Lab 1 – Gold Descriptive Paper

Ahsif A. Sheikh

CS 411

Professor Thomas J. Kennedy

March 19, 2018

Version 3

Table of Contents

1. Introduction 4
2. Product Description 5
2.1. Key Product Features and Capabilities 6
2.1.1. Vehicle Counts and Displays 6
2.1.2. Navigation 6
2.1.3. Management 6
2.1.4. Other 7
2.2. Major Components (Hardware/Software) 7
2.2.1. Out of the Box Requirements 8
2.2.2. Development Tools 8
2.2.3. Software Description 9
3. Identification of Case Study 9
4. Product Prototype Description 9
4.1.2. Hazelcast 10
4.1.3. MongoDB 11
4.1.4. Java Spring Framework 11
4.1.5. Gradle 11
4.1.6. Git 11
4.2. Prototype Features and Capabilities 12
4.2.1. Risks 13
4.2.1.1. Technical Risks 13
4.2.1.2. Customer Risks 16
4.2.1.3 User Risks 18
4.3. Prototype Development Challenges 19
5. Glossary 20
6. References 21

List of Figures

Figure 1 - Major Functional Components Diagram 8
Figure 2 - Prototype Major Functional Components Diagram..... 10

List of Tables

Table 1 - RWP and Prototype Features..... 12
Table 2 - Technical Risks 15
Table 3 - Customer Risks..... 17
Table 4 - User Risks..... 18

1. Introduction

Nationwide many campuses have encountered the same issue that Old Dominion University is currently up against. Based on research, the issue of inadequate parking is caused by a high student population, commuters and residents. Lack of parking has been a source of stress for many staff members, students and even visitors of the the campus. According to the school site, last year's total enrollment at Old Dominion was 24,828 students. Around 76% of students live off-campus, including online students. However, out of the 76%, the majority of the students commute to campus (17). Undergraduate student enrollment is 19,819, 76% multiplied by 19.819 equals 15,062.44 (17). This means that approximately 15,000 students live off campus.

Old Dominion currently offers 5 different parking garages specified for faculty, metered, commuters, and others totaling 3,013 spaces (12). Roughly 9,400 student commuters require parking at Old Dominion University daily and 1,511 faculty members (835 Full Time and 676 Part Time) (18). A student at Old Dominion decided to provide his input on the situation, "Parking at ODU sucks, there are not enough spaces for everyone and if you are a commuter you better get to class an hour early if you want a spot. It is like The Hunger Games for parking spaces. May the odds be ever in your favor." (5)

A logical solution to inadequate parking on campus would be to build more garages. This option is expensive. The national average to build a parking garage is $8.56 million (73). Furthermore, the current geographic size of Old Dominion University poses a constraint. Academic buildings hold a priority over additional parking garages. The current state of the school demands a more efficient method to utilize the existing parking without having to increase the number of parking garages. Without improvement, drivers will continue

experiencing difficulty finding parking spaces, during the hours of 10:00AM - 2:00PM, due to: lack of signage and notifications for available spaces, preferences for specific parking locations, and limited choices during peak hours.

2. Product Description

ParkODU is a web application designed for drivers that need to park at Old Dominion University. The software interface will allow users to view all current parking information related to the school so that users may find parking more efficiently. The ultimate goal of ParkODU is to automate and decrease the amount of time spent manually searching for the nearest parking vacancy. The main solution analyzes the parking availability in real-time and then helps drivers find a vacancy closest to their destination. The solution is optimized using the starting location, permit type, and the destination in order to narrow down the parking recommendations. Hence, allowing the user to save time, resources, effort and find the best parking spot.

The prototype, simulates a garage that demonstrates all of the products capabilities and features, shown in *Table 1*. ParkODU will also gather useful parking space usage statistics that will benefit Old Dominion University Transportation and Parking. It will compile data gathered in real time by various vehicle counting systems installed in the garages, parking lots and make this information open to the users.

The application will be compatible with major web browsers and will also be available for Android and iOS. The main objective of ParkODU is to inform drivers of the vehicle counts so that they can avoid driving to parking facilities that are full and go directly to another facility that has vacancies. The secondary objective of ParkODU is to provide Old Dominion University

Transportation and Parking with the data from their parking facilities for them to use in future strategic planning.

2.1. Key Product Features and Capabilities

ParkODU possesses emerging technologies that provide a less time consuming method to identifying parking vacancies through real-time monitoring with counts and displays, navigation, notifications and other management.

2.1.1. Vehicle Counts and Displays

The application will be able to track and monitor garage parking availability in real-time. These counts are organized by garage, floor, and space. ParkODU will display a detailed floor plan of each garage floor and the available places. The average vehicle counts at each location by the time of day will be displayed. We have implemented filters and allowed users to sort from garages, floors, and space by their parking permit. Also added filters that sort garages by walking distance and travel time to another building.

2.1.2. Navigation

The application will be able to utilize Google Maps navigation features. It also predicts future vehicle counts (please see algorithms to expand beyond this point). It will also allow faculty and students to import their schedule to the application for the closest parking space recommendations.

2.1.3. Management

ParkODU allows the addition, editing, or removal of a garage, floor, or space and the removal of user roles. The ParkODU App provides configuration of occupancy signage and sends notifications of the events going on at ODU that may affect parking. ParkODU is a

universal application that will interface with any vehicle detection system (so the user can pick what hardware they are most comfortable managing).

2.1.4. Other

ParkODU will utilize open source software along with mobile support. The goal is to help users become more efficient in their quest for parking. Drivers can avoid full garages and minimize time spent searching for a spot.

2.2. Major Components (Hardware/Software)

The flow of information starts from the vehicle detection technology of the customer's choice; these range from IR sensors, inductive loops, IP cameras, or other means of detection. Some systems, such as those with IR sensors installed at each parking spot that are capable of counting by space. Other systems such as inductive loops are only capable of counting spots by floor as the installation of inductive loops for each space is infeasible. The vendors providing the vehicle detection technology gather data to store on their own server. ODU would typically access the venders web portal to access the occupancy data. However, the vendors also provide API's, which allows ODU's software developers to create their own app or interfaces. The data obtained from the vendor's API's will be sent to ParkODU via requests to REST endpoint. The data stored in Hazelcast in-memory data grid for fast queries and MongDB will be used as the secondary and backup data storage. The customer will require physical or virtual machines, for the database and web application. In case the customer fails to provide machines, ODU CS department virtual machines will be used as the servers. ParkODU database, and application servers will service the web app, Android/iOS app, and the digital signs. The App stores data in MongoDB, an open source database. The information can then be displayed on signage or accessed via the web on a smartphone or computer.

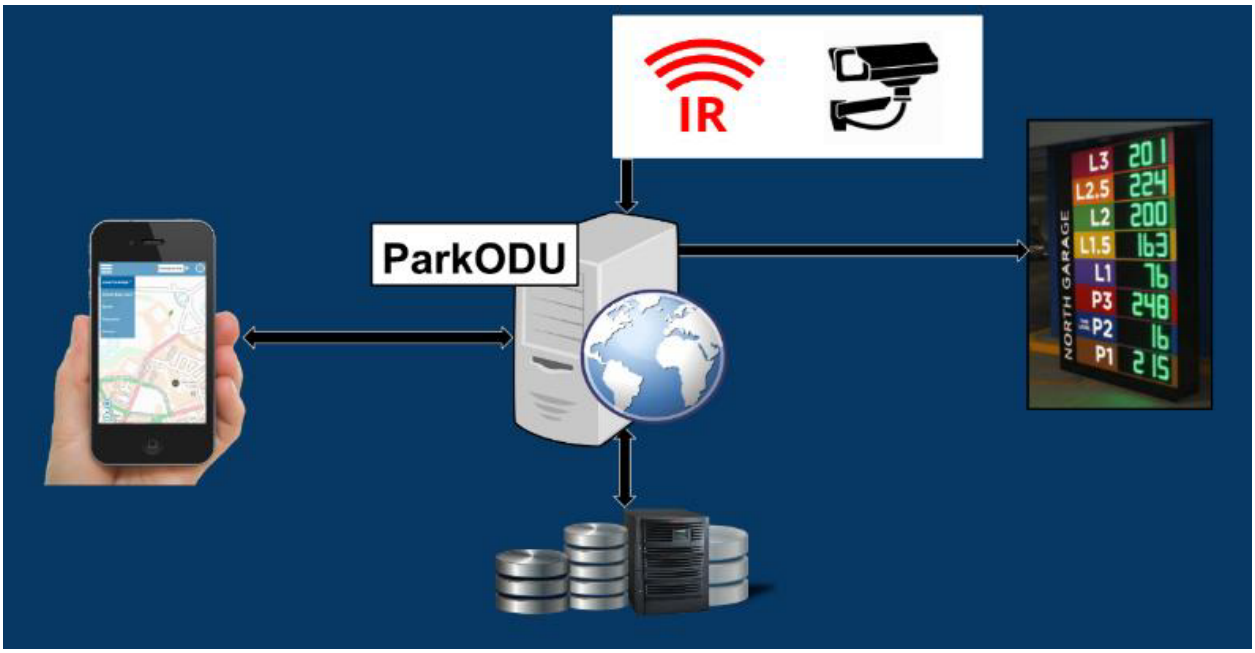

Figure 1 - Major Functional Components Diagram

2.2.1. Out of the Box Requirements

ParkODU is accessed via the web on a smartphone or computer. The customer will need to purchase the vehicle detection technology to interface with ParkODU, such as IR sensors or Cameras. Optionally, the customer is able to purchase signage and display information from ParkODU on the signs. ParkODU will be hosted on a server or cluster of servers, physical or virtual. The software is open source and available for download on our website.

2.2.2. Development Tools

The language being used to develop ParkODU is Java using Spring Framework. The IDE is IntelliJ Community Edition. Build tools are Jenkins and Gradle. The data is stored in Hazelcast and MongoDB. It will be a Third-Party API using the Google Maps API. Git is the version control platform.

2.2.3. Software Description

The software provides the tools necessary to collect, store, and communicate parking information. By providing an interface for updating vehicle counts, any vehicle detection technology is useable. ParkODU also includes the necessary algorithms to provide the functionalities listed in section 2.1.

3. Identification of Case Study

ParkODU is being designed for the department of Transportation and Parking services at Old Dominion University. Due to geographical constraints, ODU does not have space to build additional parking. Academic buildings also have a priority over additional parking lots. ParkODU offers a more effective and less costly solution to parking. Signage outside of every parking lot will make it easier for any driver, especially those visiting, to know about vacancies. ParkODU could also be adopted by universities and businesses who are in need of a more efficient way to handle their parking lots.

4. Product Prototype Description

The prototype of ParkODU will have all of the defined features and capabilities of the web app and the native Android/iOS App except support for digital signs. The data that would normally be obtained from vehicle detection systems such as inductive loops, IR sensors, and IP cameras, will be simulated. The ParkODU prototype will compile the simulated data to demonstrate all the features and capabilities of the web app and the native Android/iOS App.

4.1 Prototype Architecture (Hardware/Software)

The ParkODU prototype consists of a REST client application as the hardware for vehicle detection. It will utilize Hazelcast, MongoDB, Java Spring Framework, Gradle, and Git.

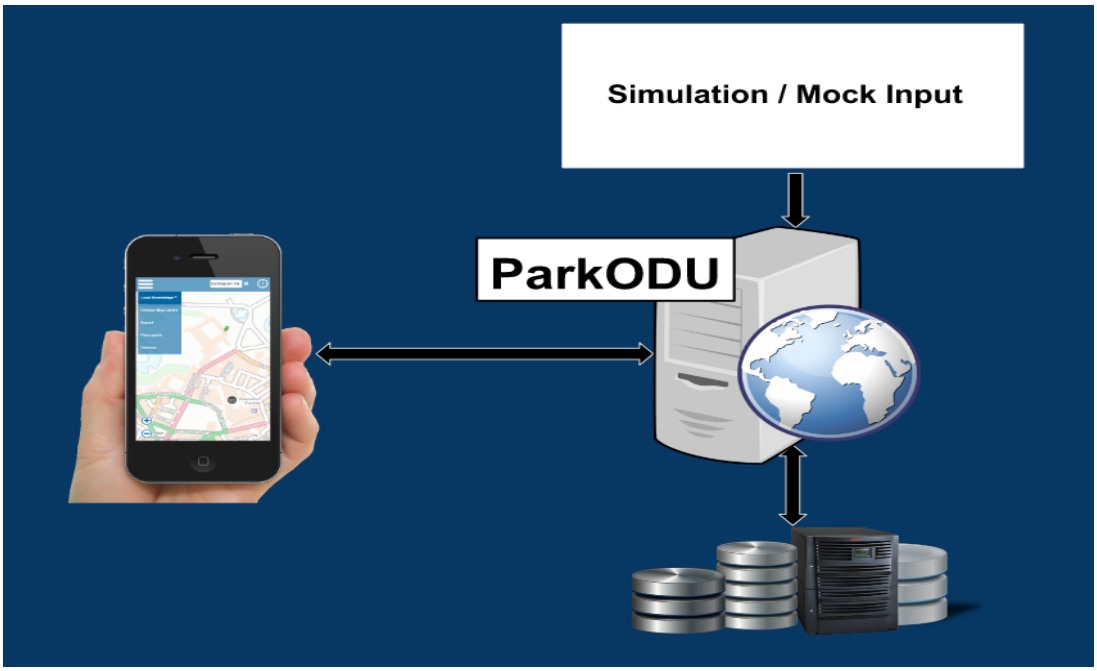

Figure 2 - Prototype Major Functional Components Diagram

4.1.1. REST

The hardware for vehicle detection will be simulated by a REST client application. The application will send requests to ParkODU REST endpoint and update the vehicle counts. The application will closely mirror the actual ODU parking traffic during weekday normal hours and also simulate special events on weekends. The ParkODU web app and native Android/iOS App will perform operations on the simulated data to demonstrate the features and capabilities.

4.1.2. Hazelcast

Hazelcast is a Java based open source in-memory data grid. Hazelcast In-Memory Data Grid is often used as an operation memory layer for databases in order to improve performance of applications, to distribute data across servers, clusters and geographies, to ingest data at very high rates, and to manage large data sets (6). Hazelcast provides high-availability access to frequently used data while offering scalable solutions in a multi-node environment. The data grid created by Hazelcast offers redundancy and increases performance.

4.1.3. MongoDB

MongoDB is a free, open-source database used for todays applications. MongoDB stores data in flexible, JSON-like documents, meaning fields can vary from document to document and data structure can be changed over time (9). Mapping is distributed within the MongoDB framework which allows the opportunity for flexible and horizontal scaling (9).

4.1.4. Java Spring Framework

Spring Framework is an open-source framework for Java-based applications. However, spring framework allows extensions for the developing of web applications along with the Java EE platform (15). This is the Spring Framework we will be using to create ParkODU. This will allow us to develop and android application and also a Web Application.

4.1.5. Gradle

Gradle is a build tool trusted by millions of developers. Gradle is a build automation system that can build programs in Java, C++, and also Python. Gradle can recognize which parts of a build are already up-to-date and offers the flexibility for incremental and multi-scale builds. It will be used to create and generate the builds of ParkODU.

4.1.6. Git

Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency (21). It provides an interface for collaborating with other teammates on a project. It allows a team to make revisions based on the most recent source code uploaded on the hub.

4.2. Prototype Features and Capabilities

Our application prototype will simulate an input and display a real-time vehicle count by floor in every garage. The prototype will provide the detailed floor plan along with navigation to the vacant space. The user will be able to import his/her schedule and the application will generate the nearest parking options. It will also analyze the users previous parking data in for improved recommendations.

The functionality and purpose are vital in showing how this application has improved the users parking experience at ODU. By providing the user with a live count of parking spaces available by garage they are able to quickly and effectively locate parking. Success has been demonstrated because the overall goal of ParkODU is to reduce the amount of time spent manually searching for parking and the user of the application will be more informed allowing this to become possible.

Feature	RWP	Prototype
Real-time vehicle counts on every level of each garage	*	*
Display floor plan to show counts by space on each floor	*	*
Display average vehicle count at each location by time of the day	*	*
Allow users to sort garages by walking travel time	*	*
Allow users to filter garages, floors, and space by their parking permit type and space types	*	*
Allow ODU parking staff to configure parking garages, floors, and spaces.	*	*
Provide directions to each garage from user's current location	*	*
Predict future vehicle counts based on the current and historical traffic pattern	*	*
Upload special event schedules and allow the apps to display notification to end users	*	*
Send data to digital signs at the entrance of every garage	*	

Table 1 - RWP and Prototype Features

The ParkODU prototype will support nearly all features and capabilities of the real working system. All of the features of ParkODU are designed to provide transparency to ODU parking availability; this will help drivers avoid full parking garages and go directly to parking garages that have available parking spaces. The prototype will demonstrate ParkODU's ability to accurately process the data received from the vehicle detection devices and display the results to end users.

4.2.1. Risks

The risk mitigations addressed by the prototype include:

- (T2) Database/Web Application Failure – the application server fails to connect to the Web, thus creating the likelihood of downtime. ParkODU integrates with Hazelcast's Management Center in order to monitor the server. Notifications will be sent in the event of any unexpected shut down.
- (T3) Software Bugs – the ParkODU prototype will utilize test driven development methods during the development phase to mitigate software bugs. Specific team members have been assigned to develop UI/UX testing standards and complete regression tests.

  (T7) Incompatible Input Format – a rule system based on regular expressions to prevent incompatible inputs from being loaded into the data store.

  (T8) Inability to Scale Under Load – with the use of Hazelcast's distributed computing platform, new members can easily be configured and added to the existing cluster.

4.2.1.1. Technical Risks

The technical risks listed in *Table 2* show various steps for mitigation towards the risks associated with the hardware and software components of ParkODU.

Risk	Description	Mitigation	Impact	Probability
(T1) Web Connection Failure	Application server fails to connect to the Web.	• Test the connection and ensure communication is regained.	Very High	Low
(T2) Database/Web Application Failure	Database/Web App failure may occur due to network settings being offline or unavailable.	• Verify the database and software communication through testing. • Establish dedicated clustered server environments for both database and web application server clusters to reduce possible downtime of ParkODU.	Very High	Low
(T3) Software Bugs	Software development opens up the possibility for bugs that may reduce functionality of the Web application.	• Software updates and debugging techniques will be administered routinely. • User Interface/User Experience (UI/UX) Testing • Regression testing and continuous integration	Very High	Very Low
(T4) Hardware Failure	Hardware including IR sensors, Garage Signage (optional), may not be functioning or require repair.	• Mark spot as hardware malfunction and send a service request to maintenance.	High	Medium

Risk	Description	Mitigation	Impact	Probability
(T5) Failure to Notify User of an Event	ParkODU is not updated with event schedules that may affect garage availability.	• Ensure ParkODU is updated with upcoming events. • Create a ScheduledTask to poll the event calendar through rest endpoints.	Very High	Low
(T6) Lack of Technical Knowledge	Minimal technical experience and/or programming familiarity needed to develop the application.	• Individually improve programming knowledge and provide training to less experienced members in area of deficiency.	Medium	Medium
(T7) Incompatible Input Format	Formatting of input is not compatible with ParkODU.	• Verify input formatting compatibility through testing.	Medium	Medium
(T8) Inability to Scale Under Load	As volume or customer count increases the database becomes slow or may fail.	• Expanding computing resources to handle the exponential growth of work with the use of database scalability. • Establish dedicated clustered server environments for both database and web application server clusters.	Very High	Very Low

Table 2 - Technical Risks

4.2.1.2. Customer Risks

Table 3 shows risks that are associated with customer based decisions on the implementation of a different solution, the incompatibility with new and existing resources, and decisions based on geographical expansion.

Risk	Description	Mitigation	Impact	Probability
(C1) University Implements a Better Solution	The university implements a solution other than ParkODU.	• Show the customer how ParkODU's benefits and features are superior to competing solutions. • Offer ParkODU as an open-source solution.	Very High	Very Low
(C2) University Does Not Allow Access to Network	ODU ITS does not allow ParkODU to run on the university's network.	• Some departments within the university run things on their own networks. • Customer determines hosting location.	Low	Very Low
(C3) Customer Unable to Maintain Servers/Hardware	Customer will be unable to maintain the hardware utilized by ParkODU.	• Customer determines the most effective hardware solution with their implementation for ParkODU.	High	Medium

Risk	Description	Mitigation	Impact	Probability
(C4) University Replaces All Garages and Lots with Other Buildings	All parking garages and lots are replaced by buildings.	• Parking will still be essential. The software will allow for reconfiguration as the university changes parking allocations.	Very High	Very Low
(C5) Customer Unwilling to Purchase Hardware	Customer may not agree to purchase hardware.	• Software will allow for multiple hardware implementations. • ParkODU will allow for manual toggling of parking space availability.	Very High	Medium
(C6) Parking Lot Replaced by Parking Garage	The University builds parking more parking garages in place of parking lots.	• Ability to add/edit/delete parking objects.	Low	Low
(C7) Customer Purchases Partially Compatible Technology	Customer purchases detection hardware that does not support a certain functionality, such as count by space.	• The software can be reconfigured to support specific customer implementation.	Medium	Medium

Table 3 - Customer Risks

4.2.1.3 User Risks

Risks associated with ParkODU end users that may potentially impact the identification of available parking spaces are shown on *Table 4.*

Risk	Description	Mitigation	Impact	Probability
(U1) User is Distracted While Using the Application	User is distracted while using ParkODU.	• Provide safety notification. • Allow for ParkODU to auto-refresh in order to display current data without any additional user interaction.	High	High
(U2) No Internet Device	The end user does not have access to an internet device, such as a Smartphone or a computer, to use the mobile or web application.	• User is able to view occupancy signage while on campus. • User can use public resources such as a public library computer to access ParkODU. • User can utilize ParkODU's historical prediction feature to print future projections.	Very High	Very Low
(U3) User Cannot Find a Parking Spot	All parking that is being monitored by the application is full.	• Inform the user parking is full and application will notify ODU Parking Services. • Provide ODU Parking Services contact information.	Very High	Very Low

Table 4 - User Risks

4.3. Prototype Development Challenges

The challenges while completing the objectives of the prototype will come from obtaining knowledge for various development tools, services and technology. The development of the ParkODU prototype will require substantial knowledge in:

- MongoDB
- Hazelcast
- Java programming
- RESTful web services
- Web programming (HTML5 and Javascript)
- Native App development for Android and iOS

It is expected that every team member will devote a significant amount of time in personal research and mentoring to understand all of the tools and services required to complete the objectives of the ParkODU prototype. Another challenge of the prototype development is the time constraint due to the complicated nature of the solution and many requirements that need to be met.

5. Glossary

Administrator - a special user with access to additional tools for user account and space management

Agile - a methodology that anticipates the need for flexibility and applies a level of pragmatism into the delivery of the finished product

Best Garage - the closest garage to the destination building with the specified minimum number of available spaces

Driver - anyone who drives and parks at ODU

Driver Entry Rate - the number of vehicles entering the garage each minute

Driver Exit Rate - the number of vehicles exiting the garage each minute

Event - an occasion which affects garage and/or space availability

Garage Rate - Driver Entry Rate - Driver Exit Rate (a positive number denotes that the garage is filling up)

Operating Hours - 7:00AM - 10:00PM

Permit - a physical decal that specifies in which spaces the vehicle is allowed to park

Predictions - a guess based on current and historical data about garage space availability

Real-time - current time

Reconfigurable - software-based creation, deletion, or editing of spaces, floors, and garages

Rush Hours - 7:45AM - 9:00AM, 12:00PM - 1:00PM, 3:00PM - 4:30PM

Sensor - any device which indicates to the software whether a space is occupied or not

Signage - signs that indicate the number of available spaces

Statistical Analysis - the ability to use sample data to form predictions

User - an entity using Park ODU

Vehicle Detection Technology - any device which indicates to the software that a vehicle has entered a specified area

6. References

Access Automation Car Park Count Systems. (n.d.). Retrieved October 10, 2017, from http://www.access-automation.co.uk/car-park-count-systems. (1)

Agile [Digital image]. (2017, May 8). Retrieved November 29, 2017, from https://www.codingmart.com/uploads/post/image/57e0c0488ca7853c76dd986e/Agile_Development_Process.pngvehicle-coun(F.4.)

Burr, David W. "Is University Parking a Common Grievance?". Parking Today Media. September 2011. http://www.parkingtoday.com/articledetails.php?id=1072. September 2017. (3)

Car counting solutions. (n.d.). Retrieved October 10, 2017, from http://www.puretechsystems.com/solutions-car-counting.html. (4)

"Hazelcast the Leading In-Memory Data Grid" Retrieved January 23rd, 2018 from https://hazelcast.com (6)

How Much Does a Parking Garage Cost? Retrieved November 02, 2017, from http://www.parking.org/2016/01/19/tpp-2013-09-how-much-does-a-structure-cost/. (7)

"IntelliJ IDEA: The Java IDE for Professional Developers by JetBrains." *IntelliJ IDEA*, Jet Brains , Retrieved January 18th, 2018, fromwww.jetbrains.com/idea/. (8)

Operating Budget and Plan. Old Dominion University. Retrieved November 02, 2017, from https://www.odu.edu/content/dam/odu/offices/budget-office/docs/opplan2017.pdf. (10)

ODU Campus Parking Map. Retrieved October 23, 2017, from https://www.odu.edu/content/dam/odu/offices/parking-and-transportation-services/docs/odu-student-parking-map-mm.pdf. (F.1.)

Parking and Traffic Procedures. Old Dominion University. Retrieved November 02, 2017, from https://www.odu.edu/content/dam/odu/offices/parking-and-transportation-services/docs/parking-transportation-rules-and-regulations.pdf. (12)

Providence Place mall enhances parking garage with $20M in improvements (2016, December 15). Retrieved October 30, 2017, from https://pbn.com/providence-place-mall-enhances-parking-garage-adds-more-pay-stations-improves-signage119194/. (F.3.)

Rogers, Emily. Dear Future ODU Students. (2017, August 28). Retrieved November 02, 2017, from https://www.theodysseyonline.com/dear-future-odu-students. (5)
Git, Retrieved October 10, 2017, from git-scm.com/. (21)

Solutions: vehicle counting. (n.d.). Retrieved October 10, 2017, from http://www.t2systems.com/solutions/vehicle-counting. (14)

"Spring: the source of modern java by Pivotal" Retrieved January 23rd, 2018 from http://spring.io (15)

Team Gold. "ParkODU." December 2017. PowerPoint presentation. (16)

The Problem at Hand - The Expansion of Parking At Old Dominion University. (n.d.). Retrieved November 02, 2017, from https://sites.google.com/a/odu.edu/the-expansion-of-parking-at-old-dominion-university/home/the-problem-at-hand. (17)

University Facts & Figures. Old Dominion University. Retrieved November 02, 2017, from https://www.odu.edu/about/facts-and-figures. Accessed November 1, 2017. (18)

Vehicle Counter. (2016, February 12). Retrieved October 10, 2017, from https://www.kiwisecurity.com/. (19)

Vehicle counting & detection systems. (n.d.). Retrieved October 10, 2017, from https://www.swarco.com/stl/Products-Services/Parking-Solutions/Parking-guidance/Vehicle-counting-detection-systems. (20)

"What Is MongoDB?" Retrieved on January 23rd, 2018 from *MongoDB*, http://www.mongodb.com/what-is-mongodb. (9)