

# POLITECNICO DI TORINO

Corso di Laurea in Ingegneria Gestionale  
Classe L8 - Ingegneria dell'Informazione



**Simulazione National Gallery Museum di Washington D.C.**

**Relatore**  
Prof. Fulvio Corno

**Candidato**  
D'Anna Filippo Maria Simone  
275791

A.A. 2021/2022

## Sommario

1	Proposta di progetto.....	1
1.1	Studente proponente .....	1
1.2	Titolo della proposta.....	1
1.3	Descrizione del problema proposto .....	1
1.4	Descrizione della rilevanza gestionale del problema.....	2
1.5	Descrizione dei data-set per la valutazione .....	2
1.6	Descrizione preliminare degli algoritmi coinvolti.....	2
1.7	Descrizione preliminare delle funzionalità previste per l'applicazione software.....	2
2	Descrizione del problema affrontato.....	4
3	Descrizione del data-set utilizzato per l'analisi.....	5
3.1	Introduzione.....	5
3.2	Tabella Locations .....	5
3.3	Tabella Objects.....	6
4	Descrizione ad alto livello delle strutture dati e degli algoritmi utilizzati .....	8
4.1	Descrizione delle strutture dati: i packages e le classi.....	8
4.1.1	it.tdp.polito.NationalGallery .....	8
4.1.2	it.tdp.polito.NationalGallery.DAO .....	8
4.1.3	it.tdp.polito.NationalGallery.model.....	9
4.2	Descrizione degli algoritmi utilizzati .....	10
4.2.1	Sezione 1: Ricerca .....	10
4.2.2	Sezione 2: Simulazione .....	12
4.2.3	Caso ENTRATA_GRUPPPPO.....	15
4.2.4	Caso CAMBIO_STANZA .....	18
4.2.5	Caso USCITA_GRUPPO .....	19
4.2.6	Stampa delle statistiche .....	20
5	Interfaccia e video dimostrativo.....	21
5.1	Sezione Ricerca .....	21
5.2	Sezione Simulazione.....	22
5.3	Mappa.....	23
6	Risultati sperimentali e esempi .....	24
6.1	Esempio 1 .....	24
6.2	Esempio 2 .....	26
6.3	Esempio 3 .....	28
7	Valutazioni sui risultati ottenuti.....	32



# **1 Proposta di progetto**

## **1.1 Studente proponente**

s275791 D'Anna Filippo Maria Simone

## **1.2 Titolo della proposta**

Simulazione National Gallery of Art di Washington D.C.

## **1.3 Descrizione del problema proposto**

Il software proposto sarà in grado di selezionare un'opera presente nel museo, in base al periodo e all'autore selezionati e di simulare delle visite all'interno del museo, in funzione delle condizioni selezionate dall'utente che usufruirà dell'applicazione.

Quest'ultima si appoggia a un database che permetterà di ottenere le informazioni di tutte le opere presenti nel museo e di creare degli oggetti stanza con.

I clienti potranno scegliere di filtrare la ricerca in funzione di :

- Autore
- Periodo Storico
- Opere prodotte

Alle visite della simulazione prenderanno parte diverse tipologie di gruppi di visitatori, questi ultimi potranno essere bambini, adulti o anziani.

Sarà possibile osservare l'andamento giornaliero del museo e l'occupazione delle singole stanze nei vari momenti della giornata, valutando se sarà necessario aumentare la sicurezza in determinati settori e se la capienza delle varie stanze verrà rispettata non creando malcontento nei clienti (o altri possibili controlli).

## **1.4 Descrizione della rilevanza gestionale del problema**

Utilizzando questo software il museo potrà analizzare scenari possibili e verosimili che fungano da modello semi-reale per l'analisi di situazioni future e da strumento sperimentale adatto a prendere decisioni di gestione.

Inoltre il software può essere applicato a qualsiasi museo che sia in grado di fornire un database adatto e la sezione di simulazione può essere ampliata con ulteriori controlli a seconda dei risultati d'interesse (prezzi dei singoli tour, guide disponibili, promozioni etc.).

## **1.5 Descrizione dei data-set per la valutazione**

Il programma utilizza il database fornito dal NGoA che contiene 130.000 + opere, nonché informazioni sull'artista, sul periodo e la loro posizione nel museo (relativa alle due costruzioni principali : ala est e ala ovest, alla stanza e alla posizione all'interno della suddetta).

<https://github.com/NationalGalleryOfArt/opendata>

## **1.6 Descrizione preliminare degli algoritmi coinvolti**

Da un punto di vista algoritmico la prima parte del software riguarda una semplice interrogazione del database, selezionando dei parametri come l'anno di inizio di produzione dell'opera, l'anno di fine e l'autore.

La seconda parte svolgerà invece una simulazione che tiene conto delle visite dei clienti e del numero di persone in ciascun gruppo al fine di valutare l'affluenza del museo e varie criticità: mancanza di audio guide, criticità nella capacità delle singole stanze , malcontento dei visitatori, recensioni positive e negative.

## **1.7 Descrizione preliminare delle funzionalità previste per l'applicazione software**

Il software sarà utilizzato dai dipendenti del museo che potranno consultare il database tramite l'interfaccia e potranno svolgere le simulazioni inserendo personalmente i parametri

Nella prima sezione all'utente sarà chiesto di inserire determinati filtri per cercare l'opera interessata (potenzialmente potrebbe essere utilizzata anche dai clienti del museo):

- Periodo Storico
- Autore
- Opere dell'autore

Elementi che saranno importati sul controller tramite database e saranno disponibili all'interno di combo box, che saranno riempite tramite la pressione dei bottoni 'Select Period' e 'Select Artist'. Infine alla pressione del bottone 'Get Info' il programma restituirà le informazioni riguardante l'opera scelta che riguardano:

- Titolo dell'opera
- Autore
- Periodo
- Descrizione dell'opera
- Dimensioni dell'opera
- Stanza in cui l'opera si trova al momento.

Per quanto riguarda la parte di simulazione, l'utente potrà selezionare la durata (in minuti) minima e massima di una visita, la permanenza (in minuti) minima e massima di un gruppo in una stanza, il numero massimo di visitatori per quel giorno, il numero di audioguide a disposizione e la capienza massima delle stanze. Inoltre dovrà essere selezionato almeno un periodo artistico pubblicizzato dal museo e che spingerà i visitatori a recarsi nelle stanze di quel periodo, con una probabilità selezionabile dall'utente tramite interfaccia. Sarà anche possibile scegliere con quale probabilità un gruppo aumenterà il proprio grado di insoddisfazione (fattore che va da 0 a 5).

Al termine della simulazione apparirà a schermo una lista di statistiche:

Il numero totale di gruppi e il numero totale di ciascuna tipologia di gruppo, il numero totale di visitatori e del totale di ogni fascia di età, i visitatori senza guida, le recensioni positive e negative e l'insoddisfazione media. Nel caso di eccessiva affluenza i clienti potrebbero mostrare malcontento sulla base di un parametro generato scelto dall'utente tra 0.1 e 1.0 che indica la probabilità di essere insoddisfatto entrando in una stanza piena. Se questo malcontento diventa eccessivo i visitatori usciranno dal museo in anticipo e lasceranno una recensione negativa, se invece la loro insoddisfazione è bassa lasceranno una recensione positiva.

## 2 Descrizione del problema affrontato

Nella gestione di una qualsiasi impresa è centrale il bisogno di sfruttare le proprie risorse al meglio e di poter avere un'idea dei possibili scenari che si presenteranno in futuro.

Attraverso numerosi strumenti si possono soddisfare le suddette necessità: processi di ottimizzazione, di previsione o di simulazione. Nel caso in analisi si è scelto di trattare quest'ultimo strumento.

Per simulazione si intende un modello fittizio che permette di valutare e prevedere lo svolgersi dinamico di eventi che prendono vita in funzione delle condizioni applicate dagli utenti. Si prestano ad essere strumenti sperimentali eccellenti sfruttando la potenza di calcolo della tecnologia moderna; la simulazione, infatti, consiste nella trasposizione in termini logico-matematico-procedurale di un "modello concettuale" della realtà.

Maggiore sarà il livello di dettaglio e complessità richiesto, maggiore sarà il tempo necessario a programmare algoritmi in grado di fornire dati sufficientemente verosimili e adatti alle analisi d'interesse.

Nel nostro caso l'applicazione in grado di simulare una giornata di visite nel National Gallery of Art di Washington D.C. e di ottenere informazioni riguardo i gruppi di visitatori che entrano nel museo, simulando le visite di ciascun gruppo in maniera autonoma e procedurale, sfruttano elementi stocastici (tipo di gruppo, età dei visitatori, la scelta del periodo di interesse) e condizioni scelte dall'utente.

Per la realizzazione si sono stabiliti alcuni principi:

- 1) Il museo apre dalle 9 di mattina alle 17
- 2) Potranno esserci un massimo di 7000 visite giornaliere con una durata che va da un minimo di un'ora e mezza a un massimo di tre ore.
- 3) I visitatori accedono al museo in Gruppi, suddivisi in 3 categorie:
  - a) Gruppo Generico
  - b) Famiglia
  - c) Gruppo Turistico

ciascuna sfrutterà parametri e algoritmi diversi che verranno trattati più avanti.

- 4) All'entrata alcuni gruppi richiederanno delle audio-guide. Il numero di audio-guide a disposizione a inizio giornata viene specificato dall'utente.
- 5) Ogni gruppo sarà composto da un diverso numero di bambini, adulti e anziani.
- 6) Il gruppo o i singoli visitatori si muoveranno liberamente all'interno del museo, seguendo le proprie preferenze riguardo i periodi, passando da una stanza a un'altra confinante.

- 7) Le stanze avranno una capienza che sarà inserita come parametro dall'utente e se una determinata stanza supererà la capienza scelta verrà segnalato attraverso un avviso che indicherà la stanza critica e l'orario in cui la capienza è stata superata.
- 8) Ogni gruppo avrà un proprio livello di insoddisfazione, dovuto all'eccessivo affollamento delle stanze, che procurerà fastidio ai visitatori e farà loro perdere il doppio del tempo nel visionare le opere di quella stanza.
- 9) Una volta completata la visita i gruppi poseranno le audio-guide e lasceranno il museo, con la possibilità di recensirlo positivamente o negativamente in funzione del grado di insoddisfazione.

Grazie ai risultati ottenuti potremmo avere un'immagine dettagliata degli scenari d'interesse, andando ad analizzare elementi di criticità come la mancanza di audioguide a disposizione, la capienza delle stanze o la soddisfazione media dei visitatori.

### 3 Descrizione del data-set utilizzato per l'analisi

#### 3.1 Introduzione

Per creare il software sono state utilizzate parti di un database fornito dal National Gallery of Art di Washington D.C., il quale afferma che è possibile usufruirne liberamente.

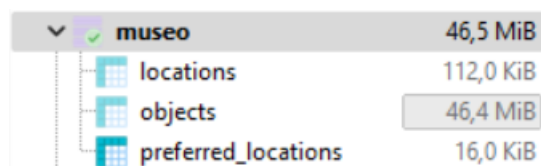
I dati forniti erano tutti in formato CSV, di conseguenza è stata svolta una conversione in formato SQL per permetterne l'importazione attraverso il software HeidiSQL.

A tal proposito è stato creato nell'ambiente di HeidiSQL un nuovo database chiamato "museo" contenente le tabelle e i parametri necessari ad accogliere il contenuto attraverso il comando import CSV.

Al fine di non affaticare l'applicazione con eccessivi calcoli si è preferito snellire il database eliminando dati errati e/o ridondanti, nonché oggetti inutili ai fini del software.

Il database "museo" è composto da 2 tabelle:

- locations
- objects



Database	Table	Size
museo	locations	112,0 KiB
museo	objects	46,4 MiB
museo	preferred_locations	16,0 KiB

Figura 3.1 Data-set completo

#### 3.2 Tabella Locations

La tabella è composta da 919 elementi e rappresenta la posizione fisica e pubblica dell'opera all'interno del museo.

Nota: le posizioni sono definite nella colonna “unitPosition” (N,S,E,W, etc), in modo tale che ogni parete della stanza abbia un locationid differente, ciò può creare descrizioni duplicate o multiple per alcuni ID.

Di seguito sono elencate le colonne contenenti informazioni utili per gli algoritmi utilizzati:

- 1) \item “locationid” rappresenta la chiave primaria che collega la tabella object alla tabella locations. Indica un codice univoco assegnato alla coppia Opera-Stanza.
- 2) \item “site” può contenere il valore “West Building” o “East Building”, a causa di un errore dei fornitori del database le opere dell’edificio est e dei piani superiori al piano principale non sono accessibili, perciò il programma si occuperà soltanto dell’edificio ovest e del “main floor”. Ovviamente fornendo un database adeguato sarà possibile ottenere dei risultati più accurati.
- 3) \item “room” contiene il codice univoco della stanza, ad esempio: G-132.
- 4) \item “publicaccess” contiene un flag: 1= accesso al pubblico, 0= accesso negato. Il database in questo caso contiene solo stanze aperte al pubblico.
- 5) \item “description” contiene la descrizione testuale della stanza, ad esempio: West Ground Floor Gallery 10.
- 6) \item “unitposition” contiene la posizione all’interno della stanza, ad esempio: N (parete Nord) o “SOUTH LEDGE”.

#	Nome	Tipo di dati
1	locationid	VARCHAR
2	site	VARCHAR
3	room	VARCHAR
4	publicacces	INT
5	description	VARCHAR
6	unitposition	VARCHAR

Figura 3.2 Tabella Locations

### 3.3 Tabella Objects

La tabella (figura 3.3) è composta da 2734 elementi e contiene le opere d'arte che si trovano all'interno di una stanza del museo, le opere che non sono esposte o che per errore non hanno un collegamento con la rispettiva location non sono state tenute in considerazione.

Di seguito sono elencate le colonne contenenti informazioni utili per gli algoritmi utilizzati:

- 1) “objectid” rappresenta la chiave primaria della tabella e indica un codice univoco assegnato a ciascun’opera d’arte. Ogni identificatore è creato dal sistema gestionale del NGA.
- 2) “locationid” rappresenta la chiave esterna che collega la tabella object alla tabella locations. Indica un codice univoco assegnato alla coppia Opera-Stanza.
- 3) “title” contiene il titolo dell’opera.
- 4) “displayDate” contiene la data corrispondente al periodo di creazione dell’opera, è in formato più adatto a un utente.
- 5) “beginyear” contiene l’anno in cui l’autore ha iniziato l’opera, leggibile dal calcolatore.



- 6) “endyear” contiene l’anno in cui l’autore ha completato l’opera, leggibile dal calcolatore (Queste date sono state fornite dal NGA per fornire una più semplice ricerca, ma alcune volte possono contenere errori/discrepanze: se un’opera ha l’attributo displayDate vuoto allora utilizzerà come beginyear e end year le date di nascita e morte dell’autore).
- 7) “medium” contiene la descrizione dei materiali utilizzati.
- 8) “dimensions” contiene le dimensioni, dell’opera e dei suoi costituenti.
- 9) “attribution” testo che descrive l’artista a cui l’opera è stata attribuita.
- 10) “classification” contiene la descrizione della categoria a cui appartiene l’opera.
- 11) “subclassification” contiene la descrizione della sottocategoria a cui appartiene l’opera.
- 12) “provenancetext” contiene la descrizione della provenienza dell’opera sottoforma di blocco di testo.



#	Nome	Tipo di dati
 1	<b>objectid</b>	<b>INT</b>
2	accessioned	INT
3	accessionnum	VARCHAR
 4	locationid	VARCHAR
5	title	VARCHAR
6	displaydate	VARCHAR
7	beginyear	VARCHAR
8	endyear	VARCHAR
9	visualbrowsertimespan	VARCHAR
10	dimensions	VARCHAR
11	medium	VARCHAR
12	inscription	VARCHAR
13	markings	VARCHAR
14	attributioninverted	VARCHAR
15	attribution	VARCHAR
16	provenancetext	VARCHAR
17	creditline	VARCHAR
18	classification	VARCHAR
19	subclassification	VARCHAR
20	visualbrowserclassification	VARCHAR
21	parentid	VARCHAR
22	isvirtual	INT
23	departmentabbr	VARCHAR
24	portfolio	VARCHAR
25	series	VARCHAR
26	volume	VARCHAR
27	watermarks	VARCHAR
28	lastdetectedmodification	VARCHAR

Figura 3.3 Tabella Objects

## 4 Descrizione ad alto livello delle strutture dati e degli algoritmi utilizzati

L'applicazione, è stata sviluppata in linguaggio java seguendo il pattern MVC (Model-View-Controller) ed il pattern DAO (Data-Access-Object) tramite i quali sono stati separati l'accesso al database, i processi algoritmici per la manipolazione dei dati e l'interfaccia utente.

### 4.1 Descrizione delle strutture dati: i packages e le classi

I packages sono utili a tenere separate le varie fasi, abbiamo infatti tre packages: uno per il model, uno per il controller e uno per il database.

#### 4.1.1 `it.tdp.polito.NationalGallery`

Il package si occupa dell'interfaccia utente e della procedura di avvio dell'applicazione tramite tre classi:

**EntryPoint:** necessario per l'avvio dell'applicazione stessa

**Main:** gestisce avvio interfaccia nel quale verranno inseriti i dati necessari.

**FXMLController:** riceve dati in Input e collega l'interfaccia con la logica applicativa tramite l'oggetto model. I occupa quindi di interpretare e validare i dati inseriti.

In questa classe avranno luogo gli algoritmi che andranno a comporre l'UI che mostrerà anche gli output.

#### 4.1.2 `it.tdp.polito.NationalGallery.DAO`

Questo package ci permette di effettuare la connessione alla base dati e di interrogare la suddetta attraverso query scritte in linguaggio SQL.

E' composto dalla classe **ConnectDB**, **MuseoDAO** e **TestDAO**(che viene usata solamente per fare dei test).

- **ConnectDB** In questa classe avviene la connessione con il database tramite pooling che permette di risparmiare tempo di apertura e chiusura di una connessione grazie alla classe `HikariDataSource`.
- **MuseoDAO** Questa classe riceve i parametri dal Model(che nella maggior parte dei casi vengono passati dal Controller) e interroga il database tramite query SQL. Crea delle strutture dati idonee all'organizzazione dei dati raccolti e li restituisce al Model.

#### 4.1.3 **it.tdp.polito.NationalGallery.model**

Questo package contiene la logica applicativa e svolge la maggior parte delle operazioni e dei calcoli.

È composto da 15 classi:

- **Autore** Contiene come unico attributo una stringa con il nome di un autore.
- **Event** Definisce la struttura, gli attributi e i diversi tipi di evento da utilizzare durante la simulazione.
- **Famiglia** Definisce una delle classi figlie della classe Gruppo: contiene una lista di membri generati casualmente secondo parametri stabiliti: una famiglia ha da 1 a 4 bambini, da 1 a 2 adulti e da 0 a 4 anziani.
- **Generico** Definisce una delle classi figlie della classe Gruppo: contiene una lista di membri generati casualmente secondo parametri stabiliti: vi sono da 1 a 10 visitatori, tra adulti e anziani e non necessita guide.
- **Gruppo** Definisce la classe padre Gruppo: contiene una lista di membri, l'insoddisfazione del gruppo, l'orario di uscita sottoforma di Duration, una lista di stanze visitate, un periodo di interesse e un nome fittizio del gruppo(usato in fase di controllo).
- **Location** Definisce il legame tra un'opera e la stanza in cui si trova.
- **MembroFamiglia** Definisce la classe figlia della classe Visitatore.
- **Model** È la classe portante del progetto, qui sono contenuti tutti i dati e le strutture necessari al corretto svolgimento dei processi. Il model collega il database e l'interfaccia utente e al suo interno viene creato il grafo che rappresenta la mappa del museo, viene avviata la simulazione specificandone i parametri, e vengono dichiarati i metodi ausiliari.
- **Opera** È la classe che contiene le informazioni delle opere estratte dal database.
- **Simulator** È la classe dove viene costruita e svolta la simulazione utilizzando i parametri specificati nel Model (e quindi nel Controller).
- **Stanza** Definisce una singola stanza: l'edificio in cui è situata, il nome della stanza, se l'accesso al pubblico è consentito, una breve descrizione, un periodo artistico, una lista di opere e la capienza.
- **Statistiche** È la classe che racchiude tutti gli output della simulazione.
- **Turistico** Definisce una delle classi figlie della classe Gruppo: contiene una lista di membri generati casualmente secondo parametri stabiliti: può avere da 1 a 15 persone e necessita di una guida per ciascun visitatore.
- **Visitatore** Definisce un visitatore.

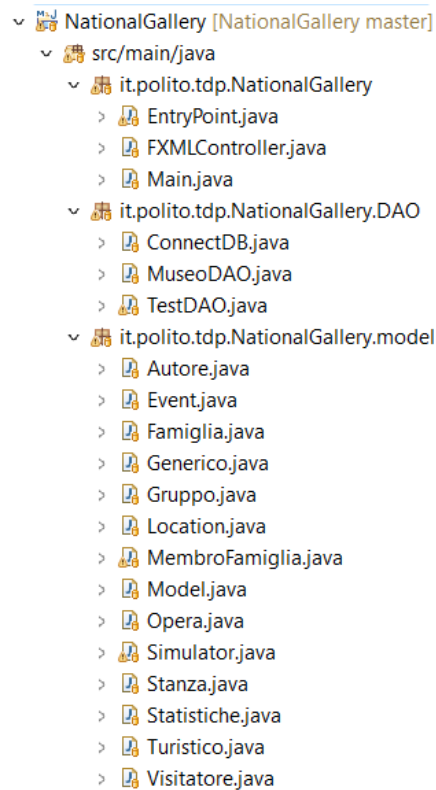


Figura 4.1 Packages e relative classi

## 4.2 Descrizione degli algoritmi utilizzati

### 4.2.1 Sezione 1: Ricerca

In questa sezione l'utente può scegliere dalle combo Box "Begin Year" e "End Year" per individuare un periodo, dopodiché potrà scegliere un autore che ha prodotto un'opera in questo periodo dalla combo Box "Artist". Dopo aver selezionato un artista, la combo Box "Work" mostrerà le opere tra cui scegliere. Premendo il bottone "Get Info" appariranno a schermo le informazioni dell'opera.

#### 4.2.1.1 Riempire le combo Box

Le combo Box dell'interfaccia vengono riempite passo passo a esclusione di quelle che indicano il periodo, che vengono inizializzate non appena viene creato il Model nel Controller.

Il Controller richiama il Model che a sua volta richiama la classe DAO che importa i dati necessari dal database grazie ai metodi: `getBeginYear()` e `getEndYear()`. Una volta selezionati gli elementi delle due combo Box, alla pressione del bottone "Select Period" verrà riempita la Box "Artist", grazie al metodo `getAutoryByAnno()` che necessita di due parametri in input, che saranno i due anni. E infine dopo aver selezionato un artista sarà possibile selezionare un'opera dalla Box "Work" riempita con le occorrenze ottenute dalla funzione `getOpereByAutoreAnno()`.

Nel Controller:

```

public void setModel(Model model) {
    this.model=model;
    this.model.inizializza();
    this.cmbBeg.getItems().clear();
    List<Integer> anniBegin=this.model.getBeginYear();
    for(Integer i: anniBegin) {
        this.cmbBeg.getItems().add(i);
    }
    List<Integer> anniEnd=this.model.getEndYear();
    for(Integer i: anniEnd) {
        this.cmbEnd.getItems().add(i);
    }
    List<String> periodi=this.model.getPeriodi();
    for(String s: periodi) {
        this.cmbPeriodiSponsor.getItems().add(s);
    }
}

```

## Nel DAO

```

public List<Integer> getBeginYear() {

    String sql = "SELECT DISTINCT beginyear FROM objects ORDER BY beginyear;";
    List<Integer> anni = new LinkedList<>();

    try {
        Connection conn = ConnectDB.getConnection();
        PreparedStatement st = conn.prepareStatement(sql);
        ResultSet res = st.executeQuery();

        while (res.next()) {
            try {
                anni.add(Integer.parseInt(res.getString("beginyear")));
            }catch(NumberFormatException e) {

            }

        }
        conn.close();

    } catch (SQLException e) {
        throw new RuntimeException(e);
    }

    return anni;
}

```

```

public List<Autore> getAutoriByAnno(int beg,int end) {

    String sql = "SELECT DISTINCT attribution FROM objects WHERE beginyear>=? AND endyear<=? ORDER BY attribution;";
    List<Autore> autori = new LinkedList<>();

    try {
        Connection conn = ConnectDB.getConnection();
        PreparedStatement st = conn.prepareStatement(sql);
        st.setInt(1, beg);
        st.setInt(2, end);
        ResultSet res = st.executeQuery();

        while (res.next()) {
            Autore a= new Autore(res.getString("attribution"));
            autori.add(a);
        }
        conn.close();
    } catch (SQLException e) {
        throw new RuntimeException(e);
    }

    return autori;
}

public List<Integer> getOpereByAutoriAnno(Autore a,int beg,int end) {

    String sql = "SELECT DISTINCT * FROM objects WHERE beginyear=? AND endyear<? AND attribution=? ORDER BY title;";
    List<Integer> opereA = new LinkedList<>();

    try {
        Connection conn = ConnectDB.getConnection();
        PreparedStatement st = conn.prepareStatement(sql);
        st.setInt(1, beg);
        st.setInt(2, end);
        st.setString(3, a.getNome());
        ResultSet res = st.executeQuery();

        while (res.next()) {
            opereA.add(res.getInt("objectid"));
        }
        conn.close();
    } catch (SQLException e) {
        throw new RuntimeException(e);
    }

    return opereA;
}

```

#### 4.2.2 Sezione 2: Simulazione

La simulazione si basa sulla creazione di un grafo che rappresenterà la mappa del museo. I vertici del grafo saranno le Stanze ottenute tramite il DAO. Le stanze saranno poi collegati da archi una per una in modo da mantenere la planimetria ed avere adiacenze corrette (impossibile da ottenere usando, ad esempio, un algoritmo che sfruttasse il database a disposizione).

```

public void creaGrafo() {
    museo= new SimpleGraph<Stanza,DefaultEdge>(DefaultEdge.class);
    Graphs.addAllVertices(museo, this.stanzeList);
    //collego la Rotunda (hub centrale del museo) con le stanze adiacenti
    //museo.addEdge(this.stanzeMap.get("M-107"), this.stanzeMap.get("M-113"));
    //museo.addEdge(this.stanzeMap.get("M-107"), this.stanzeMap.get("M-121"));
    //museo.addEdge(this.stanzeMap.get("M-107"), this.stanzeMap.get("M-127"));
    museo.addEdge(this.stanzeMap.get("M-107"), this.stanzeMap.get("M-106"));
    museo.addEdge(this.stanzeMap.get("M-107"), this.stanzeMap.get("M-134"));

    //collego la "west sculpture hall" con le stanze adiacenti

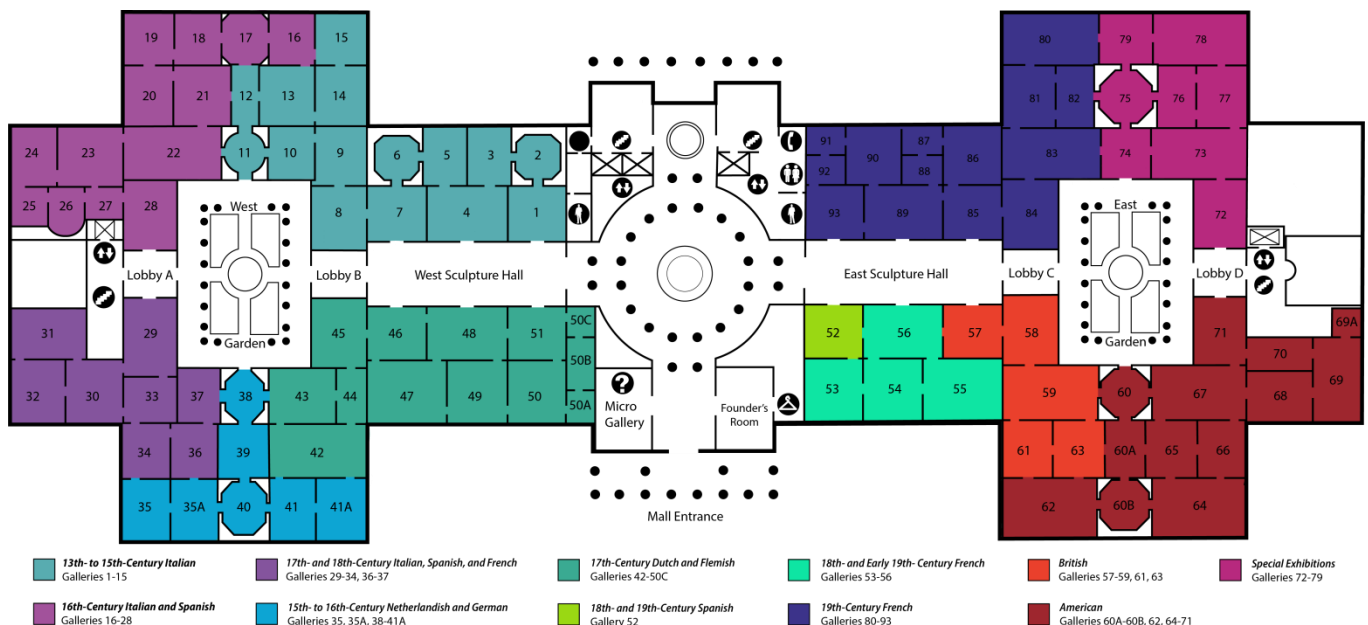
    museo.addEdge(this.stanzeMap.get("M-106"), this.stanzeMap.get("M-001"));
    museo.addEdge(this.stanzeMap.get("M-106"), this.stanzeMap.get("M-004"));
    museo.addEdge(this.stanzeMap.get("M-106"), this.stanzeMap.get("M-007"));
    museo.addEdge(this.stanzeMap.get("M-106"), this.stanzeMap.get("M-051"));
    museo.addEdge(this.stanzeMap.get("M-106"), this.stanzeMap.get("M-048"));
    museo.addEdge(this.stanzeMap.get("M-106"), this.stanzeMap.get("M-046"));
    museo.addEdge(this.stanzeMap.get("M-106"), this.stanzeMap.get("M-105"));
    //collego la "lobby B" con la "West Garden Court"

    museo.addEdge(this.stanzeMap.get("M-105"), this.stanzeMap.get("M-104"));
}

```

Una volta creato il grafo e aver assegnato a ogni stanza il periodo giusto basandosi sulla mappa del museo, è possibile dare inizio alla simulazione grazie al metodo del Model: “iniziaSimulazione()”.

La mappa in figura è stata commissionata ad un grafico che ha riprodotto l’ha riprodotta e ha aggiunto la suddivisione delle stanze in base ai periodi secondo la leggenda.



```

public String iniziaSimulazione(int audioGuide,int capienzaMax,int numVisitatoriMax,int durataMinStanza,
    int durataMaxStanza,int durataMaxVisita,int durataMinVisita,double probabilitaInsoddisfazione,
    double probabilitaSponsor) {
    Simulator simulator= new Simulator(this.museo,this.stanzeMap,this.periodiSponsorizzati,this.periodi);
    simulator.init( audioGuide, capienzaMax, numVisitatoriMax, durataMinStanza, durataMaxStanza,
        durataMaxVisita, durataMinVisita, probabilitaInsoddisfazione,probabilitaSponsor);
    simulator.run();
    return simulator.getStatistiche().toString();
}

```

Viene invocata la classe Simulator che ha i seguenti attributi:

```

public class Simulator {
    //Modello
    private Graph<Stanza, DefaultEdge> mappaMuseo;
    private List<Gruppo> gruppi;
    private int audioGuide;
    private List<String> stanzeCriticheConOra;
    private int capienzaMax;
    private int nPersone;
    private Map<String,Stanza> stanzeMap;
    private List<Visitatore> visitatori;
    private List<String> periodiSponsorizzati;
    private List<String> periodi;
    private int numVisitatoriMax;
    private int durataMinStanza;
    private int durataMaxStanza;
    private int durataMaxVisita;
    private int durataMinVisita;
    private double probabilitaInsoddisfazione;
    private double probabilitaSponsor;
    //Parametri della simulazione
    private int T_ARRIVO_MAX = 17*60;
    private int T_ARRIVO_MIN = 10*60;
    private int T_CHIUSURA= 18*60;
    //Coda degli eventi
    private PriorityQueue<Event> queue;
    //Statistiche
    private Statistiche statistiche;
}

```

Dopo aver creato il simulatore tramite il costruttore e aver inizializzato la simulazione tramite il metodo init(), usando il metodo creaEventi().

Quest'ultimo crea casualmente dei gruppi da fare entrare nel museo, assegna loro un tempo di entrata sottoforma di Duration compreso tra l'orario minimo di entrata e l'orario massimo di uscita e crea un evento di tipo ENTRATA\_GRUPPO e lo inserisce nella PriorityQueue. Inoltre, si decide casualmente se il gruppo sceglierà un periodo di interesse tra quelli sponsorizzati o uno randomico.

```

private Gruppo creaGruppo() {
    double prob=Math.random();

    if(prob<=0.33) {
        return new Turistico();
    }
    else if(prob>0.33 && prob<=0.66) {
        return new Famiglia();
    }
    else {
        return new Generico();
    }
}

```



```

private void creaEventi() {
    Duration arrivo;
    Gruppo g;

    Stanza rotonda= this.stanzeMap.get("M-107");
    while(this.nPersone<numVisitatoriMax) {
        arrivo= Duration.ofMinutes(T_ARRIVO_MIN);
        arrivo = arrivo.plusMinutes((int)(Math.random() * (this.T_ARRIVO_MAX - this.T_ARRIVO_MIN)+ 1));
        //Creo un gruppo 33% famiglia ,33% turistico etc
        g=creaGruppo();
        Double probSponsor= Math.random();
        if(probSponsor<probabilitaSponsor) {
            //al x% il gruppo sarà interessato a un periodo sponsorizzato
            sceglieSponsor(g);
        }
        else {
            //nel 30% dei casi invece sarà un periodo casuale
            g.setPeriodoDiInteresse(this.periodi.get((int)Math.random()*(this.periodi.size()+1)));
        }

        //Se un gruppo è formato da un numero di persone che supererebbe il numero massimo
        //non viene inserito e se ne crea un altro, finchè non si raggiunge la soglia e
        //si esce dal while
        if(!(this.nPersone+g.getVisitatori().size())>numVisitatoriMax)) {
            //Creo l'evento entrata associato a quel gruppo

            Duration durata = Duration.ofMinutes(this.durataMinVisita +
                (int)(Math.random() * (this.durataMaxVisita - this.durataMinVisita) + 1));
            queue.add(new Event(arrivo,EventType.ENTRATA_GRUPPO,g.getVisitatori().size(),g,durata, null,rotonda));
            g.setNomeGruppo(this.nPersone);

            System.out.println("Entrato gruppo : "+this.nPersone+" con "+g.getVisitatori().size()+" persone");
            this.nPersone+=g.getVisitatori().size();
        }
    }
}

```

Dopodichè si dà inizio alla simulazione tramite il metodo run():

```

public void run() {
    while(!queue.isEmpty()) {
        Event e = queue.poll();
        processaEvento(e);
    }
}

```

Gli eventi verranno estratti dalla Queue secondo l'ordine temporale. Inizialmente gli eventi presenti saranno soltanto del tipo ENTRATA\_GRUPPO, ma dopo aver processato ciascuno di questi si creeranno degli eventi di tipo CAMBIO\_STANZA che simuleranno lo spostamento dei gruppi nel museo.

#### 4.2.3 Caso ENTRATA\_GRUPPO

Il metodo processa Evento() ha il compito di svolgere le singole operazioni a seconda del tipo di evento. Nel caso dell'entrata di un gruppo il metodo dovrà innanzitutto fare iniziare la visita al gruppo dall'hub centrale che è la stanza M-107, chiamata "Rotunda", dopodiché incrementerà le statistiche interessate, creerà un orario di uscita idoneo e due eventi : un CAMBIO\_STANZA e un USCITA\_GRUPPO.

```

private void processaEvento(Event e) {
    switch(e.getType()) {
        case ENTRATA_GRUPPO:
            Gruppo g= e.getGruppo();
            Stanza rotonda= this.stanzeMap.get("M-107");
            rotonda.setCapienzaAttuale(rotonda.getCapienzaAttuale()+e.getnPersone());
            //aumento il numero di visitatori e aggiungo il gruppo al mondo del simulatore
            gruppi.add(g);
            this.statistiche.incrementaClienti(e.getnPersone());
            this.statistiche.incrementaGruppi(1);
            for(Visitatore v:g.getVisitatori() ) {
                switch(v.getAge()) {
                    case "adulto":
                        this.statistiche.incrementaAdulti(1);
                        break;
                    case "bambino":
                        this.statistiche.incrementaBambini(1);
                        break;
                    case "anziano":
                        this.statistiche.incrementaAnziani(1);
                        break;
                }
            }
        }

        if (g instanceof Turistico) {
            this.statistiche.incrementaTuristici(1);
            //controllo sulle audioGuide
            int diff= e.getnPersone()-audioGuide;
            if(diff>0) {
                this.statistiche.incrementaVisitatoriSenzaGuida(diff);
                audioGuide=0;
            }
            else {
                audioGuide=audioGuide-e.getnPersone();
            }
            //creo l'evento uscita dal museo,
            //se il time è anteriore all'orario di chiusura allora viene settato
            //alla fine della durata della visita
            //sennò all'orario di chiusura.
            //poi cambio la stanza perchè inizia il tour, quindi creo un evento
            //cambio_stanza
            Duration uscita= e.getTime().plus(e.getDurata());
            if (uscita.toMinutes()>T_CHIUSURA) {
                uscita=Duration.ofMinutes(T_CHIUSURA);
            }
            g.setUscita(uscita);
            esci(uscita,EventType.USCITA_GRUPPO,g.getVisitatori().size(),g,null,null);
            Stanza stanza= scegliStanza(e.getStanzaPrecedente(),e.getStanzaPartenza(),g.getVisitate(), g,null);
            Duration durataStanza= Duration.ofMinutes((int)(Math.random() * (this.durataMaxStanza - this.durataMinStanza)+2));
            Duration time= e.getTime().plus(durataStanza);
            if(time.toMinutes()<=g.getUscita().toMinutes()) {
                Event cambio= new Event(time,EventType.CAMBIO_STANZA,nPersone,g,durataStanza,e.getStanzaPrecedente(),stanza);
                queue.add(cambio);
            }
        }
    }
}

```

```

    }
    if(g instanceof Famiglia) {
        this.statistiche.incrementaFamiglie(1);
        Duration uscita= e.getTime().plus(e.getDurata());
        if (uscita.toMinutes()>T_CHIUSURA) {
            uscita=Duration.ofMinutes(T_CHIUSURA);
        }
        g.setUscita(uscita);
        esci(uscita,EventType.USCITA_GRUPPO,g.getVisitatori().size(),g,null,null);
        Stanza stanza= scegliStanza(e.getStanzaPrecedente(),e.getStanzaPartenza(),g.getVisitate(), g,null);
        Duration durataStanza= Duration.ofMinutes(
            (int)(Math.random() * (this.durataMaxStanza - this.durataMinStanza)+2));
        Duration time= e.getTime().plus(durataStanza);
        if(time.toMinutes()<=g.getUscita().toMinutes()) {
            Event cambio= new Event(time,EventType.CAMBIO_STANZA,nPersone,g,durataStanza,e.getStanzaPartenza(),stanza);
            queue.add(cambio);
        }
    }
    if(g instanceof Generico) {
        this.statistiche.incrementaGenerici(1);
        Duration uscita= e.getTime().plus(e.getDurata());
        if (uscita.toMinutes()>T_CHIUSURA) {
            uscita=Duration.ofMinutes(T_CHIUSURA);
        }

        g.setUscita(uscita);
        esci(uscita,EventType.USCITA_GRUPPO,g.getVisitatori().size(),g,null,null);
        Stanza stanza= scegliStanza(e.getStanzaPrecedente(),e.getStanzaPartenza(),g.getVisitate(), g,null);
        Duration durataStanza= Duration.ofMinutes(
            (int)(Math.random() * (this.durataMaxStanza - this.durataMinStanza)+2));
        Duration time= e.getTime().plus(durataStanza);
        if(time.toMinutes()<=g.getUscita().toMinutes()) {
            Event cambio= new Event(time,EventType.CAMBIO_STANZA,nPersone,g,durataStanza,e.getStanzaPartenza(),stanza);
            queue.add(cambio);
        }
    }
    break;
}

```

#### 4.2.4 Caso CAMBIO\_STANZA

Nel caso in cui il tipo di evento fosse un CAMBIO\_STANZA, il metodo controlla la capienza della stanza di arrivo per andare a fare le valutazioni sull'insoddisfazione e nel caso comunica il superamento della capienza, aggiungendo la stanza e l'orario alla lista delle stanze critiche.

In seguito calcola la durata della permanenza nella nuova stanza in maniera casuale tra il minimo e il massimo impostati dall'utente.

Con il metodo scegli Stanza() viene scelta casualmente una stanza adiacente, dando priorità alle stanze non visitate e del periodo di interesse.

```
case CAMBIO_STANZA:
    Gruppo gc= e.getGruppo();
    Stanza attuale= e.getStanzaPartenza();
    //controllo la capienza
    if((attuale.getCapienzaAttuale())>=capienzaMax)) {
        Double prob= Math.random();
        if(prob<=probabilitaInsoddisfazione)
            gc.aumentaInsoddisfazione();

        if(gc.getInsoddisfazione())>=5) {
            esci(e.getTime().plus(Duration.ofMinutes(1)),EventType.USCITA_GRUPPO,gc.getVisitatori().size(),gc,null,null);
            this.statistiche.incrementaRecensioniNegative(e.getnPersone());
            attuale.setCapienzaAttuale(attuale.getCapienzaAttuale()-e.getnPersone());
            break;
        }
        int t= (int)e.getTime().toMinutes();
        int hours = t / 60;
        int minutes = t % 60;
        String minutiConZero="";
        if (minutes<10) {
            minutiConZero= "0"+minutes;
        }
        //Considero la rotonda di grandezza infinita
        if(!attuale.equals(stanzeMap.get("M-107"))){
            if(minutes<10)
                this.stanzeCriticheConOra.add("STANZA CRITICA: "+attuale+" alle ore "+hours+":"+minutiConZero );
            else
                this.stanzeCriticheConOra.add("STANZA CRITICA: "+attuale+" alle ore "+hours+":"+minutes );
        }
    }

    System.out.println("Cambio stanza gruppo "+gc.getNomeGruppo());
    Stanza stanzaS= scegliStanza(e.getStanzaPrecedente(),e.getStanzaPartenza(),gc.getVisitate(), gc,null);
    Duration durataStanzaS= Duration.ofMinutes(
        (int)(Math.random() * (this.durataMaxStanza - this.durataMinStanza)+2));
    Duration timeS= e.getTime().plus(durataStanzaS);
    if(timeS.toMinutes()<=gc.getUscita().toMinutes()) {
        Event cambioS= new Event(timeS,EventType.CAMBIO_STANZA,nPersone,gc,durataStanzaS,e.getStanzaPartenza(),stanzaS);
        queue.add(cambioS);
        //aggiorno la capienza delle due stanze coinvolte nel cambio
        attuale.setCapienzaAttuale(attuale.getCapienzaAttuale()-e.getnPersone());
        stanzaS.setCapienzaAttuale(stanzaS.getCapienzaAttuale()+e.getnPersone());
    }
    else {
        attuale.setCapienzaAttuale(attuale.getCapienzaAttuale()-e.getnPersone());
    }

    break;
```

```

private Stanza scegliStanza(Stanza stanzaPrecedente, Stanza stanzaAttuale, List<Stanza> visitate, Gruppo g, Visitatore v) {
    // TODO Auto-generated method stub
    String periodoInteresse="";
    if(g!=null) {
        //si sta spostando un gruppo
        periodoInteresse= g.getPeriodoDiInteresse();
    }
    if (v!=null) {
        //si sta spostando un visitatore
        periodoInteresse= v.getPeriodoDiInteresse();
    }
    Stanza prossima= null;
    Stanza precedente=stanzaPrecedente;
    List<Stanza> possibili= new LinkedList<>(Graphs.neighborListOf(this.mappaMuseo, stanzaAttuale)) ;
    List<Stanza> possibiliPreferite= new LinkedList<>();
    int conta=0;
    //Conto quante stanze hanno in comune le visitate e le possibili
    for(Stanza sv: visitate) {
        for(Stanza sp: possibili) {

            if (sv.equals(sp)) conta++;
        }
    }
    //se il contatore è uguale alla grandezza delle possibili allora ho già visitato tutte le possibili
    if (conta==possibili.size()) {
        possibili.remove(precedente);
        //rimuovo la precedente dalle possibili per evitare eccessivi cicli
        if(possibili.size()==0) {
            //se la size di possibili è zero allora ho un vicolo cieco perchè
            //se rimuovendo la precedente la lista possibili è vuota allora la
            //precedente è l'unica stanza disponibile e la scelgo
            prossima=precedente;
        }
        else {
            //sennò ne scelgo una a caso tra le possibili anche se già visitate
            prossima=possibili.get((int)(Math.random()*(possibili.size())));
        }
    }
    else {
        //se non è un vicolo cieco lavoro con le stanze preferibili perchè hanno un periodo di interesse uguale a quello
        //del gruppo o del visitatore
        if(possibili.size()!=0) {
            for(Stanza p: possibili) {
                if(periodoInteresse.compareTo(p.getPeriodo())==0) {
                    possibiliPreferite.add(p);
                }
            }
        }
        if(possibiliPreferite.size()!=0) {
            //se ci sono delle stanze preferibili scelgo una a caso tra quelle
            prossima= possibiliPreferite.get((int)(Math.random()*(possibiliPreferite.size())));
        }
        else {
            if(possibili.size()!=0) {
                double random=Math.random();
                int indice= (int) (random*(possibili.size()));
                prossima= possibili.get(indice);
            }
            //Se non ci sono ne scelgo una a caso tra le possibili
        }
    }
    System.out.println(" dalla stanza "+stanzaAttuale+" Alla stanza "+prossima);
    visitate.add(prossima);
    return prossima;
}

```

## 4.2.5 Caso USCITA\_GRUPPO

In caso di Uscita di un gruppo, vengono restituite le audio guide (se era un gruppo turistico) e se il grado di malcontento è inferiore o uguale a 3 il gruppo lascerà una recensione positiva. In seguito il gruppo sarà rimosso dall'elenco dei gruppi che si trovano nel museo.

```

case USCITA_GRUPPO:
    Gruppo gu= e.getGruppo();
    if(!this.gruppi.contains(gu)) break;
    if(gu instanceof Generico) {
        audioGuide+=e.getnPersone();
    }
    this.statistiche.incrementaInsoddisfazioneTotale(gu.getInsoddisfazione());
    if(gu.getInsoddisfazione()<=3)
        statistiche.incrementaRecensioniPositive(e.getnPersone());
    this.gruppi.remove(gu);
    break;

```

#### 4.2.6 Stampa delle statistiche

Infine il metodo getStatistiche() interroga la classe Statistiche che fornirà i dati da mostrare all'utente, tramite un metodo toString() che è stato precedentemente modificato.

Nel simulatore:

```

public String getStatistiche() {

    this.statistiche.calcolaMediaInsoddisfazione();
    String critiche= "\nStanze critiche e orario:";
    for(String s: stanzeCriticheConOra) {
        critiche+="\n"+s;
    }

    return this.statistiche.toString()+critiche;
}

```

Nella classe Statistiche:

```


public String toString() {
    return "Statistiche:" + "\nTotale Gruppi= " + totaleGruppi + "\nDi cui:"
        + "\n#Famiglie= " + gruppiFamiglie + " #Generici= " + gruppiGenerici + " #Turistici= " + gruppiTuristici
        + "\nTotale Visitatori= " + totaleClienti + "\nDi cui:\n#Bambini= " + bambini + " #Anziani= " + anziani +
        " #Adulti= " + adulti + "\nVisitatori Senza Guida= " + visitatoriSenzaGuida+ "\nRecensioni Positive= " +
        recensioniPositive + "\nRecensioni Negative= " + recensioniNegative+ "\nInsoddisfazione Media= " +
        insoddisfazioneMedia;
}

```

## 5 Interfaccia e video dimostrativo

### 5.1 Sezione Ricerca

Simulazione National Gallery of Art

**National Gallery of Art Simulation**

Search

Simulation

Map

Begin Year

End Year

Select Period

Artist


Select Artist

Work

Get Info

# 5.2 Sezione Simulazione

Simulazione National Gallery of Art



*National Gallery of Art Simulation*

SearchSimulationMap

Max Room Capacity

Max Visitors

5

90

Sponsored Periods

Probability of choosing a sponsored period

0

25

50

75

100

Audio Guides Available

10

180

Select

Likelihood of dissatisfaction

0

25

50

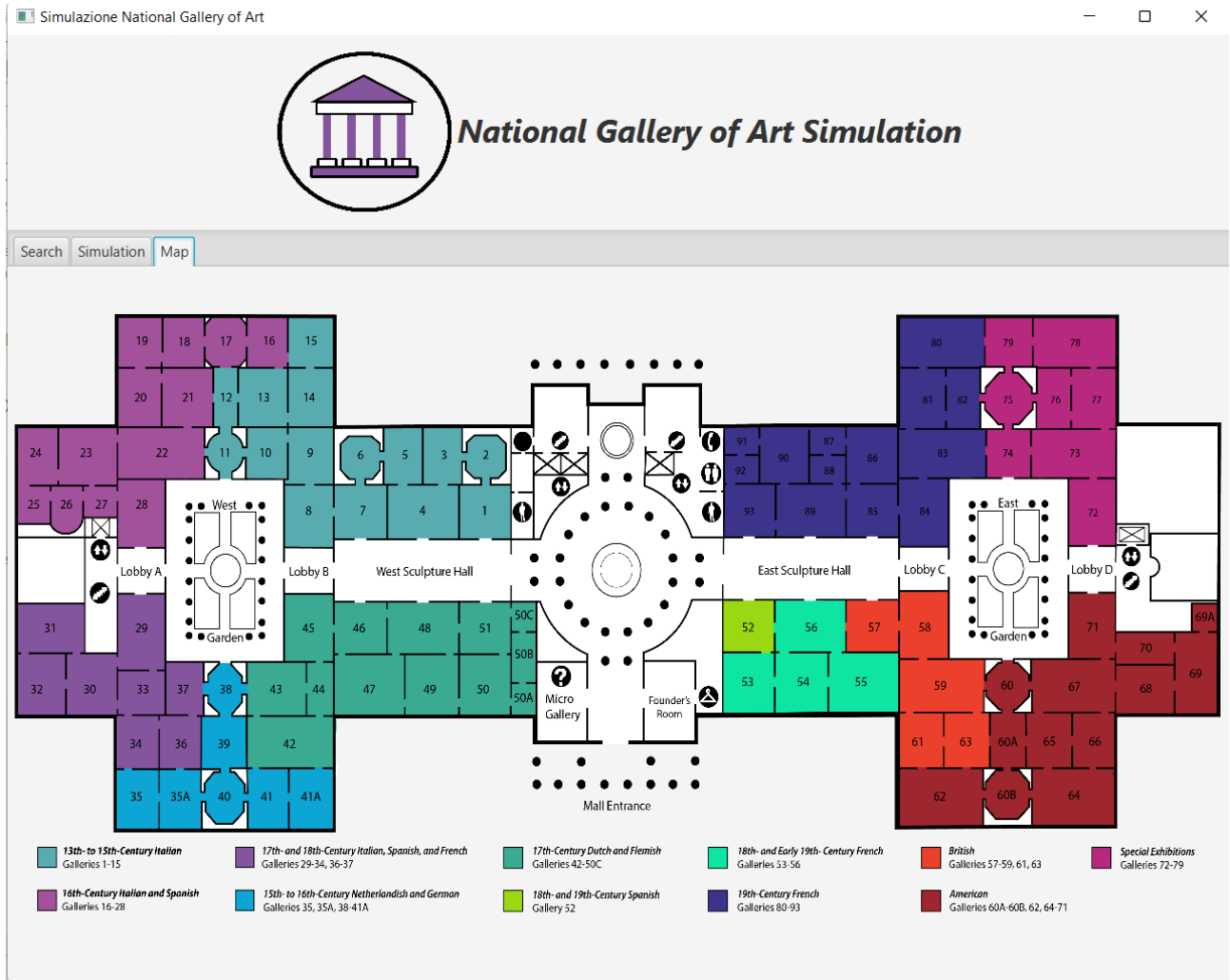
75

100

START SIMULATION



## 5.3 Mappa



Qui potrete trovare un video dimostrativo: [https://youtu.be/BAJJCblY\\_Pk](https://youtu.be/BAJJCblY_Pk)

## 6 Risultati sperimentali e esempi

### 6.1 Esempio 1

#### Ricerca

##### Input:

- BeginYear: 1435
- EndYear: 1600
- Artist: Andrea Mantegna
- Work: Portrait of a Man

##### Output:

Work 41584:

-Title : Portrait of a Man

-Attribution : Andrea Mantegna

-Period : 1470 - 1470

-Description : Painting -

tempera on hardboard transferred from canvas transferred from panel

-Dimensions : overall: 24.2 x 19.1 cm (9 1/2 x 7 1/2 in.)

framed: 37.6 x 32.5 x 3.2 cm (14 13/16 x 12 13/16 x 1 1/4 in.)

-Stanza : Stanza [room=M-013]

#### Simulazione

##### Input:

- Max Room Capacity: 100
- Audio Guides Available: 60
- Max Visitors: 700
- Min Room Stay Duration: Default (5 minuti)
- Max Room Stay Duration: Default(10 minuti)
- Min Visit Duration: Default(90 minuti) (1 ora e mezza)
- Max Visit Duration: Default(180 minuti) (3 ore)
- Sponsored Periods: “13th-to 15th Century Italian”
- Probability of choosing a sponsored period: circa 80%
- Likelihood of dissatisfaction: 75%

##### Output:

Statistiche:

Totale Gruppi= 100

Di cui:

#Famiglie= 35 #Generici= 30 #Turistici= 35

Totale Visitatori= 700

Di cui:

#Bambini= 202 #Anziani= 256 #Adulti= 242

VisitatoriSenzaGuida= 112

Recensioni Positive= 700

Recensioni Negative= 0

Insoddisfazione Media= 0.27

Stanze critiche e orario:

STANZA CRITICA: Stanza [room=M-004] alle ore 13:06

---

## Valutazione

Con una capienza così alta rispetto al numero totale di visitatori è verosimile che l'insoddisfazione media sia così bassa e che le recensioni positive siano pari al numero di visitatori. Nonostante questo la media non è zero infatti possiamo vedere che alle ore 13:06 la capienza della stanza 4 è stata superata, probabilmente a causa dell'alta probabilità di scelta di un periodo sponsorizzato, che porta la maggior parte dei visitatori a recarsi nell'ala sinistra del museo.

Dato l'esiguo numero di guide a disposizione sono stati 112 i visitatori che non hanno potuto usufruirne, corrispondenti a circa 14 gruppi turistici, che sono poco meno della metà di quelli che hanno visitato il museo quel giorno.

## 6.2 Esempio 2

### Ricerca

#### Input:

- BeginYear: 1220
- EndYear: 1796
- Artist: Jan van Eyck
- Work: The Annunciation

#### Output:

Work 46:

-Title : The Annunciation

-Attribution : Jan van Eyck

-Period : 1434 - 1436

-Description : Painting -

oil on canvas transferred from panel

-Dimensions : painted surface: 90.2 x 34.1 cm (35 1/2 x 13 7/16 in.)

support: 92.7 x 36.7 cm (36 1/2 x 14 7/16 in.)

framed: 97.31 x 42.23 x 6.99 cm (38 5/16 x 16 5/8 x 2 3/4 in.)

-Stanza : Stanza [room=M-039]

### Simulazione

#### Input:

- Max Room Capacity: 75
- Audio Guides Available: 100
- Max Visitors: 1500
- Min Room Stay Duration: 2 minuti
- Max Room Stay Duration: 5 minuti
- Min Visit Duration: Default(90 minuti) (1 ora e mezza)
- Max Visit Duration: Default( minuti) (3 ore)
- Sponsored Periods: "13th-to 15th Century Italian","18th-and Early 19th-Century French","19th-Century French"
- Probability of choosing a sponsored period: 25%
- Likelihood of dissatisfaction: 100%

## Output:

Statistiche:

Totale Gruppi= 203

Di cui:

#Famiglie= 63 #Generici= 71 #Turistici= 69

Totale Visitatori= 1500

Di cui:

#Bambini= 392 #Anziani= 555 #Adulti= 553

VisitatoriSenzaGuida= 247

Recensioni Positive= 1361

Recensioni Negative= 94

Insoddisfazione Media= 1.0492610837438423

### Stanze critiche e orario

STANZA CRITICA: Stanza [room=M-005] alle ore 11:31  
STANZA CRITICA: Stanza [room=M-005] alle ore 11:31  
STANZA CRITICA: Stanza [room=M-005] alle ore 11:32  
STANZA CRITICA: Stanza [room=M-005] alle ore 11:35  
STANZA CRITICA: Stanza [room=M-006] alle ore 11:36  
STANZA CRITICA: Stanza [room=M-007] alle ore 11:38  
STANZA CRITICA: Stanza [room=M-006] alle ore 11:46  
STANZA CRITICA: Stanza [room=M-008] alle ore 11:55  
STANZA CRITICA: Stanza [room=M-010] alle ore 12:07  
STANZA CRITICA: Stanza [room=M-005] alle ore 12:17  
STANZA CRITICA: Stanza [room=M-005] alle ore 12:19  
STANZA CRITICA: Stanza [room=M-003] alle ore 15:41  
STANZA CRITICA: Stanza [room=M-004] alle ore 15:45  
STANZA CRITICA: Stanza [room=M-005] alle ore 15:46  
STANZA CRITICA: Stanza [room=M-005] alle ore 15:46  
STANZA CRITICA: Stanza [room=M-089] alle ore 15:48  
STANZA CRITICA: Stanza [room=M-005] alle ore 15:48  
STANZA CRITICA: Stanza [room=M-089] alle ore 15:48  
STANZA CRITICA: Stanza [room=M-005] alle ore 15:48  
STANZA CRITICA: Stanza [room=M-005] alle ore 15:49  
STANZA CRITICA: Stanza [room=M-006] alle ore 15:49  
STANZA CRITICA: Stanza [room=M-005] alle ore 15:49

## Valutazione:

Rispetto al primo esempio la capienza è stata diminuita del 25% mentre l'affluenza è stata più che raddoppiata, vediamo come le recensioni positive siano diminuite e contemporaneamente notiamo un' insoddisfazione più elevata e quasi un centinaio di recensioni negative. Ciò è dovuto alla scelta di impostare la probabilità di essere insoddisfatti in una stanza piena al valore 100%, ciò significa che 94 visitatori hanno raggiunto il massimo dell'insoddisfazione e hanno deciso di uscire dal museo. Il numero è ancora gestibile ma probabilmente la capienza delle stanze è stata rispettata nel

maggior numero di casi perché la durata della permanenza in una stanza è stata dimezzata rispetto a quella di default. Invece le audio guide risultano non essere abbastanza per la quantità di visitatori, 247 restano senza. Questo dato però è particolarmente aleatorio in quanto dipende fortemente dal numero di gruppi turistici e dalla loro dimensione, infatti le oscillazioni tra le varie simulazioni sono molto grandi, nonostante i dati in input rimangano invariati.

Con gli stessi dati ma con la durata di default avremmo il doppio delle recensioni negative e l'insoddisfazione media aumenta di mezzo punto:

```
Statistiche:
Totale Gruppi= 209
Di cui:
#Famiglie= 72  #Generici= 67  #Turistici= 70
Totale Visitatori= 1500
Di cui:
#Bambini= 408  #Anziani= 590  #Adulti= 502
VisitatoriSenzaGuida= 151
Recensioni Positive= 1207
Recensioni Negative= 205
Insoddisfazione Media= 1.4736842105263157
```

Abbiamo considerato solo una parte dell'output Stanze critiche data la sua dimensione, però possiamo notare come la zona più affollata sia l'ala sinistra.

### 6.3 Esempio 3

#### Ricerca

##### **Input:**

- BeginYear: 1842
- EndYear: 1930
- Artist: Paul Gauguin
- Work: Self-Portrait

##### **Output:**

```
Work 46625:
-Title : Self-Portrait
-Attribution : Paul Gauguin
-Period : 1889 - 1889
-Description : Painting -
oil on wood
-Dimensions : overall: 79.2 x 51.3 cm (31 3/16 x 20 3/16 in.)
-Stanza : Stanza [room=M-083]
```

## Simulazione

### Input:

- Max Room Capacity: 50
- Audio Guides Available: 500
- Max Visitors: 5000
- Min Room Stay Duration: 10 minuti
- Max Room Stay Duration: 30 minuti
- Min Visit Duration: 120 minuti (2)
- Max Visit Duration: 210 minuti (3 ore e mezza)
- Sponsored Periods: Tutti
- Probability of choosing a sponsored period: 100% (in questo caso è indifferente perché abbiamo aggiunto tutti i periodi tra quelli pubblicizzati)
- Likelihood of dissatisfaction: 100%

### Output:

---

Statistiche:

Totale Gruppi= 698

Di cui:

#Famiglie= 231 #Generici= 242 #Turistici= 225

Totale Visitatori= 5000

Di cui:

#Bambini= 1323 #Anziani= 1801 #Adulti= 1876

VisitatoriSenzaGuida= 400

Recensioni Positive= 1859

Recensioni Negative= 2623

Insoddisfazione Media= 3.5730659025787967

Stanze critiche e orario:

STANZA CRITICA: Stanza [room=M-004] alle ore 10:32  
STANZA CRITICA: Stanza [room=M-089] alle ore 10:44  
STANZA CRITICA: Stanza [room=M-089] alle ore 10:45  
STANZA CRITICA: Stanza [room=M-001] alle ore 10:50  
STANZA CRITICA: Stanza [room=M-003] alle ore 10:50  
STANZA CRITICA: Stanza [room=M-089] alle ore 10:51  
STANZA CRITICA: Stanza [room=M-089] alle ore 10:51  
STANZA CRITICA: Stanza [room=M-003] alle ore 10:52  
STANZA CRITICA: Stanza [room=M-004] alle ore 10:52  
STANZA CRITICA: Stanza [room=M-002] alle ore 11:26  
STANZA CRITICA: Stanza [room=M-085] alle ore 11:26  
STANZA CRITICA: Stanza [room=M-085] alle ore 11:27  
STANZA CRITICA: Stanza [room=M-054] alle ore 11:27  
STANZA CRITICA: Stanza [room=M-007] alle ore 11:27  
STANZA CRITICA: Stanza [room=M-054] alle ore 11:28  
STANZA CRITICA: Stanza [room=M-004] alle ore 11:28  
STANZA CRITICA: Stanza [room=M-085] alle ore 11:28  
STANZA CRITICA: Stanza [room=M-004] alle ore 11:28  
STANZA CRITICA: Stanza [room=M-002] alle ore 11:28  
STANZA CRITICA: Stanza [room=M-002] alle ore 11:28  
STANZA CRITICA: Stanza [room=M-006] alle ore 16:23  
STANZA CRITICA: Stanza [room=M-006] alle ore 16:23  
STANZA CRITICA: Stanza [room=M-085] alle ore 16:23  
STANZA CRITICA: Stanza [room=M-006] alle ore 16:24  
STANZA CRITICA: Stanza [room=M-085] alle ore 16:24  
STANZA CRITICA: Stanza [room=M-085] alle ore 16:24  
STANZA CRITICA: Stanza [room=M-056] alle ore 16:24  
STANZA CRITICA: Stanza [room=M-089] alle ore 16:24  
STANZA CRITICA: Stanza [room=M-006] alle ore 16:24  
STANZA CRITICA: Stanza [room=M-007] alle ore 16:24  
STANZA CRITICA: Stanza [room=M-006] alle ore 16:25

---



STANZA CRITICA: Stanza [room=M-093] alle ore 16:36  
STANZA CRITICA: Stanza [room=M-089] alle ore 16:36  
STANZA CRITICA: Stanza [room=M-093] alle ore 16:36  
STANZA CRITICA: Stanza [room=M-006] alle ore 16:36  
STANZA CRITICA: Stanza [room=M-093] alle ore 16:37  
STANZA CRITICA: Stanza [room=M-005] alle ore 16:37  
STANZA CRITICA: Stanza [room=M-085] alle ore 16:37  
STANZA CRITICA: Stanza [room=M-089] alle ore 16:37  
STANZA CRITICA: Stanza [room=M-085] alle ore 16:37  
STANZA CRITICA: Stanza [room=M-093] alle ore 16:37  
STANZA CRITICA: Stanza [room=M-085] alle ore 16:38  
STANZA CRITICA: Stanza [room=M-089] alle ore 17:43  
STANZA CRITICA: Stanza [room=M-085] alle ore 17:43  
STANZA CRITICA: Stanza [room=M-004] alle ore 17:44  
STANZA CRITICA: Stanza [room=M-052] alle ore 17:44  
STANZA CRITICA: Stanza [room=M-003] alle ore 17:45  
STANZA CRITICA: Stanza [room=M-089] alle ore 17:47  
STANZA CRITICA: Stanza [room=M-089] alle ore 17:48  
STANZA CRITICA: Stanza [room=M-003] alle ore 17:48  
STANZA CRITICA: Stanza [room=M-089] alle ore 17:48  
STANZA CRITICA: Stanza [room=M-089] alle ore 17:49  
STANZA CRITICA: Stanza [room=M-089] alle ore 17:51

### **Valutazione:**

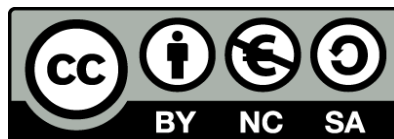
Diminuendo ancora la capacità, aumentando la durata della visita e della permanenza nelle stanze possiamo vedere come l'insoddisfazione media aumenti notevolmente, così come le recensioni negative (che indicano il numero di visitatori usciti dal museo a causa del malcontento) che sono quasi il 50% del numero di visitatori. Inoltre vediamo come qui il numero di autoguide a disposizione in un caso di default sarebbero accettabili, mentre qui lasciano molti visitatori privi di guida, questo è dovuto al fatto che la permanenza totale del museo è stata aumentata e i dispositivi vengono trattenuti più a lungo.

Se invece osserviamo le stanze critiche in vari momenti della giornata possiamo notare come non vi sia una zona particolarmente affollata, ovviamente le parti del museo più vicine all'hub centrale appariranno più spesso tra le stanze critiche perché saranno in cui confluiranno più visitatori.

Con gli stessi dati ma con una durata di default di permanenza nelle stanze avremo molte più recensioni negative e maggiore insoddisfazione in quanto i visitatori passeranno da una stanza all'altra molto più spesso generando maggiore malcontento.

## 7 Valutazioni sui risultati ottenuti

I risultati ottenuti dall'applicazioni sembrano verosimili e corretti, si riescono a calcolare dati relativi anche a migliaia di visitatori senza ritardi e in maniera quasi immediata. Ciò è sicuramente causato dal fatto è stata considerata soltanto un'ala del museo, ma il programma può essere aggiornato e migliorato, apportando modifiche riguardo la mappatura e i tipi di dati di input e output. Sono moltissime le informazioni che si possono ottenere da questo tipo di simulazioni, ed è stato difficile scegliere su cosa concentrare l'attenzione; come abbiamo potuto vedere i dati ricavati possono avere un'importanza significativa in funzione del problema aziendale affrontato, in quanto utilizzabili per valutare scenari possibili e per prevedere eventuali bisogni. Un software simile sarebbe, ad esempio, potuto essere utile durante il periodo successivo alla pandemia, durante il quale era necessario ridurre il più possibile gli assembramenti: il museo avrebbe potuto decidere la capacità massima delle singole stanze e dell'intera struttura in funzione dell'affluenza, evitando così il rischio di assembramento.



This work is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

To view a copy of this license, visit

<http://creativecommons.org/licenses/by-nc-sa/4.0/>