Proiect ISI
Treasure Hunt în București

-Echipa Impostorii-

Stoica Daniel-Cosmin 342C2
Țaran Elena-Georgiana 341C5
Fechet Ștefan 341C5

CUPRINS

CUPRINS FIGURI

1. Definirea temei

Proiectul își propune ca utilizatorii să descopere locuri din București pe baza unei metode interactive de tip treasure-hunt. Treasure hunt-ul este un minigame prin care se oferă hint-uri și ghicitori pe hartă pentru descoperirea obiectivelor.

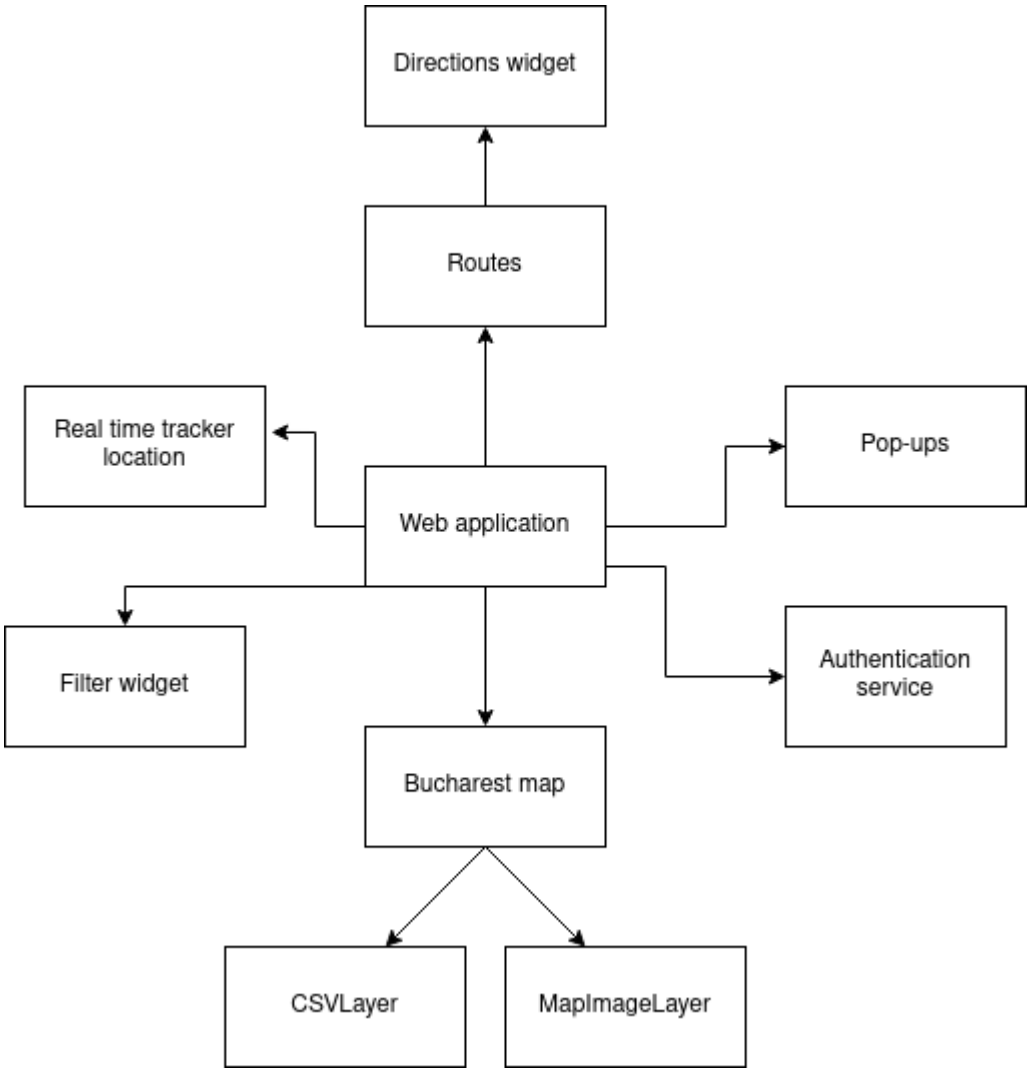
- Grup țintă: orice persoană cu vârsta cuprinsă între 10-45 ani
- Tipuri de obiective: istorice, culturale etc.
- Functionalități aplicație: actualizare locație în timp real, ghicitori și indicii pentru găsirea locației, treasure hunt-uri cu tematici diferite

2. Obiectivele proiectului

- Obiective din punctul de vedere al utilizatorului:
  - Popularizarea și explorarea anumitor zone mai puțin cunoscute din București
  - Obținerea unor informații de cultură generală într-un mod distractiv și interactiv
  - Promovarea unui stil de viață sănătos prin provocarea de a nu folosi mijloacele de transport în comun sau mașina personală
  - Încurajarea oamenilor de a petrece mai mult timp în aer liber

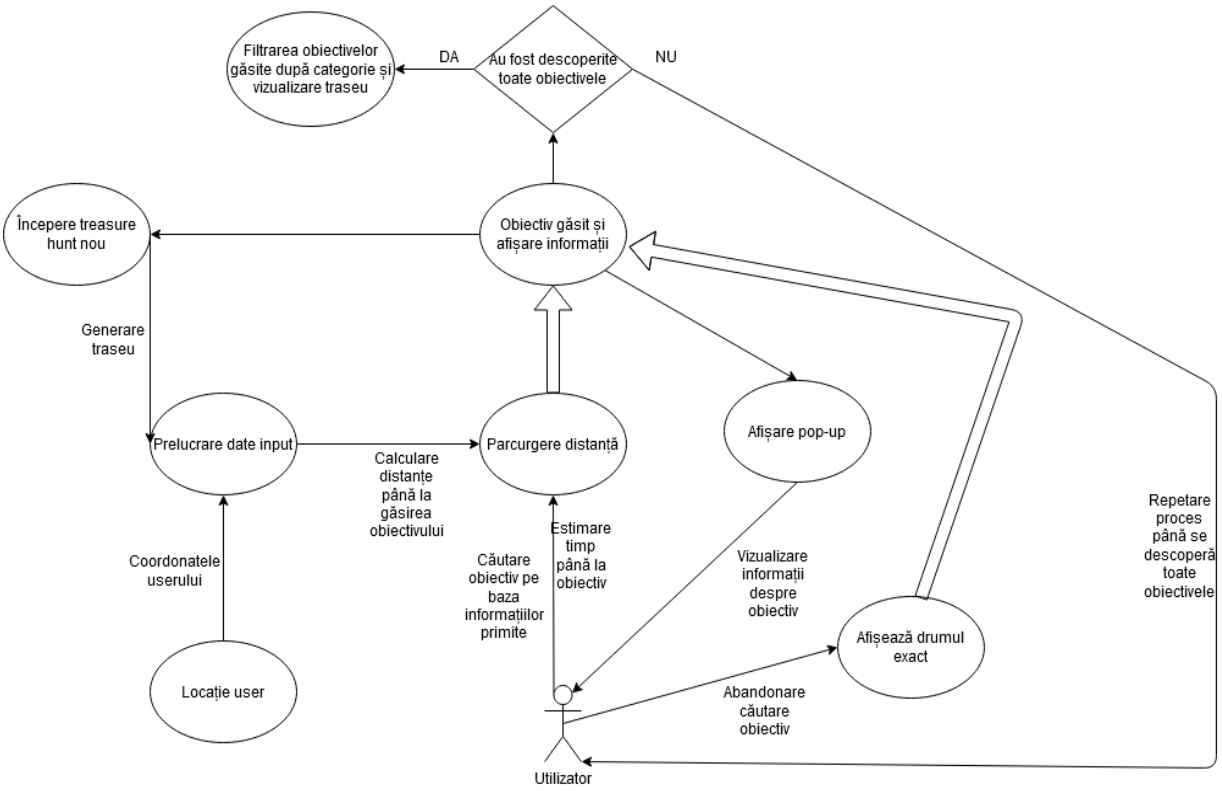
- Obiective din punctul de vedere al dezvoltatorilor:
  - Detectarea poziției utilizatorului în timp real
  - Opțiunea de a calcula ruta de la poziția curentă până la obiectiv
  - Filtrarea obiectivelor după categorie
  - Simularea mișcarii utilizatorului

3. Schema bloc a aplicației



Figură 1 - Schema bloc a aplicației

4. Diagrama use-case



Figură 2 - Diagrama use-case

5. Planificarea și documentația inițială pentru realizarea proiectului

Pentru realizarea proiectului vom utiliza JavaScript împreună cu bibliotecile ArcGIS. Fiecare treasure-hunt vă fi implementat folosind zone de tip “heat” ce vor fi stilizate în conformitate cu tematica. Utilizatorul curent va avea asignat un marker, iar poziția acestuia este aproximată în timp real. Fiecare heat zone va avea un pop-up care va conține ghicitoarea spre obiectiv. La fiecare update de locație al utilizatorului, se va calcula distanța de la el până la punctul în care trebuie să ajungă și se vor afișa mesaje corespunzătoare direcției în care se îndreaptă, modificându-se heat zone-ul. Pentru cazul în care se va dori aflarea drumului direct către obiectiv, vom implementa o rută de la locația curentă a utilizatorului până la acesta. Pentru a afișa corespunzător fiecare pop-up cu hint-uri vom reține distanța la fiecare actualizare, iar în momentul în care utilizatorul va atinge un prag, acesta va fi afișat în mod

automat. Pentru testare, vom avea salvate anumite puncte statice (se vor simula coordonatele) în care vor fi declanșate evenimentele asociate.

La momentul actual, am identificat următoarele task-uri:

- Implementare heat-zones
- Implementare tracker
- Implementare calculare distanța + afișare pop-up
- Scriere hint-uri/ghicitori
- Stilizare componente hartă
- Implementare drum complet către obiectiv

Afișarea hărții se va face 2D, folosind un ***MapView.*** Se va modifica acest ***view*** astfel încât să se faciliteze ideea de treasure-hunt. Fiecare obiectiv se află într-o bază de date, iar baza de date este sub forma unui fișier CSV. Datele vor fi adăgate pe hartă folosind ***CSV Layer*** Pentru hartă am ales să folosim un ***basemap topo-vector,*** întrucât ne facilitează desenarea clădirilor. Astfel, dacă un obiectiv va fi o clădire, acesta poate fi ușor recunoscut la găsirea lui. Pe partea de UI se vor modela anumite componente specifice temei aplicației. Spre exemplu, navigarea pe hartă nu va fi posibilă, iar widget-urile de zoom vor fi șterse.

Fiecare membru al echipei se va implica în diferite task-uri:

- ***Cosmin***: se va ocupa de implementarea tracker-ului, de desenarea unei zone de unde începe un treasure-hunt, de calcularea distanțelor și de simularea mișcării punctului care să urmeze anumite coordonate.
    - Pentru implementarea tracker-ului: un utilizator va avea asignat un punct (marker) stilizat. La intrarea in aplicație se va activa automat locația curentă (după acceptarea folosirii locației). Un utilizator nu are posibilitatea de a naviga pe hartă (navigarea poate conduce la găsirea instantă a unui obiectiv). În acest sens, se vor șterge ***widget***-urile de + - pentru zoom și se vor bloca toate acțiunile pe care un utilizator le poate face pentru navigarea pe hartă (atât prin tastatură, cât și prin mouse sau tap pe hartă). De asemenea, scale-ul/zoom-ul hărții vor fi prestabilite. Pentru a face posibilă oprirea locației, se va desena pe hartă și un ***widget al activării/dezactivării locației***
    - Fiecare obiectiv va fi cuprins într-o zonă marcată pe hartă (ori heat-zone, ori va fi desenată folosind ***Graphic layer***). Alegerea unui treasure-hunt va presupune si desenarea drumului din locația curentă a utilizatorului până la zona în care se află obiectul.
    - Calcularea distanțelor presupune o funcție care primește două coordonate (sub formă de latitudine si longitudine) și calculează, pe baza acestora, distanța în metri. Pe baza acestei distanțe se vor afișa mesaje de ghidare

- Fiind un tracking în timp real, este foarte dificil să testăm și să prezentăm funcționalități ale aplicației, astfel folosim un simulator al mișcării utilizatorului. În acest sens se va folosi un ***layer de geolocate*** care, la activarea locației, va transmite coordonate simulate markerului. Coordonatele vor fi hard-codate astfel încât punctul să se miște pe o direcție prestabilită.
- Implementarea unui ***Map Image Layer*** care să arate traficul din București în timp real, pentru a evita zonele aglomerate.

- ***Ștefan:*** se va ocupa de crearea și integrarea unui fișier csv și de styling.
  - Este absolut necesar să reunim toate punctele de interes din București într-un singur loc. Întrucât nu există niciun layer Arcgis care să realizeze acest lucru, trebuie să realizăm noi această funcționalitate. Soluția cea mai simplă din punct de vedere al implementării este crearea unui fișier csv. Acesta trebuie să conțină doar câmpurile absolut necesare funcționării optime a aplicației, și anume:

    1. Numele atracției turistice
    2. Câteva informații cu privire la importanța locației respective
    3. Latitudinea
    4. Longitudinea

    Nu este suficientă doar crearea acestui fișier și popularea lui cu infomații. Evident, aplicația trebuie sa comunice cu el și să extragă informațiile de care are nevoie. Acest pas este crucial, deoarece pe baza lui se va implementa heatmap-ul, dar și ruta din locația curentă a utilizatorului la punctul de interes.

- ***Georgiana:*** se va ocupa de realizarea rutei celei mai scurte de la locația utilizatorului la locația din treasure hunt, dar și de pop-up-uri.
  - Fiind vorba de un treasure hunt, un element esențial este reprezentat de afișarea direcției de la locația curentă la obiectivul turistic. Arcgis oferă această funcționalitate, cu mențiunea ca este necesar ca utilizatorul sa introduca credențialele pentru autentificarea pe contul de LearnArcgis, de fiecare dată când folosește această opțiune. Acest lucru este neplăcut, astfel că, trebuie configurat un proxy care să faca un bypass la acest login form care apare permanent
  - La intrare în aplicație se alege treasure hunt-ul prin pop-up. Când ajunge/intră în zona de proximitate a obiectivului, există mereu activ un pop-up care afișează mesajul cu direcția de la utilizator la acel punct.

****Alte mici funcționalități/task-uri vor fi implementate la nivel de echipă/asignate ulterior (precum legendă, descriere hint-uri ș.a.m.d.). Pe măsură ce vom implementa, se vor documenta în acest fișier.**

6. Documentație Cosmin:

Pentru implementarea trackerului s-a folosit clasa ***Track***, unde s-a precizat view-ul și s-a personalizat markerul folosind clasa ***Graphic***. Trackerul actualizează locația în timp real.

Figură 3 – Tracker Widget

S-a implementat o funcție JavaScript numită ***disableZooming*** care primește ca parametru un view, iar pentru acest view se exclud toate widgeturile, cât și controalele pentru zoom/navigate (dacă userul ar fi putut face astfel de acțiuni, îi era mai ușor să găsească obiectivul, astfel i se predefinește un zoom convenabil nemodificabil). După aceste modificări, funcția returnează noul view modificat.

Pentru mișcare, se face override la scale-ul hărții astfel încât să nu se poată vedea celelalte obiective.

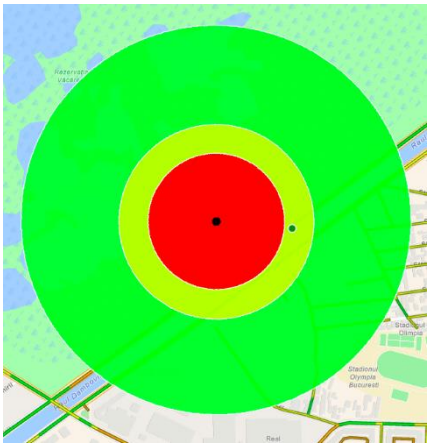
Pentru partea de marcare a zonei obiectivului s-a creat o funcție numită DrawCircle care primește 3 parametri:

View-ul unde vor fi desenate zonele marcate

Latitudinea obiectivului

Longitudinea obiectivului

Pe baza acestor parametri se desenează cele 3 zone care marchează un obiectiv și au un offset random deplasat față de obiectivul cu coordonatele (lat,lon). Dacă zonele ar fi fost cecuri concentrice cu centrul în (lat,lon), atunci obiectivul s-ar fi aflat mereu în centrul tuturor zonelor.



Figură 4 - Heat point - obiectiv

Fiecare zonă marcată are un offset față de obiectivul în sine. Offsetul se calculează în mod aleator. Pentru a adăuga o astfel de distanță, se translatează (lat,long) în radiani, se adaugă distanța și se translatează înapoi în grade. După acest calcul, se stabilesc razele zonelor și se desenează, folosind clasa ***Graphic***, pe hartă.

Pentru calculul distanței dintre două puncta a fost implementată o funcție care primește 4 parametri:

Lat1 care este latitudinea primului punct

Lon1 care este longitudinea primului punct

Lat2 care este latitudinea celui de-al doilea punct

Lon2 care este longitudinea celui de-al doilea punct

Funcția returnează distanța dintre două puncte, cu mențiunea că distanța presupune o linie dreaptă (nu se ține cont de străzi ș.a.m.d.).

Pentru testarea aplicației a fost implementat un layer de ***Geolocate*** care ajută la simularea mișcării. Acest layer este configurat/modificat pentru coordonatele din baza de date.

Pentru a evita zonele aglomerate, s-a implementat un layer pentru trafic. Acesta marchează străzile aglomerate, cât și ambuteiajele.

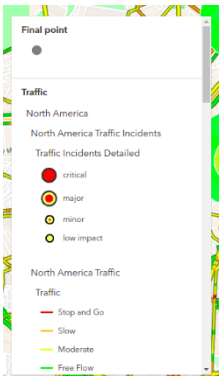
Figură 5 - Traffic layer

Unele dintre funcționalități necesită un sistem de autentificare. Pentru această situație s-a implementat un buton de login care conduce către un form al ArcGIS.

Pentru adăugarea layerelor a fost creată o funcție JavaScript numită ***displayMap***, în care se încarcă datele din baza de date, se simulează mișcarea markerului pe hartă, se creează widget-uri si layere, se declară elementele interfeței și se desenează zonele de obiective

Pentru fiecare filtrare pe baza categoriei se redesenează cercurile. Pentru citirea noilor intrări se folosește ***xhr*** și ***fileReader*** pentru parsare și se verifică tipul categoriei corespunzătoare.

Implementarea unei legende cu fiecare nume de treasure hunt ar fi făcut posibilă o rută directă pentru userii care cunosc zonele, așa că, în legendă, am scris arătat doar cum se pot citi traseele în funcție de zona geografică.



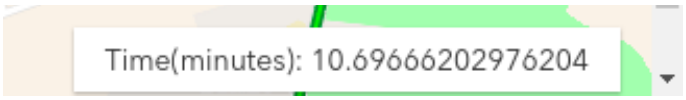
Figură 6 - Legendă

Simularea coordonatelor se bazează pe definirea unui vector de coordonate. Acestea pot fi ciclate sau se pot opri in punctul respectiv. Pentru fiecare intrare din vector a fost găsită o coordonată care sa definească o mișcare a unui utilizator.

La finalul simulării, pentru a avea o vedere în ansamblu a hărții, se face din nou override pentru scale.

Pentru fiecare update al locației pe care o pune la dispoziție layer-ul de track, se calculează un timp aproximativ până la terminarea treasure hunt-ului. Pentru a calcula acest timp, am preluat o medie de viteză cu care oamenii merg pe jos, și pentru fiecare punct de pe hartă se calculează distanța până la locația curentă.

Afișarea timpului aproximativ se face într-un ***ESRI widget***, textul fiind scris prin ***innerHTML***.



Figură 7 - Estimarea timpului până la obiectiv

Referințe:

[1] https://developers.arcgis.com/javascript/latest/api-reference/

[2] https://totalapis.github.io/api-reference/

[3] https://www.esri.com/arcgis-blog/

[4] https://doc.arcgis.com/en/

[5] https://www.esri.com/en-us/arcgis/products/arcgis-pro/resources

[6] https://gis.stackexchange.com/

[7] https://community.esri.com/

[8] https://resources.arcgis.com/en/help/

7. Documentație Ștefan

Pentru a reuni toate punctele de interes din Treasure Hunt-ul nostru a fost nevoie de o bază de date și soluția propusă a fost un fișier csv care va fi folosit apoi de ***CSVLayer***. [1]

Câmpurile pe care le conține fișierul csv sunt:

- Numele atracției turistice
- Un link de wikipedia pentru a accesa mai multe informații despre acea locație
- Longitudinea
- Latitudinea
- Ghicitoarea care duce la descoperirea obiectivului
- Un index
- Categoria din care face parte acel obiectiv: Cultural, Istoric, Natural

Target-ul userului e să parcurgă un traseu format din 5 obiective turistice. Pentru a nu avea mereu aceleași 5 obiective, se păstrează un fișiser mare (date.csv) cu toate obiectivele și pe baza lui se creează un fișier mai mic (small.csv) în care se aleg random 5 obiective.

Generarea fișierului small.csv din fișierul date.csv se face prin folosirea funcției ***XMLHttpRequest()*** cu ajutorul căruia facem un ***GET*** pentru a citi datele, care se parsează și se scriu în noul fișier care va fi descărcat ulterior. Dacă fișierul small.csv a fost deja descărcat, se va evita descărarea dublurilor printr-un alt request http.La fiecare ștergere a fișierului small.csv se va genera unul nou care va conține alte 5 obiective.

Obiectivele pot fi filtrate după categoria lor: obiectiv cultural, obiectiv natural, obiectiv istoric sau toate (fără filtrare). [2]

Figură 8 - Filter widget

Figură 9 - Filtru expandat

Butonul de filtrare a fost realizat cu ajutorul unui esri-widget, iar în momentul în care opțiunile sunt minimizate, filtrarea este reținută. Butonului i se adaugă un ***eventListener***, iar acțiunea se declanșează când utilizatorul selectează o anumită categorie. În timp ce filtrarea cercurilor concentrice s-a făcut prin ștergerea și redesenarea lor, filtrarea obiectivelor din spate afișate prin csvLayer s-a făcut pe baza unui query cu ajutorul funcției ***csv.definitionExpression.*** *[3]*

Figură 10 - Obiective nefiltrate

Figură 11 - Obiective filtrate

Referințe:

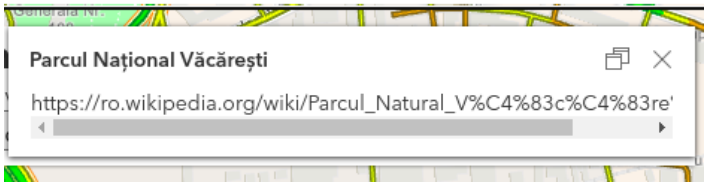
[1] https://developers.arcgis.com/javascript/latest/api-reference/esri-layers-CSVLayer.html#fields

[2] https://developers.arcgis.com/javascript/latest/sample-code/featurefilter-attributes/index.html

[3] https://developers.arcgis.com/javascript/3/jssamples/fl_layer_definition.html

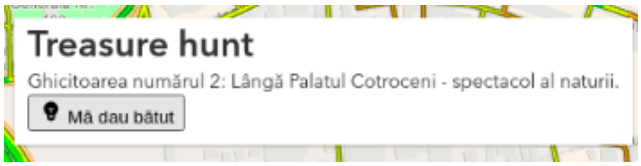
8. Documentație Georgiana:

Fiecare locație are configurat un pop-up ce conține titlul obiectivului și un link de wikipedia pentru ca utilizatorul să poată accesa mai multe informații despre acel obiectiv.



Figură 12 - Pop up obiectiv

Când utilizatorul vrea să renunțe la căutarea obiectivului sau efectiv se dă bătut, acesta poate opta pentru a i se arăta drumul către locația respectivă. Prin apăsarea butonului „Mă dau bătut" se deschide widget-ul de direcții care este implementat cu ajutorul clasei ***Directions***.

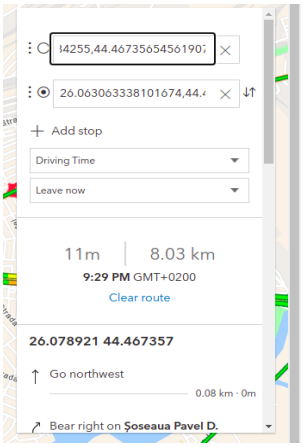


Figură 13 - Ghicitoare obiectiv

Punctul de start va fi completat automat cu locația curentă a user-ului, iar pentru punctul de final, utilizatorul va trebui să introducă coordonatele oferite ca răspuns, fără a i se divulga și numele locației.

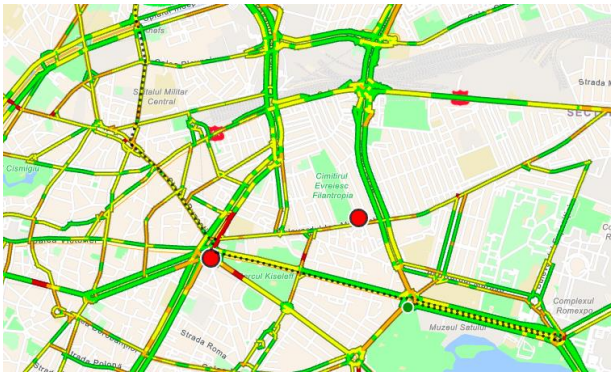
Figură 14 - Hint obiectiv

În funcție de preferință se poate afișa calcularea drumului cu mașina, pe jos etc. Pe lângă timpul estimat parcurgerii și distanța, utilizatorul primește și indicații pentru a ajunge la locația dorită. [2]



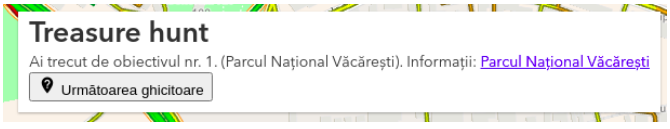
Figură 15 - Directions widget

Punctul de start, punctul de stop si punctele intermediare din afișarea rutei au fost realizare prin configurarea proprietății ***stopSymbols*** din clasa ***Directions***.



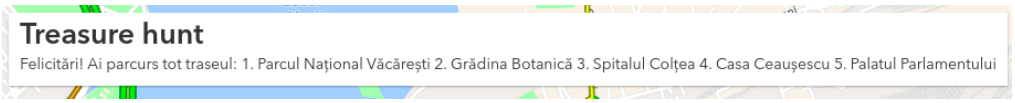
Figură 16 - Rută locație user - obiectiv

Când userul ajunge aproape de obiectivul target, mesajul din dreapta sus a ecranului se updatează, butonul „Mă dau bătut” dispare și apare butonul „Următoarea ghicitoare” care va afișa ghicitoarea următorului obiectiv din traseu.



Figură 17 - Mesaj găsire obiectiv

Atunci când toate obiectivele au fost descoperite, butoanele se dezactivează și se afișează traseul parcurs.



Figură 18 - Afișare traseu final

La pornirea aplicației stubGeoLocatorul a fost configurat să se actualizeze după niște coordonate date și când ajunge aproape de primul obiectiv să se oprească, făcându-se update cu mesajul corespunzător.

Logica butoanelor și a afișării mesajelor și ghicitorilor a fost făcută în urma parsării fișierului csv și a salvării informațiilor importante în vectori locali.

Referințe:

[1] https://developers.arcgis.com/javascript/latest/api-reference/esri-widgets-Popup.html

[2] https://developers.arcgis.com/javascript/latest/guide/driving-directions/

**** Documentațiile individuale redau distribuirea și implementarea efectivă a task-urilor fiecăruia (se pot observa anumite schimbări față de planificarea inițială sau anumite task-uri care nu au fost prevăzute înainte de începerea implementării propriu zise).**

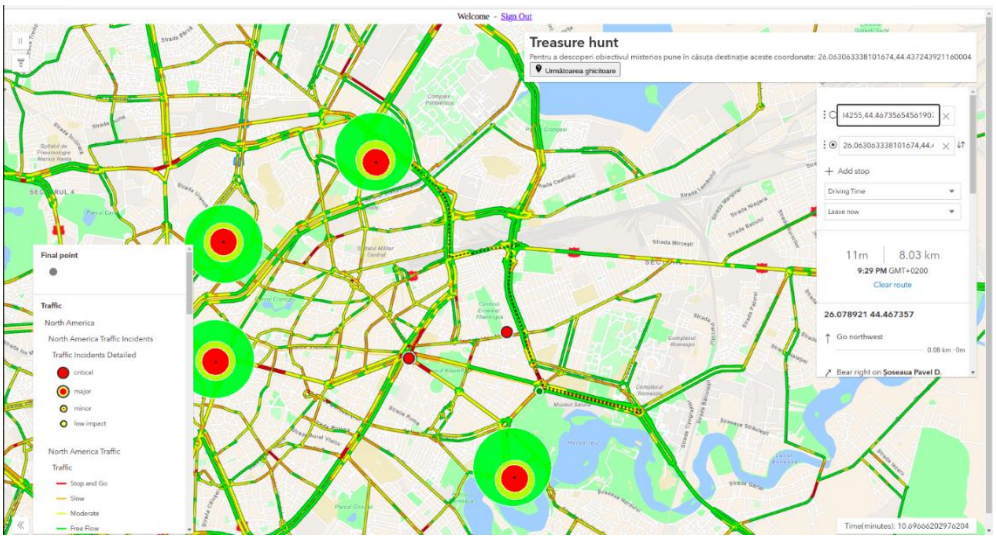
9. Concluzii și future work

Potențialul aplicației poate fi exploatat la maxim prin dezvoltare continuă. Funcționalitățile pot fi îmbunătățite și dezvoltate astfel încât să devină cât mai user-friendly. Scopul jocului își atinge punctul forte atunci când se dorește îmbinarea utilului cu plăcutul: mersul pe jos și descoperirea unor obiective ascunse, care, la rândul lor, pot fi explorate în continuare. Ca și dezvoltări viitoare s-ar putea implementa un sistem care să interconecteze toți utilizatorii și să formeze echipe pentru a atinge obiectivele. Astfel, considerăm că aplicația are un potențial care poate fi valorificat prin aducerea unor noi idei și funcționalități care să permită utilizatorilor să comunice între ei și să interacționeze eficient cu platforma.

10. Repository de GitHub

- https://github.com/CosminStoica07/Proiect_ISI/tree/master

11. Aplicația finală



Figură 19 - Aplicația finală