



Júlio Gabriel Chilela

WEB GEOGRAPHIC INFORMATION SYSTEMS (WEBGIS) FOR SMART CAMPUS AND FACILITY MANAGEMENT

Dissertation in Geographic Information Technology, Specialization in Sciences and Geographic Information Technology, guided by Dr Alberto Cardoso and Dr Driss Benhaddou and presented to the department of Mathematics of Faculty of Science and Technology at University of Coimbra

September/2016



UNIVERSIDADE DE COIMBRA

Júlio Gabriel Chilela

WEB GEOGRAPHIC INFORMATION SYSTEMS(WEBGIS) FOR SMART CAMPUS AND FACILITY MANAGEMENT

Dissertation in Geographic Information Technology, Specialization in Sciences and Geographic Information Technology,
guided by Dr Alberto Cardoso and Dr Driss Benhaddou and presented to the Department of Mathematics of Faculty of
Science and Technology at University of Coimbra

September/2016



UNIVERSIDADE DE COIMBRA

"The mind of man plans his way, but the LORD directs his steps."

Proverbs 16:9

Acknowledgement

I am truly thankful to God for the scientific knowledge He has so graciously allowed me to acquire for the purpose of contributing to the scientific community, hopefully inspiring others to reach for a better world for all of us, in this country and in my own, for generations to come.

I am also thankful to my parents Manuel Estevao and Lourdes Estevao as well as Cecilio Chilela and Angelina Napembe for taking care of me and loving me unconditionally and for their love throughout all time, even when I spent my days enveloped in books while others in my home toiled and laboured for my benefit.

I would like to offer an enormous amount of thanks to the Sapalalo's family for helping me along my way, not only in my studies but also in life. I will never forget these words, "We are doing this because we love you."

To my aunt and uncle, Estevao Chaves and Josina Chaves, thank you for bringing out the best in me.

A special thanks to my uncle and aunt Josias Chaves and Zoia Chaves to be my counselors and to advise me about the right choice to pick in my career.

To my Godmother Eliza Wendo for your prayers.

My special thanks to my grandfathers Goncalves and Eliza for tutoring me.

Especial thanks to Edivaldo Paciencia, for your magnanimous generosity

My big special thanks to my teachers, Alberto Cardoso of the University of Coimbra, in Coimbra, Portugal and Dr. Driss Benhaddou of the University of Houston, in Houston, Texas for your encouragement and support.

Last but not least, to my family and friends, thank you for directly and indirectly helping me to reach my goals step by step.

Student Number: 2014184162

Nome: Júlio Gabriel Chilela

Dissertation:

WEB GEOGRAPHIC INFORMATION SYSTEMS (WEBGIS) FOR SMART CAMPUS AND FACILITY MANAGMENT.

Key-Words:

- Web Geographic Information
- Smart Campus
- Facilities Management

Abstract

This Dissertation introduces the usage of the WebGIS for smart campus and facility management, for data visualization and to provide interaction between users - users and users - smart campus.

The goal of this project is the development of a WebGIS capable of saving, organizing, and geo-spatializing information from all over the campus and facilities. In addition, this WebGIS will also process data and manipulate devices connected to the internet using concepts and tools of Internet of Things and provide the number of people inside of buildings. The information about the number of people inside of buildings will be provided by a new brand technology knowing as smart door. This technology was created in the College of Technology, Laboratory of Wireless and Optical Network of the University of Houston. In case of a public business building (bank or shopping for instance), the WebGIS is capable, for example, to show the average time that a client can spend in that building to be attended.

In addition, this project provides a mobile application to allow users to interact with smart campus and campus facilities and to facilitate access to basic information about public transportation and data sensors which have been placed in various locations around campus.

Geographic information systems in the Web are integrated to provide maps in 3D of campus buildings and all smart campus information in layers. This system will allow smart campus administrators to interact with the devices installed in facilities around campus.

This WebGIS system will allow users to contribute to the general management of campus maintenance.

Resumo

Esta dissertação introduz o uso de Sistemas de Informação Geográfica na Web em campus inteligentes e gestão de edifícios, para a visualização de dados e para possibilitar interações do tipo utilizador-utilizador e utilizador-campus inteligente.

O objectivo do projecto éo desenvolvimento de um Sistema de Informação Geográfica na web capaz de armazenar, organizar e geo-espacializar a informação do campus inteligente e de instalações. O WebSIG vai também processar dados e manipular dispositivos conectados à internet, usando conceitos e ferramentas da Internet das Coisas e fornecer o número de pessoas em cada edifício. A informação sobre o número de pessoas em cada edifício será fornecida por uma nova tecnologia conhecida como Smart Door. Esta tecnologia foi desenvolvida no College of Technology, Laboratory of Wireless and Optical Network da Universidade de Houston. Em casos de prédios com serviços públicos (Bancos ou Lojas por exemplo), o WebSIG é capaz de mostrar, por exemplo, a média de tempo que um cliente pode demorar para ser atendido naquele edifício.

O projecto também providencia uma aplicação móvel para permitir aos utilizadores interagirem com o campusinteligente e/ou as instalações e terem acesso a informações de transporte publico, visualização de dados dos sensores distribuidos ao longo do campus, em tempo real.

Os Sistemas de Informação Geográfica na Web são integrados para disponi-bilizar mapas 3D dos edificios do campus e organizar a informação em camadas. Esse sistema vai permitir a administração do campus inteligente interagir com os dispositivos electrónicos instalados nos edificios e ao longo do campus.

O WebGIS vai possibilitar que utilizadores contribuam para a gestão geral e manutenção do campus.

Resumé

Cette dissertation presente l'utilisation des WebGIS pour le campus intelligent et la gestion des installations, pour la visualisation des donnees et pour permettre interactions du type utilisateurs-utilisateurs et utilisateurs-campus intelligent.

Ce projet a pour objectif de developper un WebGIS capable de sauver, d'organiser et de geo- localiser des informations venues de tout le cam-pus et ses installations. Dans la dependance, ce WebGIS Permettra également de traiter les données et manipuler les peripheriques connectes a l'Internet en utilisant des concepts et des outils de l'Internet des objets et de fournir le nombre de personnes a l'interieur des bâtiments. Les informations sur le nombre de personnes a l'interieur des bâtiments sera assuree par une toute nouvelle technologie Smart Door aussi intelligent. Dans le cas d'un bâtiment public des a aires (banque ou faire du shopping par exemple), le WebGIS est capable de montrer le temps moyen qu'un client peut passer dans ce batiment a être assisté.

De plus, ce projet fournira une application mobile pour permettre aux utilisateurs d'interagir avec le campus intelligent ainsi que ses installations, et faciliter l'acces a des informations simples sur le transport public et les capteurs de donnees qui ont et places dans divers endroits autour du campus, en temps reel.

Les systemes d'information geographique dans les WebGIS sont integres pour fournir de cartes des installations du campus en 3D et pour organiser les informations en couches. Ce systeme va permettre aux administrateurs du campus intelligent d'interagir avec les dispositifs installes dans les instal-lations autour du campus.

Ce systeme WebGIS permettra aux utilisateurs de contribuer a la gestion generale et a l'entretien du campus.

Samenvatting

Dit proefschrift introduceert het gebruik van geogra sch Informatiesysteem op het web van smart campus en het beheren van gebouwen, voor visualisatie van data die mogelijk maakt de interactie tussen de gebruiker-gebruiker en gebruiker-gebruiker smart campus.

Het doel van het project is ontwikkelen een Geogra sch Informatie sys-teem op de web die in staat is om opslaan, organiseren en geo-specialiseren de informatie van smart campus en faciliteiten. In verslaving, dit WebGIS ver-bruikt zal ook gegevens en manipuleren van apparaten die zijn aangesloten op het internet met behulp van concepten en instrumenten van internet van de dingen en zorgen voor het aantal mensen dat de binnenkant van gebouwen. De informatie over het aantal mensen dat in gebouwen zal worden verzorgd door een nieuwe technologie in de wetenschap zo slim deur. In het geval van een openbaar bedrijfsgebouw (bank of winkelen bijvoorbeeld), de WebGIS is in staat om de gemiddelde tijd dat een cli•ent kan besteden in dat gebouw worden bijgewoond tonen.

Het project voorziet ook een mobiele applicatie die gebruikers mogelijk maakt interageren met smart campus en / of de faciliteiten toegang krijgen tot het openbaar vervoer informatie, data visualisatie van sensoren versprei-den over de campus in de werkelijk tijd.

De Geogra sche Informatie Systemen in de Web, worden ge•ntegreerd om de 3D-kaarten van de campus gebouwen beschikbaar te stellen en organiseren de informatielagen.

Dit systeem zal het beheer van smart campus laat interageren met de genstalleerde elektronische toestellen in het gebouw langs de campus. De Web GIS zal de gebruikers mogelijk maken bijdragen aan het algemene be-heren en het onderhouden van de campus.

Contents

Acknowledgement	i
Abstract	iii
Resumo	iv
Resumé	v
Samenvatting	vi
Chapter 1	1
Introduction	1
1.1 Importance of the study	1
1.2 Problem statement	3
1.3 Purpose of the study	4
1.4 Outline of the dissertation	5
Chapter 2	6
Literature review	6
2.1 Geographic Information Systems: An explanation	6
2.1.1 Geospatial information.....	8
2.1.2 The growth of GIS	11
2.2 Web applications and WebGIS	13
2.2.1 Web Applications.....	13
2.2.2 WebGIS: an overview	15
2.3 The need for WebGIS.....	17
2.4 Internet of Things (IoT).....	18
2.5 Internet of Things and WebGIS	18
Chapter 3	20
System Tools	20
3.1 Web Development Tools.....	20
3.1.1 HTML.....	20
3.1.2 CSS	22
3.1.3 JavaScript.....	23
3.1.4 Bootstrap	23

3.2	WebGIS Tools	24
3.2.1	Cesium 3D data visualization	24
3.2.2	OpenStreetMap	28
3.3	Django.....	29
3.4	Python	30
3.5	GeoDjango	31
3.6	PostGIS	32
3.7	RabbitMQ	34
3.8	Tornado Web Server	35
3.9	WebSockets	36
3.9.1	WattDepot	36
3.10	Hardware.....	37
3.10.1	Smart Door	37
3.10.2	Raspberry Pi and GPIO	38
3.10.3	Wireless Sensors Network.....	39
Chapter 4	41
System Architecture		41
4.1	System Objectives.....	41
4.2	General Architecture of the System.....	43
4.2.1	Database logic model	44
4.2.2	Cesium-3D building modelling	45
4.2.3	The Smart Door Middleware.....	48
Chapter 5	49
WebGIS Portal: Practice case		49
5.1	Administrator.....	49
5.1.1	Add buildings in the map.....	51
5.1.2	Add Rooms	52
5.1.3	Add Devices	55
5.1.4	Main Page.....	57
Chapter 6	66
Tests and Validation		66
Chapter 7	68
Conclusion.....		68
7.1	Future work	69
References	70
APPENDIX A		73
Installation of the architecture		73
APPENDIX B		75
Raspberry pi Camera installation		75

Chapter I

Introduction

This is a project in Sciences and Geographic Information Technology, related to the master's degree in Geographic Information Technology of the University of Coimbra. The project was developed in the College of Technology, Laboratory of Wireless and Optical Network at the University of Houston, United States of América. It was guided by Dr. Alberto Cardoso, Teacher Assistant at University of Coimbra and Dr. Driss Benhaddou, Associate Professor at University of Houston.

1.1 Importance of the study

In the modern world we have seen the density of population growing very fast in many countries, and along with this phenomenon, cities are growing exponentially. Many buildings are being built and with them technologies are being used to make the buildings' residents live comfortably. Cameras, sensors and other devices have been placed everywhere throughout these cities, collecting data and sending them to one central management station.

Managing cities is not an easy task. It is more difficult when the number of residents and buildings is very huge. Many problems can occur in a short time, and usually, it is extremely important to inform urban managers about problems in as short a time as possible in order to mitigate the impact of certain situations.

Due to the need for security, city management is in need of smart systems, including alert systems in cases of disasters, such as flooding, fire, and climate changes. In the same vein, any city's infrastructure maintains several differing entities that require specialized management systems with varying technological support. For example, fire departments require alert systems while medical

centers utilize technology that enhances ambulance quality control systems, monitoring traffic patterns and route-time network analysis.

For the purposes of this research, a university campus provides, on a smaller scale, the various components found in any large city, thereby providing a model that could demonstrate its capabilities in much the same way.

University campuses are already equipped with cameras, sensors, power plants, bus systems, and other departments that present differing challenges for testing the capabilities of a smart system for facilities management. Furthermore, a large university student population of around 40,000 students offers advantages for testing smart systems, such as the efficacy of the systems' ability to detect and normalize CO₂ levels and temperature control, as well as the gathering of massive amounts of data for collection and analysis. Moreover, large student populations assist in the testing of smart doors, which can manage high occupancy levels, identify intruders by eye scanning and face recognition among other capabilities. Another important point is the availability of university public transportation systems, providing necessary information for students in real-time, and of energy management system on a university campus providing invaluable data from a central control location, assisting the smart system in decision-making, for example, in the consideration of energy storage capabilities in hazardous conditions.

A WebGIS can be developed to help in management of cities. It can provide geo-spatialized information in real-time for helping managers to get information and manipulate smart devices as well as help citizens to have important information about the city in real-time.

1.2 Problem statement

WebGIS are systems that provide geospatial information, for instance, of many smart cities in many different ways. Some are representing data of sensors around the city, monitoring environmental conditions such as CO₂, temperature, humidity, sound, light changes; others are managing smart devices, opening bridges, controlling traffic lights, borders intruders alert, cameras, air conditioning and so on. But, none of them is providing one simple information that is very useful in many areas of the society: The number of people inside of buildings. This dissertation will bring not only a WebGIS with features to manage smart environments but will also introduce the Smart Door device in the WebGIS world and explore its capabilities.

The Smart Door device contributes to control the occupancy of the building, the time that a person get in and out of the building. It provides the number of people inside of the building. A 3D WebGIS connected to this smart device (device that belongs to the internet of things portfolio) can provide not only tools to monitor environmental conditions in a city and manage smart devices but also provide the number of people inside of buildings. This information can be used for instance in fires department or to help citizens to manage their time before go to a building with public service (for instance a bank).

1.3 Purpose of the study

The goal of this study is to develop a 3D WebGIS capable of saving, organizing, and geo-spatializing information from all over the campus and facilities. In addition, this WebGIS will also process data and manipulate devices connected to the internet in real-time using concepts and tools of Internet of Things (IoT) and provide one important information to users that is the number of people inside of buildings using the Smart Door device.

The study took place in the campus of the University of Houston due to the amount of smart devices placed around the campus and the APIs available for developers. The study started with the literature review on papers with topics related to displaying 3D buildings in browsers and how to use them with the Internet of Things.

The main plan was based in the following steps:

- 1- Study how to display 3D maps on the web.
- 2- Study types of sensors and the way that they communicate with computer. Get to know the communication protocols, security, the life time of a sensor, battery duration.
- 2- Study the best way to get data from sensors and which database to use.
- 3- Design the 3D web map prototype.
- 4- Start the GIS analysis project, build end publish in a local server, analyze the behaviour and errors.
- 5- Test the scalability, increase the number of sensors and users, use different sources of information including the Smart Door device.

WebGIS will benefit from Internet of Things. This is the integration that constitutes the main topic of this study.

1.4 Outline of the dissertation

This dissertation is divided in 8 chapters and each one approach the dissertation in a different aspect.

Chapter 1 is an introductory chapter that provides an overview of the project, in addition, it gives a brief view of what this project will bring new in the WebGIS world.

Chapter 2 is a bibliography review. It resumes concepts of GIS, WebGIS and what we can find in the world of WebGIS and Internet of things. How 3D maps are being used for smart objects to control smart cities and internet of things. The dissertation presents a short resume of existing projects relating Internet of Things and WebGIS as well as the new feature that this dissertation introduces in the WebGIS world.

Chapter 3 describes every tool needed to develop the project. From web development tools to Hardware gadgets.

Chapter 4 presents the development of the project, the architecture of the system and all functionalities.

Chapter 5 is related to the results obtained with the developed system and describes the use case.

Chapter 6 This chapter is an explanation about the systems' test and validation. White and black box testing were made. The dissertation discusses what were the users reactions in terms of what is operating in the system as well as which are the improvements that the system needs.

Chapter 7 is the conclusion, summarizing the study and bringing a brief conclusion about the study and what areas the system can be improved.

Finally, the dissertation includes references and two appendices.

Chapter 2

Literature review

2.1 Geographic Information Systems: An explanation

In the set of information systems we find a very particular type named Geographic Information System (GIS). Geographic Information Systems are a combination of hardware, software and data geo-spatialized, made to manage, and save information geographically referenced.

In order to have great results, decision makers use GIS to visualize categorized information about many areas of the modern world.

Geographic information Systems are commonly used for:

"Measurements: the knowledge of distance or the spatial(aerial) extent or volume of a feature or incident will be basic but important and, using proximity analysis, GIS can establish the distance of objects relative to a theme or to other objects. Any units of measurement can be deployed, finding statistical measurements such as sum, mean, mode and standard deviation.

Distributions and relationship: Spatial distributions of object may be either random and regular or clustered, and GIS have the functionality, usually via the use of the nearest neighbor analysis, to describe distributions in these terms. Using contiguity analysis, they can also calculate the relationship between different distributions across the surface, spatially auto-correlated.

Network analysis: This analysis applies to linear features such as transport routes, rivers, pipelines and cable networks. It can establish least-costs routes, shortest path routes, and degree of connectivity, etc.

Measurement in network analysis can be in terms of monetary units, distance, and time, etc.

Temporal analysis: Spatial changes can be in absolute terms or defined over time. Thus, it is valuable to know, for instance, the varying rates of growth of an urban area over equal consecutive time periods, to identify the proportional changes in land use for a given area over time. The long-term collection of remotely sensed data greatly expedites time series analyses.

Modeling: This is a broad category that frequently includes What if scenarios or models, and this can be done for past, present or future scenarios. Optimum location analysis is a modeling procedure that attempts to optimize the location of any activity based on known inputs of the principal production functions. Digital terrain modeling allows for the inclusion of the height dimension for GIS analysis of slopes, aspect, contours and volumes.

Interpolation: This is simply the generation of missing values based on a set of known values within a study area. For instance, if a series of spot heights (altitude) are known, then it is possible to interpolate contour lines for the same area. Interpolation can be applied to a wide range of measured values.

Visualization: In GIS, visualization is used to organize spatial data and related information into layers that can be analyzed or displayed as maps, three- dimensional scenes, summary charts, tables, time-based views and schematics. (Geofferry et al, 2013, p10)"

2.1.1 Geospatial information

Geospatial is a reference of located information on the earth, or in simple terms, information with coordinates. This information can be for instance, the position of a hospital around a province, roads of a country, or the outline of a city. The Figure 2.1 and 2.2 represent Hospitals locations in Benguela province and the roads in Angola respectively.

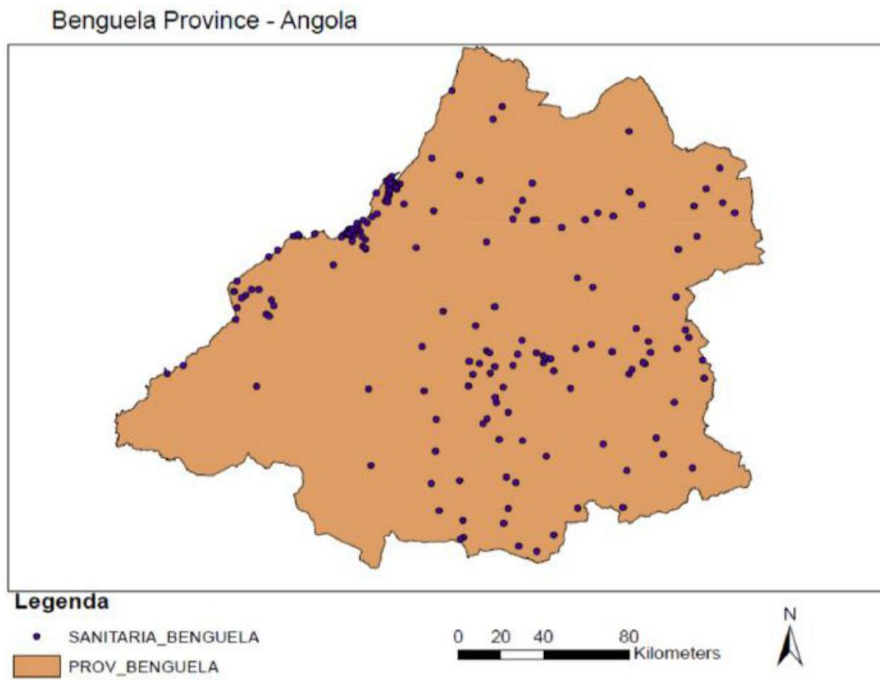


Figure 2.1: Geospatial concept. Location of hospitals in Benguela province-Angola.



Figure 2.2: Roads in Angola

Geospatial generally join some information with a specific location.

For instance, the following Figure (2.3) represents the variation of the slope in a region of Coimbra.

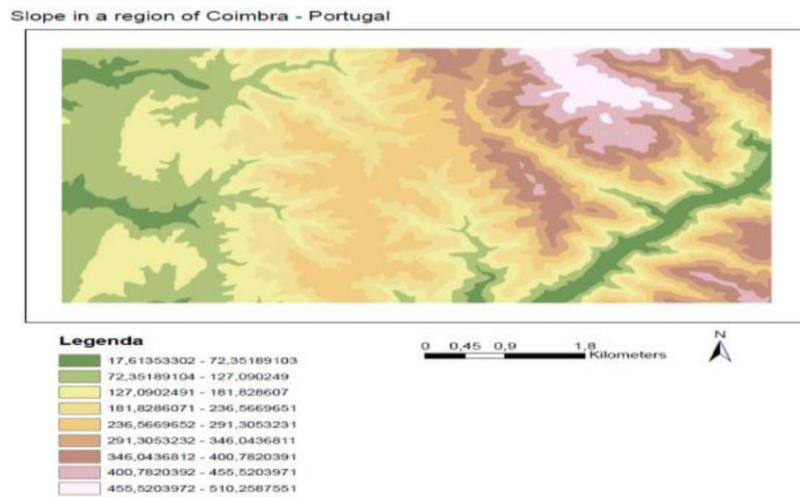


Figure 2.3: Interactive map. Slope variation in percentage- Coimbra

Internally in Geographic Information Systems, geospatial data is represented as a series of coordinates, often in the form of latitudinal and longitudinal values. Additional attributes such as temperature, soil type, height or the name of a landmark are also often present. There can be many thousands (or even millions) of data points for a single set of geospatial data (Westra, 2013).

For example the following outline of the Figure 2.4 is from Angola and it is represented by more than 5,000 points.

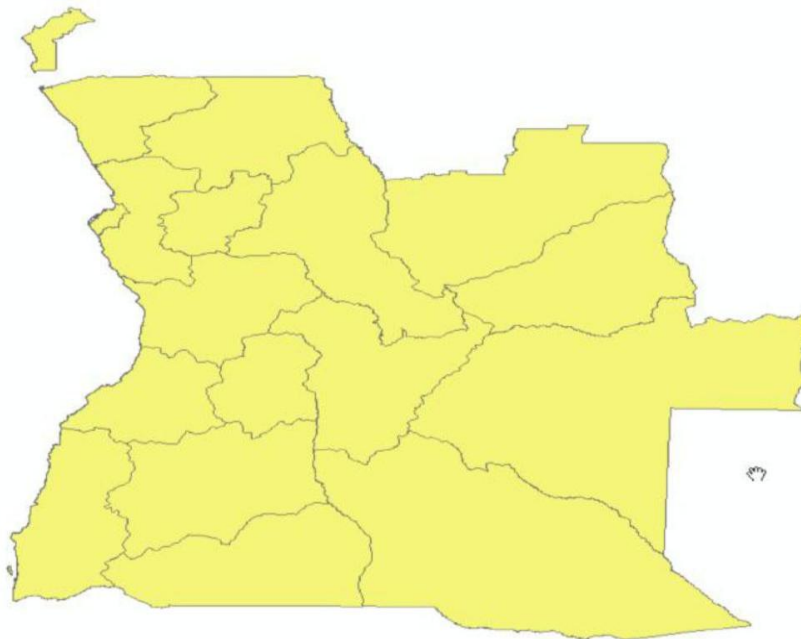


Figure 2.4: Outline of Angola

The safest way to save geospatial data is within a database, considering the large number of data that each feature supports.

2.1.2 The growth of GIS

Over time, the needs of man to know locations for hunting and to know the way back home after the hunt, has driven him to a new necessity of sketch paths and to register areas to hunt specific kinds of animals.

Over the ages, man was updating the way that he registered places of interest. Maps became more readable and understandable. With the development of technology, man started to integrate the task of building maps in computers. Computer graphics output became more developed in the 1950s (Emlinson, 1976). Along with this phenomenon, the treatment of geographic data also became more popular in computers, taking advantage in its biggest and widest resources such as databases, fast calculus and graphical visual-ization.

The Canadian Geographic Information Systems (CGIS) was the first recognized GIS (Calkins, 1976). During this time, many researchers conducted extensive studies in very different fields of GIS. Researchers faced difficult times because of the huge cost of GIS production; therefore, government institutions supported them as well as big universities using mainframe computers.

Developers were focused on the proprietary software and the applications were particularly developed for specific projects.

Mapping and measurement formats were the most commonly GIS output with some geo-spatial analyses. Applications were focused in defense, urban planning, census analyses, land and resource management.

From the 1970s, after researchers advanced GIS development, the proprietary software became popular and started growing. The largest opportunity to invest in GIS and the big potential that it became, opened the doors to many private companies investing more in GIS.

Products like ESRI's ArcInfo in 1982, Infomap by Synercom, and CARIS produced by Universal System were released (Marble, 1976).

While other GIS parallel fields were growing, GIS itself also developed rapidly thanks to the greater access to data. These common fields named geotechnology enhanced the ability to provide data through the World Wide Web. Over time, the cost of GIS

production decreased and it became more massive. Also, conventions were more realistic and possibilities of creating GIS cross-platforms were more visible.

Nowadays, Geographic Information Systems are undeniable popular gad-gets in many areas of the modern world. In a half century, Geographic Information Systems had tremendously developed and grown exponentially.

Recently the United States Department of Labor has recognized geo-technology as one of the three most important emerging and evolving fields of the twenty- first century, i.e. together with nanotechnology and biotechnology (Gewin, 2004).

Today, GIS is covering different areas and its growth is evident at more than 14 percent per year (Frost, 1994).

2.2 Web applications and WebGIS

2.2.1 Web Applications

Web applications, commonly known as web apps, are a server or client application that typically run through a web browser. These types of application normally require the browser to go to the server and load information from time to time or each time it runs.

When we refer to an application, we are talking about a specific software that performs a specific task. Nowadays, we have a myriad of web technology designed to be used across internet as we can see in the table 2.2.1.

Client Side

The client side represents the part of the application that runs on the user's side. To perform some specific tasks, some web applications are strictly de-pendent on what the user will input in a visual framework of the application.

The client side is mainly developed with hypertext Markup Language (HTML) but also along with HTML, counts with CSS as the styled language, as well as JavaScript which makes the client side dynamic.

In modern times, companies and web developer's groups developed a wide number of templates to make web developers lives easy when they are building a client side application. As an example we have the bootstrap framework.

Server / Client Side

Web application	Web technology
www.juliochilela.info	HTML, Asp.net, Postgres JavaScript, CSS, Visual Basic
www.thesis.juliochilela.info	HTML, CSS, JavaScript, Geo- Django framework, Postgis
www.gestaodeigrejas.info	HTML, JavaScript, Django framework, Posgres

Table 2.2.1: Technologies in a modern web application

Bootstrap is an intuitive and powerful mobile first front-end framework for faster and easier web development made using HTML, JavaScript and CSS (bootstrap.com, 2016).

Server Side

We can consider a server as a computer program like a web server that runs in the remote server and can be accessed by a client computer through the network.

The server side contains the database logic and sends our requests to JavaScript, then afterwards, send them to the client side. We can find many server side programming languages like PHP, Ruby, Python and Javascript. The database is hosted in this area of the application. Nowadays, there are a myriad of options, like Mysql, MariaDB, MangoDB, PostgreSQL, MS-SQL server or Oracle DB.

2.2.2 WebGIS: an overview

A WebGIS is a Geographic Information System working in a network and obeying the World Wide Web's protocols. It does not work only for Hyper Text Transfer Protocol (HTTP), Uniforms Resources Locators (URL), Hyper Text Markup Language (HTML), Web sockets, JavaScript and so on.

WebGIS structure may vary considering its systems' diversity.

Generally, WebGIS applications are tools that allow users to manipulate, interact, analyze, save, manage and exhibit all kinds of geo-spatial data, as well as edit data through the web map and visualize the results of operations.

The Internet removed the distance in cyberspace. Some of the reasons why WebGIS is so popular in the World Wide Web nowadays are:

"Global reach: Clients can share geographic information easily within their organization and with people all over the world if you wish.

Large number of users: Clients can share their apps with thousands, or even millions of users.

Low cost per user: The cost of building one WebGIS app is often cheaper than building a stand-alone desktop solution and installing it for every user.

Better cross-platform capabilities: Web apps, especially those built with Javascript, can run on desktop and mobile platforms running a wide range of operating systems, from Windows, Mac OS, and Linux to IOS, Android, and Windows Phone.

Easy to use: WebGIS apps typically incorporate simplicity, intuition, and convenience into their design. Therefore, public users can use these apps without having prior knowledge.

Easy to maintain: Web clients can benefit from the latest program and data updates each time they access a web app. The web administrator does not have to update all the clients separately.

Diverse applications: WebGIS has made building apps easier, so its applications have grown diverse. Unlocking the power of geospatial intelligence,

it now reaches millions of homes, offices, and laboratories and millions of individuals.(ESRI, 2015)"

For government, WebGIS offers an ideal channel for delivering public information services, an engaging medium for encouraging public participation, and a powerful framework for supporting decision making. For business, WebGIS helps create novel business models and reshape existing ones. It enhances the power of location-based advertising, business analysis, and volunteered geographic information, generating tremendous revenue, both directly and indirectly. For science, WebGIS creates new research areas and renews existing avenues of research (ESRI, 2015).

In daily life, WebGIS helps people decide where to eat, stay, and shop and how to get from here to there.

Web mapping is the process of designing, implementing, generating, and delivering maps on the Web and its products (ESRI, 2015).

The final product known as WebGIS is an online map that can be the output of different spatial analyses or simply representation of data in the map. Smartphones, tablets and computer browsers are updated to be a client's WebGIS map.

2.3 The need for WebGIS

Nowadays, WebGIS are being created all over the world to fulfil different needs of societies. Many of them were developed to represent cities in 2D or 3D and explore capabilities of Geographic Information Systems.

Different areas of the society uses GIS to benefit of its advantages. For instance, in Urban traffic there are WebGIS that are being used for measurement and routes, to know the distance between two points, as well as to know the short path to reach one specific location.

There are some WebGIS that are also being used in prediction. For example, according to the historic behavior of a specific kind of animal because of weather changes (temperature, precipitation and humidity), it is being possible to predict locations where the animal will go when the environmental conditions change. These kinds of map are known as maps of multi-criterial analyses.

In consideration of the slope conditions, temperature, precipitation, aspect and the distance from the rivers WebGIS are also predicting hazardous areas to live or to install one specific infrastructure in some chosen area.

City's administrators are calling upon WebGIS for urban planning and cadaster. Because of the variety of resources that it presents, it is possible to register all the city's information, like the ownership of a portion of land (with geographic coordinates), as well as the facility to know how many portionsof land one person has around the area.

Throughout GIS, it is possible to register wires and cables around the city, power poles, locations of interests and companies that are doing maintenance of different zones in the urban perimeter. Spatial analyses and modeling processes in WebGIS are also used to choose the best area to build an infrastructure, analyzing different variables such as proximity to the forest zones, slope, distance from urban areas, and exposure to the elements. This kind of information process is known as multi-criterial analysis.

2.4 Internet of Things (IoT)

The Internet of Things (IoT) is related to the evolution of the actual internet where more and more devices and humans are connected. The human being is the one who is having domain in the network, but the perspective is that autonomous machine will be ruling the world wide web.

Many type of devices and infrastructure are communicating one with another through the internet, for instance, cameras, cars, sensors, doors, bridges, traffic lights, buildings and so on. "Nowadays, it is estimated that 12.5 billion devices are connected to the Internet and that this trend will continue to reach 50 billion devices by the end of the decade" (Evans, 2011).

Internet of Things devices can be classified in three categories: Wearables, smart home devices and machine to machine.

2.5 Internet of Things and WebGIS

A new world has been created by the Internet of Things. Quantified and measurable world is becoming real through all the technologies that are presented in the market and all that are being produced. And, professionals in Geographic Information Systems' field are working everyday to be part of this evolution.

Related works about WebGIS and Internet of Thing are present in our daily life. Societies, governments and military are using the WebGIS as one of the most important tool to handle different situations.

There are some WebGIS that are providing maps of environmental information. These WebGIS are connected to small devices called as mote that are sensing our surrounding environment; and maps in real-time are showing environmental conditions such as temperature, CO₂, humidity, etc.

Some WebGIS systems are being used to monitor cities. Some cameras are placed around the city and WebGIS are connected to these cameras where administrators can record videos and see what is going on in some locations in real-time. Many of them are also connected to sensors, like sound sensors to monitor the noise pollution in many areas of the city.

One of the projects connecting WebGIS and IoT is the "3D web visualization for real-time maintenance of smart building" developed by Daniel Gaston, Christophe Joubert and Miguel Montesinos. This project tend to geo-spatialyze objects inside of buildings. The project explores the 3D visualization in the web and all of Cesium 3D visualization capabilities, providing the floor and room where the device is located.

This dissertation will discuss not only the integration of 3D WebGIS and Internet of Things but also adding a new feature called Smart Door in the 3D WebGIS world.

Chapter 3

System Tools

3.1 Web Development Tools

The web development world is filled with different programming languages and tools. It is one of the most changeable world nowadays. In this decade, programming languages are created in a short period of time and all of them have different resources that comes to make developers lives easy and fast.

In this section, the dissertation presents the web programming languages chosen to develop the system.

3.1.1 HTML

Hyper Text Markup Language (HTML) was created by Tim Berners-Lee. HTML is a markup language used to develop web pages. Browsers can interpret HTML documents. All HTML documents are simple text files that can be created and edited in any text editor like note pads, text editor Macintosh (Wikipedia, 2016).

In order to mitigate the time to produce a HTML document, There is in market different Text Editors with the most modern resources that make repetition tasks easy, insert objects in the easiest way, table elaboration in the easiest way and so on.

For example, the HTML code below is responsible to display maps of 3D buildings in a web browser:

```
<!DOCTYPE html><html lang=" en ">
```



```

<head>
<!-- Use character set. --><meta charset="utf-8">
<!-- Tell IE to use the latest, best version (or
Chrome Frame if pre-IE11). -->
<meta http-equiv="X-UA-Compatible"
content="IE=Edge,chrome=1">
<!-- Make the application on mobile take up the full
browser screen and disable user scaling. -->
<meta name="viewport" content="width=device-width,
initial-scale=1, maximum-scale=1, minimum-scale=1,
user-scalable=no">
<title>3d maps</title>
<script src="{% static 'js/jquery-1.11.2.min.js %}">
</script>
<link href="{% static 'css/bootstrap.min.css %}"
rel="stylesheet">
<script src="{% static 'js/Cesium.js %}"></script>
<style>
@import url({% static 'css/widgets.css ' %});
@import url({% static 'css/home/bootstrap.min.css ' %});
@import url({% static 'css/home/modern-business.css ' %});
@import url({% static 'css/home/font-awesome/css/font-
awesome.min.css ' %})
html, body, #cesiumContainer {
width: 100%; height: 100%; margin: 0; padding: 0;
overflow: hidden; }
</style>
</head>
<body onload="ShowBuilding();">
<div id="cesiumContainer">
<div id="controlPanel">
</div>
</body>
</html>

```

3.1.2 CSS

Cascading Style Sheets is a simple mechanism for adding style (font, colors, spacing) to web documents [5]. It represents how HTML elements will be displayed on the screen, paper or other media. CSS also save a lot of workin consideration that it can control any amount of web elements at once.

```
<style type="text/css"> #controlPanel {
  position: absolute;
  top: 5px;
  left: 5px;
  background: rgba(42, 42, 42, 0.8); padding: 5px 8px;
  border-radius: 5px; }
  label {
  cursor: pointer;
  }
  label:hover span {
  text-decoration: underline; }
  #home {
  width: 15em; height: 2em; text-align: left;
  }
  #scat {
  width: 15em; height: 2em; text-align: left;
  29
  }
  #dd {
  width: 15em; height: 2em; text-align: left;
  } #as {
  width: 15em; height: 2em;
  text-align: left; }
  #pt {
  width: 15em; height: 2em; text-align: left;
  }
```

The CSS code above will make the HTML element called Control Panel positionsbe absolute; with distance from the top 5px; left 5px; background rgba(42, 42, 42, 0.8); padding 5px 8px; border- radius 5px. This kind of way to control HTML elements can simplify the development of the website. CSS also has a tag called Class. One class is one CSS structure that can be applied to control many HTML elements at the same time. It means that any time there is need to change the color or the text font of all labels, for example, it can be done once in Class, and the change will be applied to all HTML elements that are members of class.

3.1.3 JavaScript

JavaScript is considered a programming language for HTML and Web. Is an interpreted programming language with object oriented capabilities.

The main thing to consider regarding JavaScript is that it supports numbers, strings and boolean values as primitive datatypes. It also includes built-in supports for arrays, date, and regular expression objects.

It is mostly used in web browsers, and it has scripts that interact with users, control the web browser, and alter the document content that appears within the web browser window.

JavaScript code example:

```
<script>
document.getElementById ("demo" ). innerHTML= "My F i r s t J a v a S c r i p t " ;
</script>
```

When the script above is running, it will look for the HTML element called demo and change its text with the text,"My First JavaScript".

3.1.4 Bootstrap

Bootstrap is a sleek, intuitive, and powerful front-end framework for faster and easier web development, created by Mark Otto and Jacob Thornton, and maintained by the core team with the massive support and involvement of the community [6].

Bootstrap is a very popular HTML, CSS, and JavaScript framework for developing responsive websites.

3.2 WebGIS Tools

3.2.1 Cesium 3D data visualization

One of the aims of this thesis is to investigate the use of three-dimensional visualization for buildings in web maps. This section represents a description of the fundamentals of three-dimensional visualisation.

Three-dimensional visualization is executed in some basic steps, a so-called pipeline. The visualization is determined by: projection, frame buffer handling, re-rendering scheme, transformation, rotation, scale, and scene construction. This information is interpreted by the CPU and then sent to the geometry engine (Patrik Ottoson, 2011).

Cesium is a JavaScript library focused on 3D graphics, in 3D globes creation and 2D maps, running specifically in web browsers with no plug-ins, and at the same time doing normal graphics(Cesium, 2016).

The Cesium's first step were given by Analytical Graphics, Inc. (AGI), and its maintenance is guaranteed by an open source community. It is a map library and JavaScript virtual globe. In order to obtain a hardware accelerated graphic, it uses Web Graphics Library (WebGL). It is independent from the web browser and operating system, and it is particularly designed for dynamic data visualization. It supports a 3D globe, 2D map, and a 2.5D Columbus view with special transition among views with one line of code.

It has resources to visualize dynamic geospatial data. Dynamic scenes from CZML can be animated in Cesium. It can visualize worldwide terrain from multiple sources and draw imagery layers using WMS, TMS, OpenStreetMaps, Bing, Mapbox, Google Earth Enterprise, ArcGIS MapServer, standard image files, ESRI standard and custom tiling schemes (United States Air Force, 2015).

Cesium is prepared to sketch polygons, circles, extents, billboards, ellipsoids, polylines, sensors, icons, labels, custom objects, and it consults material systems to change its appearance. The Cesium' camera is not static, and it has flight paths. For performance, it takes advantage of the low level geometric and rendering routines.

Each graphics layer can be alpha blended with the layers below it, and its brightness, contrast, gamma, hue, and saturation can be dynamically changed. It can draw the atmosphere, sun, stars, and water. It uses batch-ing, culling, and JavaScript and GPU optimizations for performance. It implements precision handling for large view distances (avoiding z fighting) and large world coordinates (avoiding jitter). It supports individual object picking (United States Air Force, 2015).

- "It has camera support including spindle, flight, and free look.
- It supports materials and textures including water, images, and solid colors. It has procedural textures, classic texture modes, and Fabric which is a Javascript Object Notation (JSON) schema for describing and combining materials. For custom drawing it contains a thin abstraction over WebGL that provides built in GLSL uniforms for common transformations. It has built in GLSL functions for ellipsoids, rays, noise, and lighting. It supports shader programs and caching.
- It supports texture and cube maps. It has dynamic texture atlas packing. It supports buffers, vertex arrays, vertexlayout, render states, frame buffers and render buffers.
- It supports geometric routines including view frustum and occlusion culling, vertex cache optimization, polygon triangulation and subdivision, tessellation, and bounding spheres and axis aligned bounding boxes. It has support for many math and geospatial functions including reference frames such as the World Geodetic System of 1984 (WGS84), International Celestial Reference Frame (ICRF), and east-north-up. It supports conversions such as longitude/latitude/height to Cartesian. It has types for cartesian, spherical, and cartographic positions as well as types for matrices and quaternions. It supports Catmull-Rom splines. It does Lagrange, Hermite, spherical linear, and linear interpolation. It supports sun and moon positions. It supports Equidistant Cylindrical and Mercator 2D map projections.
- It supports computations on ellipsoids such as computing surface normal, circles, ellipses, and tangent planes (United States Air Force, 2015)."

Listed below are some Cesium Features Checklist:

- Dynamic Geospatial Visualization
- Draws 3D models using glTF with animations and skins. Convert COLLADA to glTF using the online converter. In the figure 3.1 is depicted an example of a Cesium 3D feature.



Figure 3.1: Cesium 3D Models.

- Cesium draws and styles a wide number of geometries as seen in the Figure 3.2: polylines, bill-boards, labels, points. It draws clamps to terrain, extrudes polygons, polygons with holes, rectangles, circles, and ellipses; boxes, spheres, ellipsoids, and cylinders; corridors, polyline volumes, and



Figure 3.2: Cesium geometries.

- Draws the atmosphere, fog, sun, sun lighting, moon, stars, and water.
- Individual object picking.
- The camera has resources of navigation with mouse and touch handlers as seen in the Figure 3.3, for rotate, zoom, pan with inertia, flights, free look, and terrain collision detection.

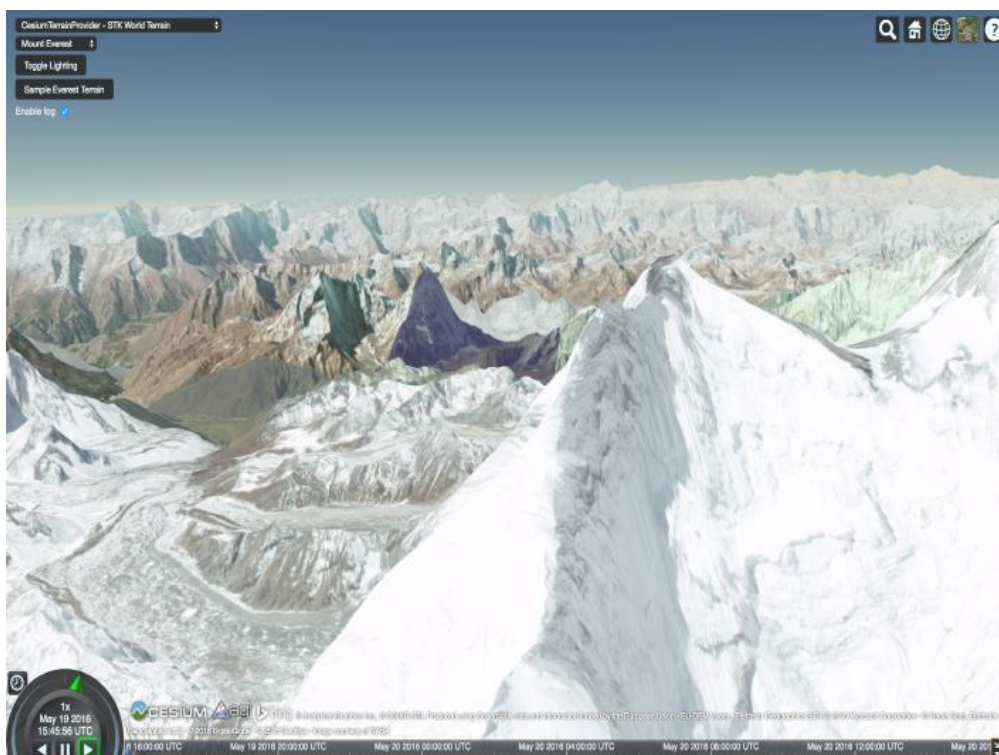


Figure 3.3: Cesium global terrain visualisation

- Batching, culling, and JavaScript and GPU optimizations for performance.
- Precision handling for large view distances (avoiding z- fighting) and large world coordinates (avoiding jitter).
- A 3D globe, 2D map, and Columbus view (2.5D) with the same API. Widgets.
- Timeline and animation widgets for controlling simulation time. -Base layer picker widget for selecting imagery and terrain.
- Selection and info box widgets for highlighting objects and displaying information.

- Geocoder widget for flying to addresses and landmarks.
- Home view widget to fly to the default camera view.
- Scene mode picker widget to morph between 3D, 2D, and Columbus view.
- Full screen widget for toggling full screen mode.
- Navigation help widget for providing mouse and touch instructions.
- Performance watch dog for monitoring the frame rate.
- Inspector widget for advanced graphics debugging.

3.2.2 OpenStreetMap

OpenStreetMap (OSM) is an opensource initiative to create and deliver free geographic information data, such as street maps, maps of locations of interest, and maps with geospatial analyses. The OpenStreetMap is an international foundation that supports the openstreetmap project (openstreetmaps.org, 2016).

This international organization does not control the OpenStreetMap project at all, and they encourage the development and the distribution of OpenStreetMaps (openstreetmaps.org). Openstreetmap is a world map in the web. It was created by a free world organization and submitted under the open license law. This community is composed of geographers whose efforts are to maintain the OpenStreetMap updated with roads data, shopping, schools, cafes, railway stations, and hospitals all over the world.

The OSM provides local knowledge, considering that contributors count with aerial images and GPS devices to confirm the veracity of information and make sure that the map is up to date. The OSM driven community is huge, and it is growing every single day. These contributors are mostly enthusiastic mappers, GIS professionals, and engineers running the OSM servers, humanitarians mapping disasters, affected areas, and many more.

One of the best things about it is that OSM is open data. It gives the freedom to use it anyway. The only thing needed is to give credits to OSM and its contributors. Any changes made need to be under the open license. The figure 3.4 represents the appearance of the openstreetmap main page.

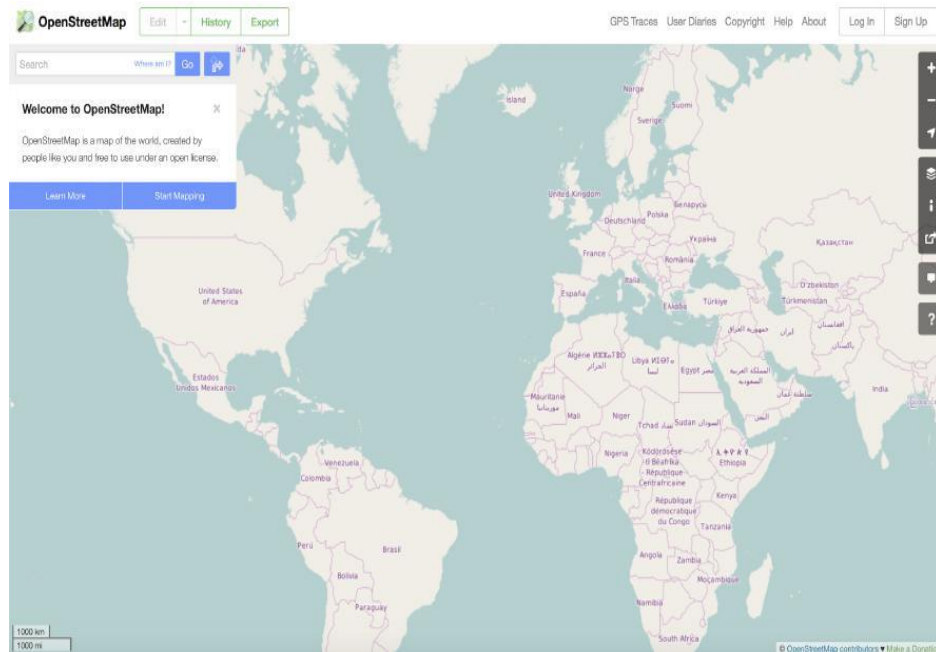


Figure 3.4: Openstreetmaps World imagery.

3.3 Django

Django is a python web framework that makes the web development easy, clean and with a friendly design. It is simple to develop web sites in an easy way without needing to reinvent the wheel. Django is under open source license, without cost as well as freedom to use, copy, modify, distribute and free access to the source code.

Some of the most precious' Django resources are listed below:

- **Django is extremely fast:** It was designed to help web developers finish their application faster and in the most practice way.
- **Very safe:** Django has a high security level. The security is a priority in Django.
- **Scalable:** Django can be trusted when used for a busiest website.
- **Extremely loaded:** It has a large amount of extras that can be used to handle common Web development tasks. Django takes care of user authentication, content administration, site maps, RSS feeds, and many more tasks right out of the box.

- **Incredibly versatile:** Companies, organizations and governments have used Django to build all sorts of things from content management systems to social networks to scientific computing platforms.

3.4 Python

Django uses Python as a main programming language. Python is an high level programming language developed by Van Rossum. Its popularity among of developers is because of its simplicity in time to write the code. It is a very good language for begginers.

Python has a very easy syntax and it does not compel us to complete every point and comma as seen in other languages like C, C++, Java and so on. But is very important to consider the indentation that concerns code alignment that is one of the few aspects to consider when we are coding with Python.

It is not necessary to declare the variable as a specific type to be used (i.e int, float, string) as seen in other programming languages because Python is a dynamic typed language. With that in mind, a variable may have is value changed innumerable times.

Its variety of data type makes Python a very special language. It supports numeric data type, string, tuples, dictionary and so on. It has resources to manipulate les, class and objects as well as a myriad of standard functions to reduce development time.

In the code below a small example of Python coding is presented.

```
>>> valor = 10
>>> while valor > 0:
...     print valor
...     valor-=1
...
10
9
8
7
6
5
4
3
2
1
```

Figure 3.5: Sample Python code

In the code above, there is a simple variable that receives the value 10 and a repetition structure WHILE to repeat and write in the screen the decreasing order of number starting at value 10.

3.5 GeoDjango

GeoDjango is one of the Django module that allows developers to work with geographic web frameworks. It makes the developers' life easy in terms of work with geospatial data in web applications.

The GeoDjango's goal is to make easy the way of the development of Web GIS applications, empowering the spatially enabled data.

Its features include: Django model fields for OGC geometries and raster data, extensions to Django's ORM for querying and manipulating spatial data, loosely-coupled, high-level Python interfaces for GIS geometry and raster operations and data manipulation in different formats, editing geometry fields from the admin [4].

GeoDjango intends to be a world-class geographic Web framework. Its goal is to make it as easy as possible to build GIS Web applications and harness the power of spatially enabled data.

GeoDjango is a very powerful tool for storing and manipulating geographic data using the Django ORM. It provides a simple API to determine distances between two points on a map, find areas of polygons, locate the points within a polygon, and much more. The figure 3.6 shows the logo of geodjango foundation.

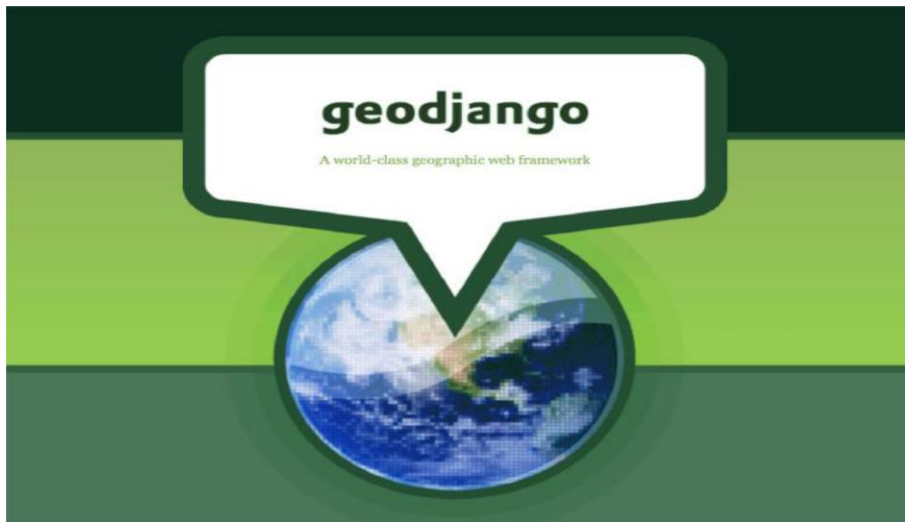


Figure 3.6: GeoDjango

3.6 PostGIS

PostGIS is a spatial database extender for PostgreSQL object-relational database. It adds support for geographic objects allowing location queries to be run in SQL.

PostGIS adds extra types (geometry, geography, raster and others) to the PostgreSQL database. It also adds functions, operators, and index enhancements that apply to these spatial types. These additional functions, operators, index bindings and types augment the power of the core PostgreSQL DBMS, making it a fast, feature-plenty, and robust spatial database management system.

PostGIS series provides:

- Processing and analytic functions for both vector and raster data for splicing, dicing, morphing, reclassifying, and collecting/uniting with the power of SQL;
- Raster map algebra for fine-grained raster processing;
- Spatial reprojection SQL callable functions for both vector and raster data, etc. Commonly, PostGIS is used by Open Source Mapping Servers. Also, conform to OpenGIS Consortium (OGC) mapping standards. Below are some acronyms for standards supported:
 - a. WMS (Web Map Service), WFS (Web Feature Service), WFS-T (Web Feature Service Transactional (ability to edit)), WCS (Web coverage

service), WPS (Web Processing service), WMTS (Web Map Tile service). Data can be extracted from PostGis database using SQL. Below are some vectors and rasters queries that can be used to extract geographic data from database.

Vector format data:

Select * from TABLE where:

- **Contains:** Returns true if and only if no points of B lie in the exterior of A, and at least one point of the interior of B lies in the interior of A.
- **ST Intersect:** Returns TRUE if the Geometries/Geography "spatially intersect" - (share any portion of space) and FALSE if they don't (they are Disjoint). For geography, tolerance is 0.00001 meters (so any points that close are considered to intersect);
- **ST Buffer:** Returns a polygon with buffer distance required; ST Distance: Returns the distance between objects;
- **ST Crosses:** Returns TRUE if the supplied geometries have some, but not all, interior points in common.
- **ST Touches:** Returns TRUE if the geometries have at least one point in common, but their interiors do not intersect. ST Area, ST Length etc...

For raster format data

- **ST NearestValue:** Returns the nearest non-NODATA value of a given band's pixel specified by a columnX and rowY or a geometric point expressed in the same spatial reference coordinate system as the raster.
- **ST Neighborhood:** Returns a 2-D double precision array of the non-NODATA values around a given band's pixel specified by either a columnX and rowY or a geometric point expressed in the same spatial reference coordinate system as the raster.
- **ST SetValue:** Returns modified raster resulting from setting the value of a given band in a given columnX, rowY pixel or the pixels that

intersect a particular geometry. Band numbers start at 1 and assumed to be 1 if not specified.

- **ST PixelAsCentroids:** Returns the centroid (point geometry) for each pixel of a raster band along with the value, the X and the Y raster coordinates of each pixel. The point geometry is the centroid of the area represented by a pixel.
- **ST PixelAsPolygon:** Returns the polygon geometry that bounds the pixel for a particular row and column.
- **ST HasNoBand:** Returns true if there is no band with given band number. If no band number is specified, then band number 1 is assumed.

3.7 RabbitMQ

RabbitMQ is a Robust messaging for applications. It is easy to use and it runs on all major operating systems. It supports a huge number of developer platforms. It is open source and commercially supported [9].

RabbitMQ is a message broker. The principal idea is pretty simple: it accepts and forwards messages. You can think about it as a post office: when you send mail to the post box you're pretty sure that Mr. Postman will eventually deliver the mail to your recipient. Using this metaphor RabbitMQ is a post box, a post office and a postman [9].

The major difference between RabbitMQ and the post office is the fact that it doesn't deal with paper, instead it accepts, stores and forwards binary blobs of data messages. In the Figure 3.7 is depicted the flow of information in RabbitMQ.

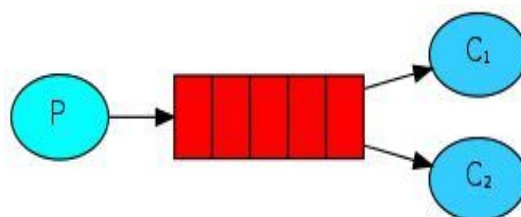


Figure 3.7: RabbitMQ Publisher and Consumer (rabbitmq.com).

3.8 Tornado Web Server

Tornado is a Python web framework and asynchronous networking library, originally developed at FriendFeed. By using non-blocking network I/O, Tornado can scale to tens of thousands of open connections, making it ideal for long polling, WebSockets, and other applications that require a long-lived connection to each user [7].

Tornado was useful for transferring values of sensors and smart door through websockets. I was able to create a WebSocket communication between the server and the clients. For each sensor, it was established a different queue to receive values and sending them to consumers. This asynchronous communication was provided by Tornado.

3.9 WebSockets

WebSocket is a protocol providing full-duplex communication channels over a single TCP connection. WebSocket is designed to be implemented in web browsers and web servers, but it can be used by any client or server application. The WebSocket protocol makes more interaction between a browser and a website possible, facilitating the real-time data transfer from and to the server [8].

WebSockets was the main protocol used to interconnect the clients with the server in a real-time communication in order to avoid databases and Json files.

3.9.1 WattDepot

To collect sensor's data and to show as a graphic, the system is using WattDepot middleware. WattDepot can support any meter that implements an HTTP-based protocol for data transmission. To simplify development, WattDepot provides client libraries for popular languages including Java, Javascript, and Python.

In WattDepot it is possible to create Organizations, and each organization has depositories. In addition, depositories can contain sensors. WattDepot allows to see graphics data from sensors. In the Figure 3.8 it is possible to see a representation of WattDepot measurements in graphics.



Figure 3.8: WattDepot sensor data visualization.

3.10 Hardware

3.10.1 Smart Door

The Smart Door device has three functions: Tracking, identifying and counting.

This project was developed for a NonIntrusive Indoor User Identification using ultrasonic sensors in the College of Technology of the University of Houston.

According to Nacer Kahalil, "User Identification is very important in the context of smart buildings because it makes it possible to know more about your users and enrich your context information and therefore more accurately predict future arrival and departure of users which will therefore be used to save energy and increase comfort by firstly opening the possibility of associating a thermal profile with each user as well as for other applications such as in the context of identifying user behavior in a commercial space such as a mall. There are different solutions in the market such as one using cameras, iris and fingerprints, however they are privacy invasive and therefore cannot be used in public areas. Other solutions include user's active engagement such as ones using wearables or smartphones and identification is dependent on whether the user carries the gadget. The idea of this project is to provide identification with high accuracy but in a nonintrusive manner which means avoid sensing any information that may infringe user's privacy".

The smart door tracks the person who is crossing the door using ultrasonic sensors. It detects if the person is getting in or out of the room or building. In addition, the smart door identifies the person who is crossing the door obtaining the width, the height and the girth in order to compare if this person that got in is the same person that was in the building or is a new one. Finally, it counts the number of people inside of building or rooms. The middleware of the smart door is available to be used by any other device or application. In the Figure 3.9 is depicted the three main functionalities of the Smart Door.

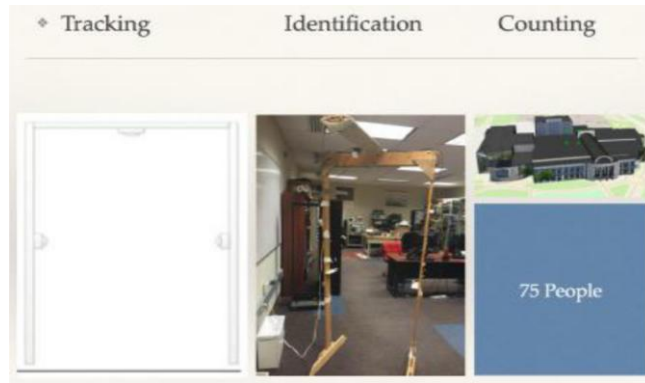


Figure 3.9: Smart Door.

3.10.2 Raspberry Pi and GPIO

The Raspberry Pi is a single-board computer created by the Raspberry Pi Foundation, a charity formed with the primary purpose of reintroducing low-level computer skills to children in the UK. The aim was to rekindle the microcomputer revolution from the 1980s, which produced a whole generation of skilled programmers [10].

The Raspberry Pi has pins along the edge of the board. These pins are a physical interface between the Pi and the outside world. At the simplest level, you can think of them as switches that you can turn on or off (input) or that the Pi can turn on or off (output). Seventeen of the 26 pins are GPIO pins; the others are power or ground pins [11]. Through these pins, sensors, cameras and others small devices can be connected and controlled by the Raspberry Pi. Sensors can measure the surrounded environment and send the values to a server. In the other hand, cameras can be connected to the Raspberry Pi and send images in real-time to the server as well. An image of RaspberryPi (Figure 3.10) and GPIO (Figure 3.11) is shown below.



Figure 3.10: Raspberry PI.

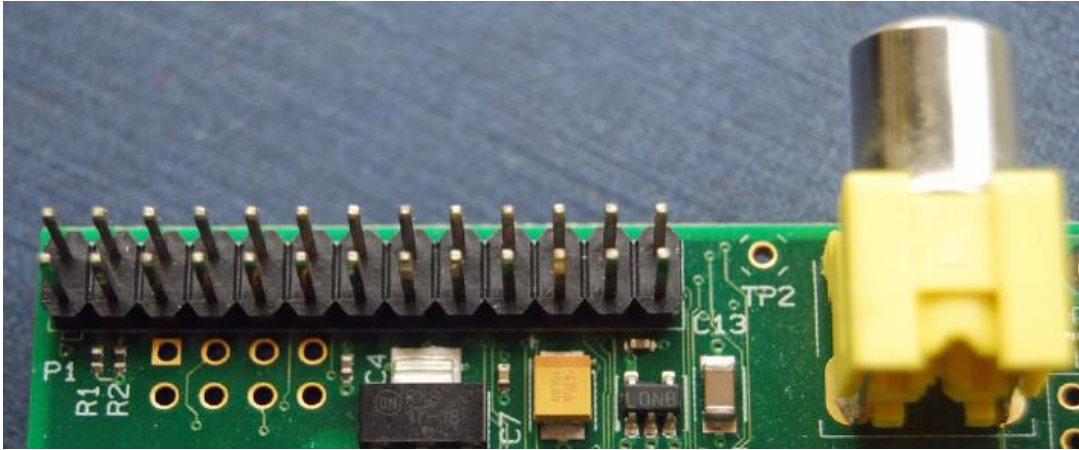


Figure 3.11: Raspberry Pi GPIO

Nowadays the Raspberry Pi is used in many different projects for Internet of things, from building automation to urban management.

3.10.3 Wireless Sensors Network

Wireless sensor network is an equipment package system that consists of one or more automatic sensors, and communication systems. Each sensor point in the sensor network is equipped with a radio transceiver or micro-controller and energy source.

Advances in sensor technology and wireless communications have made networked micro-sensors possible, where each sensor individually senses the environment but collaboratively achieve complex information gathering and dissemination tasks. A network of sensor nodes spread across a field has the capacity to provide temporal and spatial data regarding the properties of the environment.

Wireless sensor networks may be used to monitor the condition of the physical environment, such as: temperature, humidity, wind speed, precipitation, pressure, light, motion, sound, vibration, color changes, CO₂, etc., which is usually in real time mode. These sensed data are physically sent to the base station for compilation. The figure 3.12 shows the concept of smart cities.



Figure 3.12: Wireless Sensor Network [12].

Chapter 4

System Architecture

The goal of this dissertation is to develop a 3D WebGIS capable of saving, organizing, and geo-spatializing information from all over a campus and respective facilities. In addition, this WebGIS will also process data and manipulate devices deployed in smart buildings and smart cities, connected to the internet in real-time, using concepts and tools of Internet of Things (IoT) and provide one important information to users that is the number of people inside of buildings using the smart door device.

4.1 System Objectives

The following objectives were defined in order to develop a functional and well designed WebGIS that incorporates tools of the Internet of Things.

Objective 1: The goal of the system is to present a 3D map where the user can click in a 3D building, choosing the floor, opening the room and pick a device. In addition, the user can also see different devices around the city in the map and he/she can click in the device and open a menu to perform some task.

Objective 2: The second objective is to establish different tasks for different devices. If the user clicks on device, the system needs to know the type of device and show a menu with functionalities for this specific device. For instance, if the clicked device is a camera, the system needs to show a menu for the camera with: record video, take pictures, video streaming, night view and so on.

If the clicked device is, for example, an air conditioning, it will show a menu with tasks for air conditioning, like change the temperature. If it is a temperature sensor the user may be able to see the actual temperature of the room.

Objective 3: The system needs to show a page for alerts. The administrator of the system will be able to establish range values of alert. For instance, if the administrator

wants to be alerted when the temperature value of a sensor is above 32 degrees, he/she can configure it in the system and automatically, all the sensors that the value exceed 32 degrees will appears in this alert page.

Objective 4: The system needs to show the number of people inside of the building and the average time that a person can be attended in case of a public building service.

4.2 General Architecture of the System

The general architecture of the system is depicted in the figure 4.1. Each component of the software is described as well as the hardware and the database structure.

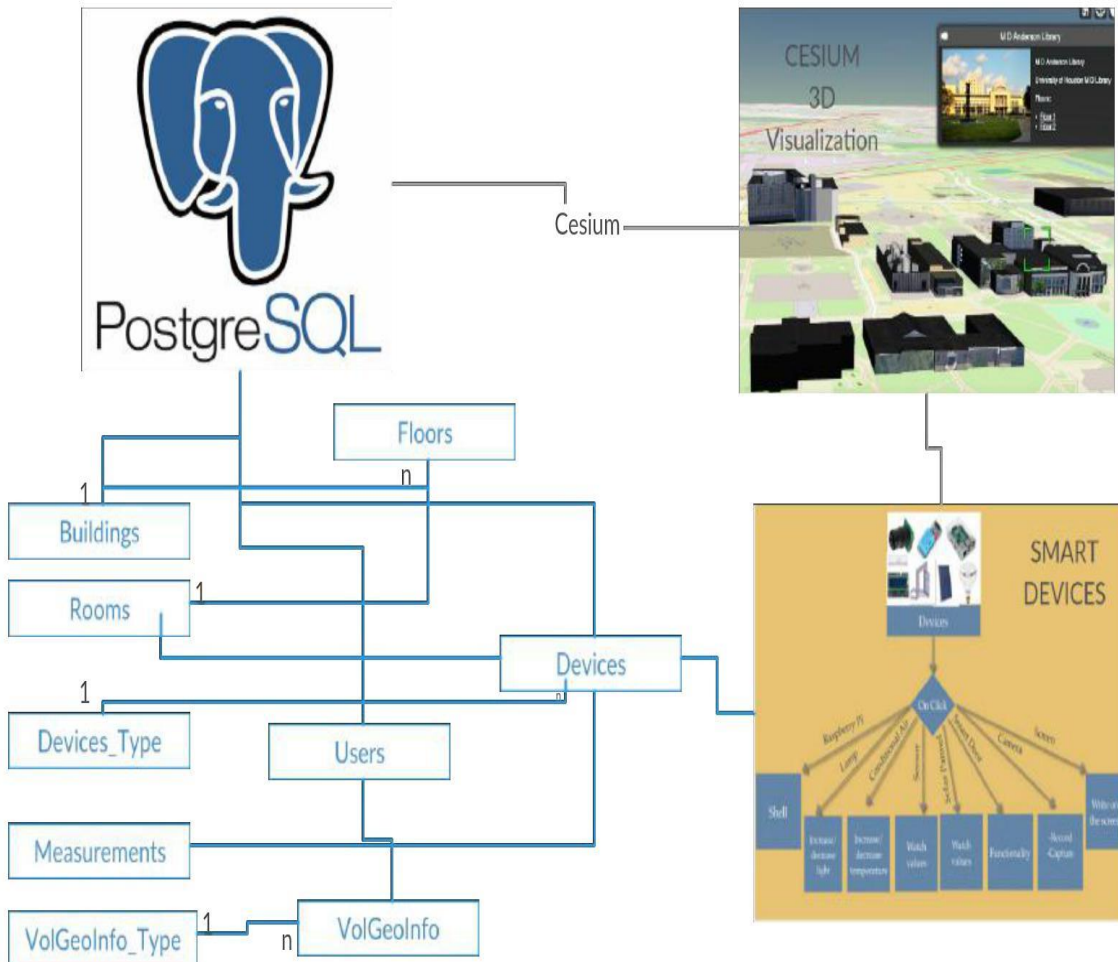


Figure 4.1: System architecture.

4.2.1 Database logic model

Each table on PostgreSQL has one specific task as we can see in the Figure 4.2 .

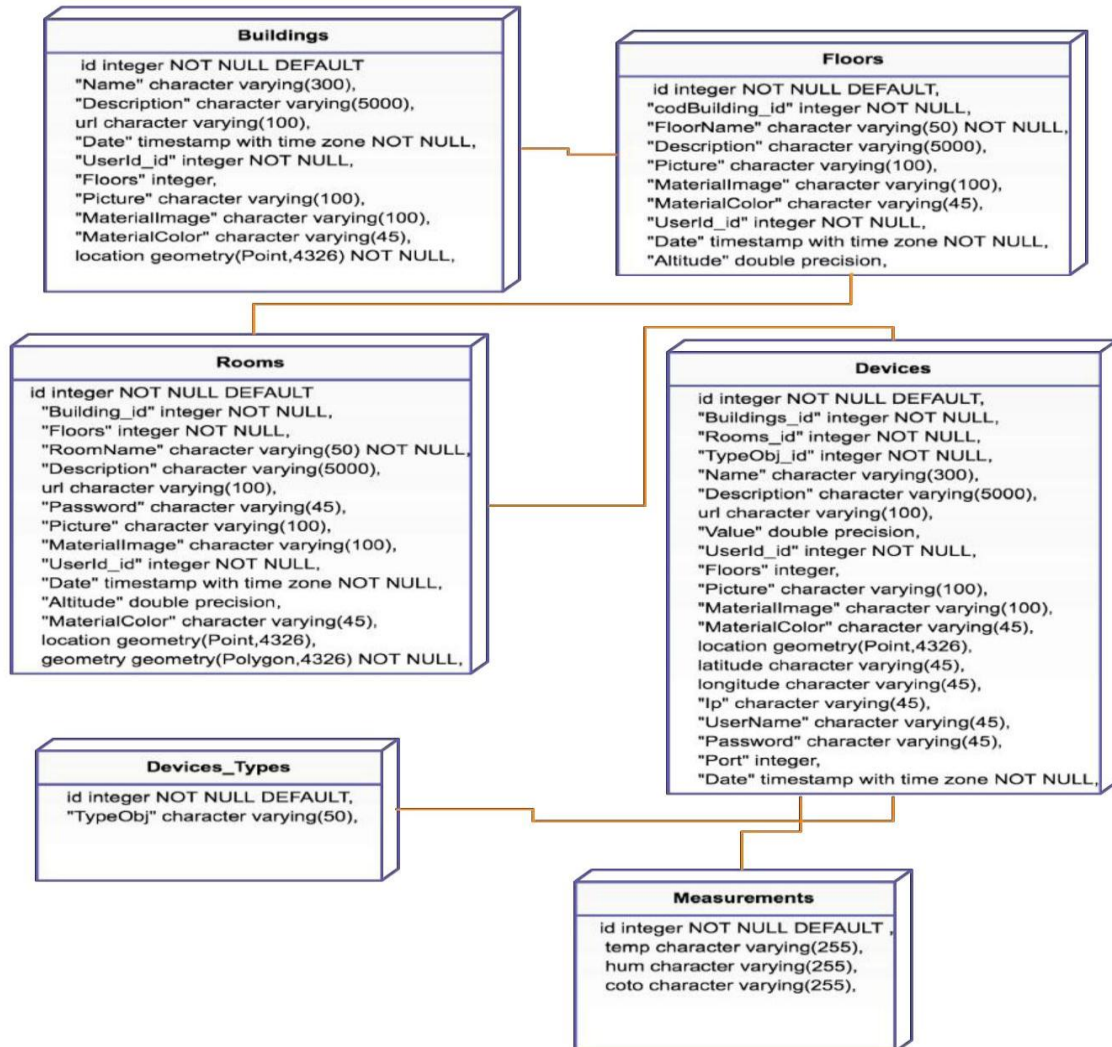


Figure 4.2: Logic Model.

The table buildings is responsible for the building settings and location. It saves the latitude and longitude for each building as well as the description of the building: the name, the number of floors, the picture to show when the building is clicked, the user that uploaded the building and the id of the building. When the administrator wants to upload a new 3D Building .gltf through Geodjango, it is mandatory to choose in the map where the building will appears.

The table floors saves the floors coordinates of each building. Every floors belongs to only one building. Some important fields to consider are the code of the building, the name and description of the floor, the color of the floor and the altitude of the floor.

Every room belongs to one floor. Each room has the id of the floor, the name of the room, the altitude, the location and the color of the room. The color is important to differentiate between two or more rooms in each floor.

The table devices saves all the locations and settings of each device. It is important to considerate that every device is of one specific type. It will help in the moment to create menus to see the devices categorized.

The system installation is described in the appendix A. On the appendix B is presented an explanation of how to install a RPI camera in a raspberry PI.

In the wireless sensor network, there are some measurements that are important to store in the database. It is done in order of having a history about the last temperature measured.

4.2.2 Cesium-3D building modelling

The Cesium 3D visualization is responsible for representing the 3 buildings in the map. It allows clicks in buildings and view features in different perspectives.

3D buildings for Cesium are made in a 3D modeling computers program. For this dissertation, 3D buildings were made in Google Sketchup program and saved with the extension .dae as we can see in the figure 4.3 .



Figure 4.3: 3D building modeling - Google SketchUp

After the 3D design as depicted in the flow chart, figure 4.5, the building is converted to a .glTF file as is seen in the Figure 4.4. This is one of the extensions that Cesium can read and display in a map. This conversion is made in the address <http://cesiumjs.org/convertmodel.html>.



Figure 4.4: 3D building .dae conversion to .glTF.

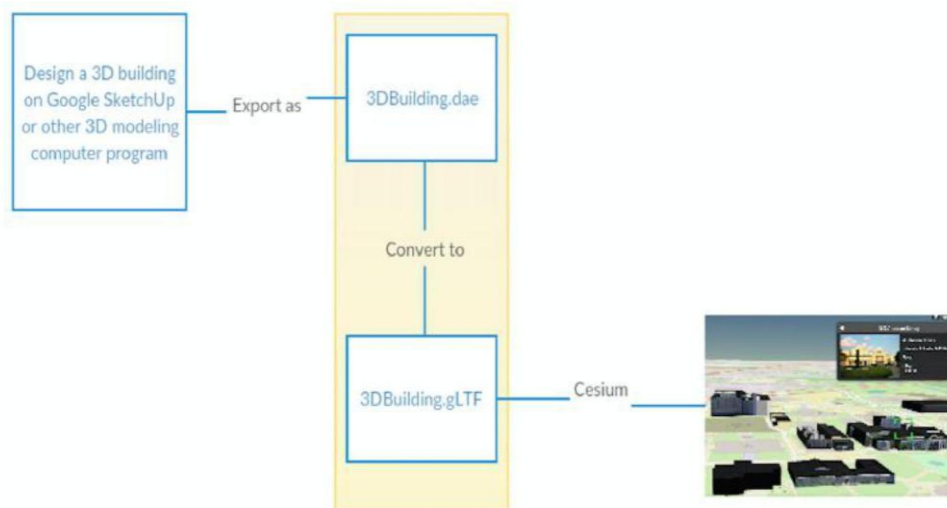


Figure 4.5: Cesium Flow Chart.

4.2.3 The Smart Door Middleware

The smart door has a middleware that sends all the values in real-time to a queue message program. The queue message program used in this project is the RabbitMQ. In the RabbitMQ there are some channels and queues where the Smart Door sends the values in real-time.

For example in the figure 4.6, there is a queue named as doorInfo (RabbitMQ). In this queue is possible to observe the id of the building, the number of people inside of the building, the time that the last person got in or out of the building and the average time of movement in the building.



Figure 4.6: Smart Door Middleware.

By the end, the WebGIS will be one of the consumers of the RabbitMQ. It will display the number of people inside of buildings in the map.

Chapter 5

WebGIS Portal: Practice case

In order to achieve the goal of having a geographic information system in the web capable of organizing, saving and geospatializing information for the campus, Django-Framework was used as the main technology in order to fulfill this objective. Considering its huge number of resources, security and capabilities to make the web development easy and friendly, Django-Framework was picked as a main web development tool to reach the goal.

Django-Framework combined with Cesium was used to build the entire web-portal and all its pages.

5.1 Administrator

The first step to have the system working, is to configure and upload every building and smart devices into the database in order to be shown in the map. To do this, it is necessary be in the administrative are of the Django. The administrator needs to access <http://localhost/admin>. First he needs to login as seen in the figure 5.1. The figure 5.2 shows the initial page of the Django admin area.

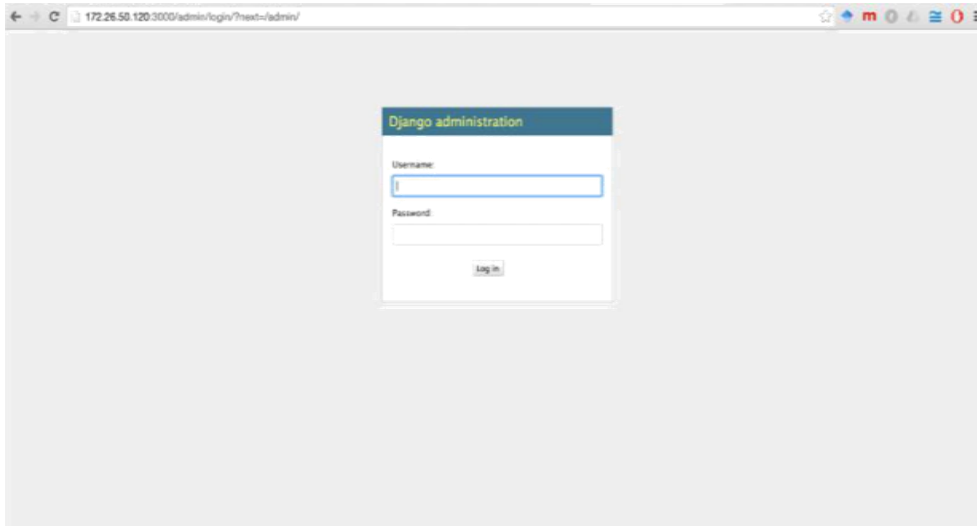


Figure 5.1: Django Admin

Once the administrator logged in, he will be able to upload buildings and configure devices as it is shown in the Figure 5.2.

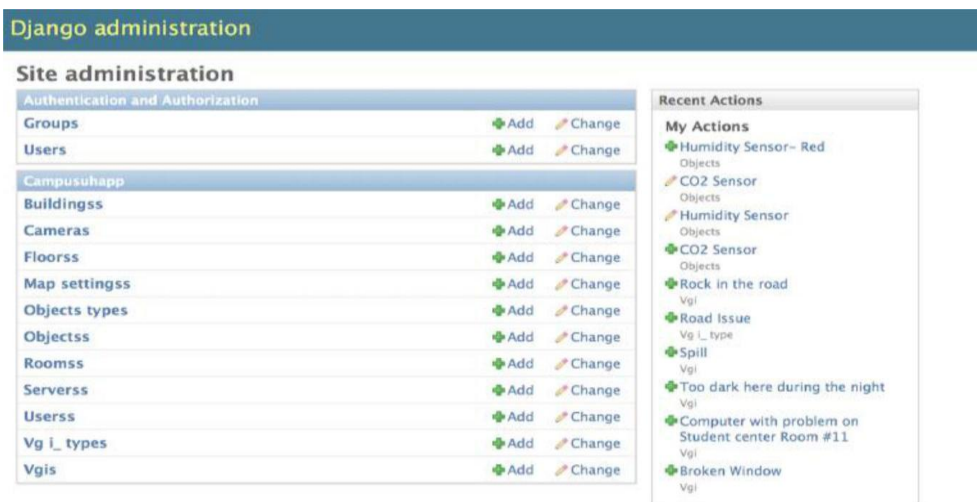


Figure 5.2: Django Admin Main Page

5.1.1 Add buildings in the map

Buildings are just .GLTF files that are converted from .DAE format. The GLTF files are the files to represent a 3D building in Cesium. 3D buildings can be made in Blender, Google SketchUp, AutoCad, Maya and so on. On the other hand, a JavaScript code used by GeoDjango does its job putting the 3D building in the right location in the map.

To add buildings in the website, the administrator needs to click in the button Add, and he/she will be redirected to the page where the information of the building is filled as well as the .gltf file and the picture that will be associated with the building. The figure 5.3 shows the required fields for adding a building.

The screenshot shows the Django administration interface for adding a building. The form is titled "Change buildings" and is located under the path "Home > Campusuhapp > Buildings > M D Anderson Library". The form contains the following fields and their values:

- Name:** M D Anderson Library (1)
- Description:** Library University of Houston (2)
- Url:** Currently: fotos/2016/01/16/Group_156.gltf Clear (3)
Change: Choose File no file selected
- UserId:** uh (4)
- Floors:** 2 (5)
- Picture:** Currently: fotos/2016/01/16/M.D._Anderson_Library.JPG Clear (6)
Change: Choose File no file selected
- MaterialImage:** Choose File no file selected (7)
- Location:** University of Houston,Tx (8)
Press "Tab" to refresh the map

The "Location" field is accompanied by a map showing the location of the building. The map is labeled "of Houston" and shows the "M.D. Anderson Library" building. Other buildings shown include "Ezekiel W. Cullen" and "Cullen Performance Hall". A red number 9 is overlaid on the map area.

Figure 5.3: Uploading buildings

The fields in the Figure 5.3 are for:

1. The first element is the name of the building.
2. The second field is for description of the building. The administrator can introduce more information about the building.
3. The third field is related to the .gLTF of the building: the 3D model.
4. The fourth field is the information about the user who is inserting data in the WebGIS.
5. The fifth field is the information about the number of floors that the building has.
6. The sixth field is for upload one picture of the building.
7. The seventh field is to upload an image that will be the base of the floors. Instead of using colors for the floors, Cesium allows to insert an image.
8. The eighth field is a text box to search for a place in the map where the 3D building will be shown.
9. The ninth element is the map to help positioning the building.



5.1.2 Add Rooms

To add a room in a building, the administrator needs to click in the button Add of the table room. The figure 5.4 shows the fields that the administrator needs to fill in order to have geometries of rooms in each building.

Django administration

Home > Campusuhapp > Roomss > Add rooms

Add rooms

Building:  

Floors:

RoomName:



Description:

Url: no file selected

Password:

Picture: no file selected

MaterialImage: no file selected


UserId:  

Altitude:

MaterialColor:

Press "Tab" to refresh the map.

Location:



Geometry:




Figure 5.4: Rooms geometries.

Explaining the figure 5.4:

1. In the first field, the administrator needs to choose the building that the room will be part.
2. The second field is the number of the floor. It is not mandatory to have floors registered. The number of floors inserted when the building was registered will be sufficient for the room registration.
3. The third and fourth fields are the name and description of the room.

4. The field altitude is the altitude of the room. When the room is viewed in 3D model it needs to have the altitude; otherwise all rooms will appear in the map as the floor room.
5. The location is the position of the room in the map. The Geodjango is configured to be in the left bottom corner of the room.
6. The geometry will represent the appearance of the room. Square, triangular and so on.

5.1.3 Add Devices

Devices (Objects) are the virtual representation of smart devices as cameras, sensors, air conditioning, lights, doors, solar panels, screens, computers and so on. These devices need to be registered before being used. To upload devices and place them in different buildings and locations of the street, the administrator need to be logged in the administrative area of the Django and click in the button Add of the objects table. In the figure 5.5 is depicted the form of inserting devices.

The screenshot shows a web interface for managing objects. The breadcrumb trail is "Home > Campusuhapp > Objectss > Humidity Sensor- Red". The main heading is "Change objects".

The form contains the following fields and elements, with red numbers 1 through 11 indicating specific parts:

- Buildings:** A dropdown menu showing "Central Power Plant" with a plus icon and a red number 1.
- Rooms:** A dropdown menu showing "Room 1 - C. Tech Second Floor" with a plus icon and a red number 2.
- TypeObj:** A dropdown menu showing "Sensor" with a plus icon and a red number 3.
- Name:** A text input field containing "Humidity Sensor- Red" with a red number 4.
- Description:** A text input field containing "Sensor of Humidity" with a red number 5.
- Url:** A text input field containing "Currently: fotos/2016/01/22/ob.png" and a "Clear" button, with a red number 6.
- Value:** A text input field containing "13.0" with a red number 7.
- UserId:** A dropdown menu showing "uh" with a plus icon and a red number 7.
- Floors:** An empty text input field.
- Picture:** A text input field containing "Currently: fotos/2016/01/22/ob_4gHjOiz.png" and a "Clear" button, with a red number 8.
- MaterialImage:** A text input field containing "Choose File" and "no file selected" with a red number 9.
- Location:** A text input field containing "University of Houston,Houston, Texas" with a red number 10. Below it is a map showing the location with a red number 11.

Figure 5.5: Register Devices.

1. The first field is to choose the building where the object is in.
2. The second field is the room of the building. In this field will only appears the rooms that belongs to the building selected in the first field.
3. The third field is to choose the type of device that is being inserted (camera, sensor, air conditioning, light, door, solar panel, screen, computer).
4. This field is to write the name of the device.
5. This field is for the description of the device.
6. The url is to pick a 3D model of the device, if, exist.
7. The seventh field is to record the user who is inserting devices in the database.
8. The eighth field is the picture that will appear when the device will be clicked.
9. Is the material image of the device. This image will be shown when there is not a 3D model of the device.
10. The tenth and eleventh elements are to choose the location in the map where the object will appear. This location is the latitude and longitude of the device inside of the room.

When the device is a computer, the system will ask to insert the IP of the computer, the user name and password in order to have ssh access through the browser.

5.1.4 Main Page

Django-Framework combined with Cesium were used to build the entire web-portal and all its pages. In the figure 5.6, the main page of the website can be seen.

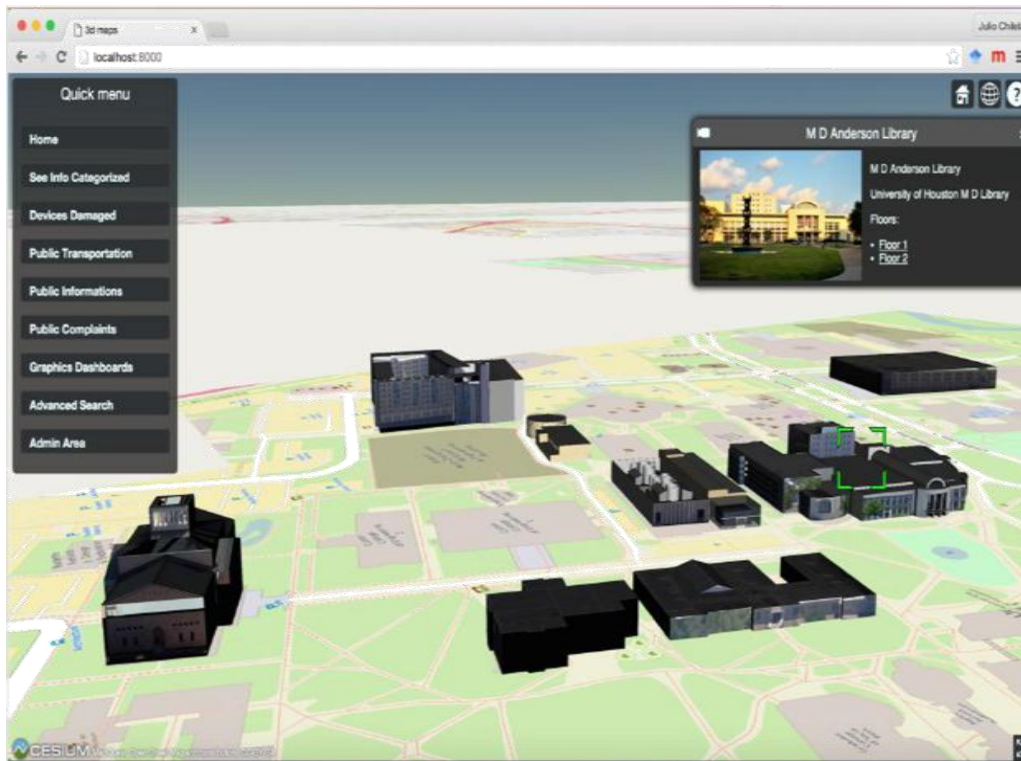


Figure 5.6: Main Page

As seen in the figure 5.6, all 3D buildings are displayed in the main web page.

If the administrator clicks in one specific building the information configured and the building picture is depicted in the right side of the screen as well as the floors. The administrator has links to the floors. For instance, if the floor 1 is clicked the map will display the rooms inside of this floor. When the rooms are opened, every device inside of that room is shown.

The figure5.7 displays a better view of the data.

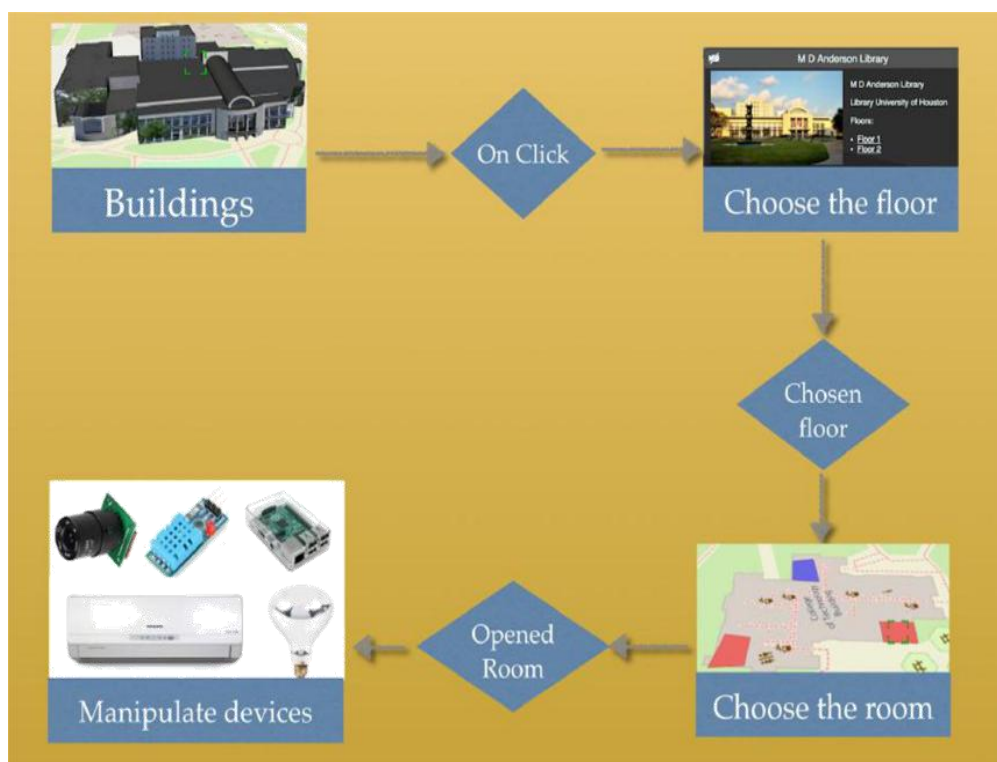


Figure 5.7: Main Page flow data

The quick menu in the left side of the figure 5.6 links the main page to different pages. The figure 5.8 shows the map of these links.

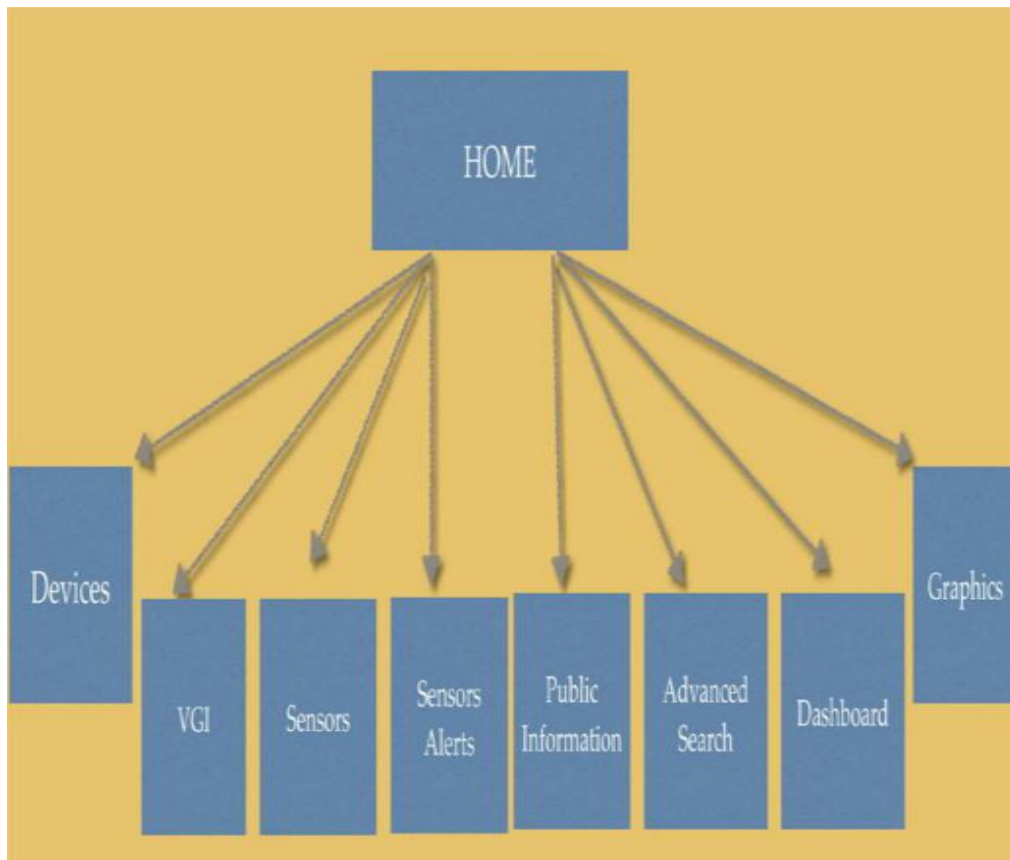


Figure 5.8: Main Page links.

As can be seen in the figure 5.8, the main page has seven links that represent all functionalities that the website affords.

- The link Devices connects the user with all devices around the campus and its facilities. The information is organized in such a way that the user will be able to easily find the device that he is looking for. All devices in the map are represented in layers. There will only appear in the map devices of the same type. If the user selects the layer Cameras, it means that all the cameras will be shown in the map.

In order to find a specific camera, the user has to click on the Advanced Search link (figure 5.9) to make possible the search for a specific location and a specific device's name.

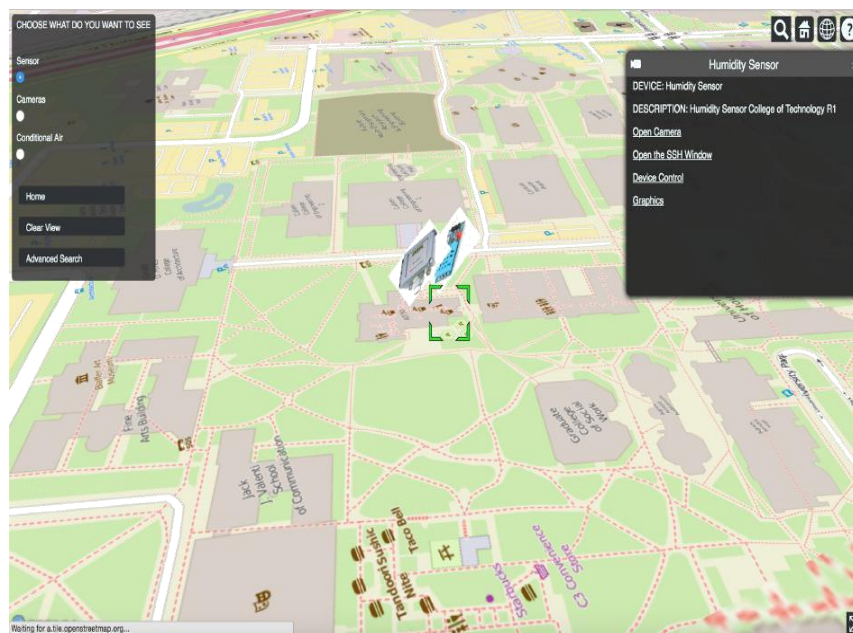


Figure 5.9: Categorized Devices.

- The link VGI (Volunteer Geographic Information) represents the information coming from a mobile application. Through the mobile application, users will inform the smart campus administrators about issues occurring on the campus. Users may report damaged properties, issues concerning to the IT department, hazards, roads and request room temperature or light changes. The Figure 5.10 shows the appearance of the page. Through this information that comes from volunteers, the main campus administrators can have control of all the campus and make decisions to fix properties and inform the community about the next steps to solve specific problems.

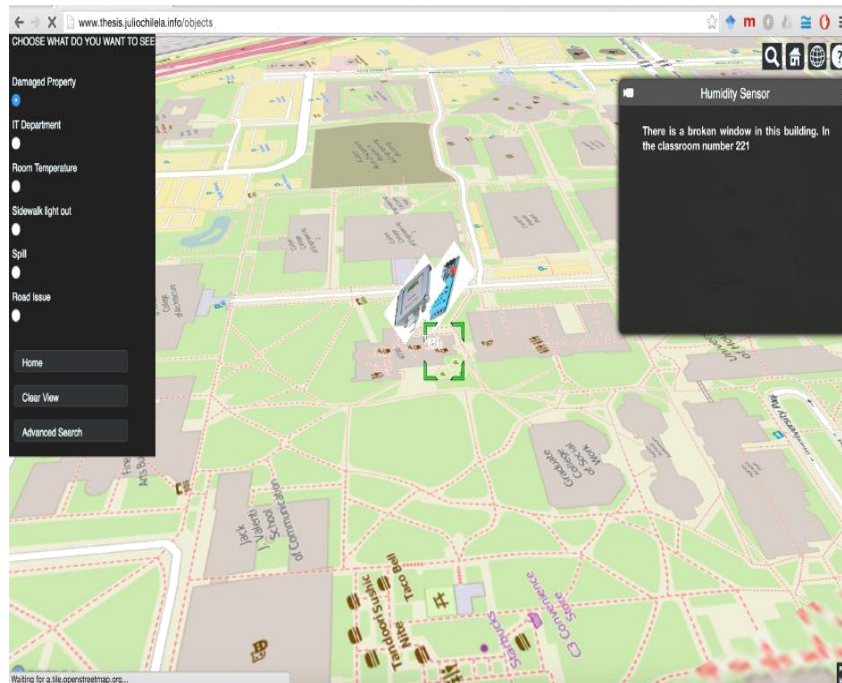


Figure 5.10: Volunteer Geographic Information.

- The link Sensors enables users to check the real-time sensors' data. The user can visualize the sensors' data in some specific location. It helps in decision making.
- Sensors alerts as seen in figure 5.11, represents all sensors according to a specific value of warning. All the sensors established as critical when the value exceeds some range, will appear as red in the map. On other hand, if the value is very close to the critical value, it will appear as yellow, informing that campus administrators they must be on alert. It helps at that moment to analyze and compare the information coming from users through VGI and the information coming from the sensors.

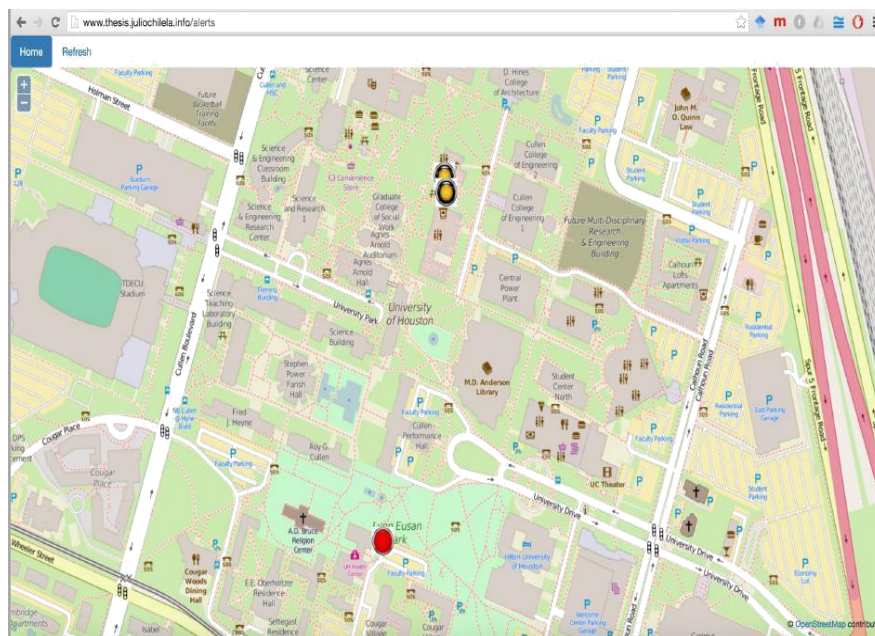


Figure 5.11: Sensors alerts.

- The link Public Information is the link that connects the campus administrators with all LCD screens on the campus. The campus administrators will be able to redirect a specific message to a specific screen or LCD on campus. Public Transportation link is to verify in real-time the position of the bus in the map. The point representing the bus will be moving around the map.
- The Dashboard link is a specific link for electrical management. In this link the smart campus administrators may cut the power supply to one specific building. In addition, main campus administrators will have the option to visualize the quantity of energy saved in the power plant and see the energy being sent from solar panels to the main power plant.
- The link Graphics is mainly used to monitor sensors' values and solar panels' values in graphics in different ways. Charts and graphics are available in this link as seen in the figure 5.12.

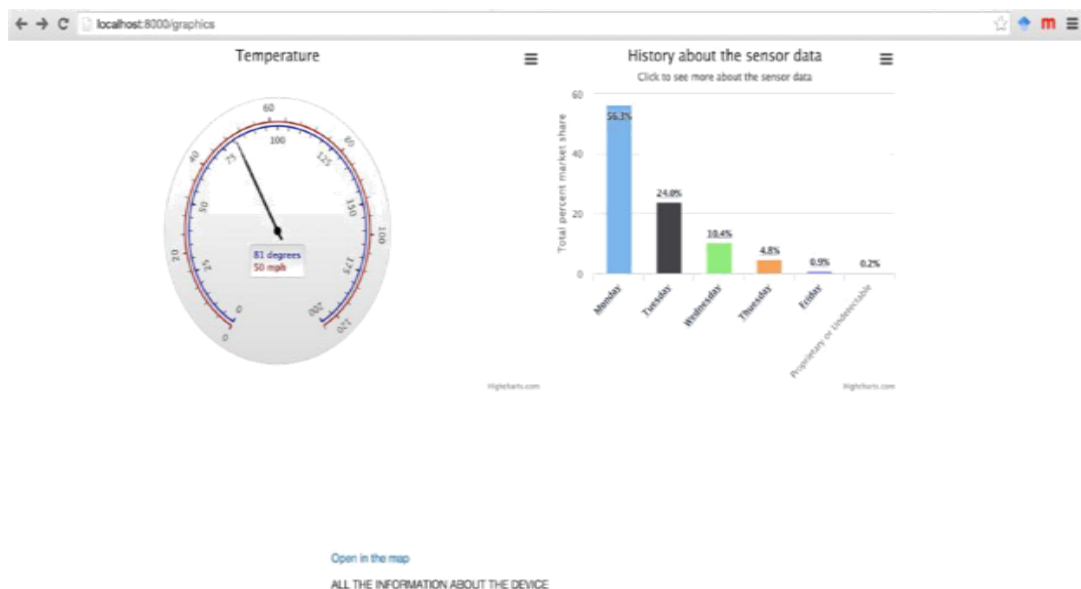


Figure 5.12: Graphics

Devices assignments

Smart devices deployed around the campus and facilities can be manipulated directly through the map. When the user clicks on the device, the map shows a menu for performing tasks.

If the clicked device is:

1. Raspberry pi: The user will have available a SHELL to access the selected computer through SSH.
2. Lamp: Users can increase and decrease the light intensity.
3. Air Conditioning: The function for changing room temperatures will be available.
4. Sensor: The user can watch values in real-time.
5. Solar Panel: Users can watch the quantity of energy being collected by the referred solar panel at a specific time.
6. Smart Door: Users can see the functionalities of the object.
7. Camera: When the administrator access the camera, They can record a video, take pictures, change the configuration of the camera, as well as put the camera in motion mode detection and download videos previously captured by the camera.
8. Screens: Smart campus has screens for public information. Through the map, the administrators will be able to write a message to specific screen, in specific location.

Figure 5.13 represents the flow chart of devices, mainly, the behavior of each device when clicked.

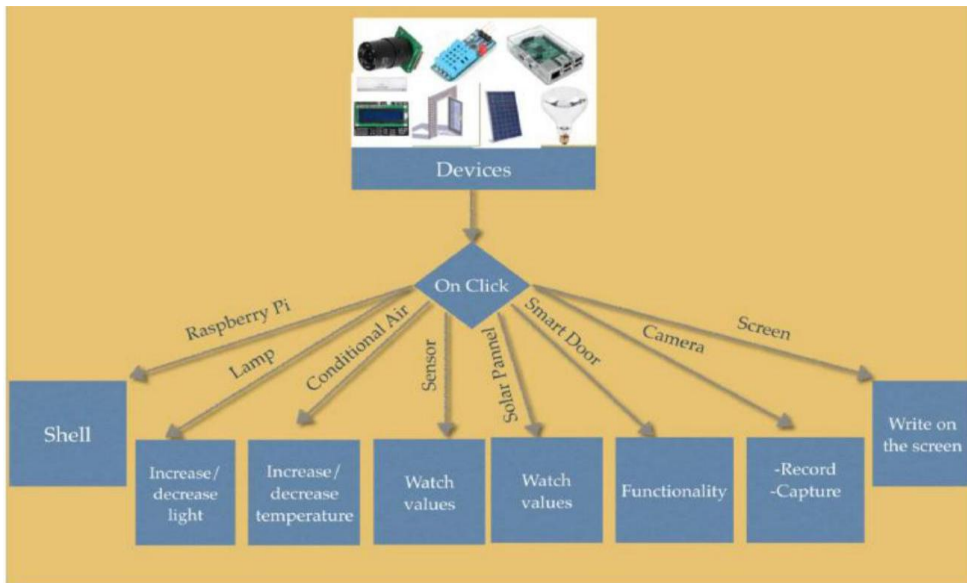


Figure 5.13: Smart Devices

All the actions that is possible to perform in a device will appears in the right side of the screen, whenever the sensor is clicked.

In the figure5.14 the maps shows a clicked computer Raspberry Pi and the options of tasks that is possible to perform in this Device. This device is equipped with a small camera and it is also possible to login to the computer by SSH to configure and change settings of the device.

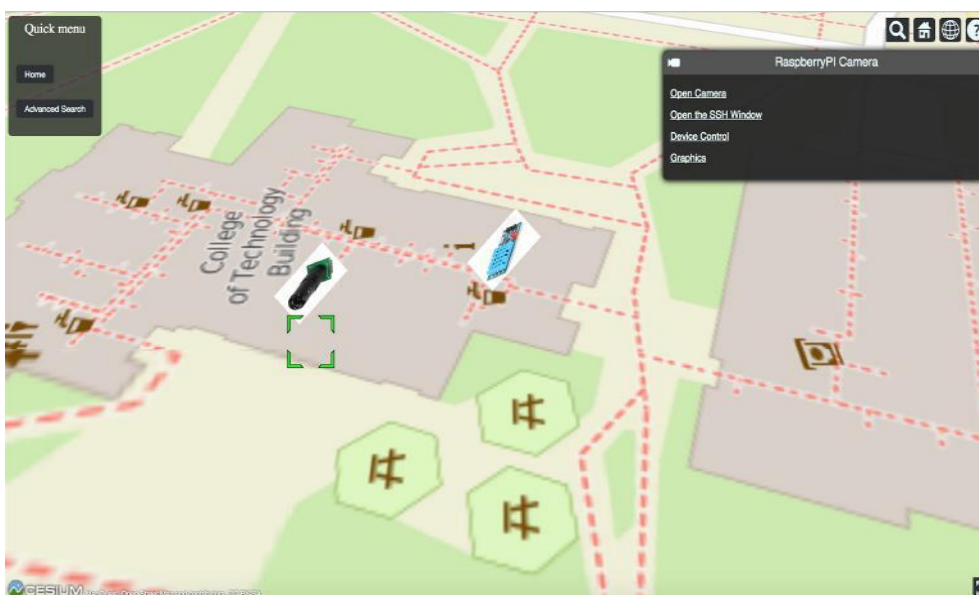


Figure 5.14: RPI Camera.

Chapter 6

Tests and Validation

The system was submitted to tests by students and employees of the University of Houston. Tests of white box were made and contributions about the behavior of the application were collected.

For example, tests of camera functionalities, air conditioning control, smart lamps control and temperature and humidity requests were made.

Faculty contributions

The two main contributions coming from the faculty were:

1. They said that the system is not fast enough when loading the main page. Every time that the user hit a link from a secondary page to load the main page, the system was taking so long. The suggestion was that once the map of the main page is loaded, it should be saved in memory cache in order that when the main page is called again, it does not need to load the map from the cesium server but from the memory cache to make the system fast.

It made me research about memory cache using Django-framework. And this contribution was solved after a few days.

2. They also claimed that the floors of the rooms does not had an altitude. If you try to render the map trying to see the floors in horizontal, the floor does not appear. They suggest to add an altitude to the floors and rooms so that, when the user renders the map to see in a 3D view, he should see that the room has an altitude.

Student contributions

Students also did tests of white box. They figured out that instead of see sensors in a 3D map, it should be good to see it in a 2D map.

This contribution was also solved a few days after the repair.

The mobile app was also unable to attach the picture from the device. They also said that every room needs to appear as an option in one choice box in the mobile app.

The option to move the pin of the map is unavailable. Sometimes the map gives different latitude and longitude of the place where the user is. It is important to move the pin to the specific location in order to report about the accurate building.

Sometimes when the map is loaded in a browser different of Google chrome the website freezes.

The goals of the project were achieved. The system can display buildings in a 3D map, manipulate devices, display sensors in alert, shows the number of people inside of buildings, shows the average time that a person can spend in building with special services and displays volunteer geographic information coming from a mobile app.

All the contributions were to increase the perform of the system and to make it more faster than what it looked. The four goals of the project were achieved.

Chapter 7

Conclusion

Managing smart campus and its facilities has been a rough task nowadays, considering the number of smart devices that are present on the smart campus and facilities, as well as the amount of information and data that those smart devices are sending to the main station every second.

WebGIS is a powerful tool for helping management, since it gives to the manager information and data geolocated and capabilities that the Internet has in terms of make the information accessible to everyone and everywhere in the world. WebGIS also can be integrated with mobile applications in order to communicate more simply with users and citizens. Cities managed and organized with WebGIS may be even more organized.

The aim of the project was reached. The system can display buildings in a 3D map, manipulate devices, display sensors in alert page, shows the number of people inside of buildings, shows the average time that a person can spend in building with special services and display volunteer geographic information coming from a mobile app.

7.1 Future work

The main application can have a mobile application that helps users inter-act with the main campus administration and to notify the administrators about ongoing issues on the campus. This application needs to be improved, because it is only sending the data without the picture of the issue. This application can have resources for taking picture about hazards, malfunctioning devices, roads and requests for temperature and lighting changes in some rooms, as well as to have access to sensors' data and public transportation. It need to be improved as well as rethink its appearance.

The main page needs to be improved, in order to make available public transportation information. Also, add in the main page a link to a page to see damaged or malfunctioning IoT devices. This page should be responsible to show every device that is not collecting or sending data to the server in order to advert administrators to repair them.

References

1. Patrik Ottoson (2007): Three Dimensional Vizualisation on the Internet
2. United States Air Force, (2015): Cesium 3D Visualization
3. Chicagodjango (2015): <https://www.chicagodjango.com/blog/geo-django-quickstart/>
[Access date: 08-09-2015]
4. W3 (2016): Cascading Style Sheets Website <https://www.w3.org/Style/CSS/>
[Access date: 07-09-2015]
5. Bootstrap(2015): www.bootstrap.com
[Access date: 07-09-2015]
6. Tornado(2016): Web Server <http://www.tornadoweb.org/en/stable/>
[Access date: 05-19-2016]
7. WebSockets (2016): <https://en.wikipedia.org/wiki/WebSocket>
[Access date: 05-19-2016]
8. RabbitMQ (2016): <https://www.rabbitmq.com>
[Access date: 05-19-2016]
9. Cox, T (2014): Raspberry Pi Cookbook for Python Programmers
10. Raspberry Pi GPIO:
<https://www.raspberrypi.org/documentation/usage/gpio/>
Access date: 05-19-2016
11. Microcontrollerslab (2016):
<http://microcontrollerslab.com/wireless-sensor-networks-wsn-applications/>
[Access date: 05-19-2016]
12. Honda, K., Shrestha, A., Witayangkurn, A., Chinnachodteeranun, R., Shimamura, H., (2009): A Field-servers and Sensor Service Grid as Real-time Monitoring Infrastructure for Ubiquitous Sensor Networks
13. Rahman, M (2012): Low Cost Configuration of Data Acquisition System for co₂ wireless sensor Network
14. Saumini, K. (2012): An integrated Framework in Geographic Information System using Wireless Sensor Network"

15. Felicísimo, A. M., (2014): "Design of a WSN for the sampling of environmental Variability in Complex Terrain"
16. Laituri, M., Kodrich, K.,(2008): Online Disaster Response Community: People as Sensors of High Magnitude Disasters Using Internet GIS
17. Geoffery Meaden, José Aguilar(2013): Advances in geographic information systems and remote sensing for fisheries and aquaculture.
18. ESRI (2016):
http://esripress.esri.com/storage/esripress/images/283/185_gtkweb_ed1p1_sample.pdf
[Access date: 05-09-2016]
19. Gaston, Joubert , Montesinos (2015): 3D web visualization for real-time maintenance of smart building

APPENDICES

APPENDIX A

Instalation of the architecture

A.1 Required hardware

A dedicated computer to host the most variates of services required to make the sistem works is necessary. The Internet connection must be avaiable in all devices through Wi-Fi or wire.

A.2 Installation of the software

A.2.1 Operating System

The server should run Ubuntu Server 14.04 or later.

Installing resources in a ubuntu server machine

1. Execute: `sudo apt-get update`

This command will update all the packages in the system to an earlyversion

2. Execute: `sudo apt-get install python-pip`

This command will install pip. Pip is a python package manager

3. Execute: `sudo pip install django==1.8`

This command installs Django in your machine

4. Execute: `sudo pip install pillow`

This command installs pillow.

Pillow supports the most commom image formats.

- 1.1.5 Execute: `sudo apt-get install apache2 libapache2-mod-wsgi`

This command installs the webserver apache and the library that supports python code

1.1.6 Execute: `sudo apt-get install python-gdal`

This command installs the gdal library

1.1.7 Execute: `sudo apt-get install postgresql-9.3-postgis-2.1` {f

This command is to install postgres and postgis in the server

1.1.8 Execute: `sudo apt-get install python-psycopg2`

This command installs the connector between postgres and python

1.1.9 Execute: `sudo -u postgres createdb postgres`

Add the following lines:

Enable PostGIS (includes raster)

```
CREATE EXTENSION postgis;
```

Enable Topology

```
CREATE EXTENSION postgis topology;
```

APPENDIX B

Raspberry pi Camera installation

1.1 Install Raspbian on your Raspberry Pi

1.2 Attach camera to Raspberry Pi and enable camera support

1.2.1 Execute: `sudo raspi-con_g`

This command opens the raspberry pi con_guration. Select enable camera and hit enter and reboot.

1.3 Execute: `sudo apt-get update`

1.4 Execute: `sudo apt-get dist-upgrade`

1.5 Execute: `sudo rpi-update`

The three commands above update your raspbian

1.6 Execute: `git clone https://github.com/silvanmelchior/RPi_Cam_`

`Web_Interface.git`

1.7 Execute: `cd RPi Cam Web Interface`

1.8 Execute: `chmod u+x *.sh`

The three commands above install RPi-Cam-Web-Interface

