Elevator Simulator Project Report

Content

|  |  |  |
| --- | --- | --- |
| Sr No | Title | Page Number |
| 1 | Introduction | 3 |
| 2 | Hierarchy Design for passenger class | 4 |
| 3 | Traffic Generator Module | 5 |
| 4 | StaticsKeeper Modules | 10 |
| 5 | System Controller Module | 11 |
| 6 | Generation for Traffic for 4 Time Units |  |
| 7 | Timing Wheel and Partition |  |
| 8 | Demo |  |

1. Introduction

The elevator system simulator must be capable of servicing different types of passengers encountered in a hospital scenario. Each Passengers has a different unique ID and every Passenger has Weight of 1 Units by default. The Other Passengers might carry extra Equipment and their weights information would be provided from a input file. A building will have multiple floors and multiple elevators will be servicing all floors.

The System Controller is an important component of the ESS and in turn is composed of the elevator traffic generator and statistics keeper. The Traffic Generator generates the traffic based on the user specified parameters. The user is responsible to supply the total number of floors and total number of elevators in the building. The number of passengers created at a given time instant should be a random number between 0 and a user defined maximum. For each new passenger created, current floor and destination floor should be randomly generated.

Statistics Keeper collects all the data needed for providing the desired statistics. It is a part of the System Controller also.

1. Hierarchy Design

**A screenshot of a cell phone

Description automatically generated**

Figure 1

Figure 1 shows the hierarchy design of Passengers class. All common attributes are shifted to Base class and child class will all inherit them. The type of inheritance is public. Support staff and Medical Staff class have a parent class which is abstract class and abstract class has pure virtual function called emergency since only support staff and medical staff can issue emergency call. Security class is also abstract and has pure virtual function called evacuate and child class securitypersonal will inherit from the pure abstract class. The Visitors and Patients inherits from Passenger class as shown in the Diagram

1. Traffic Generator

A screenshot of a cell phone

Description automatically generated

Figure 2 UML of File read class

This class does the job of reading the input file and creating the Necessary Data Structure. The Design Pattern used in this is Singleton Design Pattern as there is no need to creating multiple instances of the same class hence Singleton Design Pattern will ensure that class has one and only one instances.

A screenshot of a cell phone

Description automatically generated

Figure 3 shows the UML Diagram for Traffic Generator

Traffic Generator has FileRead class it’s a composition The job of Traffic Generator is to read the File and create the appropriate Data structure. The Design Pattern used for this class is Singelton as we only need one instance of the class.

A screenshot of a cell phone

Description automatically generated

**Figure 4** shows the relation between Both classes

|  |
| --- |
| {  0: [  ['8','4','12','10'],  ],  1: [  ['Visitors', '0.15'],  ['', '100', '0', '0'],  ['', '1', '1'],  ],  2:[  ['Patient',' 0.25'],  ['100', '0', '0'],  ['','1','3']  ],  } |

Figure 5 Shows the data Structure

The Following data Structure is generated after the File is Read. It’s a map of int and values are vector vector of string. Its similar to JSON Nested Dictionary.

Once the File is read and the Data structure is formed, we call it as PData. And then we call Generate Unique Passengers and Generate a Data Structure of Levels which is a map of int and values are vector of Passengers pointers. The Passenger pointer are generated based on their probability which is given in input file.

The Traffic Generator also has Helper class which are static methods which helps in Data type Conversion process.

**Screen Shots of Traffic Generators**

A screenshot of a cell phone

Description automatically generated

**Figure 6**: Shows main.cpp file

First Time 1 Units 5 Passengers are Generated

A screenshot of text

Description automatically generated

**Figure 7:** Shows The 5 Passengers Generated for 1 Time Unit

A screenshot of text

Description automatically generated

**Figure 8:** Shows the 6 Passengers Generated for 2 Time Unit

**IV Statics Keeper**

**A screenshot of a cell phone

Description automatically generated**

**Figure 9 Shows the UML of Statics Keeper**

**Output** of Statics Keeper

**A screenshot of a cell phone

Description automatically generated**

**Figure 10 Shows the Output of Statics Keeper**

**V System Controller**

The system controller if the main. Heart of entire Elevator Simulator System. It jobs is to generate the Traffic and perform necessary function. The Design Pattern used for System Controller is Bridge Design Pattern. The main reason for using Bridge Design pattern is to separate the interface and implementation.

A screenshot of a cell phone

Description automatically generated

**Figure 11:** shows UML of System Controller

Main.cpp

A screenshot of a cell phone

Description automatically generated

A screenshot of a map

Description automatically generated

**Figure 12:** Shows the System controller along with other classes

VI Traffic Generation for 4 Time units

A screenshot of a cell phone

Description automatically generated

Figure 13: shows screenshot of main.cpp file

A screenshot of a cell phone

Description automatically generated

Figure 14: Shows the output of Statics Keeper after the 4 Time Units

Output and ScreenShots

A screenshot of a cell phone

Description automatically generated

Figure shows the screen shots of main.cpp

A screenshot of text

Description automatically generated

Figure shows the Screen shots of the output

**A close up of text on a black background

Description automatically generated**

Figure shows for the first time slots these many passengers are generated

A screenshot of a cell phone

Description automatically generated

Figure shows the checking the levels call and task for each elevator