Team 16: GardeNet

Project Proposal and Feasibility Study

John Connell,
Anthony Jin,
Charles Kingston,
and Kevin Kredit

Calvin College ENGR 339/340
Senior Design Project
12/11/2015

Abstract

Irrigation is a labor intensive task for gardens of all sizes. For gardens and farms too large for simple timing systems and too small for industrial solutions, managers must choose between expensive Wi-Fi based solutions and large amounts of menial work. Team 16 of Calvin College's senior design class of 2016, called Team GardeNet, has resolved to fill this gap. The team proposes to design a 3G based solution that can control an arbitrarily large number of watering zones through a website interface. Basic system components include the frontend website, a cloud service to manage data storage and communications, an onsite controller, and distributed sensors and actuators. Current market offerings do not feature 3G network connectivity, and given the business survey and cost analysis, the team believes that if GardeNet comes in a 3G and Wi-Fi enabled version, it can compete profitably in the industry.

Table of Contents

Table of Figures

Table of Tables

1 Introduction

Watering gardens can be a strenuous task. Gardens need constant attention to be able to stay alive, blossom, and produce a good yield. The single most important factor in maintaining a garden is making sure that each plant gets the right amount of water to grow at its optimal level. Unfortunately, plans sometimes fall through and workers run out of time or, in the case of community gardens, volunteers do not show up, and the plants do not get the water they need. The GardeNet system is designed to solve this problem.

1.1 Project Description

The GardeNet system incorporates ease of use with reliable computing in the background to make an efficient solution. The system provides mobile friendly features that integrate with cloud server technology to easily send scheduling and shut off requests to the controller without ever having to visit the garden. A computer or mobile device can access an online dashboard to plan when to water, receive current data, and access current weather forecasts. Because the GardeNet system automates the watering, gardeners can rest easy knowing that their garden is getting all the water it needs. Note that the GardeNet system is being designed to easily integrate with current irrigation systems; it does not provide any mode of actual water delivery, as that is outside the team's area of expertise.

1.2 Requirements & Goals

The primary objective of GardeNet is to develop a basic watering control system that the user controls from a website, mobile application, or onsite input. Currently, two community gardens are interested in implementing GardeNet. Both farms give the following requirements: to be controlled through cellular data from the cloud, to have valves controlling water in many different zones at once, to have extreme reliability, and to remain cost-effective when making the system. Because GardeNet is being designed for community gardens, it must be able to be installed and maintained on a small budget. The requirements are discussed in more detail in Section 3.3 and are comprehensively listed in Appendix I: Requirements List.

2 Project Management

2.1 Team Organization

The team consists of four members, all Electrical and Computer Engineering majors: John Connell, Anthony Jin, Charles Kingston, and Kevin Kredit. The team possesses a combined two Computer Science minors, two Mathematics minors, and a Business minor. Each team member is expected to graduate in May 2016.

2.1.1 Team Roles

2.1.1.1 John Connell: Webmaster, Frontend User Interface, and Scheduling

John has past experience with website development and chose to take on the role of managing and maintaining the team's website. John will also be assuming the roles of developing the frontend user interface to the GardeNet system and managing the scheduling for the team throughout the project.

2.1.1.2 Anthony Jin: Chief Editor, Cloud Rule Engine, and Controller to Cloud Communications

Anthony has taken the role of being the final editor of all of the team's published documents. Anthony has also assumed the roles of developing the cloud server's rule engine and being the head of handling the cloud to controller/gateway communications.

2.1.1.3 Charles Kingston: Supply Chain Manager and Controller

Charles has taken the role of managing the relationships with outside and industry contacts as well as communicating with the industry sources. Charles has also been assigned with the development of the main onsite controller of the GardeNet system.

2.1.1.4 Kevin Kredit: Project Manager, Budget Officer, and Sensor/Valve Controllers

Kevin has assumed the role of project manager where he will be setting milestones and tracking the team's progress. Kevin has also been assigned to handle all of the budget details. Lastly, Kevin is the head of developing the controllers for each sensor and valve in the GardeNet system.

2.1.2 Other Important Players

2.1.2.1 Mark Michmerhuizen: Team Advisor

Professor Mark Michmerhuizen, an Electrical and Computer Engineering professor at Calvin College, serves as the project's faculty advisor. Professor Michmerhuizen brings more than two decades of industry experience to the project as well as master's degrees in Engineering and Business Administration.

2.1.2.2 Kurt Dykema: Mentor

Kurt Dykema is Technology Director at Twisthink in Holland. Twisthink has a broad portfolio of projects, including a tracking and management system for swimming coaches. Kurt brings project management tips to the table for the team as well as a connection to a potential customer (Kyle VanEerden of Eighth Day Farm; see below).

2.1.2.3 Eric Walstra: Industrial Consultant

Eric Walstra is an engineering manager at Gentex in Zeeland. He has served as a consultant in a meeting held on November 10. Eric stressed the importance of identifying areas of highest risk and highest priority; for this project, he identified sending data through a cloud service over 3G as both high risk and high priority. The team will be meeting with him again in March.

2.1.2.4 David Benjamin and Kyle VanEerden: Potential Customers

David Benjamin and Kyle VanEerden, managers of Caledonia Community Garden (CCG) and Eighth Day Farm (EDF), have helped the team steer the project in a direction that will provide the most value. David and CCG in particular represent the team's primary customer, whom the team intends to serve by actually installing a functional system.

2.2 Schedule

Currently, a work breakdown structure (WBS) is used to keep track of all project tasks. While a detailed list of the tasks is listed in Appendix II: Work Breakdown Structure, Table 1: Project Milestones lists the major milestones of the project.

Table 1: Project Milestones

Milestones	Description	Due Date
Develop Requirements	Define all technical and non-technical requirements of the design	November 27, 2015
Design	Design system architecture and top-level user interface	February 21, 2016
Test	Implement, test, and troubleshoot prototype	March 13, 2016
Refinement	Fix bugs and introduce additional features	April 24, 2016
Develop Documentation	Develop technical and user documentation	May 1, 2016
Onsite Installation	Install final system for client	May 6, 2016

John is assigned to maintain the WBS and scheduling. The team utilizes an online Gantt chart tool called Instagantt to visualize the team's WBS. In addition, John uses the online project management tool Asana to manage the team's schedule and assignments. The team meets every Monday and Wednesday for two hours to review the progress of the project and determine a list of action items for the same week. The team has also established a monthly one-hour meeting with its mentor, Kurt of Twisthink. Currently, each member of the team spends five to ten hours each week to work on various tasks for the Senior Design course and for the overall design of the GardeNet system.

2.3 Budget

Kevin manages Team 16's budget. Table 2 lists the expected installation expenses for CCG.

Table 2: CCG Installation Expenses

Item	Number of Units	Cost per Unit
Controller	1	$ 246.44
Valve Controller	0	$ 18.52
Valve/Flow Meter Controller	3	$ 46.47
Weather Station	1	$ 103.51
Cloud Server	1	$ -
Cloud Server	Monthly	$ -
3G Service	Monthly	$ 20.00
Total Installation Costs		$ 489.36

Besides the $500 budget provided by the Engineering Department, the team proposed for funding from the Eric DeGroot Engineering Fund (EDEF). Since Team 16 has two possible clients, **Error! Not a valid bookmark self-reference.** estimates their ability to serve each client based on the amount of funds potentially granted from the EDEF. Unfortunately, the funding was given to other senior design teams. With the current funding, GardeNet will still be able to supply CCG with their own version of the system.

Table 3: Ability to Serve CCG and EDF Based on Funding

	Amount Granted				
Eric DeGroot	$ -	$ 250.00	$ 500.00	$ 750.00	$ 1,000.00
Calvin	$ 500.00	$ 500.00	$ 500.00	$ 500.00	$ 500.00
Total	$ 500.00	$ 750.00	$ 1,000.00	$ 1,250.00	$ 1,500.00
	Ability to Serve Caledonia				
Basic Installation	✓	✓	✓	✓	✓
Multiple Zones	✓	✓	✓	✓	✓
Moisture Sensors		✓	✓	✓	✓
On-Site Weater		✓	✓	✓	✓
	Ability to Serve Eighth Day Farm				
Basic Installation			✓	✓	✓
Multiple Zones				✓	✓
Moisture Sensors					✓
On-Site Weater				✓	✓

2.4 Method of Approach

This section covers the design methodology and team connectivity and accountability that will be used throughout the project.

2.4.1 Design Methodology

The team decided that the design methodology that best fit the design norms and was the most efficient was the V-Model, as seen in Figure 1. The V-Model is a well-known system design methodology that is used at most major companies.

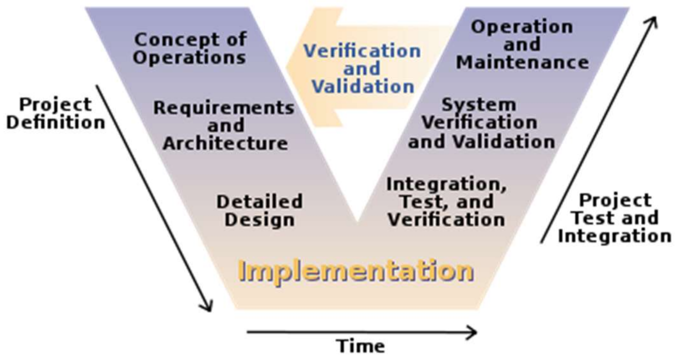
Concept of Operations
Verification and Validation
Operation and Maintenance
Project Definition
Requirements and Architecture
System Verification and Validation
Detailed Design
Integration, Test, and Verification
Project Test and Integration
Implementation
Time

Figure 1: The V Model for Systems Engineering [1]

The V-Model starts with defining system level requirements that come from both potential customers and the team itself based on the end goal of the system. After the system level requirements were defined, the team developed component level requirements that spelled out exactly how each piece of the system should communicate and interface. Then, the team moved into detailed design requirements that spelled out specific functionality of every piece of the system. After all of the requirements are developed and reviewed, the team will begin implementing the system. Because the requirements were written to be specific and testable, as soon as initial implementation is complete, the team can begin testing first detailed requirements, then component-level requirements, then system level requirements, along the way expanding and refining the design.

2.4.2 Team Connectivity and Accountability

The team decided that using Slack was the easiest way to stay in contact. Slack offers group messaging features and Asana integration. Asana, as previously mentioned, is the team's task scheduling and planning application, and Slack's integration allows the team to be notified when new tasks are created, completed, or are due soon. The team also decided on a shared OneDrive folder to store all of the documents, research, and presentations. Lastly, the team chose to make a GitHub repository that allows for simultaneous software development, file sharing, and version control.

3 Requirements

The project requirements evolved from a combination of the team's original ideas, choices of design norms, and relationships with Twisthink, CCG, and EDF.

3.1 Original Concept

The team's original concept required control of up to four zones via wired connection to the controller and 3G connectivity to the internet; controls and system performance were to be accessed from a website, mobile app, or onsite LCD interface. This concept and set of basic requirements was from here informed by the team's design norms and significantly refocused by my meetings with Twisthink, CCG, and EDF.

3.2 Design Norms

The team identified four design norms to guide the requirements making process—integrity, trust, humility and stewardship. First, integrity here refers to "delightful harmony"; the solution is to be complete and intuitive, living up to the requirements. That means that the team has to draft achievable requirements in the first place by choosing appropriate scope. Second, trust means that the system must be reliable. In order for the customer to enjoy the full benefit of the system, they must be able to run it for weeks at a time with confidence that it will behave predictably. Third, humility means that the team is not to overestimate the system's capabilities, and the system is to be simple and usable by any person. Fourth, the design is to reflect good stewardship in terms of water, finances, and volunteers' time.

3.3 Meetings with Customers and Final Concept

Meetings with the team's customers CCG and EDF gave the team a better perspective as to which elements of the original concept actually added value. From these meetings the team made two significant design shifts and added some stretch goals. First they pointed out that developing an LCD interface would add very little value for the amount of effort it would take to develop. Intuitive and useful LCD interfaces are notoriously difficult to develop and would rarely be used if the internet connection is reliable. In lieu of an LCD, the new requirement is to have “On,” “Off,” and “Delay 30 Min.” buttons paired with an LED to indicate the current state. Second, they showed that only four zones are not enough control for moderately sized gardens and that requiring wired connections severely limits system usefulness. Therefore, the team has updated the requirements to allow an arbitrarily large number of zones which can be controlled wirelessly via radio frequency (RF) communication with the controller.

Customer meetings also resulted in modification of and additions to the list of stretch goals. The team initially set stretch goals to make the system work completely off the grid via solar power and to pair the system with a “suggested setup” irrigation system. Twisthink pointed out that doing any work in the realm of irrigation systems (as opposed to control of preexisting irrigation systems) would require significantly more research outside the team’s area of expertise. On the other hand, EDF’s needs showed that the ability to control devices such as air conditioners and lights would add value. Therefore, the team no longer seeks to do any work with irrigation installations themselves, but set the stretch goal of controlling alternative devices. A summary of the basic requirements is shown in Table 4; for a full listing of system requirements, refer to Appendix I: Requirements List.

Table 4: Basic System Requirements

Category	**Initial Requirement**	**Design Norm Influence**	**Final Requirement**
Power Requirements	120 VAC input power	Must be reliable, insensitive to power surges, energy efficient	Design system to be AC and DC compatible; stretch goal to provide solar power
Control Type	Valves	Must be reliable	Valves; stretch goal to control AC units, lights
Control Zones	Up to four	Must be achievable, but also must be enough to be useful	Arbitrarily large number
Actuator-Controller Communication	Wired	Must be reliable, able to reach full extents of property	Wireless
Controller-Web Communication	3G cellular	Must be reliable, affordable, able to use without Wi-Fi	3G cellular
User Interface	Website, mobile app, onsite LCD interface	Must be simple, intuitive, complete	Website, mobile-friendly website, onsite buttons and LEDs

4 Task Specifications and Schedule

4.1 Work Breakdown Structure

Table 5 below shows the work breakdown structure with each milestone of the system and an estimate of how long it will take to finish. The percent completion comes from the time-weighted average of the expanded work breakdown structure. A full version of the work breakdown structure is listed in Appendix II: Work Breakdown Structure.

Table 5: Condensed Work Breakdown Structure with Hours Percent Completed

Number	Title	Description	Hours and %Complete
1.0	Develop Requirements	All of the research needed for the system.	20 hours, 100%
2.0	Design	Designing and Completing the System	240 hours, 4.6%
3.0	Test	Testing Functionality	30 hours, 0%
4.0	Refinement	Refine based on testing	50 hours, 0%
5.0	Develop Documentation	Create documentation for ease of use	10 hours, 0%
6.0	Onsite Installation	Install to current customers	3 hours, 0%
7.0	339 Requirements	Class work for Engineering 339	218 hours, 100%

According to the table, the project is currently at 43.6% completion when weighted by time estimates. However, the table does not include specific requirements for Engineering 340, the second semester of Senior Design. If Engineering 340 assignments are estimated to be 218 hours like the Engineering 339 assignments, then the percent completion of the project currently stands at 31.6%. Since the project started three months ago and next semester will allow more time spent on design, the GardeNet is on schedule to be finished in May.

5 System Architecture

5.1 Broad System Overview

From the broadest perspective, the system is designed to be modular and simple. The user will be able to control the whole system from their phone from any location. The inputs will come from the user on the GardeNet website. Those inputs will then be relayed throughout the system to accomplish the task specified by the user. The GardeNet website will also output the data that is being taken from the sensors for the user to see. The system components will communicate with each other as seen in the block diagram in Figure 2: System Overview Diagram. Although the block diagrams throughout this section betray the team's final choices for components, a full discussion of the component selection process can be found in Section 6.

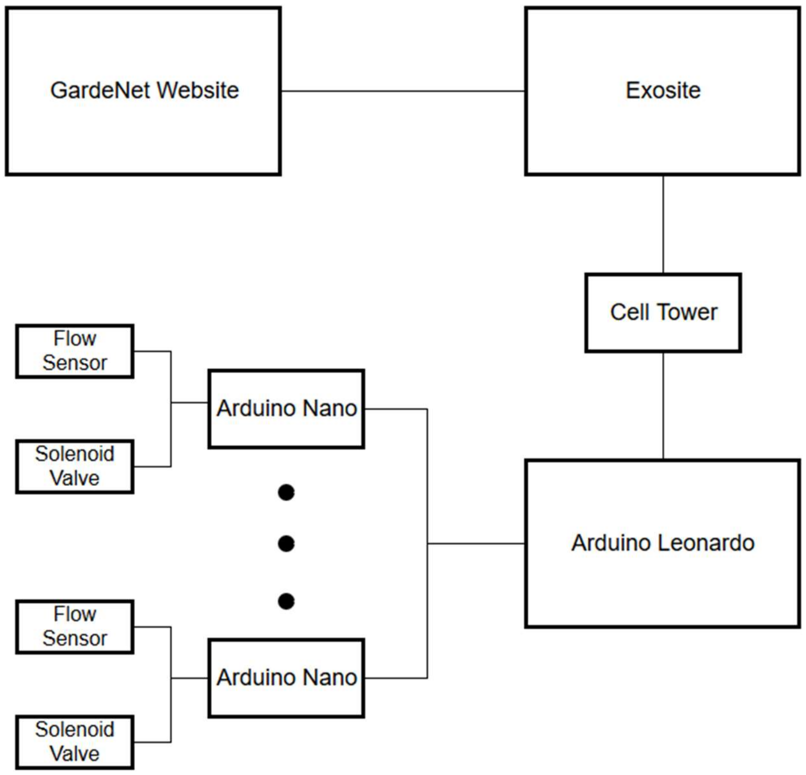

Figure 2: System Overview Diagram

5.2 Cloud Portion of the System

The cloud server acts as a hub for all of the systems data, as shown in Figure 3: Cloud System Design. The cloud server will have two primary functions: it will connect the main controller and the frontend website and it will store, process, and an act on the data that it receives.

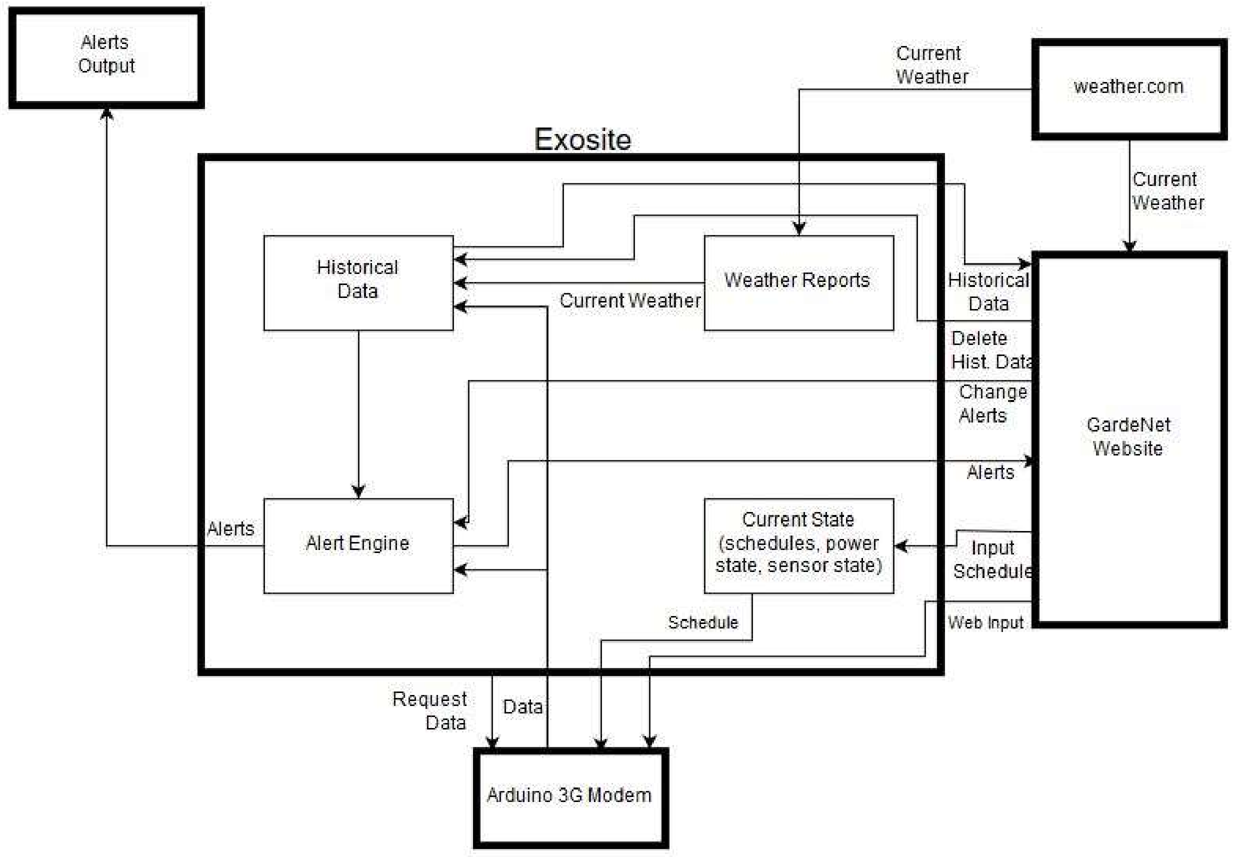

Figure 3: Cloud System Design

The cloud portion of the system will interact with the onsite system by requesting data from the main controller. Upon the cloud server receiving data from the controller, it will examine the data to determine what actions to take. If there is an issue with the data, the cloud server will send out an alert to the user containing the suspect data so the user may take action. The cloud server will also store the data that it retrieves from the controller in a historical record database. This historical database will be accessible by the user so that they may view their water usage at any time. At the end of the season, the cloud server will do a statistical analysis of the season and store it as well as display it on the GardeNet website. With consecutive season use of the GardeNet system, the cloud server will compare different season's data to find correlations and trends to display to the user. The cloud server will also pull weather reports continuously from the internet to determine a schedule to suggest to the user for the upcoming days. The user may choose to accept the schedule, or they may modify the schedule to best suit their needs. Once the user submits a schedule from the GardeNet website, the cloud server will store the schedule and send it to the main controller onsite. The team decided on sending the schedule to the main controller as opposed to the cloud server holding the schedule because if the connection between the cloud and the main controller is lost, the onsite system must remain fully functional. A user may also choose to turn certain valves on and off at any time from the GardeNet website. When the cloud server receives a user input to take action on a valve, it will send it directly to the controller to relay the message to that valve.

5.3 Main Controller

The block diagram of the main controller can be seen in Figure 4: Main Controller Block Diagram. The main controller will function primarily as a gateway between the cloud portion of the system and the valve clusters.

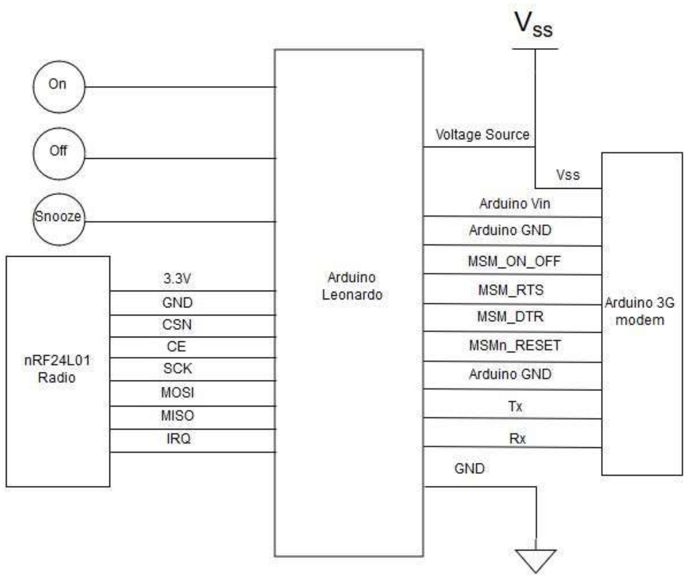

Figure 4: Main Controller Block Diagram

The main controller will be connected to a 3G cellular modem that will enable a connection to the cloud server. The main controller will also be attached to a radio module to set up a mesh network that it will be able to communicate with the valve cluster controllers on. When the main controller sets up the mesh network, it will establish itself as the master node on the network. Then, as valve cluster nodes are added to the network, the main controller will run a graphing algorithm to find the shortest path of nodes to address that node and add it to a table that stores nodes address's and paths. When a request to control a valve comes to the main controller, it will unpack the data to find which node the message is addressed to, then repackage the data and send it to that node. If the request is a schedule, the main controller will update its schedule. Lastly, the main controller will have three physical buttons: on, off, and snooze. These buttons will allow the user to physically turn on, shut off, or delay the system for a short period.

5.4 Valve Cluster

The block diagram for the valve cluster can be seen in Figure 5: Valve Cluster Block Diagram. The valve cluster controller will function as a node to on the main controller's network.

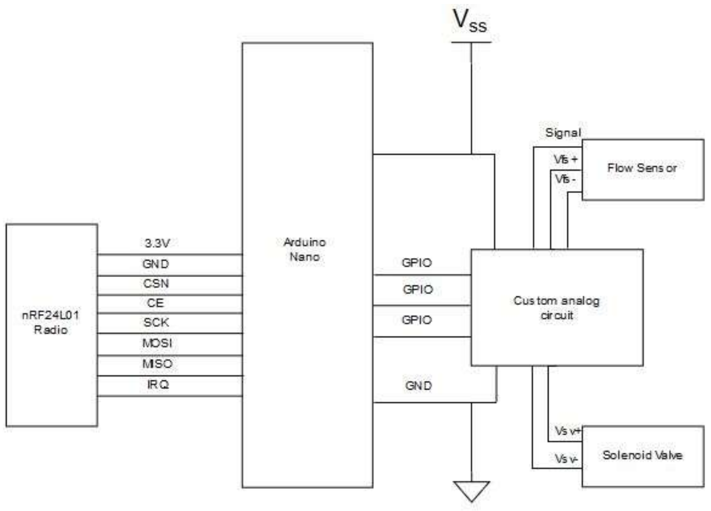

Figure 5: Valve Cluster Block Diagram

The valve cluster's controller will be connected to a radio module that will allow it to communicate with the main controller and other nodes on the network. Upon receiving a message, the valve cluster controller will first check to see if the message is addressed to it. If the message is addressed to another node, the valve cluster controller will just forward the message along. If the message is for itself, it will read the message to determine what action to take. The valve cluster controller will also be connected to a custom analog circuit that will serve the purpose of manipulating the signals between itself and the flow sensor as well as between itself and the sensor valve. Lastly, if the valve cluster controller receives a request from the main controller for its flow sensor data, it will send the readings from the sensor back to the main controller.

6 Design Process

6.1 Design Criteria

This section describes the criteria that the team used when evaluating the design alternatives. While many criteria apply to the system in general, each component also has its own special considerations. Therefore, this section begins with general criteria and then splits up into sections outlining specific criteria for each component.

6.1.1 General Design Criteria

There are some criteria that apply to all portions of the system. First is reliability, which corresponds directly to the design norm of trust—if the system is not reliable, it is worth nothing. Second is cost. Comparable systems already for sale offer competitive prices, and the target customers of urban and community gardens do not have a lot of money to spend. Third is system scalability. Because the target garden size ranges from small home installations to large gardens of several acres, the system must scale elegantly to large sizes. This corresponds to the design norm of integrity—the system shall be a joy to use for installations of all types and sizes. Another important component in a system such as this is security. This would be a general design criterion for the system before moving to production, though beyond password logins and basic encryption, it is outside the scope of this project.

6.1.2 Cloud Server Design Criteria

In addition to reliability, cost, and scalability, the criteria used to evaluate the alternative cloud server hosts were ease of use and connectivity. Ease of use here means ease with which the team can start developing on the platform due to the short time period given for design. Connectivity was weighted next because it is important that the cloud server be able to communicate with all the potential individual installations and with other websites such as weather.com and the frontend user interface.

6.1.3 Onsite Communication Design Criteria

In addition to reliability, cost, and scalability, the criteria used to evaluate the onsite communication options were ease of use and power consumption. Ease of use in this context refers both to the previous sense as ease of development and to making the system easy for the customer to install and manipulate. Because it is something that customers will interact with firsthand during setup, it must be simple to configure. This goes with the design norm of humility. Power consumption is important because onsite communication needs to use as little power as possible in order to increase battery life. In the case that the team reaches the stretch goal of running on solar power, the power savings will translate to smaller, cheaper solar panels. Note that cost is a particularly important factor here, because the cost of the communication module is multiplied for each valve; a dollar saved here is worth $10 saved elsewhere.

6.1.4 Controller/Gateway Design Criteria

6.1.4.1 Main Controller/Gateway Criteria

The additional criteria that the team used to evaluate different controller and gateway alternatives were connectivity, number of configurable pins, and power consumption. The connectivity based evaluation was the highest priority to the team because the controller serves primarily as a distributor of information between the cloud server, the valves, and the sensors. Importantly, the controller and gateway must be configurable to communicate with the chosen cloud server. The number of configurable pins was another high priority when choosing this component because the controller will be communicating to the rest of the smaller controllers, located onsite. Minimally, the chosen component must have pins that will allow it to communicate using the chosen onsite communication method. Lastly, the controller and gateway should be designed to consume as little power as possible to save money if plugging it in or to save solar costs if connecting it to a solar panel.

6.1.4.2 Valve/Sensor Controller Criteria

The additional design criteria used for evaluating the valve and sensor controllers were number of configurable pins and power consumption. Just like the main controller, the valve controllers must have enough pins to be able to communicate with the chosen onside communication method. Just like the onsite communication component, power consumption is important for battery life or solar panel size, and cost is especially important because it is multiplied by the number of zones.

6.2 Design Alternatives

The design alternatives offered in the following sections correspond to the sections analyzed in the design criteria section, with the addition in the cloud server section of a discussion on possible protocols to use to communicate between the gateway and the server.

6.2.1 Cloud Server Design Alternatives

6.2.1.1 Host Site Design Alternatives

There are three major cloud server providers that provide online platforms for IoT projects: Exosite, Aeris, and Telit. While all three host sites offer comparable reliability, scalability, and ease of use, they differ in connectivity and cost.

The biggest advantage of Exosite is its connectivity. Exosite's platform supports a variety of development boards as well as the Constrained Application Protocol (CoAP) and Hypertext Transfer Protocol (HTTP) protocols. In addition, although other features are limited, a free portal account on Exosite has unlimited data storage and can create a dashboard open to view to the public.

While providing similar functionalities as Exosite and an easy-to-use user interface, Aeris does not offer free data. In order to exploit the cloud's computational power, a specialized SIM card is also required to establish a connection node on each device using 3G or LTE cellular network.

Similar to Aeris, Telit uses specialized modems in place of SIM cards. Telit does not offer a free data plan.

6.2.1.2 Protocol Design Alternatives

The communication between two entities is impossible without a system of rules to transmit information—a communication protocol. While various protocols are available for IoT design, three of them are particularly suitable for data collection and web transfer between a cloud server and a device: Constrained Application Protocol (CoAP), Hypertext Transfer Protocol (HTTP), and Message Queue Telemetry Transport (MQTT).

CoAP is a specialized web transfer protocol for use with constrained nodes and networks in the IoT [2]. While specialized in obtaining data from devices such as sensors, CoAP includes key concepts of the Web such as URLs and Internet media type. More importantly, it has very low overhead and simplicity for constrained environment [3].

HTTP functions as a request-response protocol in the client-server computing model [4] and is intended for bandwidth-constrained applications. Although the protocol is very similar to CoAP from a developer's point of view, it aims at obtaining data from a Web API (Application Program Interface) [2].

MQTT is an extremely simple and lightweight messaging protocol, designed for constrained devices and low-bandwidth, high-latency or unreliable networks [5]. This protocol is especially suitable for mobile applications where bandwidth and battery power are at a premium.

6.2.2 Onsite Communication Design Alternatives

There are several ways to establish onsite communication within the system. One alternative is to physically wire up every system component. Even though this approach will resolve the connectivity issue and reduce the cost of data transfer, it puts a lot of constraints on the scalability of the design. While wiring up three valves to the controller within 100 feet is not a big concern, it becomes extremely cumbersome as the number of valves or the size of the garden increases. Therefore, a wireless and low-cost communication means is a better option if Team 16 intends to stress scalability.

This leads to the possibility of a personal area wireless network. Since community gardens and farms such as CCG and EDF are in the open field, where Wi-Fi does not extend, Wi-Fi is not an ideal candidate. This leads to the possibilities of MiFi, Bluetooth, or a digital radio network. MiFi is a brand name describing a wireless router that acts as a mobile Wi-Fi hotspot. Although a MiFi device can be connected to a cellular network, it can only provide internet access for up to ten devices [6]. On the other hand, a Bluetooth network will not solve the scalability problem because of its limited range and number of nodes [7].

Fortunately, an appropriate digital radio network will have the advantages of scalability, cost, and range. Team 16 have identified three RF transceivers as potential candidates: ZigBee, LoRaWAN, and nRF24L01+-based custom networks.

Based upon IEEE 802.15.4 standard, ZigBee is intended to be simpler and less expensive than other wireless personal area network such as Bluetooth or Wi-Fi. Although its stand-alone range is limited, this protocol can support industrial-size mesh networks and is very reliable due to its low power consumption and cost [8]. The disadvantages of ZigBee are higher component costs.

LoRaWAN is a Low Power Wide Area Network (LPWAN) and aims at key requirements of IoT such as bi-directional communication, mobility, and localization services. As Richard Viel, the Chief Operating Officer of Bouygues Telecom points out, the LoRaWAN technology is ideal to target battery operated sensors and low power applications as a complement to machine-to-machine (M2M) cellular. LoRaWAN is a promising candidate of M2M communication. Its topology effectively mitigates the latency between the end-devices and the central network server by making the gateway transparent [9]. However, it also poses major challenges such as sophistication and a steep learning curve to developers.

Lastly, nRF24L01+ is a highly integrated, ultra-low power RF transceiver and has similar features as ZigBee and LoRaWAN. However, the major difference among these three wireless communication schemes is the level of complexity. While ZigBee modules requires large host boards and many pins, the nRF24L01+ needs fewer pins from the host board as the chip uses simple serial peripheral interface (SPI) communication, requiring just three pins for communication and eight pins total [10].

6.2.3 Controller/Gateway Design Alternatives

6.2.3.1 Main Controller/Gateway Alternatives

The team also needs to select a development board which will serve as the gateway and main controller of their watering system as well as development boards for each individual valve. Development boards suitable for IoT design are available from many manufacturers. To name a few, Team 16 have looked at products of Arduino, Microchip, and MultiTech. In their design, the team needs a development with enough GPIO pins, processing power, and external device drivers to control the valves and sensors as well as communicate with the cloud server.

6.2.3.2 Valve/Sensor Controller Alternatives

Lastly the team needs to select a development board which will control the valves in the field. Like the main board, products from Arduino, Microchip, and MultiTech give competitive prices for small boards that can be customized to control the valves but each board needs enough pins to successfully turn on and off the valves.

6.3 Design Decisions

This section describes components that the team chose, and how each component best suits the design criteria.

6.3.1 Cloud Server Decision

6.3.1.1 Host Site Decision

The team decided Exosite best fit the design criteria. Exosite's connectivity is ranked highest because it provides the team with a vast number of devices that will be able to connect to its database and portal. Exosite also provides the team with a very high degree of scalability allowing for an arbitrary amount of devices to be connected to it. While using Exosite's full capabilities comes at a high price, the team is confident that it can implement GardeNet using only the free features. It has also been used in many of existing applications that have reported its robustness and trustworthiness meeting the design criteria and design norms the team put forth. Table 6 shows the decision matrix used to come to this decision.

Table 6: Cloud Site Decision Matrix

Factor	**Reliability**	**Cost**	**Scalability**	**Ease of Use**	**Connectivity**	**Weighted Score**
Weight (%)	40	25	10	15	10	100
Exosite	***8***	***7***	***9***	***7***	***9***	***7.8***
Aeris	8	3	10	3	9	6.3
Telit	8	5	8	8	5	6.95

6.3.1.2 Protocol Decision

The team decided that the CoAP protocol best fit the criteria specified. The protocol consumes small amounts of data reducing the cost to operate the system. Also, its ease of use with sensor data makes the protocol a prime candidate for sending sensor information in the GardeNet system. See Table 7 for the decision matrix.

Table 7: Protocol Decision Matrix

Factor	**Reliability**	**Bandwidth**	**IoT Focus**	**Ease of Use**	**Weighted Score**
Weight (%)	40	25	10	15	100
CoAP	***8***	***10***	***9***	***10***	***8.1***
HTTP	8	9	5	5	6.7
MQTT	10	9	10	5	8

6.3.2 Onsite Communication Decision

The team decided the nRF24L01+ radio best fit the design criteria. Wireless communication was chosen over wired communication because it provides better scalability and ease of use. The radio provides the user with as many zones as they need, without any long wires or extra I/O modules that would be needed with wired communication. The team decided specifically on the nRF24L01+ radio because it is the lowest cost option that still provides reliable communication at an acceptable range. ZigBee and LoRaWAN both offer longer range and higher data throughput, but at the cost of higher power consumption, cost, and complexity. The team decided that a high throughput of data was unnecessary because the GardeNet system will not need to be transferring large amounts of data. Instead, nRF24L01+ radios will allow the sensors to essentially “chirp” their data when requested. See Table 8 for the decision matrix.

Table 8: Onsite Communication Decision Matrix

Factor	**Reliability**	**Cost**	**Scalability**	**Ease of Use**	**Power**	**Weighted Score**
Weight (%)	30	30	30	5	5	100
Wires	10	10	0	10	10	7
Wi/Mi-Fi	7	2	8	8	2	5.6
Bluetooth	5	4	4	6	2	4.3
ZigBee	8	4	10	8	10	7.5
LoRaWAN	9	2	10	8	10	7.2
nRF24L01+	***8***	***9***	***8***	***7***	***8***	***8.25***

6.3.3 Controller/Gateway Decision

6.3.3.1 Main Controller/Gateway Decision

The team decided that the Arduino Leonardo best fit the design criteria. The Leonardo was selected because it easily interfaces with the NimbleLink 3G modem that comes with already loaded Verizon and FCC certifications [11]. Scott Kerstein from Arrow Electronics has also mentioned that the integration of Exosite and the NimbleLink 3G modem should be quick and seamless to set up. The Leonardo also met the design criteria for configurable pins, having extra pins on it that can configured to work as general purpose input and output (GPIO) pins as well as communication and data lines. The Leonardo was also the most cost effective decision because Arduinos are open-source, meaning that the team will not have to purchase any licenses in order to flash the board with the GardeNet code. See Table 9 for the design matrix.

Table 9: Main Controller/Gateway Decision Matrix

Factor	**Reliability**	**Cost**	**Scalability**	**Connect-ivity**	**I/O Pins**	**Power**	**Weighted Score**
Weight (%)	40	25	5	20	5	5	100
Arduino	***8***	***10***	***8***	***8***	***10***	***10***	***8.7***
Microchip	7	8	8	8	5	10	7.55
MultiTech	10	4	10	10	5	5	8

6.3.3.2 Valve/Sensor Controller Decision

The team decided that the Arduino Nano best fit the design criteria. The Nano provides the team with configurable pins so that they can communicate with the radio, the valves, and the sensors without difficulty. The Nano is also draws a small amount of power and is relatively low cost. Because it is an Arduino, it has the same qualities as the main controller with the added benefit of ensured interoperability and code similarity with the main controller. See Table 10 for the decision matrix used to come to this decision.

Table 10: Valve/Sensor Controller Decision Matrix

Factor	**Reliability**	**Cost**	**Scalability**	**I/O Pins**	**Power**	**Weighted Score**
Weight (%)	40	25	10	10	15	100
Arduino	***8***	***10***	***8***	***10***	***10***	***9***
Microchip	7	8	8	5	10	7.6
MultiTech	10	4	10	5	5	7.25

7 Integration, Test, and Debug

7.1 Component Testing Strategy

Components will be tested on an individual basis according to system requirements. Separate testable components include the frontend website (user interface to cloud server), the cloud server (website to controller), the controller (cloud server to valves), and the valves (controller to actuators). Input signals to each component shall be simulated and outputs shall be tested against expected results.

7.2 System Integration and Debug Strategy

Once all components pass their individual tests, the system shall be assembled and tested as a complete unit. System level requirements shall then be tested. Components and their interfaces shall be debugged until the system is complete or until Senior Design Night approaches.

8 Business Plan

8.1 Marketing Study

In the realm of IoT, a steady stream of products is released every year. An automated gardening system, while an interesting idea, is nothing new to the market. The GardeNet system gives the same basic functionality as a few other competitors but has advantage of running on 3G data service. Using 3G data allows the system to be used in many more locations than its competitors that only use Wi-Fi to access cloud storage. By offering GardeNet in Wi-Fi connected as well as 3G connected versions, the system also has a cost advantage that appeals to the home gardeners.

8.1.1 Competition

As it stands, the GardeNet system has two major competitors in the market, the *SkyDrop Smart Sprinkler Controller* and *HydraWise Irrigation System.* Both of these systems cost upwards of $250 and additional payments are needed to control more zones [12] [13]. Both systems promise yearly water savings and environmental friendliness. However, the scalability of these systems come at a great cost to the consumer.

The HydraWise controller offers a 6 and 12 zone controller with a 12 zone expansion module for up to 36 zones total. The total retail cost of owning 36 individual zones costs about $675. If the customer wants to measure the flow out to the zones, it is about $175 or $235 per flow meter depending on pipe size. HydraWise also adds a monthly payment option to give SMS alerts and advanced features [14].

The SkyDrop Sprinkler Controller can control up to 16 zones. The base controller costs $280 for eight zones plus an extra $60 to control the maximum 16 zones. This costs a bit less than HydraWise but is not as scalable. SkyDrop also does not currently support rain and flow sensors as it requires "a pending controller software update" [13]. The pending flow meters and rain sensors add another larger cost comparable to the HydraWise system.

The GardeNet system, equipped with 3G service, will cost about $400 plus $25 per zone. Because GardeNet prices per zone, the service is a lot more scalable to the exact number of zones needed. In the future, GardeNet will supply a Wi-Fi version to directly compete with competitors. At current estimates, the retail price of the Wi-Fi system will be around $100 plus $25 per zone. Figure 6 shows how the Wi-Fi system would be price competitive.

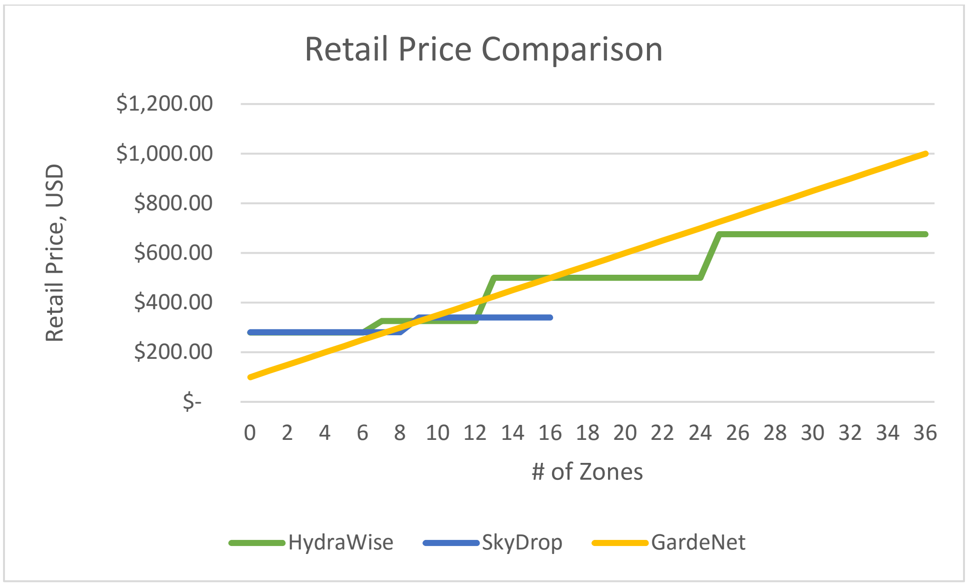

Figure 6: Retail Price Comparison of Automated Watering Systems

At lower zone amounts, the GardeNet system beats competitors by up to $150. However, the other companies have been in business for a while and have found ways to cut costs for a larger amount of zones. Because GardeNet doesn't distribute in a tier format, the retail price grows linearly with the amount of zones and is most competitive at a low number of zones.

8.1.2 Market Survey

The market for the GardeNet system consists of home gardens up to large community gardens and urban farms. Given the increase in community and urban gardening, and the stable market of home gardens, the teams is comfortable estimating sales of 500, 1000, and 2000 units in the first three years. Already, two community gardens are interested in the GardeNet system.

9 Cost Estimate

9.1 Development Costs

If the result of this project were to be taken to market, developing a professional, sale-ready product would drive several new costs. Three primary factors drive these new costs: design, test, and administrative costs. All costs are to be considered rough order of magnitude (ROM) estimates.

9.1.1 Design Costs

First, a final product would be designed to use its own circuit boards, not the open source development boards used for this project. That is for two reasons: one, the team does not own the intellectual property of the design of the development boards; and two, the development boards contain several components that the product does not use. Further, the design would have to be refined to a professional level.

According to an *IEEE Spectrum* article from 2010, printing a custom designed printed circuit board (PCB) only costs $10 per board [14]. Converting the design to an integrated circuit (IC), however, would cost $3000 per board [15]. Neither of these include non-recurring engineering costs (NRE) or components to populate the board. For the volumes being produced, developing an IC solution is not worth the cost. Therefore, given three board designs (one for each the controller, valve and sensor clusters, and weather station), four design iterations per board, and a budget of $5 per board for components, there are $180 of parts cost.

NRE in terms of engineering design hours is more difficult to estimate. A low-range estimate for design hours is three months of four team members working 40 hour weeks, or around 4000 hours. If the team's time is valued at $80 per hour, that is $160,000.

9.1.2 Test Costs

Second, a final product would require thorough testing of a complete system in various conditions. That requires several copies of each component, test equipment, and a test garden. In order to test the system properly, the team would need at least three controllers, 100 valve and sensor clusters, and three weather stations. At \$15 per board, \$180 per 3G modem, \$40 additional per valve and sensor cluster, and \$100 additional per weather station, the total cost is $3 \times (\$15 + \$180) + 3 \times (\$15 + \$100) + 100 \times (\$15 + \$40) = \$6430$. If the team use their backyards as test zones, they would still need pipes and water to test with; thus if the team has three test facilities running a total of 100 gallons per day through 5000 feet of line for a period of three weeks, the total cost is $100 \frac{gal}{day} \times \frac{\$2}{750gal} \times 21days + 5000ft \times \frac{\$65}{500ft} = \$655$. Thus the total testing costs are \$7085.

9.1.3 Administrative Costs

Third, developing a final product would be pointless without a business backing it up, and businesses require administration, floor space, utilities, and more. Assuming the team does its own administration, utilizes its basements and garages as workspaces, continue to use exclusively open source software, and uses their own utilities, they would only be optionally paying for additional time worked and for additional used utilities; thus the costs would be approximately \$15,000 additional labor and at most \$1000 additional utilities.

9.1.4 Total Development Costs

Summing the design, test, and administrative costs of bringing the product to market results in approximately \$8265 in design and test as well as \$160,000 worth of labor. By including the administrative costs and rounding up expenses to account for unforeseen troubles, the total costs are approximately \$185,000. Note that for the sake of the business plan, the team is assuming that this money would be raised through family, friends, and crowdfunding, and would not have to be repaid during the first three years of operation while the business becomes profitable.

9.2 Production Costs

Production costs consist of fixed costs such as facilities, insurance, and legal fees, and variable costs such as such as parts production, labor, and sales commissions. The financial analysis is based on a three-year operating period and a manufacturing and sales volume of 500 units during the first year of operations and 100% growth rate in the next two years. For simplicity considerations, the effect of inflation is neglected.

9.2.1 Fixed Costs

9.2.1.1 Equipment and Facilities

In order to provide technical support to the customers and be able to debug the products before delivery, the venture needs to have necessary electronic test equipment such as oscilloscopes, power supplies, multimeters, function generators, and various probes and accessories. According to the offers on an online electronics retailer, the costs of one set of equipment are: $300 for a bench top oscilloscope, $330 for a power supply, $450 for an industrial-standard multimeter, $750 for a function generator, and an additional $200 annual replacement costs for probes, test jacks, bread boards, jumper wires, and various kits [16]. The total cost is then $1830 per set of equipment. Assume that the GardeNet venture needs two sets of these electronic equipment, the total costs on equipment are $3460. Equipment maintenance costs for the next two years are estimated to be $1,000 assuming $200 annual replacement costs of the equipment and $800 for other needs in this area.

The cost of maintaining a facility is neglected because the business is modeled to utilize the team's garages and basements in the first few years while sales volumes remain small.

9.2.1.2 Insurance and Taxes

While it is difficult to accurately define the industry GardeNet falls into, a sample quote from TechInsrance for an independent contractor/web designer can be used as a reference [17]. For this kind of small business, Commercial General Liability costs between $425 to $900 and is limited to bodily injury or property damage. The insurance company also offers other specialized liability insurance such as worker's compensation and professional liability, but they are more expensive. To the bare minimum, the annual insurance cost is estimated to be $900.

As for the federal and state taxes, according to a document released by the Federal Government's Small Business Administration in 2009, the estimated average effective tax rate of small business is at

19.8% [18]. To have a more realistic estimation, assume that the GardeNet venture will be profitable, and the income tax is 40%.

9.2.1.3 Legal Fees

Legal fees vary from company to company, but for a small business such as GardeNet, the example of the entrepreneur Guy Kawasaki from Truemors is appropriate [19]. For his start-up, he was charged $4,824.13 for the following areas:

- Trademarking
- Drafting a Term of Use
- Discussion of copyright, liability, infringement, IP, and insurance issues
- Organizational resolutions and bylaws
- Stock purchase agreements

Since most of these items will likely apply to GardeNet and the venture will likely need an attorney, the cost of legal fees are estimated to be around $5,000. After the first year, it should decrease to about $1,000, incurred mainly by the cost of attorney.

9.2.1.4 Administrative Costs

Assume that for the first two years of its operation, GardeNet will need a weekly administration of 10 hours; as the sales volume increases, the administration time will increase to 15 hours per week during the third year to meet the new demand. Again set the hourly cost to be $80, the administrative costs are $41,600 for the first two years, and $62,400 for the third year.

9.2.1.5 Total Fixed Costs

In conclusion, GardeNet's total fixed costs are shown in Table 11. On top of this, the business should expect a 40% income tax if it is profitable.

Table 11: Total Annual Fixed Costs of the GardeNet Venture

YEAR OF OPERATIONS	**TOTAL FIXED COSTS**
1	$51,160
2	$44,500
3	$65,300

9.2.2 Variable Costs

The GardeNet system consists of four main physical components: a main controller, valve controllers, and optional valve/flow meter controllers, and a weather station. In addition, the company will offer two versions of the main controller: an original version compatible with 3G cellular network and a cheaper version compatible with Wi-Fi. Table 12 shows the costs and price for each of these components. Note that the overhead was calculated by increasing raw material costs by 10%, and is meant to cover shipping and warranty, and "main controller" refers to the 3G connected controller while "WiFi controller" refers to the Wi-Fi connected controller.

Table 12: GardeNet Component Costs and Prices

Component	**Costs**						
	Development	*Production*	*with Overhead*	*Labor*	*Utilities*	*Total Variable Cost*	*Price*
Main Controller	$ 246.44	$ 198.18	$ 218.00	$ 8.00	$ 0.10	$ 226.10	$ 400.00
Valve Controller	$ 18.52	$ 12.52	$ 13.77	$ 8.00	$ 0.10	$ 21.87	$ 25.00
Valve/Flow Meter Controller	$ 46.47	$ 37.47	$ 41.22	$ 8.00	$ 0.10	$ 49.32	$ 100.00
Weather Station	$ 103.51	$ 57.54	$ 63.29	$ 20.00	$ 0.40	$ 83.69	$ 150.00
WiFi Controller		$ 40.06	$ 44.06	$ 8.00	$ 0.10	$ 52.16	$ 100.00

9.2.2.1 PCB and Parts Outsourcing

Since the venture will be too small to support its own parts manufacturing workshop and assembly line, the production has to be outsourced. Referring to Table 12, component costs are expected to be lower in the production phase than those in the development phase assuming reasonable cost reduction when mass produced.

9.2.2.2 Labor

Assume the hourly cost of labor is $80 and it takes 0.25 hour to manufacture a weather station and 0.1 hour to manufacture one of the other system components, the corresponding labor costs are shown in Table 12.

9.2.2.3 Production Volume and Product Costs

Since the customers' requirements on watering zones will very likely differ, the average case should be considered when estimating production costs, as shown in Table 13. In summary, a typical GardeNet system costs $387.87 dollars. The annual production costs are shown in Table 14.

Table 13: Typical GardeNet System Costs and Prices

Component	Number	Price	Cost
Main Controller	0.4	$ 160.00	$ 90.44
Valve Controller	8	$ 200.00	$ 174.97
Valve/Met Contr	1	$ 100.00	$ 49.32
Weather Station	0.5	$ 75.00	$ 41.84
WiFi Controller	0.6	$ 60.00	$ 31.30
Total		$ 595.00	$ 387.87

Table 14: Annual Production Costs of the Typical GardeNet System

Year	Production Volume	Total Cost
1	500	$ 193,935.00
2	1,000	$ 387,870.00
3	2,000	$ 775,740.00

9.2.2.4 Customer Service and Warranty

Costs incurred in this area is included in the 10% overhead as demonstrated in Table 12.

9.2.2.5 Sales Commissions and Shipping Costs

Assuming that sale takes two hours of processing, and the corresponding cost is $80 per hour, the annual sales and administrative costs are then $80,000 for the first year, $160,000 for the second year, and $320,000 for the third year.

9.2.2.6 Total Variable Costs

The total variable costs of the business are shown in Table 15.

Table 15: Total Annual Variable Costs of the GardeNet Venture

YEAR OF OPERATIONS	TOTAL VARIABLE COSTS
1	$273,935
2	$547,870
3	$1,095,740

9.2.3 Total Production Costs

Based on the analysis of the fixed and variable operating costs, the total annual production costs are $325,095 for the first year, $592,370 for the second year, and $1,161,040 for the third year.

Table 16: Total Annual Costs of the GardeNet Venture

YEAR OF OPERATIONS	TOTAL COSTS
1	$325,095
2	$592,370
3	$1,161,040

9.3 Profitability

Given the sales numbers of 500, 1000, and 2000 units at the prices listed in Table 12, GardeNet would have a revenues (after taxes) and profits as listed in Table 17. This shows a net loss during the first year, but profitability during the second and third years of operation.

Table 17: Revenues, Costs, and Profit of GardeNet in the First Three Years

YEAR OF OPERATIONS	TOTAL REVENUE	TOTAL COSTS	NET PROFIT
1	$297,500	$325,095	($27,595)
2	$595,000	$592,370	$2,630
3	$1,190,000	$1,161,040	$28,960

10 Conclusion

The previous sections outline the method of approach, system requirements, system architecture, design alternatives, and business plan for a 3G connected garden watering system to help large gardens and small farms, implemented with a frontend website, a cloud service, an onsite controller, and wirelessly controlled sensors and actuators. If the team can develop a system to requirements with integrity, trust, humility, and stewardship, then they will have a valuable new system that brings new capabilities to the market. Given the design choices, particularly with the attention given to ease of use and development, the team is confident that the system can be implemented on schedule. Given the competitive situation outlined in the business plan and the costs outlined in the cost estimate section, the team feels confident that given the right circumstances, funding, and word of mouth marketing, GardeNet could be a profitable venture.

11 References

[1] "Systems Engineering Process," Wikipedia, [Online]. Available: https://upload.wikimedia.org/wikipedia/commons/thumb/e/e8/Systems_Engineering_Process_II.svg/420px-Systems_Engineering_Process_II.svg.png. [Accessed 14 11 2015].

[2] C. Bormann, "CoAP," [Online]. Available: http://coap.technology/. [Accessed 07 11 2015].

[3] "Exosite Documentation," Exosite, [Online]. Available: http://docs.exosite.com/coap/. [Accessed 09 11 2015].

[4] "Hypertext Transfer Protocol," Wekipedia, [Online]. Available: https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol. [Accessed 09 11 2015].

[5] "MQTT," MQTT.org, [Online]. Available: http://mqtt.org/faq. [Accessed 08 11 2015].

[6] "MiFi," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/MiFi. [Accessed 09 11 2015].

[7] "Bluetooth Marketing," bluAir, [Online]. Available: http://www.bluair.pl/bluetooth-range. [Accessed 12 11 2015].

[8] "ZigBee Wireless Standard," DIGI, [Online]. Available: http://www.digi.com/resources/standards-and-technologies/rfmodems/wireless-zigbee. [Accessed 10 11 2015].

[9] "LoRa Technology," LoRa Alliance, [Online]. Available: https://www.lora-alliance.org/What-Is-LoRa/Technology. [Accessed 12 11 2015].

[10] "nRF24L01," Nordic Semiconductor, [Online]. Available: http://www.nordicsemi.com/eng/Products/2.4GHz-RF/nRF24L01. [Accessed 12 11 2015].

[11] NimbeLink, [Online]. Available: https://nimbelink.com/skywire-evdo/. [Accessed 10 12 2015].

[12] S. LLC, "SkyDrop - Smart Sprinkler System," SkyDrop, 2015. [Online]. Available: http://www.skydrop.com/product/. [Accessed 13 November 2015].

[13] S. LLC, "SkyDrop Expansion," SkyDrop, 2015. [Online]. Available: http://www.skydrop.com/product/skydrop-expansion/. [Accessed 13 November 2015].

[14] J. Turner, "Build a Custom-Printed Circuit Board," IEEE Spectrum, 31 March 2010. [Online]. Available: http://spectrum.ieee.org/geek-life/hands-on/build-a-customprinted-circuit-board. [Accessed 11 11 2015].

[15] S. Elder, "The REAL Cost for a Custom IC," Planet Analog, 14 5 2013. [Online]. Available: http://www.planetanalog.com/author.asp?section_id=526&doc_id=559840. [Accessed 11 11 2015].

[16] "MCM Electronics," [Online]. Available: http://www.mcmelectronics.com/. [Accessed 12 11 2015].

[17] "TechInsurance Sample Quotes," TechInsurance, [Online]. Available: http://www.techinsurance.com/sample-quotes/independent-contractor/. [Accessed 13 11 2015].

[18] L. Quantria Strategies, "Small Business Administration," April 2009. [Online]. Available: https://www.sba.gov/sites/default/files/rs343tot.pdf. [Accessed 12 11 2015].

[19] G. Kawasaki, "guykawasaki.com," 5 June 2007. [Online]. Available: http://guykawasaki.com/482413_for_lega/. [Accessed 13 11 2015].

[20] HydraWise, "HydraWise," HydraWise, 2015. [Online]. Available: https://hydrawise.com/pricing/. [Accessed 13 November 2015].

12 Acknowledgements

Team GardeNet would like to acknowledge the support of several people without whom this project could not be a success: professor and advisor Mark Michmerhuizen, mentor Kurt Dykema of Twisthink, industrial advisor Eric Walstra of Gentex, component advisor Scott Kerstein of Arrow Electronics, co-chair David Benjamin of Caledonia Community Garden, town center manager Kyle VanEerden of Eighth Day Farm, and the countless others who have supported the project in one way or another.

13 Appendix I: Requirements List

1. Power
   1.1. Input Power Type
      1.1.1. Primary power source shall be 110-120 VAC
      1.1.2. Should 110-120 VAC be unavailable, the user is responsible for providing a power source
      1.1.3. It will be a stretch goal to make the system function solely on 12 VDC
      1.1.4. It will be an additional stretch goal to design a solar power system capable of providing reliable power for the system
   1.2. Power Quality
      1.2.1. Surge Protection—power surges shall not affect the system
      1.2.2. Power Failure—the system shall respond to loss of power in a predictable and recoverable manner
   1.3. Power Modes
      1.3.1. On—the system shall have simple power on procedure and be oriented towards minimizing power consumption
      1.3.2. Sleep—the design shall have "sleep mode" for extended periods when the controller will not be required for control or data collection
      1.3.3. Off—the design shall have a simple power down procedure.
2. Water Supply
   2.1. Control—the water distribution system shall be controllable via 12 VDC valves on pipe sizes up to 1.5" and up to 100 PSI
   2.2. Scope—all else regarding water sourcing, pressurization, and distribution is outside the scope of this project
3. Controller
   3.1. Valve Control
      3.1.1. Valve Communications—the valves shall be controlled by the controller via direct RF communication using custom data packets
      3.1.2. Valve States—the valves shall have only two states: ON and OFF
      3.1.3. Valve Behavior
         3.1.3.1. The valves shall turn on and off based on explicit signals from the microcontroller
         3.1.3.2. If the valves do not receive a "continue" signal from the controller once per five minutes, the valves shall shut off to prevent unwanted behavior if the controller loses power
   3.2. Local I/O
      3.2.1. Button Inputs
         3.2.1.1. The controller shall have the following button inputs: Stop, Start, Delay 30 Min., Power
         3.2.1.2. The buttons shall always be enabled and override cloud controls
      3.2.2. LED Outputs
         3.2.2.1. The controller shall minimally have the following LED outputs
            3.2.2.1.1. Green "running" light: solid if on, blinking if delayed, off if off
            3.2.2.1.2. Red "connection" light: solid if connected to cloud server, blinking if unconnected, off if system is off
   3.3. Sensor Control
      3.3.1. Sensor Communications
         3.3.1.1. The sensors shall send data to the controller via direct RF communication
         3.3.1.2. The controller shall request a reading from the sensor at regular periods depending on the sensor type

3.3.1.3. The controller shall request a reading from the sensor when queried from the cloud

3.3.2. Sensor States—the sensors shall have two states: ON and OFF

3.4. Gateway Operation

3.4.1. Gateway Communications

3.4.1.1. The term "gateway" refers to the device via which the controller communicates with the cloud server

3.4.1.2. The gateway shall operate on cellular data service which balances speed and longevity with economy (e.g., 2G will soon be unsupported, 4G is expensive)

3.4.1.3. Transmission shall be encrypted

3.4.2. Data Transmitted

3.4.2.1. Current State

3.4.2.1.1. The gateway shall update the cloud server whenever it changes state

3.4.2.1.2. State changes are any alteration or deviation from the expected schedule

3.4.2.2. Sensor Data—the gateway shall transmit sensor data when (a) every half hour during standard operation and (b) when queried from the cloud server

3.4.2.3. Control Input—the gateway shall update the cloud when input settings have changed due to onsite user input

3.4.3. Data Received

3.4.3.1. Cloud Input—the gateway shall receive control input from the website or mobile app though the cloud server

3.4.3.2. Query for information—the gateway shall receive queries for data from the cloud server when users go online to check the status of the garden

4. Sensors

4.1. Number of Sensors

4.1.1. Each valve shall have one pressure sensor and one flow rate sensor

4.1.2. Each installation shall have one weather sensor

4.2. Sensor Communications with the controller—sensor shall communicate with the controller via direct RF communication when queried by the controller

4.3. Power—sensors shall run for at least two weeks on battery power. It shall be a stretch goal to generate power via a micro solar panel

5. Valves

5.1. Amount of Valves

5.1.1. There shall be one valve per zone

5.1.2. The system shall handle an arbitrarily large number of zones

5.2. Valve Control Signals

5.2.1. All valve control signals shall be initiated by the controller

5.2.2. If the controller does not send out a "continue" signal at least once per five minutes, then the valves shall shut off in order to prevent flooding in the case the controller loses power

5.2.3. Valves shall be designed so that they have minimum static power consumption

5.2.4. Signals shall be sent wirelessly up to at least 500m

5.3. Valve Capabilities

5.3.1. Flow rate—the valves shall handle all flow rates

5.3.2. Leakage—the valves shall have as little leakage as possible

5.3.3. Pressure—the valves shall handle pressure up to at least 100 PSI

5.3.4. Power—the valves shall run for at least two weeks on battery power. It shall be a stretch goal to generate power via a micro solar panel

6. Cloud Server
    6.1. Rules Engine
        6.1.1. Onsite commands shall overrule online controls
        6.1.2. Rule Types—the user shall be able to specify:
            6.1.2.1. Run for X number of minutes
            6.1.2.2. Run until X gallons of water delivered
            6.1.2.3. Combinations of (1) and (2)
        6.1.3. Design Goals
            6.1.3.1. A fully functioning rules engine will keep the user updated on all things going in in the garden
            6.1.3.2. The rules engine shall not result in "positive feedback" results; runtimes and control signals shall be controlled and predictable
    6.2. Alerts
        6.2.1. Type
            6.2.1.1. Email
            6.2.1.2. SMS text message
        6.2.2. Frequency
            6.2.2.1. Maximum
                6.2.2.1.1. Emails < 20/day
                6.2.2.1.2. SMS text message < 10/day
        6.2.3. Type and frequency shall be modifiable by the user
    6.3. Dashboard
        6.3.1. The "dashboard" feature of Exosite (or similar cloud service) shall be present but shall be the secondary viewing option to the website interface
7. Website and Mobile App
    7.1. Views
        7.1.1. Administrator View—shall contain:
            7.1.1.1. All components available in the Passive View
            7.1.1.2. Interface for control
        7.1.2. Passive Viewer View—shall contain:
            7.1.2.1. Overall system state
            7.1.2.2. Historical usage data
    7.2. Controls
        7.2.1. Interface to monitor and control individual valves
        7.2.2. Master switch to control system power
        7.2.3. Interface to setup watering schedules
    7.3. Usability
        7.3.1. Website
            7.3.1.1. Detailed and aesthetically pleasing interface
            7.3.1.2. Shall be usable on first attempt by a new user
        7.3.2. Mobile Devices
            7.3.2.1. Shall have simplified version of website functionality
            7.3.2.2. Shall consist minimally of a mobile-friendly version of the full website
            7.3.2.3. It shall be a stretch goal to develop a standalone app

14 Appendix II: Work Breakdown Structure

Table 18: Work Breakdown Structure with Hours Percent Completed

Section Number	***Title***	***Description***	***Time Estimate***
1.0	***Develop Requirements***		20 hours, 100%
1.1	***Power***	Choose the kind of power used for the system and develop "modes" for efficiency	20 hours, 100%
1.2	***Water Supply***	Determine the kind of source, the output flow rate, and quality of water	10 hours, 100%
1.3	***Controller***	Find a controller to control the LCD, sensors, and valves and provide sufficient gateway operations.	20 hours, 100%
1.4	***LCD***	Choose a user friendly GUI on the LCD and provide communications to controller	20 hours, 100%
1.5	***Sensors***	Find sensors that can detect rainfall and give data to the controller	20 hours, 100%
1.6	***Valves***	Use valves that can take in control signals in order to operate with the system	20 hours, 100%
1.7	***Website--Server***	A cloud server is needed that can handle large amounts of data sent between the controller and the website	20 hours, 100%
1.8	***Website--User Interface***	The user interface of the website must be able to handle scheduling, commands, and alerts to the user	10 hours, 100%
2.0	***Design***		240 hours, 4.6%
2.1	***Implement Requirements***		150 hours, 0%
2.1.1	***Hardware Installation***	Connect the controller to the valves and sensors	30 hours, 0%
2.1.2	***App Development***	Configure an application to control the controller from the cloud	30 hours, 0%
2.1.3	***Enable remote control***	Remote control connected to controller to send signals to valves and sensors	30 hours, 0%
2.2	***Low Level Debug***		180 hours, 0%
2.2.1	***Functionality***	Actively reconfigure system so that the system remains functional	50 hours, 0%
2.2.2	***Communication***	Fix communication errors that may occur internally and externally	60 hours, 0%
2.2.3	***Security and Reliability***	Create dedicated admin users to make all scheduling decisions	10 hours, 0%
2.3	***Purchase Materials***		20 hours, 54%

2.3.1	***Development Boards***	Purchase the researched development board	20 hours, 80%
2.3.2	***Hardware***	Purchase all other necessary hardware	20 hours, 33%
2.3.3	***Others***	Purchase miscellaneous things to operate the system	20 hours, 50%
2.4	***Website***		100 hours, 10%
2.4.1	***Admin/Viewer Use***	Create different views of the website for admin and viewer use. Viewers can only see data from garden. Admins can send data to the controller.	90 hours, 0%
2.4.1.1	***Garden stats***	Real-time stats sent to the viewer and admin	30 hours, 0%
2.4.1.2	***Remote Control***	Control the garden valves from the website.	60 hours, 0%
2.4.2	***About team***	Create a website about the team	10 hours, 100%
2.4.2.1	***339 Requirements***	Make sure the website contains all necessary requirements by the Engineering 339 class	10 hours, 100%
3.0	***Test***		30 hours, 0%
3.1	***Test Bottom Level Requirements***	Test website interface usability, test website interaction with cloud, cloud interaction with controller, controller interaction with devices	10 hours, 0%
3.2	***Test Middle Level Requirements***	Test website control of individual devices (e.g. successful data collection, able to open/close valves)	10 hours, 0%
3.3	***Test Top Level Requirements***	Test in actual use-case--with running water, controlled remotely	10 hours, 0%
4.0	***Refinement***		50 hours, 0%
4.1	***Troubleshoot Bugs***	Root out problems found in testing	10 hours, 0%
4.2	***Improve Scalability***	Improve code and system design for maximum generalization	50 hours, 0%
4.3	***Include WOW Factors***	Add special features to happily surprise clients	50 hours, 0%
5.0	***Develop Documentation***		10 hours, 0%
5.1	***Technical Documentation***		10 hours, 0%
5.1.1	***Functional diagrams***	Draft diagrams showing the functional workings of the system	4 hours, 0%
5.1.2	***Schematics***	Show schematics, pseudocode, and full system specifications	2 hours, 0%
5.2	***User Documentation***		10 hours, 0%
5.2.1	***Website***		2 hours, 0%

5.2.1.1	***Use***	Have website map and describe intended usage	2 hours, 0%
5.2.1.2	***Maintenance***	Hand over website control	2 hours, 0%
5.2.2	***Controller & Device***		6 hours, 0%
5.2.2.1	***Installation***	Describe installation process	2 hours, 0%
5.2.2.2	***Use***	Describe proper use	2 hours, 0%
5.2.2.3	***Maintenance***	Describe necessary maintenance (e.g. safe power-down)	2 hours, 0%
5.2.2.4	***Storage***	Describe portions that require special long term winter storage	2 hours, 0%
6.0	***Onsite Installation***	Install	3 hours, 0%
7.0	***339 Requirements***		218 hours, 100%
7.1	***Work Breakdown Structure***	Identify and quantify tasks, put into schedule	20 hours, 100%
7.2	***Project Brief***	Prepare document introducing project to an industrial reviewer	10 hours, 100%
7.3	***Oral Presentation I***		8 hours, 100%
7.3.1	***Project Overview***	Present to the class about the project in its beginning stages	8 hours, 100%
7.4	***Oral Presentation II***		16 hours, 100%
7.4.1	***Progress Report***	Update given progress since last presentation	8 hours, 100%
7.4.2	***Draft***	Prepare draft	8 hours, 100%
7.4.3	***Final Submission***	Prepare final version	8 hours, 100%
7.5	***Fridays at Calvin Poster***	Have poster describing the work for high school visitors	4 hours, 100%
7.6	***PPFS***		60 hours, 100%
7.6.1	***Draft***	Write first draft of PPFS	40 hours, 100%
7.6.2	***Final Submission***	Make edits, add sections, and improve on draft PPFS for final submission	20 hours, 100%