UCF Computer Science Senior Design

D.R.O.N.E.S
Damage Reporting On Necessary and Essential Structures

Angelo Montagner
Christopher Grenci
Ernest Wheaton
Contents
1. Project Summary ........................................................................................................... 1
  1.1 Project Description ..................................................................................................... 1
    1.1.1 Drones for inspections ....................................................................................... 1
    1.1.2 Help with safety ............................................................................................... 1
  1.2 Project Objectives ....................................................................................................... 2
    1.2.1 Create a system to collect data on a site inspection ............................................ 2
    1.2.2 Report on collected data and display neatly ...................................................... 2
  1.3 Technical approach .................................................................................................... 2
    1.3.1 Android application ......................................................................................... 3
    1.3.2 Database ........................................................................................................... 3
    1.3.3 Analytics Engine ............................................................................................. 3
    1.3.4 Web Portal ....................................................................................................... 4
2. Technical content ............................................................................................................ 4
  2.1 Project Significance and Motivations ......................................................................... 4
    2.1.1 Project Identification ....................................................................................... 4
    2.1.2 Project Significance ....................................................................................... 4
    2.1.3 Angelo’s Motivations .................................................................................... 4
    2.1.4 Chris’s Motivations ...................................................................................... 5
    2.1.5 Ernie’s Motivations ....................................................................................... 5
    2.1.6 Broader Impacts ............................................................................................. 6
  2.2 Technical objectives, goals, specifications, and requirements .................................... 6
    2.2.1 Android application ....................................................................................... 6
    2.2.2 Server ............................................................................................................. 6
    2.2.3 Analytics Engine ........................................................................................... 7
    2.2.4 Web Portal ..................................................................................................... 7
    2.2.5 Goals ............................................................................................................. 7
    2.2.6 Specifications ............................................................................................... 7
    2.2.7 Requirements ............................................................................................... 8
  2.3 Research and investigations ....................................................................................... 8
    2.3.1 Drones Considered ....................................................................................... 8
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.3.2 Programming Languages</td>
<td>9</td>
</tr>
<tr>
<td>2.3.3 Industry Research</td>
<td>19</td>
</tr>
<tr>
<td>2.4 Detailed design content</td>
<td>25</td>
</tr>
<tr>
<td>2.4.1 User interface</td>
<td>26</td>
</tr>
<tr>
<td>2.4.2 Application/User Interface Mockups</td>
<td>28</td>
</tr>
<tr>
<td>2.4.3 Maps</td>
<td>50</td>
</tr>
<tr>
<td>2.4.4 Server</td>
<td>52</td>
</tr>
<tr>
<td>2.4.5 Database</td>
<td>53</td>
</tr>
<tr>
<td>2.4.6 Analytics</td>
<td>53</td>
</tr>
<tr>
<td>Prototype Hierarchical Task Analysis (HTA)</td>
<td>56</td>
</tr>
<tr>
<td>2.5 Explicit Design Summary with diagrams</td>
<td>57</td>
</tr>
<tr>
<td>2.5.1 API Endpoints</td>
<td>57</td>
</tr>
<tr>
<td>2.5.2 Database Structure</td>
<td>67</td>
</tr>
<tr>
<td>2.6 Build, prototype, test, and evaluation plan</td>
<td>68</td>
</tr>
<tr>
<td>2.6.1 Detailed prototyping plan</td>
<td>68</td>
</tr>
<tr>
<td>2.6.2 Detailed Testing plan</td>
<td>69</td>
</tr>
<tr>
<td>2.6.3 Testing Rubric</td>
<td>69</td>
</tr>
<tr>
<td>2.7 Setting up Android Programming Environment</td>
<td>71</td>
</tr>
<tr>
<td>2.7.1 Setting up DJI SDK Permissions</td>
<td>75</td>
</tr>
<tr>
<td>2.7.2 Getting a DJI developers Key</td>
<td>76</td>
</tr>
<tr>
<td>2.8 Facilities and Equipment</td>
<td>77</td>
</tr>
<tr>
<td>2.8.1 Drones Considered</td>
<td>77</td>
</tr>
<tr>
<td>2.8.2 Testing areas</td>
<td>83</td>
</tr>
<tr>
<td>3. Administrative content</td>
<td>84</td>
</tr>
<tr>
<td>3.1 Budget</td>
<td>84</td>
</tr>
<tr>
<td>3.2 Milestones</td>
<td>86</td>
</tr>
<tr>
<td>3.3 Project Summary</td>
<td>87</td>
</tr>
<tr>
<td>4 Questions and Concerns</td>
<td>88</td>
</tr>
<tr>
<td>4.1 Weather Concerns</td>
<td>88</td>
</tr>
<tr>
<td>4.2 Legal Concerns</td>
<td>88</td>
</tr>
<tr>
<td>4.3 EULA Considerations</td>
<td>89</td>
</tr>
</tbody>
</table>
5. References ................................................................................................. 90
1. Project Summary
This project is student funded and student led project. The project is devoted to
devising a method whereby, drones can be safely automated, by the technical
definition, not the cultural definition. This project does not seek to create a self-
aware drone, rather to create a drone that, given a set of instructions, will
execute those instructions. The project will use this drone to assist in inspections
of buildings. The main intent of this use will be in attempting to make the
workplace safer.

1.1 Project Description

1.1.1 Drones for inspections
Our project will utilize the DJI drone and Android platforms to assist with
structural inspections. The project has been split into several parts. These parts
will have their own “project leads”. These project leads will be team members in
charge of all logistical portions of the part of the project. The project is divided as
such, drone, application, and Web services. The drone will be handled by
Angelo Montagner, the application will be handled by Ernest Wheaton and the
Web services will be handled by Chris Grenci. The project parts will come
together to automate, with supervision, inspections. These inspections will be
controlled by an android application that will define the inspections and send all
reports to the Web service and the Webster vice will display the reports. The
user will then be able to view these reports. The application will initially be on
android, and written in java. The server will be on a school virtual machine, until
such a time that a school virtual machine is no longer available or advantageous
to the team, and written in golang. Our project will utilize the android
development tools, such as, Android Studio and the Android
Ubuntu server distribution and the DJI drone platform.

1.1.2 Help with safety
The Project was designed as an answer to companies attempting to cut down on
costs due to employee injury and increase workplace safety. The project was
presented and incepted with employee safety in mind. Our primary goal for the
project is to decrease workplace injuries as a result of dangerous conditions in
buildings to be inspected. These injuries are harmful to workers, obviously, as
well as, to the companies that are the employers to the injured. According to the
OSHA website, the top 4 causes of workplace death and injury are falls,
electrocutions, struck by objects and caught in between. Falls alone account for 40% of the workplace deaths in America and occur almost exclusively in construction and inspections.[1] This is an unacceptably high number and this project seeks to help reduce it.

1.2 Project Objectives

We have two primary objectives for our project. Firstly, we are going to create a system to automate the inspections of construction sites and other similar applications that require regular or frequent maintenance. Secondly, we will format and analyze the data our application collects to give our users an accurate and quick way to determine any issues on their work site.

1.2.1 Create a system to collect data on a site inspection

The system will use the DJI drone camera and GPS locator to collect data from the inspection site. The camera will be used to capture picture data at predefined point. The GPS locator will be used to capture latitude, longitude and height data at picture capture points. This is to allow for easy reporting, as the pictures will be stored according to a hash function of the GPS location. The system will be controlled by an android application that will store information to the device until the end of the inspection. The data will then be transferred to the database.

1.2.2 Report on collected data and display neatly

The system will be able to retrieve the data from the database to the android device. The data retrieved will be displayed in a user friendly way, so that the user will be able to understand the data. The data will be displayed as a list of images, all clickable for resizing, identified by their GPS locale and date. The data will also contain, if the user wishes, some form of image composition of the images, taken at the same position over multiple dates, which will represent the change over time of the picture. The data will then be removed from the device, to conserve space on the device.

1.3 Technical approach

In order to create a fully operational system for inspections using drones, we are building an application that will handle most possible scenarios. This system will be primarily comprised of the following three parts: a mobile application for the Android operating system for communication with the drone, a database for
storage and analysis of the data collected by our users, and a website interface to display the collected data and analytics.

1.3.1 Android application

The Android application will be the core of our system. The application will give users a usable User Interface so that they may traverse the application. Through this User Interface, the user will be able to initiate all actions. These actions include, accessing previous inspections stored on the database, access scheduled inspections and create flight plans. Users will be able to create flight paths around their work site. On these flight paths they will be able to designate locations where the drone will take a picture and log data. When the flight is completed, the application will then generate a preliminary report based on the collected data, which will then be sent to the database for a more thorough analysis. This analyzed data may be accessed through the User Interface and displayed to the screen. The user will also be able to plan and access previously planned inspections, and execute these plans.

1.3.2 Database

The database element of our system will complement the Android application by managing the storage and retrieval of collected data. It will sanitize and then store the raw report data received from the Android application. It will periodically check for new raw data, which will then be retrieved and sent to the analytics engine for analysis. The analyzed data will be stored in a part of the database separate from the raw data. It will be capable of serving both raw and analyzed reports to the web portal upon request. The reports will include location data, timestamp, pictures, and the flight plan the report was generated from.

1.3.3 Analytics Engine

The server side portion of our system will include an analytics engine for providing our users with meaningful data from the inspections they execute. It will use computer vision and machine learning algorithms to identify regions of the images that contain important objects and then determine if those objects need maintenance action or further human inspection. The analytics engine will be run upon receiving new inspection data from the Android application, and the results of the analysis will be stored as a report in the database for on-demand retrieval by our users.
1.3.4 Web Portal
The web portal will display the analyzed data received from the database in a neat and helpful fashion. Users will be able to identify problems on their work site at a glance, with more detailed information available to them for any section they desire.

2. Technical content
This section will include all specifics of the project. Everything programming languages considered to the specifics of the User Interface is included below. We will be using these specifics as the framework for building our project.

2.1 Project Significance and Motivations

2.1.1 Project Identification
This project is called D.R.O.N.E.S. This acronym means Damage Reporting On Necessary and Essential Structures, which explains our project perfectly. The project goal is to create a medium on which companies can do basic inspections and reporting on structures. This reporting includes pictures, coordinates and height. The project would be considered an extra tool for a building inspector.

2.1.2 Project Significance
The project will allow for safe observation of structures that normally have people physically inspecting them. The project will also use analytics on data collected to determine if human inspection is necessary on the structure. This allow for less waste of human resources and less risk of the inspector incurring injuries to themselves due to workplace injuries.

2.1.3 Angelo’s Motivations
Drones have always been a topic of interest. Creating an application around drones enables me to apply my knowledge of programming to an area in need of assistance. I first became interested in drones when I started experimenting with Internet of Things (IoT) devices. Building applications for IoT devices involves thinking outside of the box to create something useful and feasible. Drones also have a large number of use cases that are also very interesting and many are well on their way to make changes in the industry.

My main goal is to contribute on a long term project and get invaluable experience working on a team. This project also allows me to get hands on
experience on the lifecycle of an application. I will like to see what it takes to go from an idea to a fully developed project. Ideally, I will like to put what I learned so far in school into practice. Primarily, I will like for our team to create a product that has been properly made and adjusted to a real use case. In this case, Drones will be used to help inspectors do their job and do it in a safe way.

2.1.4 Chris’s Motivations

When this project was presented, I saw the idea of inspection safety improvements as more of a guiding factor for an interesting experiment in programming AI and Computer Vision tasks related to drone operation. Robots in a wide variety of forms are becoming increasingly pervasive throughout our society and I believe the AI technology controlling them will only become more interesting and complex over time. I believe that a major push for a world full of robotics technology is just beginning, and I want to contribute whatever I can to making it happen. I see this project as the first steps on the path towards a graduate degree and eventual career developing AI for robotics.

2.1.5 Ernie’s Motivations

This project was presented to the class as a way to reduce the risk to engineers performing routine structural maintenance. Also, this project presents a way to bridge the gap, in my mind, between software and hardware. The challenges faced by engineers attempting to assess damage done to structures are, especially tall structures, are paramount. They must wear heavy protective equipment, climbing and safety gear and cameras or other reporting equipment. This presents many challenges by itself, while also, putting the engineering life at risk. I saw this project as an opportunity to not only help make engineers’ lives safer and easier, but to also possibly help create a company based off this idea. I also saw opportunities to expand my knowledge in areas that I have little to no experience in. I enjoy expanding my knowledge into new areas, especially areas that will have an impact on my professional life. One of these areas is drone programming. Because of drones' multifaceted use, many companies are starting to invest in and research for drone applications. This is an excellent environment to start a company and I esteem it prudent to have experience in helping start a company. Working on this project will also expand my professional interaction knowledge. I volunteered to be the project manager for the team, and this is a new experience for me. A project of this scope will give me a relatively safe arena to learn about project management and how to professionally interact with my peers. My personal goals for this project include learning, hands on, about an emerging technology, gain experience in starting a startup and reducing workplace health risks.
2.1.6 Broader Impacts
We hope our project will provide the inspections industry with a new tool to safely conduct their inspections. These inspections are potentially dangerous and require an engineer or inspector to place themselves in these situations. Our project hopes to reduce these risks by removing the person from the potentially dangerous situation. As mentioned in the beginning of this paper, falls are responsible for 40% of workplace injuries, which occur almost exclusively in the construction industry. This 40% translates to almost 350 deaths a year.[1] This project aims to reduce this number, even a little. Other ways our project could be impactful, is in environmental monitoring. The project will measure change between pictures and could be used in environmental studies.

2.2 Technical objectives, goals, specifications, and requirements
The Technical objective of this project is as such: To create a system to perform aerial inspections using quadcopter drones via the development of an efficient communication between an Android application, Server, Database and Analytics Engine.

2.2.1 Android application
The Android application will be developed in the android studio environment, along with the Java SDK and Android SDK. The objective of this application is to take user inputs and translate them to GPS coordinates that will direct the drone where to go and what to do. The application will then gather all information from the drone, such as, pictures and coordinate visited. The application will locally store for temporary viewing and eventually sending off the data to a server. The application, as it stands, will not download the pictures to the main storage of the phone, rather, to a file in the program's memory. This will enable the application to track what files are its own and delete all of these images at the shutdown of the application.

2.2.2 Server
The Server expose a Responsive State Transfer API for use by the other modules of the project. It will provide user and request authentication to ensure the security of user data. It will communicate with a MySQL Database for data storage. This database will store the information gathered from the inspections, also referred to as a flight plan in this document, and make available for viewing
from the mobile client and from the web portal. These interactions will be handled by golang.

2.2.3 Analytics Engine

The Analytics Engine will produce reports based on data collected by the Android application. It will use several methods to provide an accurate summary of the inspection site. It will use Machine Learning algorithms to further refine the results it generates over time.

2.2.4 Web Portal

The web portal, should the team deem it necessary to develop, will be a standard login page, that the user will enter their credentials that have been registered in the application. After correctly entering the login information, the user will then be shown, is specific discrete areas, their future inspections and their completed inspections with the respective reports. These reports will include all data gathered from the flight and displayed in order to facilitate an easy understanding of the data. This Web Portal will be a copy of the Android application, rather than the other way around because the experience is mainly devoted to the Android platform. For an idea of what the Web Portal would look like, refer to the mockups of the application.

2.2.5 Goals

The goals of this project are creating a safer work environment for people in the inspections and create a robust, new technology in a new, growing technology area.

2.2.6 Specifications

1. The project will use the Android Application Platform for mobile development
2. The project will use a school supplied virtual machine for server hosting
3. The project will use Java for the mobile application programming language
4. The project will use Go for the server side programming language
5. The project will use a Unix based Operating System for the server
6. The project will use the android libraries for network communications
7. The project will use the DJI SDK for drone communication
8. The project will use the DJI Phantom 3 Advanced drone as the testing drone
2.2.7 Requirements

1. The android application shall be able to send commands to drone
2. The android application shall be able to save pictures and metadata from drone
3. The android application shall be able to send collected data to server
4. The server shall be able to receive and store raw data and analyzed reports efficiently
5. The server shall be able to determine when new raw data needs to be analyzed
6. The analytics engine shall be able to create meaningful reports from data
7. The web portal shall be able to display reports in efficient, human-readable form.
8. The android application shall make the user aware of the existing laws and advise them to follow the above mentioned laws

2.3 Research and investigations

This section will explain and support all decisions made about the project specifications and requirements. Not all hardware and software is created equal, and as this section will show, the team had to test and research many options before we selected, what the team deemed as, the best fits.

2.3.1 Drones Considered

We have researched into many possible drone platforms and their advantages and disadvantages. After researching which drone companies make the best consumer drones. The companies we decided were the best fit for our project were Parrot and DJI. These companies both provide relatively inexpensive drones with customizable SDKs and both a strong ecosystem of developers for the drones. Both drones' SDKs also supported both iOS and Android. This was a huge selling point to us, as multiplatform support would be a huge advantage to us. The drone we first researched was the DJI Matrice 1000, which is the DJI’s developer kit drone. This drone is completely customizable, with upgrades to drone processing power, indoor wall detection and different cameras available. Unfortunately, this drone is priced at $4000 dollars, which is beyond this project’s current budget. Next, we decided to research and obtain a Parrot Bebop drone. This drone is the most inexpensive drone we considered and we viewed it as the perfect candidate for a prototype. However, we discovered, throughout the process of researching for the prototype, that the SDK was missing some very important features and that it was also incomplete. Also, the developer's had
devoted more time to the development of the iOS SDK and the Android SDK was severely under documented. These disadvantages led us back to the DJI Phantom 3 Advanced. This drone is a little more expensive than the Parrot Bebop 2, coming to be around $1000. This drone has a fully supported sdk, a better camera that is attached to a gimbal, GPS location services, and wall detection. The DJI SDK also supports forms of autonomous flight.

2.3.2 Programming Languages

The following portion of the document outlines all programming languages that were considered for use in the project, along with reasons we did or did not use them in the final product.

2.3.2-1 Server Development

2.3.2-1.1 PHP

The first language we considered for our server side application was PHP. PHP is a commonly used language for creating server-side programs and APIs and is very well documented. However, we decided against PHP for server side because, as a team, we did not possess much, if any, experience in this language. Also, even though this language is a very popular one for server programming, the team wanted to investigate any new server application languages that may provide the latest or enhanced implementations.

Figure 2.3.2 - 1: PHP Logo [2]
2.3.2-1.2 ASP.NET/C#

The next language we researched was ASP.NET on top of C#. This implementation for server side applications, while older, is less common. We decided against this language because of the lack of complete documentation and the apparent limitations of the language as a server side language.

Figure 2.3.2 - 2: ASP.NET Logo [3]

2.3.2-1.3 Node.js

Node.js is a newcomer to server-side applications. It is an attempt to implement all of the typical server operations in JavaScript. While it has become a popular way to create server-side APIs and applications, it has many limitations that require several additional packages to overcome. For example, without a lot of extraneous code, every function call is created as a new thread. If synchronization is required, it takes a large amount of extra code to guarantee. We decided against this language because we believe that the limitations of JavaScript and lack of proper thread management was not an optimal solution for the project.

Figure 2.3.2 - 3: Node.js Logo [4]
2.3.2-1.4 Go

Go is another somewhat new language. It is an open-source programming language maintained by Google. Unlike Node.js and PHP, it is a compiled language. It was designed for both easy development and fast compilation. We chose to use it because it provides almost all of the functionality we require as part of its standard packages. This includes http request handling, image manipulation, and database communication. We have also found several non-standard packages that provide increased functionality for our needs.

Figure 2.3.2 - 4: Golang Logo [5]

2.3.2-2 Authentication

2.3.2-2.1 OAuth

OAuth2 is a robust system for determining the authenticity of all data transmitted to and by the server. It is a design specification, not a library. Many companies such as Google, Facebook, and Amazon provide API calls to perform authentication using their OAuth protocols. We determined that the limited focus of our project did not require such a robust authentication protocol, as we do not plan to allow users to authenticate from outside services.

Figure 2.3.2 - 5: OAuth2 Logo [6]
2.3.2-2.2 JSON Web Tokens

JSON Web Tokens are a secure way to guarantee the authenticity of a data payload transmitted over HTTP. Each JSON Web Token has three parts: a header, a payload, and a footer. The header simply describes the request as using the JSON Web Tokens format. The payload is where all of the operational data is contained. The footer contains the signing data. We chose to use this because it is a simple way to ensure our data is coming from an approved source. It does not require the large amount of overhead that a protocol such as OAuth requires; but because it is not a protocol itself, it can be easily integrated with additional security methods.

2.3.2-3 Mobile application

2.3.2-3.1 Android Java

Our choice of development platform has limited our mobile development language. The Android Development Kit restricts us to using Java for the mobile application. This works in our favor as all of the project members have at least 2 years of academic experience. Also, Java is a widely used language and there is a plethora of professional and open source help sites and resources. Also, Java has many widely used 3rd party, open source libraries that enable Http data transfer, Json data structuring and many more that we may or may not use in the project. We will be using java to build most of the project, save the server.
Figure 2.3.2 - 6: Android Logo [7]

Figure 2.3.2 - 7: Java Logo [8]
2.3.2-3.2 DJI SDK

The DJI SDK is a software development kit designed for industry application developers for Unmanned Aerial Vehicles (UAVs). The DJI SDK simplifies the application development process by taking care of flight control elements such as attitude algorithms, stabilization, battery management, signal transmission and flight safety, allowing developers to focus on their core application.

Currently, DJI provides three different SDKs:

1. The Mobile SDK for mobile devices
2. The Onboard SDK used to communicate between on-board devices and the flight controller.
3. The Guidance SDK uses along onboard devices to sense and determine obstacles in the surrounding environment.

![DJI SDK Hierarchy](image)

**Figure 2.3.2 - 8: DJI SDK Hierarchy [9]**

<table>
<thead>
<tr>
<th>Targeting Users</th>
<th>Mobile SDK</th>
<th>Onboard SDK</th>
<th>Guidance SDK</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mobile Developer</td>
<td>Embedded Software Developer, Flight controller</td>
<td>Aircraft, Robotics developer, Researcher</td>
<td></td>
</tr>
<tr>
<td>Feature(s)</td>
<td>HD Live Video Feed, Intelligent Navigation and Flight Control</td>
<td>Aircraft real time attitude control, transmitting sensors data</td>
<td>Ultrasonic Ranging, Binocular Obstacle Recognition</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------------------------------------------------</td>
<td>-------------------------------------------------------------</td>
<td>-----------------------------------------------</td>
</tr>
<tr>
<td>Flight Platforms</td>
<td>Phantom 3 Professional, Phantom 3 Advanced, Inspire 1, M100</td>
<td>M100</td>
<td>M100 or generic flight platform.</td>
</tr>
<tr>
<td>User Cases</td>
<td>Aerial Imagery, Intelligent Navigation, Live Video Feed</td>
<td>Autonomous flight, Data collection and analysis, Parking Inspection</td>
<td>Distance Measurement, Visual Tracking</td>
</tr>
<tr>
<td>Required Skill Set</td>
<td>iOS/Android development</td>
<td>C++, C</td>
<td>C++, C</td>
</tr>
<tr>
<td>Tools</td>
<td>XCode/Eclipse, Android Studio, DJI PC Simulator</td>
<td>N1 Assistant Software, DJI PC Simulator</td>
<td>Guidance Assistant Software and Drivers</td>
</tr>
<tr>
<td>Firmware</td>
<td>Phantom 3 Professional, Phantom 3 Advanced and Inspire1 Beta</td>
<td>M100 Beta firmware</td>
<td>Guidance firmware</td>
</tr>
<tr>
<td>----------</td>
<td>-------------------------------------------------------------</td>
<td>-------------------</td>
<td>-------------------</td>
</tr>
</tbody>
</table>

Figure 2.3.2 - 9: DJI SDK Comparison Chart. Details the features of the SDK packages that are available for development using DJI Drones [8]

Figure 2.3.2 - 10: DJI Logo [10]

2.3.2-4 Analytics

2.3.2-4.1 OpenCV

OpenCV is an open-source library that provides many common computer vision functions. It is the industry standard library for writing computer vision applications. It supports a variety of image and color formats. It is most commonly used with C or C++ because the bindings it provides are written in C++, but it is also usable with Java, Python, and MATLAB. It contains a wide
variety of computer vision functions, including several machine learning algorithms. It is mostly geared towards real-time applications [16]. We are considering it for use in the image analysis portion of our project.

Figure 2.3.2-11: OpenCV Logo [16]

2.3.2-5 Database

2.3.2-5.1 MySQL
MySQL is a popular relational database system. It is supported by a wide variety of programming languages and management tools. It is very easy to configure for both usability and security. Like all SQL databases, data is organized into databases and tables. Each table has columns which determine the data to be stored in that table, and tables can be connected using the SQL JOIN function. We chose to use a relational database over a non-relational database because we have a clear view of the data we will be storing and how it will be categorized.
2.3.2-5.2 MongoDB

MongoDB is a popular NoSQL database system. Instead of tables which can be connected using a JOIN function, data is organized into documents. There is no pre-set structure to the database so each document can be different depending on the data to be entered. This allows for a more dynamic approach to data storage, but can be slower to access related data if it is not part of the main document. We chose not to use it because we are using a very structured approach for our data storage.
2.3.3 Industry Research

About: Kespry designs commercial-grade drone systems that allow a business to capture, view and analyze aerial imagery and survey data using an automated and proprietary drone technology. Early Kespry commercial customers include leaders in aggregates, mining, construction and surveying. The Kespry Drone System is an automated commercial-grade solution that allows businesses to easily plan drone flights, capture aerial data, and view high-resolution maps and surveys of construction and mining sites.
Source:

2.3.3-1 Articles/News Research

Article: Michigan testing drones for bridge inspections

Drones have started to get state approval for performing inspections. According to Steven Cook, a Michigan Department of Transportation engineer.

“A traditional bridge inspection for example typically involves setting up work zones, detouring traffic and using heavy equipment,” Cook said. “(Drones) can get in and get out quickly.”
It is estimated that traditional inspections take “eight hours, a crew of four people and heavy equipment at a cost of about $4,600. The same inspection with a drone takes two people just two hours at an estimated cost of about $250”[13]

2.3.4 - Legacy diagrams
These diagrams are the original diagrams that the team drew up for our concept. We are including them in this design document because they provide some lower level information flow concepts that we will implement and they show the direct interactions of portions of the project.
Diagram 2.3.4-1

This diagram shows the overall development process, development blocks and flow of information. Each of these blocks represent either a specific area of development, an important meeting or information flow. The only meeting is the first team meeting and it has no bearing on the information flow of the diagram, and is only included as a reference.
Diagram 2.3.5-2

This Diagram shows the loose development process and information flow of the drone software. The mobile application and output handler blocks are included to help define the flow of information.
Diagram 2.3.5-3
This is the information flow diagram of the mobile application. This is very simplistic because the developer tools are well known and very robust.
Diagram 2.3.5-4
This diagram shows a simplistic development cycle that follows the waterfall method and an information flow chart. The lower three states represent the flow of information between the applications and drone. The upper two states represent the waterfall method that will lead into the development of the drone platform application.
2.4 Detailed design content

The project will be divided into several smaller portions to aid in development. These parts are outlined below, we will only give a brief description here. These parts are: The User Interface, the database, the analytics engine and the connections between them. The User interface will be the portion of the project that handles all interactions between the user and the machine. This is not limited to the mobile User Interface, this portion will include a web portal interface. The database will store all information gathered from the project and required to run the project. The analytics engine will analyze the data, extrapolate from the data and generate reports. The connections between all of the parts will be handled by each part of the project, but because of the complexity of the data, we are devoting a significant amount of time for development of these connections. The details of all portions mentioned above are outlined below.

![Use Case Diagram](image.png)

Figure 2.4-1: Use Case Diagram. This shows all of the functions our application will be performing and the relationship between them.
2.4.1 User interface

The user interface will be different depending on the mode that the application will be in. The application will start in a menu where the user may select what mode to use. There will be a local mapping mode and a global mapping mode. The first mode will have the user take a top-down picture of the general area where the flight path will be planned. This view will allow for the user to plan the flight and point out any obstacles in the terrain. The user will then be able to “draw” a no-fly zone or point to avoid where obstacles exist. The user will also be able to point out specific areas for the drone to take a picture. The point will be travelled to by the drone in the same order that they were placed on the map. These points will be placed by either drag and drop or selecting a point and touching an area of the map. These point will gather latitude and longitude information from the map, then a user will enter an altitude. This altitude will be capped to follow the law. These points will give the user options for the type of picture they would like to take. The drone will be able to take a picture in a user defined direction at a point, at a specified altitude with the camera facing down, also known as a terrain picture, and the drone will be able to fly a maneuver, most likely a circle in the beginning, and the user will specify a time interval that the drone will take the pictures every user defined time slice. The first of the options will be simple. The user defines a point, then defines an angle, relative to north, and defines a height, in meters. All the altitudes will be in meters because of the dji SDK defines altitudes in meters. The next picture option will be just as simple as the first. The terrain picture will have the user define a point, then define an altitude. The final option potentially could be a list of options. This is the list of maneuvers that the drone will be able to perform, with a user defined time interval. This time interval will define when the drone will take pictures, in seconds, along the maneuver. The options for these maneuvers will be a circle with user defined radius in meters, a cylinder with user defined radius and height intervals, and a square with user defined length that will be the arms of the square. The final option will be, if time permits us to develop, to allow the user to define a shape with the time intervals and variable height. The flight plan will be confirmed, then the user will have the option to save the flight plan. Once the flight plan has been executed the application will move into a “reporting” mode, which will display basic analytics and the pictures. These pictures will be viewable from the application. The application will then prompt the user to send the flight data to the user’s account. The global mapping mode will be a similar to the local mapping. This mode, however, will interact with the built-in google maps API. This mode is detailed in the section below, however, this mapping
mode will be overviewed here. The User Interfaces of the two modes will be similar, both allowing for the user to select navigation and picture points. The global mapping mode will get the individual navigation points from the google maps API, instead of a relative GPS location. The following pictures are the mockups for the User Interface design, and while they may not reflect the final User Interface design, they will show the flow of the User Interface. From the splash screen to the screens detailing every action of the drone, this section will give a detailed analysis of what the application will do. Each of the mockups is given their own page, with a descriptive paragraph underneath to give the reader an idea of what actions can be performed and what the purpose of the portion of the application is. These mockups will support the above description with pictures, so as to facilitate an easier understanding of the application and its uses.
2.4.2 Application/User Interface Mockups

Figure 2.4.2 - 1: This shows how the application icon will show up on the device. The icon is subject to change.
Figure 2.4.2 - 2: This is the splash screen that will display upon entry into the application. This will allow time for the application to connect with the database and prepare for the user login or creation of a new user.
Figure 2.4.2 - 3: This is the login screen. The user will have a choice between logging in as a verified user and selecting the option to register themselves as a user. The registration is important, because the application will verify the drone that you are using and will also provide security for the data being stored.
Figure 2.4.2 - 4: Here is the new user registration screen. The user is asked for information used to identify and for contacting purposes. The name and organization is used for user identification and will be displayed on a welcome screen for the user to ensure that they have entered the correct credentials and are fully aware of who they are logging in as. The email will be used as a point of contact for password troubleshooting and other general use.
Figure 2.4.2 - 5: This screen shows the GUI to register a drone or reuse a drone that has already been registered. This screen will run a scanner in the background when the screen is initialized to detect any drones in the area and will generate the available drones accordingly.
Figure 2.4.2 - 6: This is the Drone Registration page. Some of the above information will be filled out if a drone is detected. The info that will not be auto filled, if a drone is detected, is the “Owner”, “Organization” and “Drone Name” fields. If no drone is detected, then the user may fill out some of this information to assist the application in discovering the drone. However, this method is not advised, as the user may not easily obtain some of the information needed.
Figure 2.4.2 - 7: This is the home page for the application. The user will be able to select the “Fly Drone” button which will lead into the menus described below. The other option is to view the planned, or upcoming, inspections.
Figure 2.4.2 - 8: This view shows the options that will be given to the user if they select the “fly drone” button on the main menu. The “Fly to Waypoints” option will take the user to the google maps mode and the “from map” button will take the user to the local mapping mode.
Figure 2.4.2 - 9: This view will be loaded if the user selects the google maps option in the above described view. This will allow users to load past plans they have created, and execute them. Or the user may create a plan from here.
Figure 2.4.2 - 10: This view shows what the local file explorer might look like. The user will select an inspection, then the user will be able to access all pictures and data related to the specified inspection. The user may also be able to search the data by date.
Figure 2.4.2 -11: This view shows the view loaded when a user selects the “Load a Flight Plan” button. The user will select one of the flight plans, as shown above. The application will load the flight plan and related locale.
Figure 2.4.2 - 12: This view shows a user using the search function to find an inspection by name.
Figure 2.4.2 - 13: This is a mockup of what a local mapping might look like. In this screen, the user has yet to press the camera icon, which takes a picture of the local and uses the GPS location of the drone to calculate the GPS locations of the navigation and camera points.
Figure 2.4.2 - 14: This view shows the user adjusting the camera settings. The settings shown above are only placeholder settings. There may be more.
Figure 2.4.2 - 15: This view shows the video options available. The same statement stands for the options as above.
Figure 2.4.2 - 16: This view shows the general settings that may be available for the user to adjust.
Figure 2.4.2 - 17: This is the view that the application will enter once the user has selected the “local mapping” option and taken a top down picture of the locale of the area being inspected. This view will allow the user to accept the picture shown or to retake the picture if the picture is not acceptable for the user’s purposes. This picture will then be used in the next few screens to create a flight plan.
Figure 2.4.2 - 18: This view shows the “no-fly” zones. A user will be able to define areas that the drone will not fly in. These are placed so that the drone will not accidentally drift into places that may damage the drone.
Figure 2.4.2 - 19:
Figure 2.4.2-20: This is the view showing the flight plan with navigation points and a picture point. The navigation points show the beginning and end of this short flight. Also, the navigation point in the middle shows a change in altitude. Not shown here are the menus that pops up when a point is selected. This menu give all the options for the point.
Figure 2.4.2 - 21:
Warning: Please be sure to check weather conditions before flying the drone.
2.4.3 Maps

The mapping portion of the User Interface will be gathered from contextual data or from an outside source. The contextual gathering will be system to send a drone to the middle of a site to be inspected, fly the drone to the maximum height allowed by law and capture an image of the site. The application will then calculate the GPS of every navigation or camera point, relative to the center GPS point, to map out the inspection site. However, these calculations would put stress on the application, as it would be a computationally heavy operation. In our application, we will include documents, or some other way of notification, so that the user may be notified of the slowdown that may occur when the application attempts these calculations. Gathering the mapping data from an outside source would involve us using the android google maps libraries to display the map. This has the advantage of being less computationally heavy on the application and requiring less time to create the plan. However, this option will provide outdated maps, as google maps only updates their images every 1 to 3 years.[14] Both methods have clear advantages and disadvantages. Ideally, we would like to implement both methods to provide users with both solutions. However, we will implement the google maps solution first due to the simplicity of the implementation, then time permitting, we will implement the local mapping solution.
Figure 2.4.3 - 1: This is an early mockup of the mapping interface with the Google maps android API. The yellow points are the above mentioned navigation/picture points. Missing in this mockup are the navigation path, separately defined picture and navigation points and user defined no-fly zones. This mock up provides a general high level picture of the mapping interface. Direct implementation was not the main goal of this mock up. Some of the Ideas from this version are still present in the current version of the project, as mentioned in the User Interface portion of the Detailed design Content.
2.4.4 Server

The server will provide a means for the application to connect and transmit data reliably. The application will send the data over some network. When the server receives this data it will categorize it and store it in the database for future analysis and retrieval. The server will expose a Responsive State Transfer API to provide endpoints for the application and website to communicate with the server. This allows for access to the server without regard for user location, as long as the user has some form of internet connection. The API endpoints and application will package data to be transmitted as a JSON object to be sent with a HTTP POST request. This data will include the user who is submitting the data, lists of GPS coordinates for flight paths, pictures taken at specific points along that path, the date a flight path was executed and the date that a picture was taken. The user will then be able to access all data from all flights that they have executed. This data will be accessible from a website. The data will be displayed as raw and as an analyzed report. This will allow for the user to determine the status of the structure that they are inspecting. The database will then send the collected data to an analytics engine. The server will also provide authentication using the JSON Web Tokens library. Each request sent to the server will be required to have the proper header, body, and signature.

A concern that has arisen, in regards to the server, is that of the school network and need for a mobile application to handle the vpn. The school’s network has limited direct access to the virtual machine. To overcome this, the school has recommended that users wishing to connect to UCF internal computer resources from outside the school network, use the CISCO anyconnect. UCF recommends this application for desktop uses, but the team found an android version. After some testing, this application has become the solution for the UCF vpn issue. As long as we keep our server hosted on the UCF network, a user will have to have download the mobile CISCO anyconnect application. The user will then have to connect to the UCF vpn and sign in with either student or faculty credentials. This requirement is the main advantage to off campus server hosting, but until such a time as we have adequate funding or UCF deems it prudent to not provide such resources, we will continue hosting on the UCF network.
2.4.5 Database

The server will be hosting a MySQL database for storage of user data and reports. The data will be data provided by the user and data collected from the drone. This data consists of GPS coordinates, pictures, flight paths and user account data. The Flight paths are a collection of GPS coordinates stored as they are visited and the user data is email, name, and company. The user data will be used for user identification and user recovery, in the case of lost or stolen passwords. The application will require such data for user login and data viewing.

2.4.6 Analytics

The reports we prepare for our users will be generated by a server-side analytics engine. The server will perform some image processing to show the changes in the pictures to the user. We may also implement a machine learning aspect to help identify problem areas early on. The image processing will compare the images taken at a certain spot and compare the changes over the lifetime of collection. It will then provide a layer in the most recent picture to show where the most change has happened. This change over time view will be helpful for users that wish to monitor possible degradation of structural elements over a period of time, most likely weeks or months, and attempt to provide solutions so as to stop or reverse the degradation of the structural elements.

The entire project can be summarized by the diagram that we have provided below. When attempting to debug or add to the existing project, please use these reference materials to understand the existing flow of information and to place the component in the correct place in the information flow.
Figure 2.4.6-1: Data Flow between System Components
This diagram shows the basic flow of information in our project. This is an updated, condensed version of the diagrams that are presented in the "past diagrams" section. This diagram will also provide the basis of our prototype. The entire project will be built off of this foundation.
Figure 2.4.6-2: This is the class diagram for the initial Android application prototype. This is Android application will directly interact with the DJI Drone. It will consist of three classes which includes the login, main activity, and connector class. The main activity will be the launcher activity containing the initialization of several drone components such as DJI Base Product and InitUI. The InitUser Interface will be the main primary method which interacts with the components from the Connector class.
Our application will make use of Hierarchical Task Analysis (HTA) to layout and plan out user tasks. Primarily, hierarchical task analysis is used to decompose a high level objects into a series of smaller hierarchical objects. In return, the Hierarchical Task Analysis will give us a brief overview of what the prototype will be composed of along with an idea of its main functionalities.

Prototype Hierarchical Task Analysis (HTA)

1. Login Page
   a. New user
   b. Existing user
2. Drone Selection
   a. Select existing Drone
   b. Register new Drone
3. Home
   a. Fly Drone
   b. To Do Inspections
4. Flight
   a. Free Flight
      i. Camera View
         1. Camera Settings
         2. Video Settings
         3. General Settings
      ii. Capture
      iii. View Gallery
         1. GridView of captured images
         2. Delete
   b. Create New Flight Plan
      i. From Map
      ii. From waypoint selection
   c. Load Flight Plans
5. Browse Flight Plans
   a. Filters Flight Plans
      i. Filter by date
      ii. Filter by last modified
      iii. Filter by creation
      iv. Filter by Complete/Incomplete
   b. Search Flight Plans
2.5 Explicit Design Summary with diagrams

Figure 2.5-1: This illustration explains the general flow of recording and capturing images on our first Android application prototype. Our initial prototype will have the option to playback the images in single preview mode or multiple preview mode.

2.5.1 API Endpoints

We will be exposing several API endpoints from the server for the Android application and Web application to interact with. As we are using a Responsive State Transfer (REST) model, these endpoints will be accessed using the standard HTTP request structure. To access any of these endpoints, the entity sending requests must first be authenticated with the server. Authentication will be managed and verified using JSON Web Tokens. To ensure parity between different versions of the server and client, all API endpoints will be prefixed with a version number in their URL. For example: .../api/v1/users. All requests will return
a HTTP status code to determine their success or failure, even if no other data is required.

<table>
<thead>
<tr>
<th>Endpoint</th>
<th>Request Type</th>
<th>Function</th>
<th>Data Returned</th>
</tr>
</thead>
<tbody>
<tr>
<td>/users</td>
<td>POST</td>
<td>Handles user login requests</td>
<td>Authentication token on successful login</td>
</tr>
<tr>
<td>/users/create</td>
<td>POST</td>
<td>Handles user creation requests</td>
<td>Confirmation of success or failure</td>
</tr>
<tr>
<td>/reports</td>
<td>GET</td>
<td>Retrieves all reports owned by the current user.</td>
<td>List of available reports</td>
</tr>
<tr>
<td>/reports</td>
<td>POST</td>
<td>Submits new data to the server for analysis</td>
<td>Confirmation of success or failure</td>
</tr>
<tr>
<td>/reports/{ID}</td>
<td>GET</td>
<td>Retrieves a specific report</td>
<td>All data associated with specified report</td>
</tr>
<tr>
<td>/flightplans</td>
<td>GET</td>
<td>Retrieves all flight plans owned by the current user.</td>
<td>List of available flight plans</td>
</tr>
<tr>
<td>/flightplans</td>
<td>POST</td>
<td>Submits a new flight plan to the server for storage</td>
<td>Confirmation of success or failure</td>
</tr>
<tr>
<td>/flightplans/{ID}</td>
<td>GET</td>
<td>Retrieves a specific flight plan</td>
<td>The specified flight plan object</td>
</tr>
</tbody>
</table>

Figure 2.5.1-1: Table detailing the endpoints for the server API. Shows the URL path, the type of HTTP request expected, the function performed by the endpoint, and the data that will be returned in the HTTP response.
Figure 2.5.1-2: Sequence Diagram for user login. Details the flow of control for requesting verification of user credentials. Returns an authentication token for the user to use if successful or a HTTP failure response if not.
Figure 2.5.1-3: Sequence Diagram for new user account creation. Details the flow of control for entering user details, verifying that all required data is present, and verifying that there is no existing account with that data. Returns a HTTP success or failure response.
Figure 2.5.1-4: Sequence Diagram for loading list of reports. Details the flow of control for loading a user’s saved reports. Returns a list of all reports the user has access to as a HTTP response, or a HTTP failure response if the user has no reports saved.
Figure 2.5.1-5: Sequence Diagram for submitting new data. Details the flow of control for submitting new inspection data to the server and verifying that the data is complete. Returns a HTTP success or failure response.
Figure 2.5.1-6: Sequence Diagram for loading a single report. Details the flow of control for selecting a report from the menu and requesting the selection from the server. Returns a HTTP response with the requested report or a HTTP failure response if no report exists with the specified ID.
Figure 2.5.1-7: Sequence Diagram for loading list of flight plans. Details the flow of control for loading a user’s saved flight plans. Returns a list of all flight plans the user has access to as a HTTP Response, or a HTTP failure response if the user has no flight plans saved.
Figure 2.5.1-8: Sequence Diagram for submitting a new flight plan. Details the flow of control for saving a flight plan to the server and verifying that the flight plan data is correct. Returns a HTTP success or failure response.
Figure 2.5.1-9: Sequence Diagram for loading a flight plan. Details the flow of control for loading a previously authenticated user’s stored flight plans. Returns the flight plan with specified ID as a HTTP response, or a HTTP failure response if no flight plan exists for that ID.
2.5.2 Database Structure

We will be using a MySQL database to store our users' data. This is a relational database system, so we will use SQL’s JOIN function to connect stored data to its owner. We will have several tables in our database. Firstly, we will have a “User” table. This will contain a Unique Identifier (UID) for each user, along with their full name and email address. Each user entry will own devices, flight plans, and reports, which will have separate table structures. The “Device” table will contain the relevant information about devices that have been registered on our service. This includes a Unique Identifier for the device, its model name and number, and a reference to its owner’s UID. The “Flight Plan” table will contain the information related to flight plans that have been created using our service. This includes a Unique Identifier for the flight plan, the name and a reference number to the work site it is for, and a reference to its owner’s UID. It will also be linked to from the NavPoints that comprise the flight plan. These NavPoints will have their own table structure. This will contain a Unique Identifier for the navigation point, a GPS or relative coordinate, and a Boolean indicating if the navigation point is a location where pictures will be taken. The “Report” table will contain all of the data generated from our analytics engine. This includes a Unique Identifier for the report, a reference to the UID of the flight plan the report was generated from, the raw data the report was generated from, and the analyzed data we have generated. The “Raw Data” table will contain all data generated from inspections as it was before being sent to the analytics engine. This includes a Unique Identifier for the dataset, a reference to both the report and device that own the data, and a modified form of the navigation point data to include images that have been taken at each point. The structure of the data model described here can be seen in Figure 2.5.2 below.
This diagram details our model for the data we will be collecting and manipulating in our project. We currently plan to use a relational database for storage. Raw Data is collected from the drone, NavPoints and Flights plans are gathered from the User via the application and the rest are built by the software.

2.6 Build, prototype, test, and evaluation plan

2.6.1 Detailed prototyping plan

Our Project will require that the prototyping follow a specific format, which is explained in this paragraph. We will prototype the basic connections and build from there. This allows us to build the foundation, make sure that foundation is secure, and build from there. The following paragraph shows the flow of the project building. We will first build the connections between the application and the server. We will then build the connection from the application to the drone. Then we will build the server and drone interactions at the same time. Building these two together will streamline the process of implementing the exact data.
being transferred. The next step will be building the application and server authentication. After this, we will build a simple web portal. Finally, we will build the analytics engine and attach it to the existing web portal.

### 2.6.2 Detailed Testing plan

As mentioned above, this project will have many building steps and after each step, the person responsible for building the part. The initial prototype has been built and tested. We will follow this model forward as it provides opportunity to go back and fix and reevaluate any issues with the specified portions. The method is reminiscent of the waterfall development method, with the ability to go back and fix any model that may be problematic. This, however, will take team approval, as we will want to decide if we want to change an established module or modify the goal of the current module.

<table>
<thead>
<tr>
<th>Part to Test</th>
<th>Date Started</th>
<th>Date Completed (Planned)</th>
<th>Date Completed (Actual)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connections</td>
<td>3/22/16</td>
<td>4/1/16</td>
<td>4/1/16</td>
</tr>
<tr>
<td>Server/Drone</td>
<td>3/25/16</td>
<td>5/8/16</td>
<td></td>
</tr>
<tr>
<td>Authentication</td>
<td>6/16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>application UI</td>
<td></td>
<td>9/16</td>
<td></td>
</tr>
<tr>
<td>Web Portal</td>
<td>7/16</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Analytics</td>
<td>10/16</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 2.6.2-1: Testing Schedule

### 2.6.3 Testing Rubric

The rubric for evaluation of the tests performed. These are the metrics by which we will grade our own work and also, the metrics that we will present for the review board for consideration. All of the following metrics have been approved by the team and will be followed by the team. Ideally, all parts of the project will have a full functionality pass. However, if time presses, the team will accept at most 2 Partial functionality passes. This is a worst case scenario, and as such, will only be accepted in the event that the team runs out of time.
<table>
<thead>
<tr>
<th>Test</th>
<th>Pass (full functionality)</th>
<th>Pass (Partial functionality)</th>
<th>Fail</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connections</td>
<td>The application can download and upload images from the server</td>
<td>The application can pass and receive some data from and to the server</td>
<td>No data is passed from the application or server</td>
</tr>
<tr>
<td>Server/Drone</td>
<td>Flight plans are made and stored in the database and can be accessed from the application and executed on the drone. All data discussed is being passed to and from the drone</td>
<td>Flight plans are made and stored in the database and are downloaded on the application. Partial data is being passed from the drone</td>
<td>Flight plans are inaccessible from the application. Little to no data is being transmitted.</td>
</tr>
<tr>
<td>Authentication</td>
<td>The server allows only the application and web Portal access to its data.</td>
<td>This is a pass/fail here, there is no middle ground</td>
<td>The server allows everyone</td>
</tr>
<tr>
<td>application UI</td>
<td>All buttons, bars and other interactive functions work with their respective commands. All status reporting portions correctly report and the User Interface is easy to work with.</td>
<td>All core buttons work. All status reporting works. User Interface is responsive.</td>
<td>The User Interface is unresponsive and no data is being shown.</td>
</tr>
<tr>
<td>Web Portal</td>
<td>Displays Reports and Pictures</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Analytics</td>
<td>TBD</td>
<td>TBD</td>
<td>TBD</td>
</tr>
</tbody>
</table>

Figure 2.6.3-1: Testing Rubric
2.7 Setting up Android Programming Environment

Before installing Android Studio, make sure you have JDK 6 or higher installed—the JRE alone is not sufficient.

When developing for Android 4.0 (API level 19) and higher, you will need to install JDK 7.

To check if you have the correct version of the JDK installed, open a terminal and type `javac -version`. If the JDK is not available or the version is lower than version 6, download the Java SE Development Kit 7.

Start a new Android Studio Project. Give the application and company domain any name you like, click next.

Set the minimum SDK version as API 19 and click next.

Select "Empty Activity" and click next. Press 'Finish'.
Download and unzip the SDK package downloaded from the DJI website.

<table>
<thead>
<tr>
<th>Mobile SDK</th>
<th>Overview</th>
<th>Features</th>
<th>Get Started</th>
<th>Documentation</th>
<th>Downloads</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Software</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>iOS SDK</td>
<td>v3.1.1</td>
<td></td>
<td></td>
<td></td>
<td>ZIP</td>
</tr>
<tr>
<td>iOS SDK Sample</td>
<td>v3.1.1</td>
<td></td>
<td></td>
<td></td>
<td>GitHub</td>
</tr>
<tr>
<td>iOS SDK Release Note</td>
<td>v3.1.1</td>
<td></td>
<td></td>
<td></td>
<td>PDF</td>
</tr>
<tr>
<td>Android SDK</td>
<td>v3.1.0</td>
<td></td>
<td></td>
<td></td>
<td>ZIP</td>
</tr>
<tr>
<td>Android SDK Sample</td>
<td>v3.1.0</td>
<td></td>
<td></td>
<td></td>
<td>GitHub</td>
</tr>
</tbody>
</table>
Next, import the SDK Module.
In the ‘Source Directory’ field, find the DJI-SDK-LIB folder location. Press Finish.

Next, right click on the 'application' module in the file directory to the left.

Click 'Open Module Settings'.

Navigate to the 'Dependencies' tab. Press the green plus sign, click 'Module Dependency', and select ‘:DJI-SDK-LIB’.

By now, the ‘DJIISDKLIB’ should appear on the project tree.
2.7.1 Setting up DJI SDK Permissions

Inside the Manifest file, all projects should include the following permissions:

```xml
<uses-permission android:name="android.permission.VIBRATE" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.WAKE_LOCK" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
<uses-permission android:name="android.permission.MOUNT_UNMOUNT_FILESYSTEMS" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-feature android:name="android.hardware.camera" />
<uses-feature android:name="android.hardware.camera.autofocus" />
<uses-feature
    android:name="android.hardware.usb.host"
    android:required="false" />
<uses-feature
    android:name="android.hardware.usb.accessory"
    android:required="true" />
</!-- DJI SDK -->
```

As shown above, there are several important permissions that need to be included for all products using the DJI SDK. Our application will require all these permission plus additional ones depending on how our requirements change.

Primarily, the application will require access to the internet, access to location, internal and external storage, and most importantly the camera hardware. Additionally, the DJI SDK uses the android.hardware.usb.accessory and android.hardware.usb.host method calls to connect a Drone controller. The DJI controller is used to interact with the phone.
2.7.2 Getting a DJI developers Key

To obtain a DJI developer key, the user must be a registered developer. To register as a DJI developer: [http://developer.dji.com/en/user/applications](http://developer.dji.com/en/user/applications).

Next, once you have entered the application information on the DJI website, DJI will request email confirmation and verification of application.

After the application key has been generated, the Android application manifest needs to be updated.

Under the `<meta data tag android:name="com.dji.sdk.API_KEY">`, enter the application key on the `android:value` as shown below. The application key will need to be changed if the project/package name changes.
2.8 Facilities and Equipment

This Section breaks down specific hardware the team considered. We give a full compare and contrast table, giving our thoughts on features included and excluded from certain platforms. We will also give reasoning for our choice of hardware platform. Also included in this section, is our testing facilities. Because of the nature of this project, we will need a considerable amount of space and, ideally, a building or some structure to test our project on.

2.8.1 Drones Considered

The drones we have considered so far, have been the DJI Matrice 1000, phantom advanced and professional, Parrot Bebop 2 and the Hoverfly Livesky. The table below describes the key considerations we made when deciding on the drone platform for our project. These considerations were based on cost,
included features, programmability and other possible features or lack thereof that would influence our decision.

<table>
<thead>
<tr>
<th>Drone</th>
<th>Price</th>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>DJI Matrice 100</td>
<td>$4000</td>
<td>Drone platform designed for development</td>
<td>Expensive and does not come as a ready to go package</td>
</tr>
<tr>
<td>DJI Phantom 3 Pro</td>
<td>$1200</td>
<td>4k Camera, robust SDK, Ready to go out of box</td>
<td>None that apparent to the team</td>
</tr>
<tr>
<td>DJI Phantom 3 Advanced</td>
<td>$800</td>
<td>Same as the Pro version</td>
<td>Slightly cheaper camera than the Pro</td>
</tr>
<tr>
<td>Parrot Bebop 2</td>
<td>$350</td>
<td>Inexpensive</td>
<td>SDK is poorly documented and still has key pieces in development</td>
</tr>
<tr>
<td>Hoverfly Livesky</td>
<td>$10000</td>
<td>Commercial Drone platform, advanced imaging systems</td>
<td>Beyond our budget, even if corporate funding was available</td>
</tr>
</tbody>
</table>

Figure 2.8.1-1: The table above shows the pros, cons and pricing of the drone listed on the far left of the table. This table does not show results, only arguments for and against each drone.
This drone is the developer’s dream. It has all of the options of a custom built drone, without having to do the legwork of developing an SDK for the custom drone. It has many expansion options, shown above from left to right, increased computation power, more efficient battery, stronger/lightweight propellers, sound recording, ultrasonic wall detectors, bottom mounted laser, for altitude detection and options between 2.7k and 4k cameras. Unfortunately, this drone is out of our personal budgets, and it is inconceivable that we would be able to afford it by the end of this semester.
This drone has features, such as, ultrasonic wall detection, 4k camera, tri-directional gimbal and a custom camera. If we had to choose the perfect pre-built drone, this would be it.
This drone has many of the features of the Pro with only a camera that has less resolution. The camera has a 2.7k camera, while the Pro boasts a 4k. Ultimately, we choose this drone because the photos we would be using would not need to be 4k.
Figure 2.8.1-5: Parrot Bebop 2 (img src: macpricesaustralia.com.au)
This is the parrot drone, which is the cheapest of the drones we considered. It has a 1080p camera, no gimbal, no built in GPS location tracker and poorly developed sdk. We considered this drone because it is cheap and easily replace, but unfortunately, the SDK was in such disarray, that we could not conceivably develop on this platform.
The Hoverfly Live Sky drone is an advanced commercial drone with a multitude of optional imaging modules available. Unlike the other drones we have considered, it is connected to its power source and control system with a tether.

2.8.2 Testing areas

The flight testing areas are to be decided. The testing areas depend on allowances by local law enforcement. Many areas that contain buildings to test our system on contain, in some form of implementation. These local law enforcement agencies usually to not take kindly to 3rd parties taking unauthorized pictures and keeping them. Hence, the need for authorization by the local authorities.
3. Administrative content

The following sections will outline and explain the administrative content of the project. This content includes budgeting, milestones and a Project summary. The budget is outlined and explained in as much detail as is provided to us, as this project is a student funded project, and therefore, we do not have a strict predefined budget. The milestones are described more in depth, so as to facilitate a more structured project. Most student funded and led projects suffer from poor planning and milestone structure. The team wanted to avoid this pitfall and we attempted to over plan for the project. Finally, the Project summary. The summary will summarize all parts of the project from an extremely high level view.

3.1 Budget

The project has gone beyond our original budget because we found that the bebop drone was inadequate for our development purposes. This caused us to search for a new drone, and we found 2. Both drones are manufactured by DJI. These drones we selected are the DJI Phantom 3 Pro, which costs $1200, and the DJI Phantom Advanced, which costs $900.

This project will be more costly than most Computer Science Senior Design Projects, due to the fact that we require hardware that is not currently supplied by the University. Due to the nature of the project, the team may seek to capitalize on our project. We will be designing our project in a way that will keep future customers experience in mind.

Our team will also be developing a mobile application. We currently developing in the Android environment. Developing for the apple platform would require the initial purchase of a developer’s license which goes for $99, which we want to avoid due to being a self-funded project. Also, if we choose to extend the mobile platform to iOS then we will require the use of Macs which will be provided by the University. Specifically, we will be using the Android Standard Development Kit (SDK). As of today, the Android Standard development kit is free of charge. This is our current budget for our drone.

If we get funding for the project, we would seek more advanced drones for development. The drone we would want to eventually acquire would be the Matrice 100 from DJI. This drone is a developer’s model of their phantom series and has an onboard and mobile SDK and is fully customizable. Unfortunately, this will not be the drone we are prototyping due to the cost of the hardware. The drone itself would cost $3300 and all other parts costing $600, with the total cost
coming to nearly $4000. As mentioned before, because we are a student funded project this is out of our reach until possible funding becomes available.

The total cost of the project, if we get sponsor funding, would come out to be $4500. Without the funding we would cut project costs by not buying the DJI Matrice 100 Drone and would total at $900. This number represents the actual cost, as of the submission of this document.

---

**Cost Analysis Chart**

<table>
<thead>
<tr>
<th>Name</th>
<th>Cost (in dollars)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DJI Phantom 3...</td>
<td>$900</td>
</tr>
<tr>
<td>DJI Matrice 100</td>
<td>$4500</td>
</tr>
<tr>
<td>Parrot Bebop 2</td>
<td>$350</td>
</tr>
<tr>
<td>Apple dev lisce...</td>
<td>$99</td>
</tr>
<tr>
<td>Total</td>
<td>$5,849</td>
</tr>
<tr>
<td>Actual</td>
<td>$900</td>
</tr>
<tr>
<td>Actual (if funded)</td>
<td>$5,499</td>
</tr>
<tr>
<td>Original plan</td>
<td>$449</td>
</tr>
</tbody>
</table>

---

**Figure 3.1-1: Cost Analysis Chart**

The graph shows the current costs we have encountered so far for our budget. We have funded ourselves so far, and we have purchased the DJI Phantom 3 Advanced. The total is the sum of all costs, whether they are actual costs or not. The Actual column shows the costs so far for the project, which only includes the DJI Phantom 3 Advanced. The Original plan column is the sum of the bebop, our original prototype drone, and the apple development license. And finally, if we attract corporate sponsorship, we will request the funds to buy both an apple development license and the Matrice 1000 drone. This total is represented in the Actual (if funded) portion. The project may or may not use all of the above mentioned hardware, and therefore required this unorthodox breakdown of costs.
This project can be completed without full funding. The project will move forward with whatever drone is in our possession by the end of the spring semester. This gives us time to properly develop, test, and revise all the portions of our project where it is necessary. The full schedule and milestone breakdown will be discussed below, after the numerical breakdown of the budget.

As of the submission of this document, the team has acquired a DJI Phantom 3 Advanced. This drone will be the drone that all product testing will occur on, and will be the presentation drone. If the project becomes a company, we plan on extending the platform to the Matrice 1000, as that platform will allow for many upgrades to the current platform. This platform would include many hardware and software upgrades, that otherwise are inaccessible with any other drone.

<table>
<thead>
<tr>
<th>Name</th>
<th>DJI Phantom 3 Advanced</th>
<th>DJI Matrice 100</th>
<th>Parrot Bebop 2</th>
<th>apple dev license</th>
<th>total</th>
<th>Actual</th>
<th>original plan</th>
<th>Actual (if funded)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cost (in dollars)</td>
<td>$900</td>
<td>$4,500</td>
<td>$350</td>
<td>$99</td>
<td>$5,849</td>
<td>$900</td>
<td>$449</td>
<td>$5,499</td>
</tr>
</tbody>
</table>

Figure 3.1.2: This table shows the breakdown of the above budget and graph.

### 3.2 Milestones

<table>
<thead>
<tr>
<th>Milestone</th>
<th>Date Planned</th>
<th>Date Completed</th>
</tr>
</thead>
<tbody>
<tr>
<td>First Meeting</td>
<td>2/15/16</td>
<td>2/15/16</td>
</tr>
<tr>
<td>Project Defined</td>
<td>2/17/16</td>
<td>2/17/16</td>
</tr>
<tr>
<td>Drone 1 Acquired</td>
<td>3/2/16</td>
<td>3/2/16</td>
</tr>
<tr>
<td>Drone 2 Acquired</td>
<td>3/23/16</td>
<td>3/24/16</td>
</tr>
<tr>
<td>Start Prototyping</td>
<td>3/23/16</td>
<td>3/25/16</td>
</tr>
<tr>
<td>Finish Documentation</td>
<td>4/28/16</td>
<td></td>
</tr>
<tr>
<td>Finish Application Prototype</td>
<td>5/31/16</td>
<td></td>
</tr>
<tr>
<td>Begin development for final product</td>
<td>6/1/16</td>
<td></td>
</tr>
<tr>
<td>Event</td>
<td>Date</td>
<td></td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>-------</td>
<td></td>
</tr>
<tr>
<td>Finish “Behind the Scenes” of the application</td>
<td>8/1/16</td>
<td></td>
</tr>
<tr>
<td>Finish user interface of the application</td>
<td>9/1/16</td>
<td></td>
</tr>
<tr>
<td>Begin Testing/Bug hunting</td>
<td>9/1/16</td>
<td></td>
</tr>
<tr>
<td>Testing ends</td>
<td>11/16</td>
<td></td>
</tr>
<tr>
<td>End of term Presentation</td>
<td>11/16</td>
<td></td>
</tr>
<tr>
<td>End of Project presentation</td>
<td>12/16</td>
<td></td>
</tr>
</tbody>
</table>

Figure 3.2.1: Project Schedule

The above graph is the basic milestone chart for the project. This graph is intentionally at a high level, so as, not to lose track in all the detail. The project so far is on track and we have only encountered a few delays, due to drone uncertainty and delays in drone delivery. The team will commit to enforcing this milestone schedule so that we may complete our project on time. Also, the graph shows that there is about a month where we do not have any activities planned. This is also intentional, so as to provide an opportunity for mistakes and delays. This buffer time, if not used for the core project, may be used for adding features that are not required, but would be interesting problems to solve and be useful if the project is commercialized.

3.3 Project Summary

In summary, our team will be creating a platform on which, owners of any DJI brand drone will be able to conduct safety inspections. While our team is gearing this towards construction inspections, there is nothing stopping casual users or industries outside of the target audience using this software. The team’s main goals are to increase the workplace safety, gain team experience in new and common technologies and to possibly gain experience in entrepreneurship. The technologies that we will be using are the DJI phantom 3 advanced, the android application and phone platforms, golang, Linux and Java.

The application we are creating will be used to semi automate the drone, for the purpose of inspecting hard to reach or dangerous areas. The application will be written in java and the server will be an UCF hosted, Linux VM with an API
written in golang. The server will contain a SQL database that will contain user data. For the definition of this server data, please read the paper, specifically section 2.3.2 database. The data contained will be retrievable from the server to the application. The server will also contain a rudimentary image recognition/machine learning module to analyze the picture data. There is a possibility that the data will be viewable from a web browser via a web portal. This web portal will imitate the application in its data display.

Core to our project are the people that the project will be helping. The project will be helping many people putting themselves in danger. The project seeks to decrease the danger, and possible injuries, in the inspection industry. In complete summary, the project will be an android project that will communicate with a Linux server via the internet. The drone will be controlled by the android application.

4 Questions and Concerns

This section is devoted to answering any and all outside questions and concerns that we have encountered during the development of the project. These questions are condensed into two parts: Weather and Legal. If your question is not covered by either of these sections, please make one of the team members aware of your question and we will do our best to answer the question and provide its answer in our documentation.

4.1 Weather Concerns

During our presentation, the team was asked at least 2 questions about operation in extreme weather and whether or not the drone will detect inclement weather and fly to safety. The team discussed these concerns and we have come to the conclusion that, as part of our EULA, the operator of the application and drone shall take full responsibility when flying the drone, whether or not they are aware of inclement weather in their vicinity. It is our recommendation, therefore, that the user of the drone, make themselves aware of all weather patterns in their area of operation, so that the drone may operate normally and safely.

4.2 Legal Concerns

Also, through the course the presentation, the team was asked how we were going to handle the legal issues surrounding drone, more specifically, how we
were going to handle the users attempting to fly above the legal 400ft limit and users attempting to operate the drone out of sight and in close proximity to government buildings, stadiums, etc. where drone flight is not permitted. The first issue is handled by the DJI sdk. The programming of the drone will not allow the drone to fly above the legal limit. The next concern will again be handled by our EULA. It will state that “users attempting to operate the drone outside of an operator’s line of sight are operating the drone outside of its intended use and outside of the bounds of the law (FAA Advisory Circular 91-57) and therefore shall be subject to above mentioned law, and we, also identified as the developers and owners of this software, are not held responsible for any damages that may occur from misuse or use contrary to the law established in the location of operation. Also, in conjunction to the previous statement, if the user attempts to operate the drone and mobile application in an area where drone flight is not permitted, the user shall be responsible to authorities that may try to enforce these restrictions”.

4.3 EULA Considerations

To make the user aware of these clauses in our EULA, the application will provide several popups that will detail the EULA and ask the user to accept the EULA. If the user does not accept, the application will close and will not be made accessible until the user does accept the agreement. This is the team’s solution to address the concerns described above and voiced during our presentation and discussion. The EULA will include provisions, such as the aforementioned legal snippet, so as to make the user aware of the specifics of the applications intended use and what constitutes as activity beyond its intended use.
5. References


[15] MongoDB Logo, Retrieved April 2016, from https://www.mongodb.com/assets/MongoDB_Brand_Resources/MongoDB-Logo-7db53b4037f9953e9df4f694c758141fcfa022cee1d2ff50749d133d0b70b692.jpg