Department of Computer Science and Engineering

University of Puerto Rico at Mayagüez

Report of Project 1

Emergency Helper System

Dr. Manuel Rodríguez Martínez

Dong Wang

Mar/03/2020

TABLE OF CONTENTS

I. Introduction

This report is a reference manual that explains the Emergency Helper System (EHS). It will show what is the main idea of the database design. Besides, it will display the Entity-relation diagram, which shows the general picture how the database system works. In this diagram, each entity and relation will be explained.

(Phase 2 or later)

The rescue system provides a dashboard. Its graphical user interface and functions will be explained too. In the fourth part, it shows the model-view-controller system.

II. Emergency Helper System (EHS)

The Emergency Helper System is also called EHS, which is the application that manages the data by relational database system. This system is exposed to client applications through a REST API. The backend of the system is relational, and allows the user to browse categories for resources (i.e., water, food, medications, tools, heavy equipment, etc.), and search for specific items and obtain who is supplying items, and who needs the items. Also, it provides location (GPS) and address of the person in need and that of the supplier. You will provide some free resources while others have a cost.

Besides, the system has a simple Web-based UI to search and view resources, with the option to locate the person and supplier in a Google map. Also it provides a Web-based dashboard indicating relevant statistics for supply/demand of resource categories, and trending data (e.g., charts that show if resources are increasing or decreasing).

III. Entity-Relation Diagram (ERD) of EHS

The image above is the entity-relation diagram for EHS. From the diagram, as a uer, there are three types of registrations. In other words, there are three types of users that can login into the system related to their identities and purposes. There are three identities that a person can be. They are administrator, user, supplier.The section “merge” is the integrator that receives all the users and divides them into certain categories. Each system administrator can regulate at least one user. Each user has one administrator to check and monitor their behavior.

The administrator can login into the backend system to get all the information of supplier and users, delete the inappropriate users or suppliers . However, adding a new user or supplier to the database is not allowed. Either, the administrator has no privilege to update the information of each consumer or supplier. The operation of adding and updating can only be implemented by consumers themselves. These actions would effectively limit the malbehavior from administrators. In this way, we can minimize the risk of attacking the rescuing information.

A consumer has several options on the website. 1) A consumer can register, login, delete, revise his/her information. The operation of “delete” and “register” can be implemented by a consumer. 2) The consumer can check the information of each type resource with availability and its supplier information. 3) A consumer can make a reservation or purchase. Based on the consumer’s need for a certain item. The system will measure if the need surpasses the remaining amount. If so, the engineer needs to make a purchase from the supplier. If not surpass, then the customer can make a reservation for the item. Also each consumer must have one or more payment methods to purchase the certain material.

A supplier can register and login with implementing these tasks. 1) A supplier can check the resource price, and check availability of each item.

IV. API Introduction

@app.route('/')
def home():
This is the home page to load the information of home index

@app.route('/PartApp/parts', methods=['GET', 'POST'])
def getAllParts():
Here the parts mean different types of resources that people may need.
The request of ‘GET’ is to check the information of all information about the condition of the resource. And the ‘POST’ is to revise the information of the resource.

@app.route('/PartApp/parts/<int:pid>', methods=['GET', 'PUT', 'DELETE'])
def getPartById(pid):

This method can help search a single one item based on its number. Besides, a person can revise the information of the item and delete it.

```
@app.route('/PartApp/parts/<int:pid>/suppliers', methods=['GET', 'POST'])
def getSupplierByPartId(pid):
```

This API is to display the resource based on the given name of suppliers. The ‘POST’ is about making changes on the resource. Based on the certain supplier. Still I plan to grant this privilege to this administrator.

```
@app.route('/PartApp/suppliers', methods=['GET', 'POST'])
def getAllSuppliers():
```

This method allows customers to check information of suppliers to purchase and the reservation of resources to make a request. This ‘Get’ request can let the customer see the availability of each resource. Then he/she can decide to make a reservation or make a purchase.

```
@app.route('/PartApp/customers', methods=['GET', 'POST'])
def getALLCustomers():
```

This method allows the administrator to check and delete the information of the customer.

References

[1] (n.d.). Retrieved from https://www.postgresql.org/docs/
[2] “Workstation Player : Run a Second, Isolated Operating System on a Single PC with VMware Workstation Player.” *VMware*, 12 Dec. 2019, www.vmware.com/products/workstation-player.html.
[3] “Download Ubuntu Desktop: Download.” *Ubuntu*, ubuntu.com/download/desktop.