<Nume proiect/ acronim>

# Documentul de proiectare

Cuprins

[1. Introducere 1](#_Toc160527836)

[1.1 Scopul documentului 1](#_Toc160527837)

[2. Prezentare generală și abordări de proiectare 2](#_Toc160527838)

[2.1 Prezentare generală 2](#_Toc160527839)

[2.2 Presupuneri/ Constrângeri/ Riscuri 2](#_Toc160527840)

[2.2.1 Presupuneri 2](#_Toc160527841)

[2.2.2 Constrângeri 2](#_Toc160527842)

[2.2.3 Riscuri 3](#_Toc160527843)

[3. Considerațiii de proiectare 4](#_Toc160527844)

[3.1 Obiective și linii directoare (ghiduri) 4](#_Toc160527845)

[3.2 Metode de dezvoltare 4](#_Toc160527846)

[3.3 Strategii de arhitectură 4](#_Toc160527847)

[4. Arhitectura Sistemului și Proiectarea Arhitecturii 6](#_Toc160527848)

[4.1 Vedere logică 6](#_Toc160527849)

[4.2 Arhitectură hardware 6](#_Toc160527850)

[4.3 Arhitectură software 6](#_Toc160527851)

[4.4 Arhitectura informațiilor 7](#_Toc160527852)

[4.5 Arhitectura de comunicații interne 7](#_Toc160527853)

[4.6 Diagrama de arhitectură a sistemului 8](#_Toc160527854)

[5. Proiectarea sistemului 9](#_Toc160527855)

[5.1 Proiectarea bazei de date 9](#_Toc160527856)

[5.1.1 Obiecte de date și structuri de date rezultante 9](#_Toc160527857)

[5.1.2 Fișiere și baze de date 9](#_Toc160527858)

[5.2 Conversii de date 9](#_Toc160527859)

[5.3 Interfețe utilizator 10](#_Toc160527860)

[5.3.1 Intrări 10](#_Toc160527861)

[5.3.2 Ieșiri 10](#_Toc160527862)

[5.4 Proiectarea interfețelor cu utilizatorul 10](#_Toc160527863)

[6. Scenarii de utilizare 11](#_Toc160527864)

[7. Proiectare de detaliu 12](#_Toc160527865)

[7.1 Proiectare hardware de detaliu 12](#_Toc160527866)

[7.2 Proiectare software de deatliu 12](#_Toc160527867)

[7.3 Proiectare detaliată de securitate 13](#_Toc160527868)

[7.4 Proiectare de detaliu pentru performanța sistemului 14](#_Toc160527869)

[7.5 Proiectare detaliată a comunicațiilor interne (între componente) 14](#_Toc160527870)

[8. Controale pentru verificarea integrității sistemului 15](#_Toc160527871)

[Anexa A: Gestiunea modificărilor documentului 16](#_Toc160527872)

[Anexa B: Acronime 17](#_Toc160527873)

[Anexa C Documente la care se face referire 18](#_Toc160527874)

## Introducere

Instrucțiuni: Furnizați informații de identificare pentru sistemul existent și/sau propus sau situația pentru care se aplică Documentul de Proiectare a Sistemului (de exemplu, numele complet și acronimele proiectului de dezvoltare, sistemul sau situația existentă și propusă, după caz), și evoluția așteptată a documentului. De asemenea, descrieți orice considerații referitor la securitate sau confidențialitate asociate cu utilizarea acestui document..

Documentul de Proiectare a Sistemului descrie modul în care cerințele funcționale și non-funcționale înregistrate în Documentul de Cerințe se transformă în specificații de proiectare a sistemului, mai tehnice, pe baza cărora se construiește sistemul. Se documentează proiectarea sistemului la un nivel înalt și specificațiile detaliate de proiectare.

Se descriu obiectivele și considerațiile de proiectare, se furnizează o prezentare generală la nivel înalt a arhitecturii sistemului și se descrie proiectarea datelor asociate cu sistemul, precum și interfața om-mașină și scenariile operaționale.

Proiectarea sistemului la nivel înalt este descompusă în continuare în specificații detaliate de proiectare pentru fiecare componentă a sistemului, inclusiv hardware, comunicațiile interne, software, controalele de integritate a sistemului și interfețele externe.

### Scopul documentului

*Instrucțiuni: Furnizați scopul documentului. Acest document ar trebui să fie adaptat pentru a se potrivi nevoilor unui proiect specific.*

Prin acest livrabil se documentează și urmăresc informațiile necesare pentru a defini eficient arhitectura și designul sistemului, în scopul de a oferi echipei de dezvoltare îndrumare asupra arhitecturii sistemului ce urmează să fie dezvoltat.

Documentele de proiectare sunt produse incremental și iterativ pe parcursul ciclului de viață al dezvoltării sistemului, în funcție de circumstanțele particulare ale proiectului de tehnologie informațională (IT) și de metodologia de dezvoltare a sistemului utilizată.

Publicul țintă este managerul de proiect, echipa de proiect și echipa de dezvoltare. Anumite părți ale acestui document, cum ar fi interfața cu utilizatorul (UI), pot fi împărtășite cu clientul/utilizatorul și cu alte părți interesate ale căror contribuții/aprobări sunt necesare în UI.

## Prezentare generală și abordări de proiectare

Această secțiune descrie principiile și strategiile care vor fi utilizate ca ghiduri în momentul proiectării și implementării sistemului.

### Prezentare generală

Instrucțiuni: Prezentați succint contextul sistemului și abordarea sau organizarea de bază a proiectării. Oferiți o prezentare generală a arhitecturilor sistemului și software-ului și a obiectivelor de proiectare.

### Presupuneri/ Constrângeri/ Riscuri

#### Presupuneri

Instrucțiuni: Descrieți orice presupuneri sau dependențe legate de sistem, software și utilizarea sa. Acestea pot viza probleme precum: software sau hardware asociat, sisteme de operare, caracteristicile utilizatorilor finali și posibilele și/sau probabilele modificări ale funcționalității.

#### Constrângeri

Instrucțiuni: Descrieți orice limitări sau constrângeri globale care au un impact semnificativ asupra designului hardware-ului, software-ului și/sau comunicațiilor **s**istemului și descrieți impactul asociat. Astfel de constrângeri pot fi impuse de oricare dintre următoarele (lista nu este exhaustivă):

* Mediu hardware sau software
* Mediu utilizator final
* Disponibilitatea sau volatilitatea resurselor
* Conformitatea cu standardele
* Cerințe de interoperabilitate
* Cerințe de interfață/protocol
* Cerințe de licențiere
* Cerințe pentru depozitarea și distribuția datelor
* Cerințe de securitate (sau alte reglementări similare)
* Limitări ale capacității de memorie sau ale altor resurse
* Cerințe de performanță
* Comunicații de rețea
* Cerințe de verificare și validare (testare)
* Alte mijloace de a aborda obiectivele de calitate
* Alte cerințe descrise în Documentul de Cerințe

#### Riscuri

Instrucțiuni: Descrieți orice riscuri asociate cu designul sistemului și strategiile propuse de reducere a acestora.

## Considerații de proiectare

Instrucțiuni: Descrieți problemele care trebuie abordate sau rezolvate înainte de a încerca să elaborați o soluție de design completă.

### Obiective și linii directoare (ghiduri)

Instrucțiuni: Descrieți orice obiective, linii directoare, principii sau priorități care domină sau încorporează designul sistemului și al software-ului. Exemple de astfel de obiective ar putea fi: accentul pe viteză versus utilizarea memoriei; sau utilizare, aspect sau "să se simtă" similar cu un produs existent. Linii directoarele includ ghiduri și convenții de codare. Pentru fiecare ghid, descrieți motivul, cu excepția cazului în care este implicit evident. Descrieți orice politici și/sau strategii de design care nu au implicații arhitecturale majore (adică nu afectează semnificativ organizarea generală a sistemului și structurile sale de nivel înalt), dar care totuși afectează detaliile interfeței și/sau implementarea diferitelor aspecte ale sistemului (de exemplu, alegerea unui produs specific de utilizat).

### Metode de dezvoltare

Instrucțiuni: Descrieți succint metoda sau abordarea folosită pentru designul sistemului și al software-ului (de exemplu, structurat, orientat pe obiecte, prototipare, J2EE, UML, XML, etc.). Dacă una sau mai multe metode formale/publicate au fost adoptate sau adaptate, includeți o referință către o descriere mai detaliată a acestor metode. Dacă au fost luate în considerare mai multe metode serios, menționați fiecare astfel de metodă, împreună cu o explicație succintă a motivului pentru care întreaga metodă sau o parte din ea a fost utilizată sau nu. Descrieți orice contingente care ar putea apărea în designul sistemului și al software-ului care ar putea schimba direcția de dezvoltare. Posibilitățile includ lipsa acordurilor de interfață cu agențiile externe sau arhitecturile instabile în momentul în care documentul este pregătit. Abordați orice soluții de rezervă posibile sau planuri alternative.

### Strategii de arhitectură

Instrucțiuni: Descrieți orice decizii și/sau strategii de design care afectează organizarea generală a sistemului și structurile sale de nivel superior. Aceste strategii ar trebui să ofere o înțelegere a principalelor abstractizări și mecanisme utilizate în arhitectura sistemului. Descrieți raționamentul utilizat pentru fiecare decizie și/sau strategie (posibil referindu-vă la obiectivele și principiile de design anterior menționate) și modul în care au fost echilibrate sau compensat obiectivele sau prioritățile de designAtunci când descrieți o decizie de design, discutați orice alte alternative semnificative care au fost luate în considerare și motivele pentru respingerea lor (precum și motivele pentru acceptarea alternativei în cele din urmă alese).

Exemple de decizii de design ar putea include (dar nu sunt limitate la) următoarele:

* Utilizarea unui anumit tip de produs (limbaj de programare, bază de date, bibliotecă, produs comercial disponibil pe piață etc.)
* Reutilizarea componentelor software existente pentru implementarea diferitelor părți/caracteristici ale sistemului
* Planuri viitoare pentru extinderea sau îmbunătățirea software-ului
* Paradigmele interfeței utilizatorului (sau modelele de intrare și ieșire ale sistemului)
* Paradigmele interfeței hardware și/sau software
* Detectarea și recuperarea erorilor
* Politicile de gestionare a memoriei
* Bazele de date externe și/sau gestionarea și persistența datelor
* Date distribuite sau control asupra unei rețele
* Abordări generalizate pentru control
* Concurgență și sincronizare
* Mecanisme de comunicare
* Managementul altor resurse

## Arhitectura Sistemului și Proiectarea Arhitecturii

Simulatorul de trafic este organizat într-o arhitectură modulară, component-based, tipică aplicațiilor dezvoltate în Unity, unde fiecare funcționalitate principală este separată într-un subsistem. Scopul acestei organizări este de a asigura scalabilitate, reutilizare, mentenanță ușoară și o separare clară a responsabilităților. Aplicația nu interacționează cu aplicații externe în mod direct, dar se bazează pe pachete Unity, precum Splines Package pentru generarea strazilor, Unity NavMesh navigația pentru vehicule simple, Odin Inspector pentru creare rapidă de UI în inspector pentru tool-uri custom.

Sistemul este împărțit în mai multe componente, astfel avem acest tabel cu fiecare sistem principal și rolul acestuia în proiect :

| **Componentă** | **Responsabilitate** |
| --- | --- |
| RoadSystemManager | Gestionarea generării străzilor și intersecțiilor folosind splines. |
| IntersectionManager | Detectarea și gestionarea intersecțiilor. Crearea de semafoare și reguli asociate. |
| TrafficSignalController | Controlul comportamentului semafoarelor (timer sau inteligent). |
| TrafficSpawner | Generarea vehiculelor pe baza punctelor de start/stop. |
| VehicleController | Controlul individual al vehiculelor în simulare. |
| UIManager | Gestionarea interfeței grafice și feedback-ului către utilizator. |
| SimulationManager | Coordonarea pornirii/opriri simularii și validarea tuturor condițiilor. |

Această decompunere permite:

* Separarea logicii de simulare (trafic, reguli) de partea grafică (UI, input).
* Testabilitate crescută pe componente izolate.
* Extensibilitate (ex: dacă vrei să adaugi un nou tip de semafor sau de vehicul, o poți face ușor fără să afectezi alte componente).
* Control clar al dependențelor și o arhitectură ușor de întreținut.

### Vedere logică

Organizarea sistemului este următoarea:

* UIManager – se ocupă de interfața cu utilizatorul, validări și afișarea erorilor.
* RoadBuilder – permite crearea de drumuri cu ajutorul spline-urilor și generarea de mesh-uri.
* IntersectionManager – detectează și generează intersecțiile pe baza punctelor comune dintre drumuri.
* TrafficSignalController – configurează semafoarele pe intersecții și le sincronizează.
* SimulationManager – controlează logica generală a simulării (pornire, oprire, flux vehicule).
* VehicleSpawner – instanțiază vehiculele și le trimite spre destinație conform punctelor Start/Stop.

### Arhitectură hardware

Simulatorul este proiectat să ruleze local, pe un singur sistem de procesare (centralizat). Acesta este în general un calculator. Tot fluxul de procesare, afișare grafică și logică de simulare se desfășoară în aceeași aplicație Unity rulată pe dispozitivul utilizatorului.

### Arhitectură software

Simulatorul de trafic urban este dezvoltat în motorul Unity, folosind limbajul de programare C#. Arhitectura software este împărțită în mai multe straturi și componente logice pentru a asigura modularitate, mentenabilitate și extensibilitate.

Structura pe straturi:

* Stratul de prezentare (UI)
* Stratul logicii aplicației
* Stratul de date interne (scenarii, salvări)
* Componente de AI și rutare
* Stratul de simulare fizică și grafică

Componente software utilizate :

| **Componentă** | **Descriere** | **Versiune** | **Licență / Producător** |
| --- | --- | --- | --- |
| **Unity Engine** | Motorul de jocuri care gestionează grafica, fizica, scriptingul | 2022.3 LTS | Unity Technologies, licență personal / educațională |
| **C#** | Limbajul de programare principal pentru logica jocului și UI | .NET Standard | Microsoft |
| **TextMesh Pro** | Afișare text UI vectorial de înaltă calitate | inclus | Unity |
| **Cinemachine** | Controlul camerei și tranziții dinamice în timpul simulării | inclus | Unity |
| **NavMesh** | Sisteme de navigație și pathfinding pentru vehicule | Unity built-in | Unity |
| **Unity Input System** | Gestionarea input-urilor (mouse, tastatură, gamepad) | 1.5+ | Unity |
| **URP** | Universal Render Pipeline – randare performantă, optimizată | 12+ | Unity |
| **Visual Studio** | IDE pentru scrierea și depanarea codului C# | 2022 | Microsoft |
| **SourceTree / Git** | Gestionare versionare sursă și colaborare | Git | GitHub |

### Arhitectura informațiilor

În cadrul simulării de trafic, informațiile stocate și gestionate de sistem vor include mai multe tipuri de date care sunt esențiale pentru funcționarea corectă a aplicației. Aceste informații vor fi utilizate pentru a crea și gestiona simularea, inclusiv pentru intersecții, drumuri, semafoare și vehicule.

Date despre drumuri și intersecții:

* Descriere: Aceste date includ informațiile despre drumurile generate de utilizator (spre exemplu, tipul de drum, numărul de benzi, curburile, direcțiile de trafic) și intersecțiile dintre acestea.
* Formatul datelor: Datele vor fi stocate în format electronic (în Unity, într-o bază de date internă sau fișiere JSON/XML) care permite gestionarea dinamică a acestora. Acestea pot include coordonate ale nodurilor care definesc drumurile, lățimea benzilor, limitele de viteză și tipul de intersecție.
* Furnizori de date: Utilizatorul va furniza aceste date prin interfața aplicației, folosind funcționalitățile de creare a drumurilor și intersecțiilor.

Date despre semafoare:

* Descriere: Datele referitoare la semafoare includ tipul semaforului (clasic sau inteligent), duratele semaforului, modul de operare (pe timer sau pe bază de flux de trafic).
* Formatul datelor: Aceste informații vor fi stocate tot în format electronic (JSON/XML) și vor fi asociate fiecărei intersecții create de utilizator. Datele se vor salva și vor putea fi accesate pentru simularea semnalizării.
* Furnizori de date: Utilizatorul va furniza aceste date prin alegerea tipului de semafor și configurarea acestuia pentru fiecare intersecție.

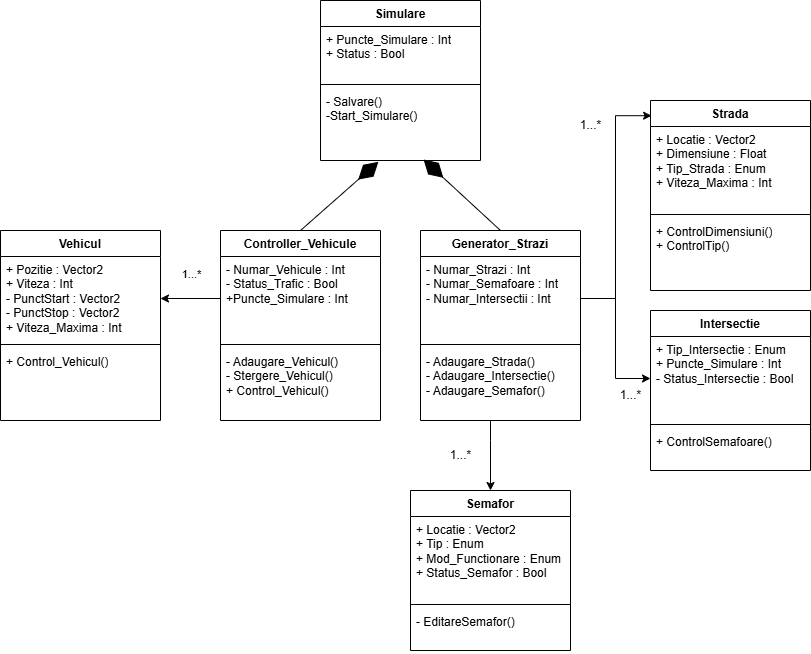
Date despre vehicule și fluxul de trafic:

* Descriere: Informațiile despre vehicule includ datele vehiculului (tipul, viteza, direcția) și configurațiile punctelor de start/stop (pentru a defini de unde pleacă vehiculele și unde ajung).
* Formatul datelor: Datele vor fi stocate în format electronic (JSON, fișiere de configurare sau variabile Unity) care includ coordonatele de start și stop, precum și detalii despre numărul de vehicule generate pe minut și ruta acestora.
* Furnizori de date: Utilizatorul furnizează aceste date atunci când definește punctele de start/stop și numărul de vehicule pe minut.

### Arhitectura de comunicații interne

Deoarece este o aplicatie standalone, toate componentele comunică între ele într-o rețea internă simplă, fără a necesita infrastructură externă. Comunicarea între UI și motorul de simulare se face prin apeluri funcționale directe în C#, iar componentele fizice ale simulării (vehicule, semafoare) sunt actualizate pe baza logicii din motorul de simulare. Aplicația se va baza pe resursele interne ale calculatorului (CPU, RAM și GPU), fără a depinde de un server sau o bază de date externă.

### Diagrama de arhitectură a sistemului



## Proiectarea sistemului

### Proiectarea bazei de date

Deoarece aplicația este standalone și nu utilizează un DBMS clasic, datele vor fi gestionate intern, în memorie, folosind structuri de date în C#. Aceste structuri de date vor stoca informațiile necesare simulării, fără a exista o bază de date externă. Datele vor fi stocate în fisiere de tip json.

#### Obiecte de date și structuri de date rezultante

Structuri de date principale:

* Stradă: Reprezintă drumuri sau străzi. Fiecare obiect va conține numele, numărul de benzi și limita de viteză. Este un obiect C# cu validări specifice pentru unicitatea numelui.
* Intersecție: Reprezintă intersecțiile între două sau mai multe străzi. Conține informații despre străzile implicate și semafoarele asociate.
* Semafor: Gestionarea semafoarelor pentru fiecare intersecție. Poate avea stări de schimbare (roșu, galben, verde).
* Vehicul: Reprezintă vehiculele care circulă în simulare, cu caracteristici precum viteza, poziția și direcția.

#### Fișiere și baze de date

Fișierele interne vor fi gestionate de aplicație sub formă de obiecte C# care vor fi stocate în memorie sub forma de json.

##### Baze de date

Nu există baze de date tradiționale DBMS, deoarece aplicația funcționează complet în memoria internă a sistemului. Datele sunt gestionate în structuri de date C#.

##### Fișiere non-DBMS

Fișierele non-DBMS vor fi folosite pentru a salva starea simulării între sesiuni, de exemplu:

* Fișiere de configurare: Vor conține setările utilizatorului pentru strada, intersecțiile și semafoarele. Acestea vor fi fișiere de tip text (XML sau JSON) și vor fi citite la inițializarea aplicației pentru a restaura setările salvate anterior.
* Fișiere temporare: Fișierele temporare vor fi utilizate pentru a stoca date de simulare care sunt procesate și șterse după terminarea simulării.

### Conversii de date

Datele sunt stocate in fisiere de tip json in acelasi folder cu programul, vor exista decat conversii de la JSON la obiecte C# sau invers.

### Interfețe utilizator

Aplicația va avea o interfață grafică (GUI) în Unity, cu următoarele clase de utilizatori:

* Operatorul de simulare: Va interacționa cu aplicația pentru a configura drumul, intersecțiile și semafoarele, precum și pentru a vizualiza simularea.
* Utilizatorul final: Poate fi un operator care doar vizualizează simularea și poate modifica parametrii de bază (de exemplu, viteza maximă pe stradă).

#### Intrări

Utilizatorii vor introduce date prin intermediul unui GUI:

* Ecrane de introducere a datelor: Utilizatorul va introduce informații pentru drumuri, intersecții și semafoare. Valori ca numele străzii, numărul de benzi și limitele de viteză vor fi introduse manual.
* Tipuri de înregistrări de intrare: Datele vor fi validate în timp real (de exemplu, numele străzii trebuie să fie unic).

#### Ieșiri

Ieșirile sistemului vor include:

* Rapoarte vizuale: Afisarea în timp real a simulării cu vehicule care se deplasează pe străzi și intersecții. Se vor include semafoarele și viteza vehiculelor.
* Raport de eroare: Dacă există conflicte în configurare (de exemplu, două străzi cu același nume), se va genera un mesaj de eroare.

### Proiectarea interfețelor cu utilizatorul

Interfața utilizatorului va fi creată cu Unity, oferind un GUI interactiv:

* Ecran principal: Vizualizarea drumurilor și intersecțiilor, cu butoane pentru a modifica parametrii.
* Ecran de simulare: Afișează fluxul de trafic și starea semafoarelor. Utilizatorii pot începe sau opri simularea, ajustând parametrii pe parcurs.

Aceste interfețe vor fi interactive și ușor de utilizat, cu butoane clare și mesaje de eroare pentru a ghida utilizatorul.

## Scenarii de utilizare

### Scenariul 1: Crearea rețelei de drumuri

**Descriere:**  
 Utilizatorul configurează infrastructura rutieră pentru simulare.

**Flux de evenimente:**

Utilizatorul deschide aplicația și accesează modul "Creare drumuri".

Selectează instrumentul de desenare a drumurilor bazat pe splines.

Trasează manual străzile pe hartă, specificând punctele cheie și curbura drumului.

Setează proprietăți pentru fiecare drum: nume unic, număr de benzi, limită de viteză.

Salvează rețeaua de drumuri într-o configurație internă.

**Erori posibile:**

Nume de drum duplicat → sistemul notifică utilizatorul să introducă un nume unic.

Puncte de spline inconsecvente → sistemul solicită ajustarea drumului.

### Scenariul 2: Definirea și configurarea intersecțiilor

**Descriere:**  
 După crearea drumurilor, utilizatorul configurează intersecțiile pentru a modela punctele de întâlnire.

**Flux de evenimente:**

Sistemul detectează automat zonele de intersecție bazate pe drumurile desenate.

Utilizatorul selectează o intersecție din listă sau direct pe hartă.

Alege tipul intersecției: cu semafor sau fără semafor.

În cazul alegerii "cu semafor":

Configurează semnalele luminoase pentru fiecare direcție.

Specifică durata fiecărei stări (roșu/verde/galben) sau optează pentru control inteligent (bazat pe flux).

**Erori posibile:**

Drumuri lipsă într-o intersecție → sistemul alertează utilizatorul să corecteze geometria.

### Scenariul 3: Configurarea fluxului de trafic

**Descriere:**  
 Utilizatorul definește punctele de intrare și ieșire pentru vehicule și stabilește parametrii de generare.

**Flux de evenimente:**

Accesează modul "Configurare trafic".

Selectează un punct de pornire (start) pe unul dintre drumuri.

Selectează un punct de destinație (stop) pentru vehicule.

Specifică:

Tipul vehiculului (mașină mică, camion, autobuz).

Numărul de vehicule generate pe minut.

Intervalul de generare și viteza medie.

Confirmă configurarea și salvează scenariul.

**Erori posibile:**

Destinații invalide (nu există drum continuu între start și stop) → sistemul semnalează eroarea și sugerează corecturi.

### Scenariul 4: Inițierea și rularea simulării

**Descriere:**  
 Utilizatorul începe simularea traficului bazată pe configurațiile create.

**Flux de evenimente:**

Accesează modul "Simulare".

Apasă butonul "Start simulare".

Sistemul începe generarea vehiculelor conform scenariului definit.

Vehiculele navighează pe străzi, respectând regulile semafoarelor și limitele de viteză.

Utilizatorul poate:

Vizualiza simularea în timp real (perspectivă de sus sau urmărire vehicul).

Ajusta manual durata semafoarelor.

Opri sau relua simularea.

**Erori posibile:**

Coliziuni nerezolvate sau erori de rutare → sistemul alertează și oferă soluții automate de corectare.

### Scenariul 5: Ajustări dinamice în timpul simulării

**Descriere:**  
 Utilizatorul poate modifica în timp real parametrii simulării pentru a testa scenarii alternative.

**Flux de evenimente:**

În timpul rulării, utilizatorul deschide panoul de control al semafoarelor.

Modifică durata de semnalizare verde/roșu pentru o anumită intersecție.

Apasă "Aplică modificări" → noile setări sunt implementate instantaneu.

Monitorizează impactul asupra fluxului de trafic.

**Erori posibile:**

Configurații contradictorii între semafoare → sistemul previne salvarea și indică necesitatea revizuirii.

### Scenariul 6: Salvarea și încărcarea simulării

**Descriere:**  
 Permite utilizatorului să salveze progresul pentru sesiuni viitoare.

**Flux de evenimente:**

Utilizatorul apasă butonul "Salvează simulare".

Sistemul serializează toate datele (drumuri, intersecții, semafoare, flux vehicule) într-un fișier JSON.

La o sesiune ulterioară, utilizatorul apasă "Încarcă simulare".

Alege un fișier JSON existent → sistemul reconstituie întreaga configurație.

**Erori posibile:**

Fișier JSON corupt sau incompatibil → sistemul anunță imposibilitatea încărcării și propune utilizatorului să reîncerce sau să încarce alt fișier.

### Scenariul 7: Gestionarea erorilor și alerte

**Descriere:**  
 Sistemul oferă feedback utilizatorilor pentru a preveni erori de configurare.

**Flux de evenimente:**

La fiecare acțiune de configurare (creare drum, adăugare semafor), sistemul validează automat datele introduse.

În caz de erori, aplicația afișează un mesaj descriptiv și evidențiază problema pe hartă sau în listă.

Utilizatorul corectează erorile și poate continua configurarea fără restartarea aplicației.

## Proiectare de detaliu

### Proiectare hardware de detaliu

**Calculator necesar**:

CPU: Quad-core (minim i5 10th gen sau echivalent AMD Ryzen).

RAM: 8 GB (recomandat 16 GB).

GPU: Compatibil DirectX 11 sau 12, minim GTX 1650.

**Dispozitive suplimentare**: Monitor standard, mouse și tastatură.

**Arhitectură**: Toate componentele software rulează local.

### · Proiectare software de detaliu

**Servicii principale**:

RoadSystemManager: gestionează generarea străzilor.

IntersectionManager: gestionează logica de creare a intersecțiilor.

TrafficSignalController: controlează logica semafoarelor (timer sau flux dinamic).

TrafficSpawner: generează vehicule.

VehicleController: gestionează deplasarea vehiculelor.

UIManager: gestionează toate interfețele cu utilizatorul.

SimulationManager: gestionează starea generală a simulării.

**Limbaj**: C# (standard .NET utilizat de Unity).

**Framework**: Unity 2022.3 LTS + extensii Splines Package, NavMesh, Cinemachine.

**Persistență date**: Salvare/încărcare date în fișiere JSON.

### Proiectare detaliată de securitate

Aplicația fiind standalone, nu există transfer de date prin rețea, însă:

Salvarea fișierelor JSON va fi validată pentru a evita coruperea datelor.

Structurile interne vor avea validări de tip (ex. verificarea numelor unice pentru drumuri).

### Proiectare de detaliu pentru performanța sistemului

**Estimări de volum**: Suport pentru rețele de 50-100 de drumuri și câteva sute de vehicule simultan.

**Cerințe de disponibilitate**: Minimizarea crash-urilor Unity prin validări riguroase la intrări utilizator.

**Back-up**: Posibilitatea de salvare manuală sau automată a simulărilor în JSON.

### Proiectare detaliată a comunicațiilor interne (între componente)

**Comunicare inter-componente**:

Directă în memorie (apeluri C#).

Managerii componentelor sunt singletoni sau referințe injectate.

**Format date schimbate**:

Structuri interne C# pentru drumuri, intersecții, semafoare, vehicule.

Pentru salvare: JSON.

**Topologie**: Centralizată (toate componentele într-o singură aplicație locală).

## Controale pentru verificarea integrității sistemului

Instrucțiuni: Furnizați specificații de proiectare pentru următoarele nivele minime de control și orice controale suplimentare adecvate sau necesare:

* Securitate internă pentru a restricționa accesul la datele critice doar pentru acele tipuri de acces necesare de către utilizatori/operatori
* Proceduri de audit pentru a îndeplini cerințele de control, raportare și perioade de reținere pentru rapoartele operaționale și de management
* Piste de auditare a aplicațiilor pentru a audita dinamic accesul la recuperare la datele critice designate
* Tabele standard care urmează să fie utilizate sau solicitate pentru validarea câmpurilor de date
* Procese de verificare pentru adăugarea, ștergerea sau actualizarea datelor critice
* Capacitatea de a identifica toate informațiile de auditare prin identificarea utilizatorului, identificarea terminalului de rețea, dată, oră și datele accesate sau modificate.

Anexa A: Gestiunea modificărilor documentului

Instrucțiuni: Furnizați informații despre modul în care dezvoltarea și distribuția documentului va fi controlată și urmărită. Utilizați tabelul de mai jos pentru a furniza numărul de versiune, data versiunii, autorul/deținătorul versiunii și o scurtă descriere a motivului pentru crearea versiunii revizuite.

Tabel 1 – Înregistrarea modificărilor asupreaa documentului curent

| versiune | Data | Autorul/Deținătorul | Descriere |
| --- | --- | --- | --- |
| <X.X> | <ZZ/LL/AAAA> | <nume autor> | <Descrierea modificării> |
| <X.X> | <ZZ/LL/AAAA> | <nume autor> | <Descrierea modificării> |
| <X.X> | <ZZ/LL/AAAA> | <nume autor> | <Descrierea modificării> |

Anexa B: Acronime

*Instrucțiuni: Furnizați o listă de acronime și traduceri literale asociate utilizate în cadrul documentului. Enumerați acronimele în ordine alfabetică folosind un format tabular, așa cum este ilustrat mai jos.*

Tabel 2 - Acronime

| Acronim | Forma completă |
| --- | --- |
| <Acronim> | <Forma completă> |
| <Acronim> | <Forma completă> |
| <Acronim> | <Forma completă> |

Anexa C Documente la care se face referire

*Instrucțiuni: Sintetizați relația acestui document cu alte documente relevante. Furnizați informații de identificare pentru toate documentele folosite pentru a ajunge la și/sau referite în acest document (de exemplu, documente conexe și/sau asociate, documente prealabile, documentație tehnică relevantă, etc.).*

Tabel 3 – Documente la care se facce referire

| Nume document | Locație sau URL | Dată emitere document |
| --- | --- | --- |
| < Nume document > | <Locație sau URL> | <ZZ/LL/AAAA> |
| < Nume document > | <Locație sau URL> | <ZZ/LL/AAAA> |
| < Nume document > | <Locație sau URL> | <ZZ/LL/AAAA> |