Szakdolgozat

Formai követelmények:

A szakdolgozatot kemény kötésben kell leadni, 1 példányban.

Fedőlapjának színe fekete, aranyszínű feliratokkal.

Lap: A4-es méret, színe fehér

Betűméret: 12 pont

Sorok: sorkizárt igazítás, 1,5-es sortávolság

Margó:

- belső: 3,5 cm

- külső: 2,5 cm

- alsó: 2,5 cm

- felső: 2,5 cm

Oldalszám: az oldalszámozást a tartalomjegyzéktől kezdve az irodalomjegyzékkel bezárólag folyamatosan kell végezni. Jelölése arab számokkal történik (a tartalomjegyzék oldalainak az oldalszámát nem szokás feltüntetni).

## A dolgozat fő fejezetei (1. szintű címsorok) új oldalon kezdődjenek.

A szakdolgozat a hallgató önálló munkája, melyben be kell tartani a hivatkozások és idézések standard szabályait.

(lapszámozás, címsorok, tartalomjegyzék, stílusok, futó fejléc, ábraszámozás, stb.)

Belső fedőlap

Témabejelentő

Tartalomjegyzék

Tartalomjegyzék

[Oldalszám 1](#_Toc323154875)

[A dolgozat fő fejezetei (1. szintű címsorok) új oldalon kezdődjenek. 1](#_Toc323154876)

[Bevezetés 4](#_Toc323154877)

[Mit csinál ez a program 4](#_Toc323154878)

[Milyen környezetben 4](#_Toc323154879)

[Milyen eszközökkel 4](#_Toc323154880)

[Milyen hozzáállással készült 4](#_Toc323154881)

[Felhasználói dokumentáció 5](#_Toc323154882)

[Mire való a szoftver 5](#_Toc323154883)

[Célközönség 5](#_Toc323154884)

[Futtatási környezet 5](#_Toc323154885)

[Üzembe helyezés 5](#_Toc323154886)

[Használati útmutató 5](#_Toc323154887)

[Rendszerfunkciók 5](#_Toc323154888)

[Rendszerüzenetek 5](#_Toc323154889)

[Fejlesztői dokumentáció 6](#_Toc323154890)

[Megoldási terv 6](#_Toc323154891)

[Rendszer Architektúra 6](#_Toc323154892)

[Adatbázis 6](#_Toc323154893)

[Csomag, Modul és Osztályszerkezet 6](#_Toc323154894)

[Felhasználói Felület 6](#_Toc323154895)

[Megvalósítás 6](#_Toc323154896)

[#Ez mi a fene, cleancode-ról meséljek egy egész fejezetet? Oké... 6](#_Toc323154897)

[Tesztelés 7](#_Toc323154898)

[Junit 7](#_Toc323154899)

[MVP / Database interface / Sycron vs Asyncron hívások 7](#_Toc323154900)

[RDF Adatbázis mérete 7](#_Toc323154901)

[Eeh...? Eredményhelyesség - kihagyós 7](#_Toc323154902)

[Hatékonyság elemzés: GWT Speed Tracer 7](#_Toc323154903)

[Irodalomjegyzék 8](#_Toc323154904)

# Bevezető

A Program

A programom egy weboldal, aminek két célja van. Az egyik, hogy bárki kényelmesen találhasson magának és ajánlhasson másoknak utazási célt, és ott található érdekes látnivalókat, szórakozóhelyeket. A másik, hogy népszerűsítse és egyszerűvé tegye az országon belüli utazást.

Az első célt egy térképes kereső biztosítja. Az oldalon egy térképen lehet úticélokat keresni és kinézni. A kinézett utazási célok magyarországi falvak vagy városok, ahova el lehet jutni vonattal. Ezeken az utazási céloknál meg lehet nézni a látnivalókat, szórakozóhelyeket, múzeumokat, és egyéb érdekes helyeket, illetve bárki hozzáadhat egy általa ismert új érdekességet is.

A másik célt a kereséssel összefűzött elvira MÁV menetrendje biztosítja. Az utazás keresésekor időpontot és a vonatokon igénybevehető kedvezményeket is ki lehet választani. Ezekkel az adatokkal az utazással kapcsolatos vonat információkat is megjeleníti a honlap, például a vonatok indulását, érkezését, a jegy árát, indulási vágány számát és egyéb hasznos adatokat mutat meg, minden keresés alkalmával.

Környezet

A feltelepített program az internetről bárhonnan elérhető, és bármelyik korszerű böngészővel használható. A program Java (1.6) nyelven készítettem, és egy war fájlként futtatható egy tetszőleges web-szerveren, például Glassfish vagy Apache Tomcat application server segítségével.

Eszközök

A keretrendszer amiben az alkalmazást fejlesztettem a Google Web Toolkit(GWT)[[1]](#endnote-1), ami roppantul leegyszerűsíti a webes alkalmazások fejlesztését, és elfedi a böngészők közötti különbséget. A keretrendszer biztosítja kliens és a szerver közötti egyszerű kommunikációt RPC[[2]](#endnote-2) segítségével, így ezt választottam az összekapcsolásukra. Sajnos ezt a technológiát nem használhattam ki a MÁV Elvira rendszerének lekérdezéséhez. Az Elvira egy REST API-n keresztül kérdezhető le, így a tőle szükséges adatokat ezen keresztül szereztem be. A program egy saját RDF adatbázist használ az útvonalak és városok meghatározásához, ezt bővíti a felhasználók által megadott adatokkal.

Hozzáállás

A szakdolgozat írásakor nagyon nagy hangsúlyt fektettem a kód minőségére és átláthatóságára. A munkámra nagy hatással volt Robert C. Martin[[3]](#endnote-3) és Martin Flower[[4]](#endnote-4) könyvei, amikben az olvasható, tiszta kód készítését írják le. Igyekeztem a szakdolgozatnak minden részét modulárisan, kicsi és könnyen érthető elemekből felépíteni, és bár nem sikerült mindenhol megvalósítani az elveket, de mindenhol törekedtem a kódom olvashatóságára és szépségére, az öndokumentáló változó, függvény és osztálynevek használatára.

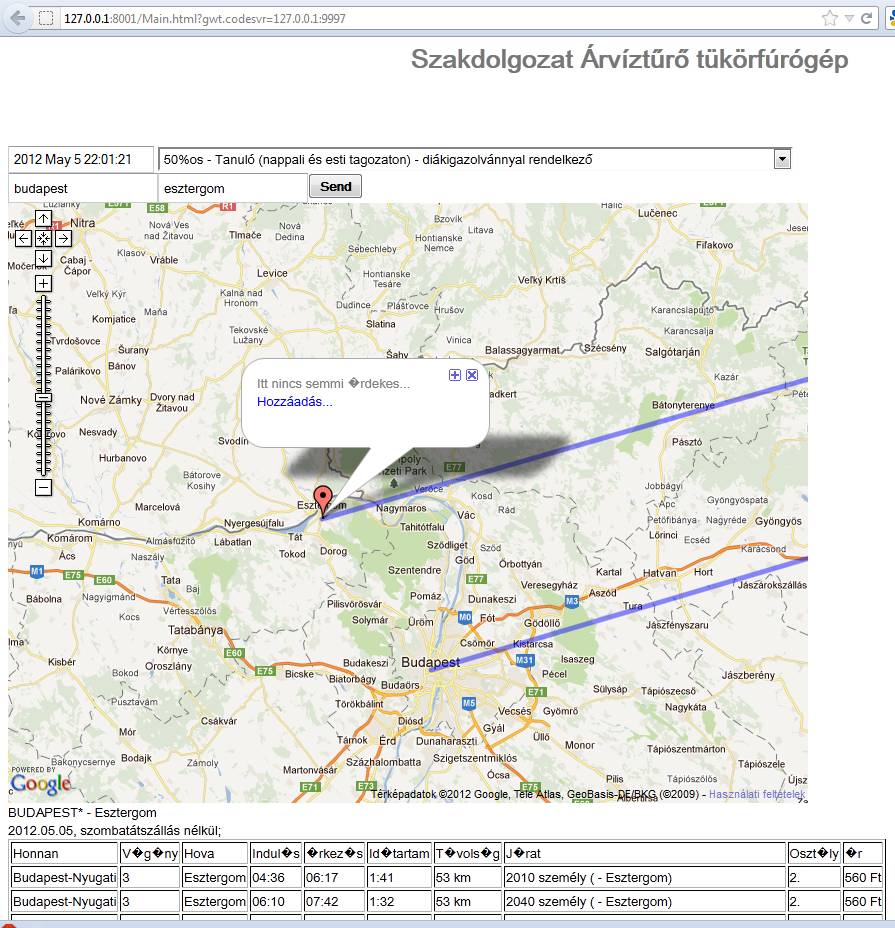
# Felhasználói dokumentáció

Bevezető

Ez a program egy webes alkalmazás ami információkat biztosít olyan kirándulásokhoz, utazásokhoz, ahol az utazók vonattal terveznek menni. A honlapon a célállomások érdekességei közt lehet böngészni, és új érdekességeket hozzáadni.

Mivel a weblapnak szükségtelen a felhasználók adatait eltárolni a feladat ellátásához, a honlap használata regisztrációhoz illetve bejelentkezéshez nem kötött, bárki szabadon használhatja. A széles felhasználói közönség miatt a felhasználói felületet a lehető legegyszerűbbre lett tervezve.

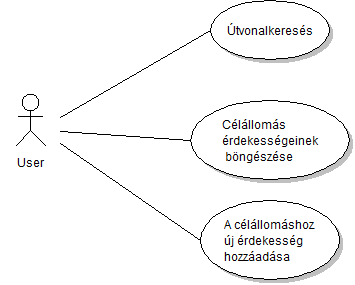
TODO Ide rakni egy olyan képet a programról, amin már nem ilyen ronda.



. Felhasználói felület

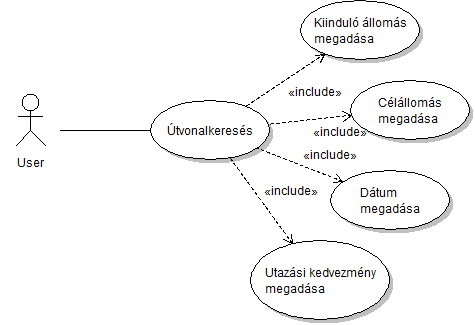
## Felhasználói esetek

A felhasználó alapvetően három interaktív dolgot csinálhat a honlapon, azaz olyasmit, amihez klikkelni vagy gépelni kell:



. Felhasználói esetek

Ebből az útvonalkeresés részletesen úgy történik, hogy a felhasználó kitölti a kereséshez szükséges mezőket. Az „honnan” és „hova” mezőket kötelező kitölteni, különben a keresés nem lehetséges. A dátum és a kedvezmények kiválasztása nem kötelező: ebben az esetben aznapi dátummal és kedvezmény nélkül, teljes árú vonatjeggyel érkeznek meg az adatok az oldalra.

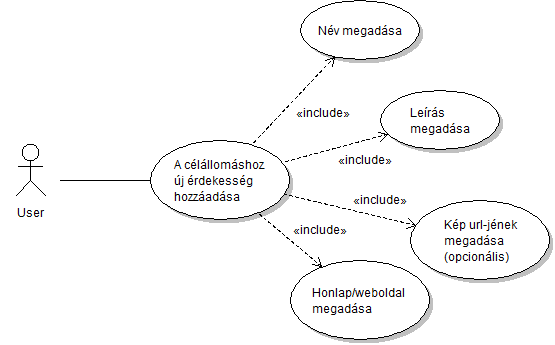


. Útvonalkeresés

Sikeres útvonalkeresés után a térképen megjelenik a vonat által bejárt út. Ennek kezdete a kezdeti, a jelölővel megjelölt vége pedig a célállomás.

A második, azaz „célállomások böngészése” felhasználói esetben a felhasználó a térképen a sikeres útvonalkeresés után a térképen felbukkanó jelölőre kattintva nyithatja meg annak a városnak a listáját, amin a jelölő áll. A felugró ablakban a listában szereplő elemekre kattintva nyithatja meg az egyes helykről szóló rövid leírásokat, valamint a leírásban szereplő linkre kattintva az adott érdekesség honlapját.

Ennek a listának a legalsó kék „Új hely hozzáadása” elemével lehet új érdekes helyet hozzáadni a listához. Ekkor ki kell tölteni a hely nevét, rövid leírását, az érdekesség honlapját, valamint opcionálisan egy neten található kép webcímét megadni.



. Új érdekesség hozzáadása

Amennyiben ez a lista üres, akkor a hozzáadási funkción kívül csak egy felirat látszik, közölve a felhasználóval, hogy „Itt nincsen semmi érdekes...”.

## Futtatási környezet

### Kliens

A weboldal minden modern böngészőben működik, ahol a „modern” a jelenleg letölthető legfrissebb böngészőket jelentik. Ezen kívül a böngészőben engedélyezve kell lennie a javascript kód futtatásának, és mivel honlapról van szó, természetesen szükséges az internet hozzáférés is.

### Szerver

A szerver futtatási környezete egy tetszőleges Web Application Server(WAS)[[5]](#endnote-5) (például Glassfish vagy Tomcat), amire a war kiterjesztésű fájlban lévő alkalmazást fel kell telepíteni. Ennek a telepítésnek a részletei változóak, attól függően, hogy milyen WASra telepítünk. A szerveroldali program, mint minden Java program, egy Java Virtual Machine (JVM)-ban fut, aminek virtuálismemória mérete legalább 500 MB méretű kell hogy legyen.

## Üzembe helyezés

A szerver üzembe helyezéséhez megfelelően fel kell paraméterezni az alkalmazás JVM-jét, valamit

## Használati útmutató

Általános felhasználói tájékoztató (például a szokásostól eltérő képernyő-, billentyű-, illetve egérkezelés leírása, teendők hibaüzenetek esetén stb.).

## Rendszerfunkciók

A rendszer funkcióinak ismertetése. A feladat jellegéből fakadóan célszerű lehet ezt folyamatszerűen, képernyőképekkel alátámasztva bemutatni. A funkciókat ajánlatos a felhasználói szintek szerint csoportosítani. Itt vegyük figyelembe, hogy a leírás a fejlesztői dokumentációban meghatározott részfeladathoz illeszkedik-e, az ott meghatározott funkciókat/használati eseteket írja-e le?

## Rendszerüzenetek

A rendszer futás közbeni üzenetei (hibaüzenetek, figyelmeztető üzenetek, felszólító üzenetek stb.) és azok magyarázata – az esetleges üzemeltetési teendőkkel együtt. Itt vegyük figyelembe, hogy tartalmaz-e biztonsági, illetve hibaelhárítási előírásokat?

Egyéb, a szoftver használatához szükséges információk.

# Fejlesztői dokumentáció

## Megoldási terv

Ez a fejlesztői leírás része, a rendszerterv, amelyből az alkalmazás célja, felépítése és működése megérthető, ez alapján az alkalmazás forráskódja lényegében elkészíthető.

### Rendszer Architektúra

Rendszer architektúrájának leírását (alrendszerek, rétegek bemutatása, az alkalmazott szabványok, technológiák, fejlesztő módszerek megadása, felhasznált eszközök és kész komponensek definiálása). Az értékelésnél vegyük figyelembe, hogy mennyire válnak szét az alkalmazás rétegei (például felhasználói felület, logika, adatforrás)?

### Adatbázis

Az adatbázis – feltéve, hogy van – leírását. Érdemes egy áttekintő diagammal szemléltetni a táblákat és a köztük levő kapcsolatokat, majd külön táblázatokban megadni az egyes táblák mezőszerkezeti leírását, az esetleges tárolt eljárások, függvények, triggerek, stb leírását.

### Csomag, Modul és Osztályszerkezet

Modul és/vagy osztályszerkezet (fontosabb modulok és/vagy osztályok és azok metódusai, továbbá ezek kapcsolatának) leírását. Az egyes csomagok fő eljárásait illetve a fontos osztályok fő metódusait bemenő-, kimenőadat, tevékenység hármassal jellemezni kell.

### Felhasználói Felület

A felhasználói felület – feltéve, hogy van – tervét (a képernyő- és listaterveket, valamint a menütervet). Legyen egy áttekintő ábra, amely mutatja a képernyők (ablakok, weblapok) közti navigálási lehetőségeket, irányokat. Ki kell emelni a fontosabb felhasználói eseménykezeléseket.

## Megvalósítás

A fejlesztői leírásnak a megvalósításról szóló része bemutatja, hogy milyen döntéseket kellett hozni a terv megvalósítása során (adatábrázolás, felhasznált komponensek, kódban alkalmazott nyelvi elemek, stb). A dokumentáció ne tartalmazza a forrásprogramot (legfeljebb csak fontosnak ítélt részleteit), elég azt a mellékelt adathordozón elhelyezni. A megvalósítás a fentieken kívül tartalmazza a komponens tervet (az alkalmazás fizikai komponenseinek kapcsolatrendszerét) és azok telepítésének módját.

### #Ez mi a fene, cleancode-ról meséljek egy egész fejezetet? Oké...

A forráskód tartalma, szerkezete megfelel-e a tervnek?

 Mennyire ismeri a hallgató az adott fejlesztő eszközt (pl. korszerű, hatékony nyelvi elemek vannak-e túlsúlyban, vagy ehelyett bonyolult, nehézkes, körülményes és leginkább terjengős forráskódot eredményező nyelvi elemek jellemzik a kódot)? Indokoltak-e a választott nyelvi elemek használata?

 Milyen a forráskód külalakja, mennyire áttekinthető (strukturáltság, bekezdések, tagolások, kommentezés stb.)?

 Mennyire módosítható a kód. Alkalmazza-e a hallgató a kód-újrafelhasználás nyelvi eszközeit (függvények, származtatás, generikus elemek)?

 Törekszik-e a hatékony adatábrázolásra?

 Mennyire öndokumentáló a kód, vagyis a választott azonosítók (pl. változónevek) mennyire beszédesek, konvencionálisak, a megjegyzések mennyire segítik a kódértést?

 Tartalmazza a szükséges ellenőrzési, hibakezelési funkciókat, általában megoldott-e a kivételkezelés?

 Mennyire gazdálkodik jól az emberi és gépi erőforrásokkal, így például a felhasználó idejével és türelmével, a lemezkapacitással és a memóriakapacitással?

## Tesztelés

Ez is a fejlesztői leírás része, amelynek a tesztelési szempontokat kell bemutatnia, és a tesztelés során szerzett tapasztalatokat összegeznie valamint a szoftver skálázhatóságáról készített elemzést kell tartalmaznia.

Az értékelésnél vegyük figyelembe, hogy a dokumentáció:

### Junit

 Tartalmaz-e tesztelési terveket, teszteseteket (Ezeket csoportosíthatja rendszerteszt és modultesztek szerint illetve fekete és fehérdoboz megközelítéssel)?

### MVP / Database interface / Sycron vs Asyncron hívások

 Beszámol-e olyan tanulságokról, amelyek alapján meg kellett változtatni a korábbi implementációs döntéseket, esetleg a terv egyes elemeit (az ilyen tapasztalatok nem rontják a dolgozat értékét)?

### RDF Adatbázis mérete

 Tartalmazza-e nagy adattömeg melletti futtatások értékelését?

### Eeh...? Eredményhelyesség - kihagyós

 Elemzi-e a program által adott eredmény helyességét (különösen olyan optimalizációs feladatok esetén, ahol több helyes megoldást valamilyen célfüggvénnyel lehet rangsorolni)?

### Hatékonyság elemzés: GWT Speed Tracer

 Elemzi-e a program futásának hatékonyságát?

## 

# Irodalomjegyzék

GWT dokumentáció

Robert C. Martin – Clean Code

Martin Flower – Refactoring

1. GWT: Google Web Toolkit – Keretrendszer webes alkalmazások fejlesztéséhez [↑](#endnote-ref-1)
2. RPC :Remote Procedure Call – Elosztott rendszerek kommunikációjához használt technológia [↑](#endnote-ref-2)
3. Robert C. Martin – Clean Code – Könyv illetve előadássorozat a jól olvasható kód írásáról [↑](#endnote-ref-3)
4. Martin Flower – Refactoring – Köny a kód „refaktorálásáról”, struktúrájának javításáról [↑](#endnote-ref-4)
5. WAS – Web Application Server – Speciális Java alkalmazások futtatási környezetét biztosító alkalmazásszerver [↑](#endnote-ref-5)