **Törlés gomb stílus (crudDeleteBtn):**  
  A törlés gombokhoz rendelt stílus, amely piros háttérszínt és fekete szöveget alkalmaz. A betűméret és a szöveg stílusa megegyezik a többi CRUD gombbal.
* **DataGrid stílus (DataGridStyle):**  
  A DataGrid megjelenését szabályozza, beleértve a háttérszínt, a sorok színét és a görgetősávok megjelenését. A háttérszín #ffb500 és az alternatív sorok háttérszíne #332761. A sorok és oszlopok méretezése is be van állítva.
* **DataGrid oszlopfejléc stílus (DataGridHeaderStyle):**  
  A DataGrid oszlopfejléceinek megjelenését szabályozza, beleértve a háttérszínt #19094e és a szövegszínt, valamint a betűméretet.
* **DataGrid sorfejléc stílus (DataGrid):**  
  A sorfejléc színét szabályozza, amely #19094e háttérszínt alkalmaz, és fehér szövegezést biztosít a sorfejléceknek.

**LoginView/RegisterView:**

* *A megjelenítést* ***XAML*** *stílusokkal definiálják.*
* *A* ***titleText*** *stílus egy nagy méretű, vastag betűs, középre igazított fehér színű szövegeket hoz létre.*
* *A* ***normalText*** *stílus közepes méretű, tördelhető, középre igazított fehér szövegeket definiál.*
* *A* ***button*** *stílus 50 px magas és 200 px széles, lekerekített gombokat hoz létre, amelyek alapértelmezetten átlátszóak, fehér kerettel és fehér szöveggel. Egérrel való fölé húzáskor a háttér sötétvörösre, a szöveg fehérre változik.*
* *A* ***button2*** *stílus egy kisebb, 45 px magas és 120 px széles változata a button stílusnak, hasonló hover és kattintási effektusokkal.*
* *A* ***textHint*** *stílus halványszürke, kisebb méretű szövegeket hoz létre a felhasználói beviteli mezők segítésére.*
* *A* ***textBox*** *stílus egy minimalista szövegbeviteli mezőt biztosít, átlátszó szegéllyel és közepes méretű, szürkés szöveggel.*
* *A* ***mainbutton*** *egy 200 px széles, lekerekített, mélyvörös gomb, amely föléhúzáskor sötétebb árnyalatot kap.*
* *A* ***mainbutton2*** *ennek egy kisebb, 120 px széles és 45 px magas változata hasonló hatásokkal.*
* *A* ***textError*** *stílus egy középre igazított, piros hibaüzenet megjelenítésére szolgál, fehér háttérrel és félkövér szöveggel.*

# **Az oldalsó menü (desktop)**

**Komponens állapota:**

**MainView.xaml.cs:**

* ***\_viewModel****:  
  A MainViewModel típusú változó, amely a nézetmodell példányosításáért felelős. A komponens adatait (pl. felhasználó, aktuális oldal, menü állapot) tartalmazza, de itt nem közvetlenül változó, hanem inkább egy adatforrást ad a nézet számára.*
* ***TgBtnIsChecked****:  
  Ez az állapotváltozó a navigációs panel állapotát jelzi, hogy kinyitva van-e a menü (értéke true vagy false). A* ***ToggleButton*** *kezelése ezen alapul.*
* ***CurrentView****:  
  Az aktuálisan megjelenített oldal (pl. HomePage, DragAndDropTablePage stb.). Ezt a változót a komponens frissíti, amikor az alkalmazás különböző oldalai között navigál.*
* ***SuperAdminMenuVisibility****:  
  Ez az állapotváltozó szabályozza a szuperadmin menü láthatóságát. Ha az adminisztrátor jogosultságai lehetővé teszik, a menü láthatóvá válik (Visibility.Visible), különben elrejtődik (Visibility.Collapsed).*

**MainViewModel.cs:**

* ***\_pageInstances****:  
  A választott oldalak tárolására szolgáló szótár, amely megőrzi az oldalakat, hogy ne kelljen őket újra létrehozni.*
* ***CurrentAdmin****:  
  Az aktuális adminisztrátor adatainak tárolására szolgáló változó.*
* ***CurrentView****:  
  Az aktuálisan megjelenő nézet (oldal).*
* ***TgBtnIsChecked****:  
  A navigációs menü állapotának tárolása.*
* ***SuperAdminMenuVisibility****:  
  A szuperadmin menü láthatóságának állapotváltozója.*

**Függvények:**

**MainView.xaml.cs:**

* ***ListViewItem\_MouseEnter****:  
  A navigációs menüpontok fölé húzott egérre reagál, és láthatóvá teszi vagy elrejti az egyes menüpontok eszköztippjeit.*
* ***ToDoTable\_MouseLeftButtonDown****:  
  A "Feladatok Tábla" menüpontra kattintáskor az oldalváltás történik.*
* ***Map\_MouseLeftButtonDown****:  
  A "Térkép" menüpontra kattintáskor az oldalváltás történik.*
* ***Administration\_MouseLeftButtonDown****:  
  Az "Adminisztráció" menüpontra kattintáskor az oldalváltás történik.*
* ***GodAdmin\_MouseLeftButtonDown****:  
  A "Rendszergazda" menüpontra kattintáskor az oldalváltás történik.*
* ***Statistic\_MouseLeftButtonDown****:  
  A "Statisztikák" menüpontra kattintáskor az oldalváltás történik.*
* ***Profile\_MouseLeftButtonDown****:  
  A "Profil" menüpontra kattintáskor az oldalváltás történik.*

**MainViewModel.cs:**

* ***LogoutClicked****:  
  A felhasználó kijelentkezését végző függvény, amely a WebSocket kapcsolatot is lezárja, eltávolítja a tokeneket, és átirányítja a felhasználót a bejelentkezési oldalra.*
* ***OnPageSelected<TPage>()****:  
  A megfelelő oldal (TPage típusú) betöltését és a nézet frissítését végzi el.*

**Stíluskészítése:**

* *A megjelenítést a ResourceDictionary segítségével alkalmazzuk.*
* ***ImageBrush (Ikon megjelenítéshez):***

*Ez a képforrás ikon, amely a* ***menüelemek*** *fölötti* ***hover*** *állapotban kerül alkalmazásra, amikor a felhasználó az egérrel fölé viszi.*

* ***Szöveg stílus (font\_style):***

*Ez a stílus a TextBlock elemekhez van rendelve, és a menüpontok szövegének kinézetét szabályozza.*

* ***ToolTip stílus (tt\_style):***

*A ToolTip stílusát szabályozza, amely az eszköztippek megjelenésére vonatkozik a menüpontoknál, ha az egér fölé viszik őket.*

* ***ToggleButton stílus (tb\_style):***

*A* ***ToggleButton*** *stílusa, amely a menü gombok interakcióit kezeli.*

**Kommunikáció:**

* ***Stílusok és Trigger-ek Kommunikációja:***

*A menüpontok* ***stílusai*** *(tb\_style és tt\_style) biztosítják, hogy a felhasználói interakciók (pl. hover, kattintás) vizuális visszajelzésekkel párosuljanak. A ToggleButton elemek a ControlTemplate és a Triggers segítségével vezérlik az interakciókat, míg a* ***ToolTip*** *a felhasználónak információt ad a menüpontok funkcióiról.*

# **REST API:**

# **A tervezett projekt bemutatása:**

* Projekt címe:  
  ClearDrive
* Rövid esettanulmány, amely bemutatja a problémát, amire a pályázók szoftveres megoldást keresnek:  
  Nagy tapasztalattal rendelkező, szegedi gépjárművezető oktató hívta fel a figyelmünket arra, hogy a mindennapi (szegedi) közlekedés során fellépő rendellenességek (veszélyek: kidőlt fa az úton, fa letört ága veszélyezteti a parkoló autókat; hibák: jelzőlámpa meghibásodott, úthibák, nem látszik az útfelfestés, nem látszik a tábla/lámpa; illetve illegális szemétlerakatok) bejelentése kizárólag telefonon keresztül lehetséges. Ezen felül valamennyi ezen problémák közül más-más fenntartóhoz tartozik, amelyek között magánszemélyek számára elérhető központi bejelentési lehetőség nincs, így nehezítve a hiba bejelentését, elhárítását. Ezt a problémát lenne hivatott az általunk tervezett szoftver áthidalni.

* Rövid leírás arról, hogy a tervezett szoftver hogyan fogja megoldani az ismertetett problémát:  
  A közlekedés résztvevői:
  + jelenteni tudják az utakon talált veszélyeket és hibákat (veszélyek: kidőlt fa az úton, fa letört ága veszélyezteti a parkoló autókat; hibák: jelzőlámpa meghibásodott, úthibák, nem látszik az útfelfestés, nem látszik a tábla/lámpa; illetve illegális szemétlerakatok),
  + tájékoztatva legyenek a közlekedési akadályokról az utakon,
  + megtervezzék a veszély- és hibamentes útvonalat,
  + a fenntartók gyorsan és hatékonyan tudják elhárítani a keletkezett hibákat.
* Az alkalmazás tervezett főbb képességei:
  + Veszélyek/akadályok bejelentése a hatóságoknak
  + Hibajegy státuszának követése
  + Tájékozódás az útviszonyokról
  + Fenntartóként jelentések felülvizsgálata
  + Továbbítás az illetékes szerveknek
  + Értesítések
* Kik lesznek az applikáció egyes felhasználói és mit csinálnak a rendszerben:  
  A közlekedési rendszer web, mobil és desktop felhasználói:  
  a közlekedés résztvevői:
  + sofőrök,
  + gyalogosok,

illetve a hivatalok valamint azok szakemberei:

* Csongrád Megyei Állami Közútkezelő KHT.
* Csongrád Megyei Közlekedési Felügyelet
* Közterület-felügyelet
* Szegedi Közlekedési Kft.
* Szegedi Környezetgazdálkodási Nonprofit Kft.
* MVM Lumen

A felhasználók feladatai a rendszerben:

* Közlekedés résztvevői: Jelentik az utakon észlelt veszélyeket, hibákat, zavaró tényezőket.
* Hivatalok: A jelentéseket felülvizsgálja és dönt azok kijavításáról.
* Szakembereket küldenek ki a helyszínre, annak érdekében, hogy kijavítsák a felülvizsgált hibákat (pl.: kidőlt oszlopok, nem működő közvilágítás, szemét összeszedése).
* Az applikáció tervezett felépítése:  
  *Web:*
  + Tájékozódás az útviszonyokról, naprakész térképen megjelölt pontokon láthatják a hiba típusát, majd rákattintva a részleteit
  + Bejelentés hatóságoknak és más felhasználóknak: hibajegyek => fényképek és/vagy videók feltöltése, szöveges hibabejelentés, térképen való helyszínmegjelölés típusokkal
  + Hibajegy útjának nyomon követése (vizsgálat alatt, elfogadva, elutasítva, megoldás alatt)
  + Statisztika: adott városok útviszonya viszonyítva az átlaghoz és bejelentések száma viszonyítva más felhasználókhoz

*Mobil:*

* + A webalkalmazás kiterjesztése mobil platformra
  + A mobil appon belül fénykép/videó készítése majd feltöltése
  + Felugró ablakban értesítést kap a felhasználó az alkalmazástól a hibajegy állapotával kapcsolatban
  + Hibajegy nyomonkövetése (vizsgálat alatt, elfogadva, elutasítva, megoldás alatt)
  + Adott helyzetünk által láthatjuk a körülöttünk lévő környezetet, problémákat

*Desktop(kizárólag intézmények számára):*

* + Hibajegyek felülvizsgálata (elfogadva, elutasítva, megoldásra vár, kiadva, megoldva)
* Liftbeszéd:  
  A közlekedést igénybe vevők számára, akik szeretnének gördülékenyebben közlekedni az országban a ClearDrive-al, egy közlekedésfelügyeleti szoftverrel, amely segítségével az utakon lévő hibákat, közlekedési akadályokat és észrevételeket jelenteni lehet. Nem úgy, mint a Waze ahol csak forgalmi információkról lehet tájékozódni, amelyek nem jutnak el a hatóságokhoz és általában nincs mindenhol megfelelően frissítve. Ezzel szemben a mi programunk a **ClearDive** teljesen ingyenes, a jelentés minden esetben hatósági felülvizsgálatra kerül és megfelelő időközönként frissítjük annak érdekében, hogy a közlekedés résztvevői időben tájékoztatva legyenek.

# **AB-tervek-leírás\_ClearDrive:**

**ClearDrive adatbázis felépítése**

**Tervezés**

**Egyedek:** Felhasználó, Hibajegy, Szervek, Adminisztrátor

**Tulajdonságok:**

Felhasználó (**userID**, felhasználóNév, jelszó, email, telszám, rangNév)

Hibajegy (**hibaID**, leírás, státusz, fénykép, hibatípus, prioritás, szélesség, hosszúság)

Szervek (**szervID**, név, cím, email, telszám)

Adminisztrátor (**adminID**, név, jelszó, email, telszám)

HibaFeltőltés (**userID**, **hibaID**, bejelentésIdeje) - kapcsolótábla a Felhasználó és a Hibajegy táblák között

**Egyedek és tulajdonságaik (adattípusok):**

Felhasználó

**userID** szöveg (CHAR(36)) - elsődleges kulcs

felhasználóNév szöveg (VARCHAR(256)) – a felhasználó neve

jelszó szöveg (LONGTEXT(256)) – a felhasználó jelszava

email szöveg (VARCHAR(256) – a felhasználó email címe

telszám szám (TINYINT) - 11 db szám

rangNév szöveg (INT(11)) – a felhasználó rangjának neve

Hibajegy

**hibaID** szám (INT) – AUTO\_INCREMENT – elsődleges kulcs

leírás szöveg (TEXT) – a hiba leírása

státusz szöveg (VARCHAR(256)) – a hiba státusza

fénykép

hibatípus szöveg (VARCHAR(256)) – a hiba típusa

prioritás szám (TINYINT) - a hiba prioritása

szélesség szám (INT) - a hiba szélessége

hosszúság szám (INT) - a hiba hosszúsága

Szervek

**szervID** szám (INT) – AUTO\_INCREMENT – elsődleges kulcs

név szöveg (VARCHAR(256)) – a szerv neve

cím szöveg (VARCHAR(256)) – a szerv címe

email szöveg (VARCHAR(256)) – a szerv email címe

telszám szám (TINYINT) - 11 db szám

Adminisztrátor

**adminID** szám (INT) – AUTO\_INCREMENT – elsődleges kulcs

név szöveg (VARCHAR(256)) – az adminisztrátor neve

jelszó szöveg (VARCHAR(256)) – az adminisztrátor jelszava

email szöveg (VARCHAR(256)) – az adminisztrátor email címe

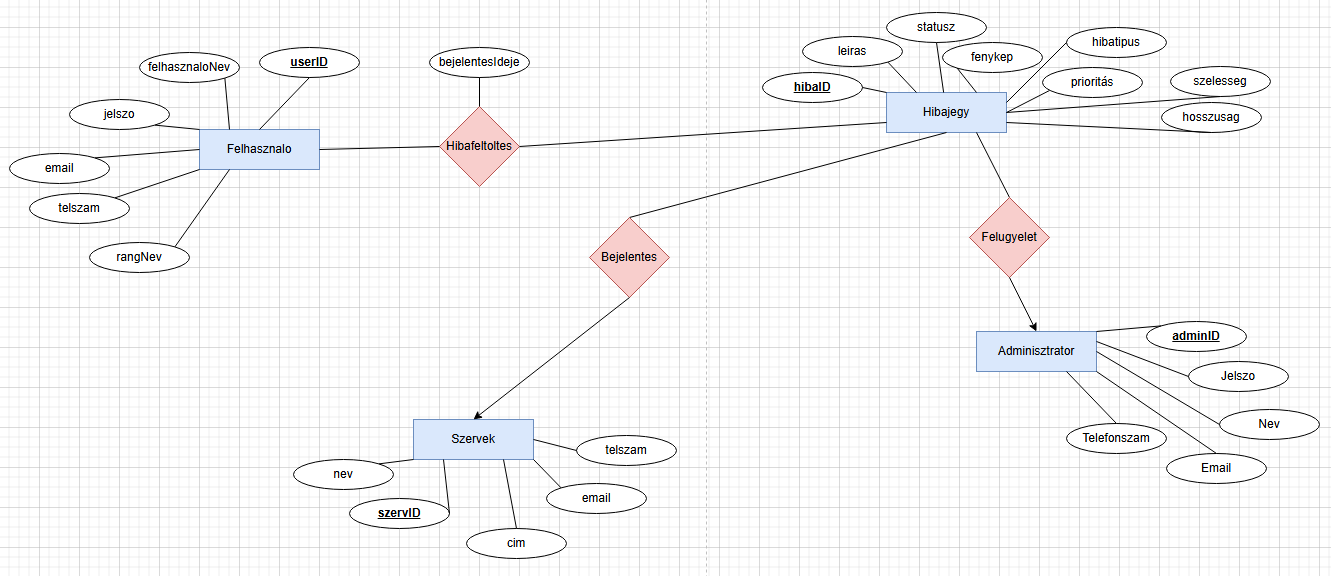
telszám szám (TINYINT) - 11 db szám

Hibafeltőltés

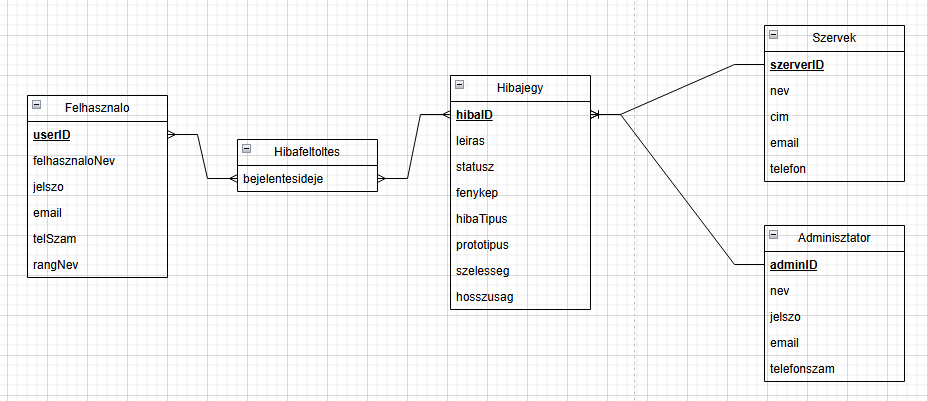
**userID** szám (INT) – idegen kulcs

**hibaID** szám (INT) – idegen kulcs

bejelentésIdeje idő (TIME) – év, hónap, nap, óra, perc-ben megadott érték

**Az adatbázis E-K diagramja:**

**Az adatbázis Bachmann-ábrája:**

Felhasználókat tart nyilván és egyben a felhasználókhoz tartozó hiba jegyeket is. A weboldal és a mobil lehetőséget ad arra, hogy egy űrlapon az adott hiba adatait kiválaszthassuk és megadjuk, a hiba helyszínét, majd a hibához tartozó összes tulajdonságot megadjuk (leírás, státusz, fénykép(nem kötelező, de ajánlott), hibatípus, prioritás, szélesség(a térképre kattintva automatikusan kitöltésre kerül a mező), hosszúság(a térképre kattintva automatikusan kitöltésre kerül a mező)). Lehetőség van továbbá újabb hiba felvételére űrlapon. A felhasználók regisztráció után jelenthetnek be hibákat. Regisztrált és nem regisztrált felhasználóknak is lehetőségük van az adatbázisban tárolt hibák elérésére. Kereshetnek adott feltöltés idejére, hiba típusra, státuszra, leírásra.Fejlesztési terület lehet, hogy a hibákat különböző típusba soroljuk, pl. NaturalDisaster, RoadProblem, EnvironmentalIssue, Other. Ehhez újabb táblát kellene létrehozni (Hibatípusok), amelyet az Hibajegy táblához kellene kapcsolni. Így hiba típusra is rá lehet keresni.

**Az asztali alkalmazás funkciói (adminisztrátor)**

* Új felhasználó regisztrálása.
* Bejelentkezés az alkalmazásba.
* Hibajegyek rendezése táblázatban, közben megtudja nézni a hibajegyek leírását, rájuk kattintva megjelenik a térkép amely megjelöli a hiba helyét.
* A felhasználó adatait tudja módosítani, törölni a felhasználót.
* A hiba adatait tudja módosítani, törölni a hibát.

**A webes alkalmazás funkciói (felhasználó)**

* Új felhasználó regisztrálása.
* Bejelentkezés a felületre.
* A térképre kattintva hibákat tud feltölteni a felhasználó, megadva leírását, típusát, képet tud hozzá csatolni.
* Figyelemmel tudja kísérni a feltöltött hibajegyei állapotát.

**A mobil alkalmazás funkciói (felhasználó)**

* Új felhasználó regisztrálása.
* Bejelentkezés a felületre.
* Vezetés közben egy gombra kattintva gyorsan elmenti a telefon helyzetét és később ki tudja tölteni a hiba adatait.
* A térképre kattintva hibákat tud feltölteni a felhasználó, megadva leírását, típusát, képet tud hozzá csatolni.
* Figyelemmel tudja kísérni a feltöltött hibajegyei állapotát.

# **Sziiv-orai:**

**ClearDrive**

*Kovácsházi Ádám, Krisztin Benedek László, Surányi István*

## Szegedi SZC Vasvári Pál Gazdasági és Informatikai Technikum*, 6722 Szeged, Gutenberg u. 11*

1. **Bevezetés**

Nagy tapasztalattal rendelkező, szegedi gépjárművezető oktató hívta fel a figyelmünket arra, hogy a mindennapi (szegedi) közlekedés során fellépő rendellenességek (veszélyek: kidőlt fa az úton, fa letört ága veszélyezteti a parkoló autókat; hibák: jelzőlámpa meghibásodott, úthibák, nem látszik az útfelfestés, nem látszik a tábla/lámpa; illegális szemétlerakatok, tilosban parkoló autók) bejelentése kizárólag telefonon keresztül lehetséges, munkaidőben. Ezen felül valamennyi ezen problémák közül más-más fenntartóhoz tartozik, amelyek között magánszemélyek számára elérhető központi bejelentési lehetőség nincs, így nehezítve a hiba bejelentését, elhárítását. Ezt a problémát hivatott az általunk tervezett szoftverrendszer áthidalni. A közlekedést igénybe vevők számára, akik szeretnének gördülékenyebben közlekedni az országban a ClearDrive-al, egy közlekedésfelügyeleti szoftverrel, amely segítségével az utakon lévő hibákat, közlekedési akadályokat és észrevételeket jelenteni lehet. A mi programunk a ClearDive teljesen ingyenes, a jelentés minden esetben hatósági felülvizsgálatra kerül és megfelelő időközönként frissítjük annak érdekében, hogy a közlekedés résztvevői időben tájékoztatva legyenek.

1. **Probléma megoldásának menete**

A fejlesztett szoftver rendszer célja, hogy egy hatékony megoldást biztosítson a közlekedési akadályok, veszélyek és hibák bejelentésére, amelyet a közlekedés résztvevői könnyen használhatnak. Az alábbi lépések mentén valósult meg a rendszer fejlesztése:

### 2.1. Tervezési fázis

A fejlesztés során elsődleges szempont volt egy olyan rendszer kialakítása, amely lehetővé teszi a különböző platformok (Asztali, Mobil, Web) zökkenőmentes működését egy egységes backend szolgáltatás mellett. Ennek érdekében:

* Egy .NET C# alapú backendet terveztünk, amely MySQL adatbázist használ az adatok tárolására,
* A backenden egy REST API-t valósítottunk meg,
* A felhasználói felületekre különböző technológiákat alkalmaztunk:
  + **Blazor**: Webes alkalmazás, amely reszponzív felhasználói felületet biztosít,
  + **MAUI**: Mobil alkalmazás, amely Android és iOS rendszereken is futtatható,
  + **WPF**: Asztali alkalmazás Windows rendszerekre, amely a fenntartó szervek számára lesz hasznos.

### 2.2. Fejlesztési fázis

A fejlesztés agilis módszertannal történt, amely során:

* **Backend fejlesztése**:
  + A rendszer alapja egy C# nyelven írt REST API, amely az adatbázis lekérdezéseket és az adatok validálását végzi,
  + MySQL adatbázist alkalmaztunk a bejelentések, felhasználók és visszajelzések tárolására
* **Frontend fejlesztése**:
  + Blazor technológiával készült el a webes felület.
  + A MAUI keretrendszert használjuk a mobil alkalmazás fejlesztésére,
  + A WPF alkalmazás a hatósági szerveket szolgálja.

### 2.3. Tesztelés és hibajavítás

A fejlesztett alkalmazásokat kiterjedt tesztelésnek vetettük alá:

* **Egységtesztek** biztosították az egyes modulok megfelelő működését.
* **Integrációs tesztek** segítettek a backend és frontend kommunikáció validálásában.
* **Felhasználói tesztelés** során visszajelzéseket gyűjtünk, amelyeket felhasználunk a végleges verzióban.

A fenti fejlesztési folyamat eredményeként létrejött egy stabil, többplatformos rendszer, amely hatékonyan segíti a közlekedési problémák bejelentését és azok kezelését.



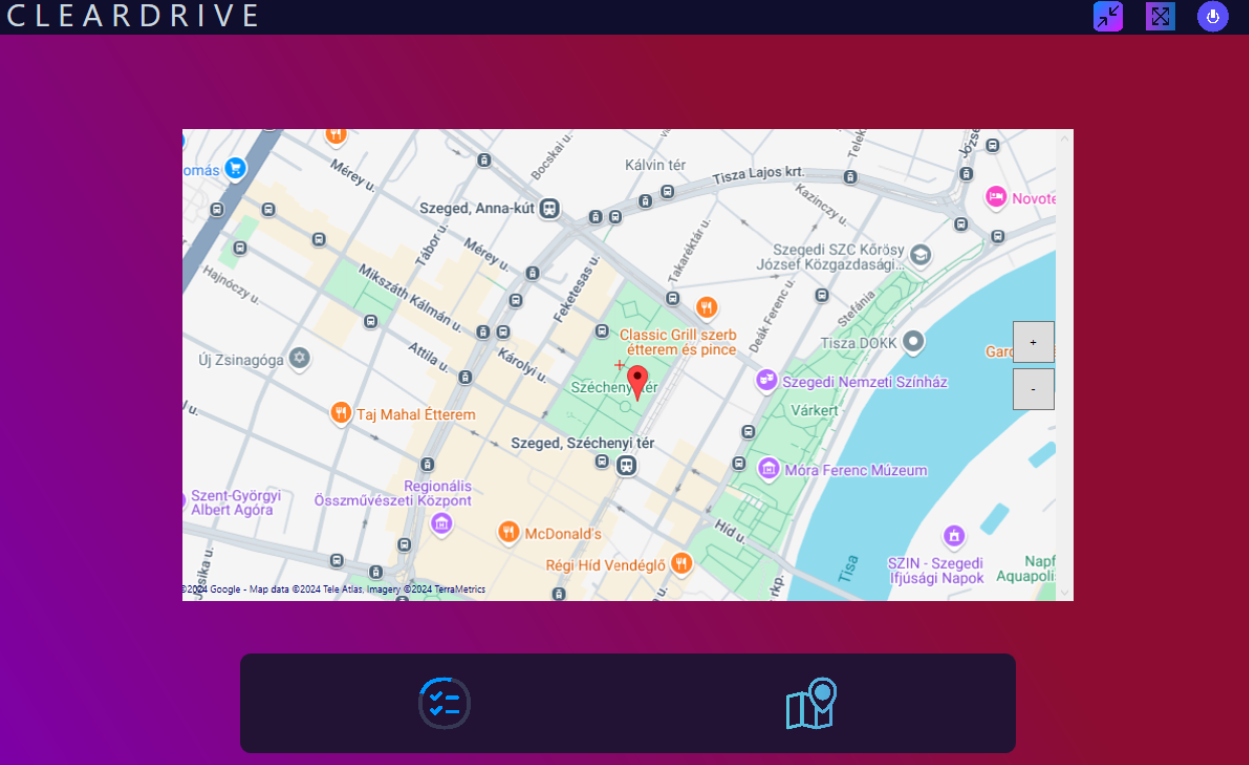
**3. Elért eredmények**

A fejlesztési folyamat során létrehozott rendszer számos előnnyel rendelkezik a közlekedési hibák és veszélyek hatékony bejelentésében és kezelésében. Az alábbi eredményeket értük el:

* **Felhasználóbarát bejelentési felület**: Az alkalmazás egyszerű kezelőfelületet biztosít a bejelentések gyors és pontos leadására.
* **Platformfüggetlen működés**: A rendszer webes (Blazor), mobil (MAUI) és asztali (WPF) verzióban is elérhető, így széles körben használható.
* **Valós idejű értesítések**: A felhasználók és hatóságok értesítést kapnak a friss bejelentésekről, lehetővé téve a gyors reagálást.
* **Biztonságos és skálázható rendszer**: A .NET és MySQL alapú architektúra biztosítja a megbízható működést.

A fejlesztett rendszer nemcsak a bejelentések kezelésére, hanem a közlekedésbiztonság növelésére is alkalmas, mivel a felhasználók és hatóságok közötti kommunikációt jelentősen megkönnyíti.

|  |  |  |  |
| --- | --- | --- | --- |
| **Munkafolyamat neve** | **Ki (kik végzik)?** | **Lépések** | **Hibalehetőségek,**  **speciális dolgok** |
| **Térkép** | Felhasználó | A térkép menüpontra nyomva megjelenik a térkép | Hibásan jelenik meg a térkép |
| **Fotó/Videó feltöltés** | Felhasználó | A telefont a gépre csatlakoztatva, majd a weboldalra felmenve lehet feltölteni a megfelelő menüben | Nem érzékeli a gép a bedugott eszközt |
| **Elérhetőségek** | Felhasználó | Egy menüpontra kattintva megjelennek a céggel kapcsolatos elérhetőségek (pl.: telefonszám) | Az elérhetőség hibásan van megadva (régi telefonszám) |
| **Statisztika** | Felhasználó | Egy menüpontra kattintva lekérdezi a felhasználó adatait(pl.: mennyi bejelentése volt eddig) | Az adatok hibásan/-más adatai jelennek meg |



# **Forgatókönyvek:**

# **Záródolgozat megvalósítása felhasználó központú módszerrel**

## **A tervezés után áttérünk a program megvalósításra.**

## **5.a. Szoftver funkciók és szolgáltatások (munkafolyamatok) rendszerezése**

<*Rendszerezze az egyes alkalmazások főbb funkcióit, kezdje a legegyszerűbben megvalósítható funkcióktól és haladjanak az összetettebb funkciók felé>*

**A ClearDrive web alkalmazásban**

1. Tájékozódás az útviszonyokról

2. Statisztika elemzése

3. Elérhetőségek

4. Kép/videó feltöltés

5. Térkép alapján útvonal készítés

**A ClearDrive mobil alkalmazásban**

1. Térkép felülete és az alapján a(z) veszélyek/akadályok észrevétele

2. Kép/videó feltöltés

3. Térkép alapján útvonal készítés

4. Hibajegy státuszának követése

**A ClearDrive desktop alkalmazásban**

1. Jelentések felülvizsgálata, majd hibajegy készítése

2. Szakemberek kiküldése feladat elvégzésére

3. Jelentések felülvizsgálata után értesítés a felhasználóknak a hibajegy státuszáról

## **5b. Forgatókönyvek**

## **< *A táblázat kitöltése segít meghatározni a programban elérhető fő funkciókat, menüpontokat.***

## ***Mind a három alkalmazásra (desktop, mobil, web) fogalmazzák meg őket!***

## ***Az első oszlopba írjuk azt a tevékenységet (menü pontot) ami leír egy tevékenységet.***

## ***A következő oszlopba írjuk, hogy a tevékenységet a felhasználó milyen lépések sorozataként oldja meg. Ezek leírásánál lehet használni a GUI elemek megnevezését (pl. egy legördülő listában kiválasztja).***

*A harmadik oszlopba kerülnek a műveletek során fellépő váratlan események, hibalehetőségek.*>

**Desktop alkalmazás**

Kovácsházi Ádám, Surányi István, Krisztin Benedek László

|  |  |  |  |
| --- | --- | --- | --- |
| **Munkafolyamat neve** | **Ki (kik végzik)?** | **Lépések** | **Hibalehetőségek,**  **speciális dolgok** |
| **Térkép** | Admin | A térkép menüpontra nyomva megjelenik a térkép | Hibásan jelenik meg a térkép |
| **Todo táblázat** | Admin | A 3 oszlop közül az elsőben vannak az elvégzendő feladatok, azoknak az átvétele | A feladatokat nem lehet átvenni |

**Web alkalmazás**

Kovácsházi Ádám, Surányi István, Krisztin Benedek László

|  |  |  |  |
| --- | --- | --- | --- |
| **Munkafolyamat neve** | **Ki (kik végzik)?** | **Lépések** | **Hibalehetőségek,**  **speciális dolgok** |
| **Térkép** | Felhasználó | A térkép menüpontra nyomva megjelenik a térkép | Hibásan jelenik meg a térkép |
| **Fotó/Videó feltöltés** | Felhasználó | A telefont a gépre csatlakoztatva, majd a weboldalra felmenve lehet feltölteni a megfelelő menüben | Nem érzékeli a gép a bedugott eszközt |
| **Elérhetőségek** | Felhasználó | Egy menüpontra kattintva megjelennek a céggel kapcsolatos elérhetőségek (pl.: telefonszám) | Az elérhetőség hibásan van megadva (régi telefonszám) |
| **Statisztika** | Felhasználó | Egy menüpontra kattintva lekérdezi a felhasználó adatait(pl.: mennyi bejelentése volt eddig) | Az adatok hibásan/-más adatai jelennek meg |

**Mobil alkalmazás**

Kovácsházi Ádám, Surányi István, Krisztin Benedek László

|  |  |  |  |
| --- | --- | --- | --- |
| **Munkafolyamat neve** | **Ki (kik végzik)?** | **Lépések** | **Hibalehetőségek,**  **speciális dolgok** |
| **Térkép** | Felhasználó | A térkép menüpontra nyomva megjelenik a térkép | Hibásan jelenik meg a térkép |
| **Fotó/Videó feltöltés** | Felhasználó | Az alkalmazásban engedélyezi a kamera használatát | Véletlen rákattintani a ‘nem engedélyezem’ gombra ezzel letiltani a kamera használatot |

# **Felhasználóközpontú tervezés:**

# Vizsgaremek tervezés felhasználó központú módszerrel

Projekt témája: Közlekedés során észlelt problémák elhárítása

Projektben kifejlesztett szoftver rendszer elnevezése: **ClearDrive**

Projekt címe: **ClearDrive** - Web, Desktop, Mobil

## 1. Összefüggések és célok.

A biztonságos közlekedésért: veszélyek(kidőlt fa az úton, fa letört ága veszélyezteti a parkoló autókat) és hibák (jelzőlámpa meghibásodott, úthibák, nem látszik az útfelfestés, nem látszik a tábla/lámpa) jelentése, helyük megjelölése, tájékoztatás, illetve illegális szemétlerakatok megjelölése

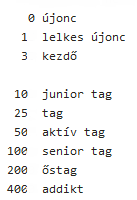
A közlekedés felügyeleti szoftver rendszer célja, hogy az azt használó felhasználók:

* tájékoztatva legyenek a közlekedési akadályokról az utakon,
* jelenteni tudják az utakon talált veszélyeket és hibákat,
* a felhasználók megtervezzék a veszély- és hibamentes útvonalat az országon belül

Ezen felül az illetékes szervek feladatot kapnak a munkahelyükön, hogy javítsák az elbírált hibákat.

A **ClearDrive** szoftverrendszer web, mobil és desktop alkalmazásában:

* **web**:
  + Tájékozódás az útviszonyokról, naprakész térképen megjelölt pontokon láthatják a hiba típusát, majd rákattintva a részleteit
  + Bejelentés hatóságoknak és más felhasználóknak: hibajegyek => fényképek és/vagy videók feltöltése, szöveges hibabejelentés, térképen való helyszínmegjelölés típusokkal
  + Hibajegy útjának nyomon követése (vizsgálat alatt, elfogadva, elutasítva, megoldás alatt)
  + Statisztika: adott városok útviszonya viszonyítva az átlaghoz és bejelentések száma viszonyítva más felhasználókhoz
  + Felhasználói fiók:összesített bejelentések száma, ebből mennyi lett kijavítva, rang rendszer:



* **mobil**:
  + A webalkalmazás kiterjesztése mobil platformra
  + A mobil appon belül fénykép/videó készítése majd feltöltése
  + Felugró ablakban értesítést kap a felhasználó az alkalmazástól a hibajegy állapotával kapcsolatban
  + Hibajegy nyomonkövetése (vizsgálat alatt, elfogadva, elutasítva, megoldás alatt)
  + Adott helyzetünk által láthatjuk a körülöttünk lévő környezetet, problémákat
* **desktop***(kizárólag intézmények számára):*
  + Hibajegyek felülvizsgálata (elfogadva, elutasítva, megoldásra vár, kiadva, megoldva)
  + Felhasználóknak való visszajelzés
  + Rendszer statisztika(mennyien használják napi szinten, aktívan a rendszert), hátralévő bejelentések megtekintése
  + Továbbítás az illetékes szerveknek

## Csongrád Megyei Állami Közútkezelő KHT.

## Csongrád Megyei Közlekedési Felügyelet

## Közterület-felügyelet

* + - Szegedi Közlekedési Kft.

## Szegedi Környezetgazdálkodási Nonprofit Kft.

* + - MVM Lumen

## 2.Kik a rendszer használói? Mi a feladatuk a rendszerben?

A közlekedési rendszer web, mobil és desktop felhasználói: a közlekedés résztvevői (pl.:

sofőrök, gyalogosok) illetve a hivatalok valamint azok szakemberei:

## Csongrád Megyei Állami Közútkezelő KHT.

## Csongrád Megyei Közlekedési Felügyelet

## Közterület-felügyelet

* Szegedi Közlekedési Kft.

## Szegedi Környezetgazdálkodási Nonprofit Kft.

* MVM Lumen

A felhasználók feladatai a rendszerben:

**Közlekedés résztvevői:** Jelentik az utakon észlelt veszélyeket, hibákat, zavaró tényezőket.

**Hivatalok:** A jelentéseket felülvizsgálja és dönt azok kijavításáról. Szakembereket küldenek ki a helyszínre, annak érdekében, hogy kijavítsák a felülvizsgált hibákat(pl.: kidőlt oszlopok, nem működő közvilágítás, szemét összeszedése).

Feladatok (vörös út):

**mobil**:

* 1. Térkép felülete és az alapján a(z) veszélyek/akadályok észrevétele
* 1. Kép/videó feltöltés
* 2. Térkép alapján útvonal készítés
* 3. Hibajegy státuszának követése

**desktop**:

* 1. Jelentések felülvizsgálata, majd hibajegy készítése
* 2. Szakemberek kiküldése feladat elvégzésére
* 3.Jelentések felülvizsgálata után értesítés a felhasználóknak a hibajegy státuszáról

**web**:

* 1. Tájékozódás az útviszonyokról
* 2. Statisztika elemzése
* 3. Elérhetőségek
* 3. Kép/videó feltöltés
* 3. Térkép alapján útvonal készítés
* 4. Hibajegy státuszának követése

**mobil/desktop/web**:

* 1. Dizájn

## 3. (elképzelt) Interjú készítés a felhasználókkal

### 3.a Szcenáriók (a felhasználók jelenleg milyen tevékenységet végeznek és hogyan?)

b0kovada, b0kriben, b0surist

|  |  |  |
| --- | --- | --- |
| Felhasználó megnevezése! | Felhasználó tevékenysége | Jelenleg hogyan csinálja? |
| Közlekedés résztvevői | Közlekedés közben megfigyelt hibákat és észrevételeket jegyzi, az alkalmazáson keresztül jelenti | 1. Megfigyeli a hibákat 2. Telefonon diszpécserrel kommunikál a megadott időben 3. Várja a hiba javítását, amelyről nem kap visszajelzést   (de általában ki sem javítják) |
| Hivatalok | A hibajegyeket megkapja, felülvizsgálja és dönt azok kijavításáról majd kijavítja | 1. Tájékozódik az utak állapotáról bejövő telefonhívások által 2. Kiadja a feladatot a talált rendellenességről az   adott szakembernek   1. A szakember kimegy a megadott helyre és   elvégzi feladatát |

### 3.b „Pain point” – Fájdalompontok és megoldási javaslatok!

b0kovada, b0kriben, b0surist

|  |  |
| --- | --- |
| Kényes, problémás kérdések és válaszok az interjú során! | Megoldási javaslatok! |
| Mi van ha nagyobb a probléma, mint ahogy azt a bejelentő gondolta, és a rendelkezésre álló eszközök segítségével nem lehet kijavítani? | Akkor jelzünk egy olyan alvállalkozónak, akiknek van rendelkezésre álló eszköze a probléma javítására, esetlesegen azoknak a bérlése is szóba kerülhet. |
| A mobil alkalmazás jelentős internetforgalmat generál, ha azonnal a helyszínen akarom feltölteni az úttal kapcsolatos észrevételeimet? | Nem feltétlenül, az alkalmazás beállításainál be lehet kapcsolni az adatcsökkentett módot mellyel alacsony felbontású média válik csak elérhetővé, így jóval kevesebb mobilinternetet használ az alkalmazás. |
| Lesz-e a hivatalok részére külön funkció mely segítségével letilthatják a gyanús, spam felhasználókat? | Igen lesz, egy külön menüpontra kattintva meg tudják nyitni azt a fület amelyben ezt megtehetik. |

### 3.c. A képzeletbeli felhasználók tulajdonságainak összefoglalása

b0kovada, b0kriben, b0surist

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Szereplő | Tulajdonságai | Céljai | Fenntartásai (félelmei) | Tipikus idézetek az interjúból |
| Közlekedés résztvevői | Általában sietnek a célhelyre, valamint törekednek a szabályos közlekedésre. | A lehető leggazdaságosabban és leggyorsabban szeretnének eljutni a célba, mások akadályoztatása és járművük épségének megőrzése mellett. | A probléma pontos helyének elfelejtése, mobileszköz kompatibilitása. | “azt gondolom, az alkalmazás biztonságosabbá fogja tenni a mindennapi közlekedést” |
| Hivatalok | Lassú munkavégzés, nem érdekük hamar befejezni a munkát.  Nem kapkodják el a munkát (lassú döntéshozatal), nem tartják nagy gondnak a kisebb kátyúkat. | Elhárítani az úthibákat, tábla és egyéb problémákat.  A problémák felülvizsgálata majd az alkalmazottak kijelölése a problémák megoldására. | Nagyobb a probléma, mint ahogy azt a bejelentő gondolta, és a rendelkezésre álló eszközök segítségével nem lehet kijavítani.  Sok hamis jelentés érkezik, mely időt vesz el a dolgozótól | “ennek segítségével a programban sokkal gördülékenyebben megy majd az útkarbantartók értesítése” |

### 4. Liftbeszéd

A közlekedést igénybe vevők számára, akik szeretnének gördülékenyebben közlekedni az országban a CleraDrive-al, egy közlekedésfelügyeleti szoftverrel, amely segítségével az utakon lévő hibákat, közlekedési akadályokat és észrevételeket jelenteni lehet. Nem úgy, mint a Waze ahol csak forgalmi információkról lehet tájékozódni, amelyek nem jutnak el a hatóságokhoz és általában nincs mindenhol megfelelően frissítve. Ezzel szemben a mi programunk a **ClearDive** teljesen ingyenes, a jelentés minden esetben hatósági felülvizsgálatra kerül és megfelelő időközönként frissítjük annak érdekében, hogy a közlekedés résztvevői időben tájékoztatva legyenek.

A ppt-hez:

2-es pont:

Számozottan mi a legfontosabb a felhasználói köröknek. (vörös utak)

Felhasználó központú tervezés:

* emberek megfigyelése
* kérdések
* telefonálás során érdeklődés
* eltérő igények megfigyelése
* felhasználói profilok(persona)
* vörös utak(legfontosabb feladatok - hierarchia)
* ***felhasználó feladataira összpontosítás***
* sok idő
* agilis módszer
* felhasználók megfigyelése a szoftver használata közben, hatékonyságot mérjük(hány embernek sikerült, milyen gyorsan, élmények)
* ***a felhasználói viselkedés empirikus mérése***
* folyamatos tesztelés (papír alapon)
* megfelelő dizájn
* elektronikus prototípus
* ***ismétlődő tervezés***

**A PPT elkészítése:**

## Prezentáció készítés a TDD módszerről

A csoport egy prezentációt készít a TDD módszerről és előadja azt!

1. A prezentáció oldalain csak szakkifejezések és kódok állhatnak!
2. A prezentáció oldalhoz tartozó szöveget külön kell megírni, nem jelenhet meg a vetítésben!
3. A prezentációt elő kell adni, mindenki több diát kell elmondjon! A beszéd közben nem nézheti a vetítést!
4. A dia legalább 6 legfeljebb 12 diából álljon!

A dia a jövőbeli érettségi / szakmai vizsga bemutató része, ezért komoly munka az elvárt hozzáállás!

### 

### 

### 

### 

### 

### 

### 

### 

### 

### 

### **UML Osztálydiagram alap**

1. **Felhasználó**
   * **Attribútumok:**
     + felhasználónév: String
     + jelszó: String
     + email: String
     + szerep: Szerep (pl. Közlekedés résztvevő, Hivatal)
   * **Metódusok:**
     + jelentVeszelyt()
     + követHibajegyet()
     + statisztikaMegjelenites()
2. **Hibajegy**
   * **Attribútumok:**
     + id: Int
     + leírás: String
     + statusz: Statusz (pl. vizsgálat alatt, elfogadva)
     + helyszín: String
     + fénykép: String
   * **Metódusok:**
     + hozzaadHiba()
     + frissitStatusz()
3. **Térkép**
   * **Attribútumok:**
     + pontok: List<Pont>
   * **Metódusok:**
     + megjelolHelyszin(hiba: Hibajegy)
     + keresHibakat()
4. **Pont**
   * **Attribútumok:**
     + lat: Float
     + lng: Float
     + hibaTipus: String
5. **Adminisztrátor**
   * **Attribútumok:**
     + adminId: Int
   * **Metódusok:**
     + felulvizsgalHibajegyek()
     + kikuldiSzakember()
     + statisztikaMegjelenites()
6. **Statisztika**
   * **Attribútumok:**
     + bejelentettHibak: Int
     + javitasAlattHibak: Int
   * **Metódusok:**
     + szamolStatisztika()

### **Kapcsolatok**

* **Felhasználó** -> **Hibajegy**: 1  
  (Egy felhasználó több hibajegyet is jelenthet)
* **Hibajegy** -> **Térkép**: 1:1 (Egy hibajegy egy adott helyszínhez kapcsolódik)
* **Adminisztrátor** -> **Hibajegy**: 1  
  (Az adminisztrátor felülvizsgálja a hibajegyeket)
* **Felhasználó** -> **Statisztika**: 1:1 (Minden felhasználónak van saját statisztikája)