Immersive Experiences for Zoos and Aquariums

Design Document

sdmay21-02

Client

True360

Advisor

Dr. Alexander Stoytchev

Team Members

Daniel Barnes - Meeting Scribe

Aryan Joshi - Report Manager

Austin Nebel - Chief Engineer

Samuel Shifflett - Meeting Facilitator

Noah Syens - Test Engineer

Thomas Powell - Chief Engineer (UI)

Sebastian Vang - Product Owner

sdmay21-02@iastate.edu

http://sdmay21-02.sd.ece.iastate.edu

Revised: November 15th, 2020

Executive Summary

This document provides the overall design and plan that the group *sdmay21-02* will be taking in order to complete the project for our client, *True360*. The document is split up into six different sections, and each section describes a specific part of the project. The sections are as follows:

1. Introduction - This section introduces the project, explaining what it is, why it is needed, and what is required of our group
2. Project Plan - This section describes how we will go about completing the project during our second semester of Senior Design
3. Design - This section illustrates how our project will be created, both on the software side, and the visual side
4. Testing - This section describes how our project will be tested to ensure correct functionality, both with automated and manual testing
5. Implementation - This section describes a preliminary implementation of what we will be creating during the second semester
6. Closing Material - This section ties all the previous sections together, and includes any miscellaneous information that did not fit in any other sections.

Development Standards & Practices Used

Technology we will use in this project:

- Node.js
- Electron
- Express
- Mocha/Sinon/Chai
- SQL Database
- Spectron
- Postman

Engineering Standards:

- Test Driven Development will be conducted
- Follow all best practices for security, ensuring that user data is secure and encrypted
- Code should be written in a way that it is easy to maintain and understand
- All code must be reviewed by at least 2 other developers

Summary of Requirements

- Create a desktop application that allows users to upload WOW! Moments to various platforms by right clicking the videos
- Create a centralized web server that allows for communication between True360 and various zoos and aquariums.

Applicable Courses from Iowa State University Curriculum

List all Iowa State University courses whose contents were applicable to your project.

- COM S 228
- COM S 252
- COM S 309
- COM S 319
- COM S 327
- COM S 329
- COM S 339
- COM S 363
- COM S 409
- CPRE 230
- CPRE 231

New Skills/Knowledge acquired that was not taught in courses

List all new skills/knowledge that your team acquired which was not part of your Iowa State curriculum in order to complete this project (still in progress).

- Electron
- Registry Editing
- API Development
- Functional Testing
- Code Reviews
- Product Deployments
- Cybersecurity Practices

Table of Contents

List of figures/tables/symbols/definitions

<u>Figures and Tables:</u>

<u>Definitions:</u>

WOW! Moments - Video moments when animals are doing something out of the ordinary in which the general public would enjoy seeing.

1 Introduction

1.2 PROBLEM AND PROJECT STATEMENT

Problem:

True360 aims to provide an immersive experience for customers going to zoos and aquariums. They do this by creating special moments called "WOW! Moments". These WOW! Moments are recordings from within a zoo or aquarium exhibit that signify something interesting happening. True360 provides the WOW! Moments to Zoo customers in the form of 360 degree video. The Zoo can then take the moment, and either share it through VR headsets at the zoo, in order to provide an immersive VR experience, or they can publish the video on various social media platforms to encourage customers to come to the zoo.

Unfortunately, there is a problem in that there is no way to publish these videos easily across multiple platforms within a user-friendly desktop application, while also allowing True360 to be a part of the publishing process. Current solutions fail to address all use cases needed by True360. True360 needs a way to be able to track which of the WOW! Moments have been published, and be able to obtain insights into video metrics such as views, likes, number of comments, and more. Without this, True360 isn't able to engage their business customers, and prove that these WOW! Moments are actually a valuable product, and eventually, it could cause True360 to lose customers.

Creating a way for users to upload WOW! Moments that is easy to use, extendable in functionality, and that allows True360 to tailor to their own specifications is crucial to the business needs of True360.

Project Statement:

To solve this problem, we will be developing a cross-platform desktop application that will allow users to publish videos to certain social media platforms. Users will be able to right click on a WOW! Moment directly from their computer, and seamlessly publish their video by just clicking "Publish WOW! Moment", selecting the desired platform, and filling out information about the video. Quality of life is improved for the user due to no longer needing to interact with all the websites of the different platforms they might

want to publish on, or no longer needed an additional license with a third party publishing product. By creating a product unique to True360, the company is able to be involved in that publishing process.

We will also be creating a database and centralized cloud server to hold publication data. This will allow True360 to know exactly where WOW! Moments are being published. They will be able to query APIs for these publishing platforms and provide analytics to businesses to optimize their upload strategies. This creates a better relationship between True360, and their customers, ensuring a positive experience for all parties involved.

1.3 OPERATIONAL ENVIRONMENT

We will develop a desktop application that can run on both Windows 10 and MacOS Catalina in a business environment. The on premises server will be handled by the zoo in a controlled environment with heat dissipation and dust filtering.

1.4 REQUIREMENTS

General:
- Automated tests will be run on the systems before it is deployed to production.
- All features and code within the systems will be documented with comments within the code.
- All code must be approved by at least 2 other developers before being merged into a codebase.
- All UI designs will be approved by True360 before they are submitted to production.

Cloud System:
- Create a cloud based Express web server that is able to receive HTTP POST requests from the desktop app that contain video publication data.
- Create a cloud based SQL database that the cloud based Express web server can store information from the HTTP Post requests messages.
- Contains a */publish* POST endpoint that adds a published video to the SQL database
- Create a cloud based process that runs daily to retrieve views, likes, and shares from all the videos that were published through the True360 Desktop Application across multiple social media platforms.
- Contains a */support* POST endpoint that receives a name, description, and application version number, and then formats and sends the information in an email to support@true360.com
- Contains a */requestEdit* POST endpoint that receives a video ID/Title, and associated zoo, and formats and sends the information in an email to editing@true360.com

Desktop App:
- Allows for a right click "Publish WOW! Moment" that has an arrow to a specified social media to open the application
- Desktop Application can run on Windows 10 and MacOS Catalina
- Allows users to publish a selected video to YouTube, Facebook, Vimeo, and Hootsuite in the optimal video format for each platform.
- Allows users to configure login credentials to multiple accounts per platform, and then select a specific account when publishing
- The Desktop Application will be built in Electron to offer cross compatibility on MacOS and Windows
- API keys and passwords for connected publishing platforms will be encrypted and hashed using the NIST standard and stored on the Network Share Server

- Desktop app will automatically update when the user launches it if update is available
- Desktop app will hit the */publish* endpoint on the cloud web server on successful video publish. It will supply the video URL
- Uninstallation of the Desktop Application should remove all the files associated with the Desktop Application on the user's computer as well as the right click option from the popup menu on the OS but keep the corresponding information within the cloud database.
- Users that have the Desktop Application will have an option in the Application to send a support request to the cloud based web server that will notify the necessary individuals of the support request by sending a */support* POST request to the cloud web server.
- The Desktop Application will include a "Help" section to aid users in understanding how to properly use it.
- Adds Metadata for copyright, publication info, and tags to videos before publishing
- Users should be able to submit "project" requests to True360 through the cloud based web server at */requestEdit* that notifies the editing team certain videos they would like True360 edit.

Non-Functional Requirements:
- The product shall appear easy and simple to use.
- The product shall perform swiftly for the user.
- The product shall be usable by users with limited computer experience.
- The product shall be built to anticipate any future software updates.
- The product shall not display any offensive terms or symbols.
- The product shall comply with legal regulations.
- The product shall be built to last.

1.5 Intended Users and Uses

To attain an end product that performs well and provides the best user experience, we must make sure that we fulfill the uses and needs of the users specified below.

Intended Users:
- Zoo/Aquarium Management
- Zoo/Aquarium Staff
- True360 Administration

Intended Uses:
- Uploads 360 captured camera footage of animals (WOW! Moments) to specified media platforms.
- Extendability to add additional social media platforms.
- Documentation of the video publication data.
- Email True360 "Project Edit Requests" with selected video(s).
- Submit Bug Reports to True360

1.6 Assumptions and Limitations

Assumptions:

- Users will have either the latest versions of Windows 10 or MacOS Catalina installed.
- True360 app will be extensible to support other social media and VR kiosks.
- True360 app will be easy and seamless for the user experience.
- Detected WOW Moments! will already be stored on the zoo's local server.
- End users will know how to navigate to their directory to find the videos.
- End users have existing social media accounts to publish videos to.
- Zoos will have enough internet bandwidth to successfully upload 8k resolution videos.
- The end product will eventually be deployed across multiple zoos in the US.

Limitations:

- There is no budget for developing the True360 app.
- The end product will not support mobile OS.
- Most of the development team has little to no experience developing Electron apps.
- True360 app is limited to the Electron framework.
- True360 app is not guaranteed to work with older builds of supported OS's.
- The end product will only be used by zoos in the US and that are contracted with True360.

1.7 Expected End Product and Deliverables

A desktop-based commercial application that will streamline the zoo employee experience for publishing WOW! Moments to various social media platforms. The application will be installed via an ease-of-use downloadable installer which will deploy the software onto the employee's machines, and it will be designed to work on both Windows 10 and Mac OS Catalina. Once installed, the user will be able to access WOW! Moments in a shared drive on their local network which is connected to the True360 server located at the zoo.

Once in the server's WOW! Moment directory, the user will be able to right-click on any mp4 files generated by the True360 system and have the option to open the True360 Desktop Application to publish that video. If the user decides to publish a video, several mp4 metadata fields will automatically be attached to it and the video will be published to the platform of choice. From this context menu the user will also be able to send True360 personalized support requests as well as configure their social media account configurations..

There will only be one initial configuration aspect of the application which will be to link the app with the zoo's social media accounts. This configuration will live on the True360 server at the zoo and all computers that have the True360 software installed will access this file. This will allow for the configuration process to be efficient for the end-user. The application will then receive automated updates at any given time to both improve the user experience and to provide security fixes. This software will be delivered by the end of April, 2021.

2 Project Plan

Project Goal:

True360 has tasked us with creating a desktop application for zoos that are partnered with True360. The Desktop Application will allow for employees at the zoo to publish videos that are "WOW" moments to the zoo's corresponding social media platforms and also maintain a database of publishing data that lives within True360's AWS cloud account.

Elements of the Overall System:

Desktop Application:

The Desktop Application will be the tool that employees at the zoo will use to publish their videos to their corresponding social media platforms.

AWS Cloud Account:

The AWS Cloud Account will be the centralized area where the publishing data will be stored so that True360 can monitor usage of the Desktop Application.

On Premises Server:

The On Premises Server holds the video files that can be published as well as configuration files that will link their social media accounts to their Desktop Application.

Tasks:

Source Control Setup:

- Create Repositories for Desktop Application Code and Cloud Based Scripts
- Create Branching Strategy for Each Repository

Cloud Environment Setup:

- Setup Virtual Machine what will Run the Cloud Based Web Server
- Setup SQL Database that is able to be Linked to Cloud Based Web Server
- Create Recurring Process to Retrieve Interaction Data with Videos Published through True360 Desktop Application

Desktop Development and Testing Environment Setup:

- Setup Testing Environments for Mac OS Catalina and Windows 10 Operating Systems
- Build Basic Electron App that Each Team Member can Test on their Machine
- Organize Code for Electron App in Source Control Repository into Front-End and Back-End Categories

Operating System Registry Development and Testing Setup:

- Develop Process for Testing Right Click Feature on Mac OS Catalina and Windows 10

Right Click Feature Development:

- Develop Basic Right Click Functionality on MacOS Catalina and Windows 10
- Integrate Basic Electron Application to Open with Right Click Feature

Backend Desktop Application Development:

- Develop Backend of Desktop Application to Send HTTP Post Requests to Localhost Server
- Change Basic HTTP Post Requests to Send Requests to Cloud Based Web Server

Cloud Web Server Development:

- Develop Web Server to Receive HTTP Post Requests with Multiple Endpoints
- Develop Process to Ingest Basic Data Into Cloud Based SQL Database

Testing of Desktop Application Linked with Cloud Web Server and Database:

- Create Action Button in Desktop Application to Send Basic HTTP Post Request to Cloud Based Web Server
- Test Ingesting of Basic HTTP Post Request to Cloud Based SQL Database from Cloud Based Web Server

Social Media API Integration Development:

- Develop Basic Publishing of Video to Youtube, Facebook and Vimeo Using Each Platform's API
- Develop HTTP Post Requests to Send Responses of Successful/Failed Publishing to Cloud Based Web Server

Cloud Database Schema Design for Social Media Response Messages:

- Create Table Designs for Each Social Media's Response Message
- Create Specific Endpoints for Each Social Media Response
- Create Ingestion of Each Social Media Endpoint Into Cloud Based SQL Database

Mac OS Catalina and Windows 10 Installer Generation:

- Test Installer Generation for Mac OS Catalina and Windows 10
- Test Installation on Mac OS Catalina and Windows 10
- Test Uninstallation on Mac OS Catalina and Windows 10
- Test Desktop Functionality on Mac OS Catalina and Windows 10

Develop CI/CD Process for Updates to Desktop Application:

- Develop Process to Check Version of Desktop Application with Latest Version of Code Base in Source Control Repository
- Develop Process to Either Update Application Immediately or Delay Update

Develop Process for Building Social Media Configurations:

- Develop Process to Retrieve Social Media Configurations from Network Share Server
- Develop Process to Add Multiple Accounts for Each Social Media Platform

Develop Process for Parsing Social Media Configurations:

- Develop Process to Parse Social Media Configurations that Allows Users to Publish to Corresponding Social Media Platforms

Develop Process for Updating Social Media Configurations:

- Develop Process for User's to Update Their Social Media Configurations

Create UI That is Linked to the Backend Functionality of the Desktop Application:

- Implement Designs of UI
- Link Action Buttons to Backend Functionalities

Test Functionality of Entire Desktop Application and Cloud Based System Working Together:

- Test Functionality of Desktop Application Using UI for Each Social Media Platform
- Test Functionality of Cloud Based Web Server and Ingestion of Data Into Cloud Based SQL Server
- Test Functionality of Daily Process that Retrieves Interaction Data for Each Video Published Through True360 Desktop Application

2.2 Risks And Risk Management/Mitigation

The scores associated with each risk is the percentage chance of affecting the project negatively. The risks with a score of **0.5** or above will have a risk mitigation plan associated with the risk.

Risks

- There is no budget for developing the True360 app: **Score 0.2**
- The end product will not support mobile OS: **Score 0.1**
- Most of the development team has little to no experience developing Electron apps: **Score 0.6**
  - *Mitigation Plan:* The Desktop Team will create a basic Electron application in the beginning to better understand how to move forward with Development during the rest of the project.
- True360 app is limited to the Electron framework: **Score 0.4**
- True360 app is not guaranteed to work with older builds of supported OS's: **Score 0.3**
- The end product will only be used by zoos in the US and that are contracted with True360: **Score 0.2**
- The Desktop Application will rely heavily on the strength of the internet connection at the zoos: **Score 0.4**
- The Desktop Application will rely on the user's computer resources for video processing and social media uploads, which will require a computer with good processing speed: **Score: 0.4**
- Development process may be modified due to the pandemic: **Score 0.5**
  - *Mitigation Plan:* The team has designed the project and project timeline in mind with the idea that we will not be meeting in person, and all communications and meetings will be online.
- Iowa State University goes online and the team now has to work remotely from Ames: **Score 0.5**
  - *Mitigation Plan:* The project team will not be greatly affected by this due to the project not needing to deal with any hardware as it is solely software-based. The team will all be able to find internet access outside of Ames.
- The client, True360, is shut down: **Score 0.1**

- The team members working on this project change: **Score 0.2**
- Team does not have the prior experience necessary for the complexity of this project: **Score 0.3**
- Team has personality clashes which leads to problems with management: **Score: 0.2**
- Team has the wrong approach to completing the project: **Score: 0.2**
- Team uses the wrong programming language in completing the project: **Score: 0.3**
- Team members are unable to perform to the best of their ability (e.g. low motivation, become ill): **Score: 0.1**
- Team mismanages the project tasks: **Score: 0.4**
- A team member drops the class and we have fewer people to complete the project: **Score: 0.1**
- The client tries to change the requirements of the project on us: **Score: 0.2**
- The team experiences feature creep in the development of this project: **Score: 0.4**
- The time it takes to complete tasks has been underestimated: **Score: 0.3**
- Features are implemented poorly and lead to more work than expected: **Score: 0.3**
- The team fails to communicate effectively between its members: **Score: 0.1**
- The team fails to communicate effectively with the client: **Score: 0.1**
- The end product will be rejected by users: **Score: 0.3**
- The team does not have access to the necessary resources to complete the project: **Score: 0.2**
- The team does not have the financial resources necessary to complete the project: **Score: 0.1**
- The team experiences technical difficulties that disrupt the project development: **Score: 0.2**
- The team experiences internet outages that disrupt the project development: **Score: 0.1**
- The APIs used for social media platforms could change: **Score: 0.3**

2.3 PROJECT PROPOSED MILESTONES, METRICS, AND EVALUATION CRITERIA

The milestones are derived from the tasks in section 2.1. Since we are building the system from the ground up, these are our milestones and evaluation criteria:

Phase 1: Environment Setup with Managed Source Control Process

- Source Control Setup: The application will be maintained with a git repository and any merge requests will be approved.
- Cloud Environment Setup: Setting up a database that will be hosted on the cloud.
- Desktop Development and Testing Environment Setup: Setting up testing environments so that the application can be tested thoroughly across each developers system.
- Operating System Registry Development and Testing Setup: Developing a test for right clicking functions on Windows and MacOS.

Phase 2: Basic Functionality Development

- Right Click Feature Development: End users will be able to right click the context menu and select options with our app.
- Backend Desktop Application Development: Develop backend to send HTTP requests to the local server.
- Cloud Web Server Development: Develop a cloud web server to receive HTTP post requests.
- Testing of Desktop Application Linked with Cloud Web Server and Database: Set up basic testing of sending data to the web server.

Phase 3: Finalizing Advanced Functionality Development

- Social Media API Integration Development: Basic publishing of videos to various social media using their API.

- Cloud Database Schema Design for Social Media Response Messages: Creating specific endpoints for each social media response.
- Mac OS Catalina and Windows 10 Installer Generation: Testing installation and functionality on Windows/MacOS.
- Develop CI/CD Process for Updates to Desktop Application: Checking for application updates.
- Develop Process for Building Social Media Configurations: Developing a process to retrieve configurations and adding multiple accounts for each social media platform.
- Develop Process for Parsing Social Media Configurations: Developing a process to parse social media configuration.
- Develop Process for Updating Social Media Configurations: Developing a process to allow users to update social media configuration.

Phase 4: Complete Functionality with UI Development and Testing

- Create UI That is Linked to the Backend Functionality of the Desktop Application: Create a UI that end users will interact with that is linked to the desktop application backend.
- Test Functionality of Entire Desktop Application and Cloud Based System Working Together: Testing functionality for the UI linked with multiple social media platforms, video data analytics retrieved and ingestion of data into the cloud.

2.4 PROJECT TIMELINE/SCHEDULE

The following timeline is how our team plans to complete this project. The different colors represent different phases of the project throughout its development lifecycle. Each task is assigned to either the Cloud Team or Desktop Team, as well as tentative team member assignments. Each task also shows the tentative start and end dates.

True360 Desktop Application Development

True360
Project Lead

Project Start: Mon, 1/25/2021

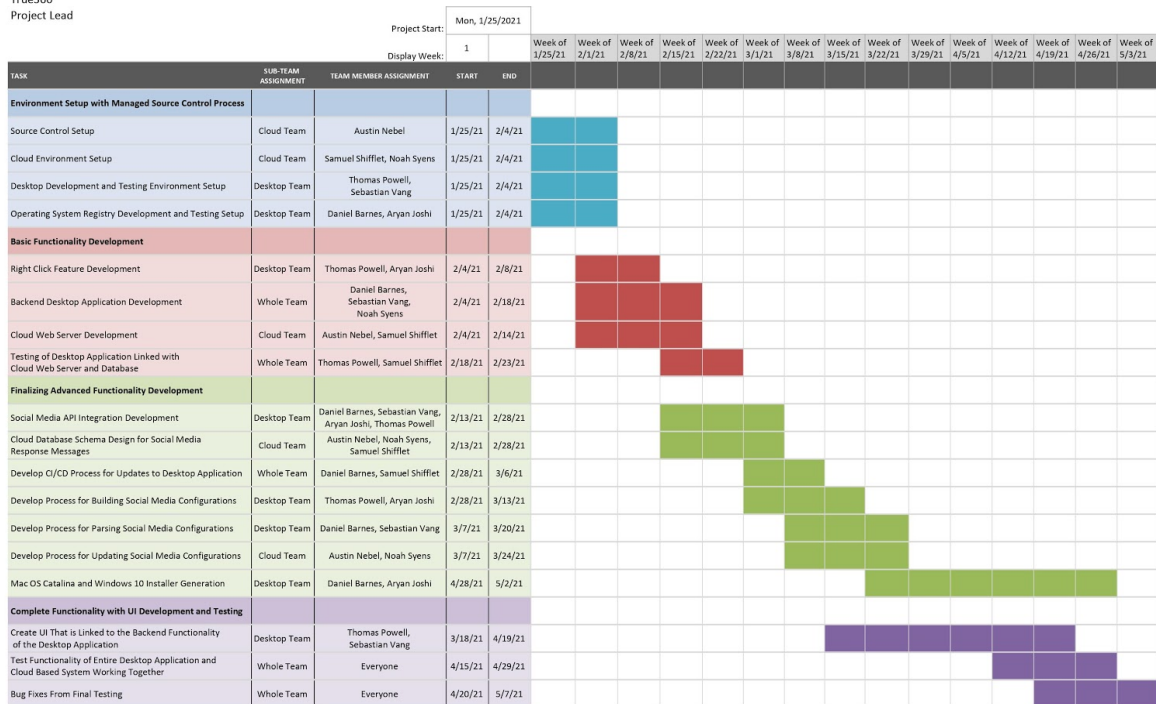| TASK	SUB-TEAM ASSIGNMENT	TEAM MEMBER ASSIGNMENT	START	END
Environment Setup with Managed Source Control Process				
Source Control Setup	Cloud Team	Austin Nebel	1/25/21	2/4/21
Cloud Environment Setup	Cloud Team	Samuel Shifflet, Noah Syens	1/25/21	2/4/21
Desktop Development and Testing Environment Setup	Desktop Team	Thomas Powell, Sebastian Vang	1/25/21	2/4/21
Operating System Registry Development and Testing Setup	Desktop Team	Daniel Barnes, Aryan Joshi	1/25/21	2/4/21
Basic Functionality Development				
Right Click Feature Development	Desktop Team	Thomas Powell, Aryan Joshi	2/4/21	2/8/21
Backend Desktop Application Development	Whole Team	Daniel Barnes, Sebastian Vang, Noah Syens	2/4/21	2/18/21
Cloud Web Server Development	Cloud Team	Austin Nebel, Samuel Shifflet	2/4/21	2/14/21
Testing of Desktop Application Linked with Cloud Web Server and Database	Whole Team	Thomas Powell, Samuel Shifflet	2/18/21	2/23/21
Finalizing Advanced Functionality Development				
Social Media API Integration Development	Desktop Team	Daniel Barnes, Sebastian Vang, Aryan Joshi, Thomas Powell	2/13/21	2/28/21
Cloud Database Schema Design for Social Media Response Messages	Cloud Team	Austin Nebel, Noah Syens, Samuel Shifflet	2/13/21	2/28/21
Develop CI/CD Process for Updates to Desktop Application	Whole Team	Daniel Barnes, Samuel Shifflet	2/28/21	3/6/21
Develop Process for Building Social Media Configurations	Desktop Team	Thomas Powell, Aryan Joshi	2/28/21	3/13/21
Develop Process for Parsing Social Media Configurations	Desktop Team	Daniel Barnes, Sebastian Vang	3/7/21	3/20/21
Develop Process for Updating Social Media Configurations	Cloud Team	Austin Nebel, Noah Syens	3/7/21	3/24/21
Mac OS Catalina and Windows 10 Installer Generation	Desktop Team	Daniel Barnes, Aryan Joshi	4/28/21	5/2/21
Complete Functionality with UI Development and Testing				
Create UI That is Linked to the Backend Functionality of the Desktop Application	Desktop Team	Thomas Powell, Sebastian Vang	3/18/21	4/19/21
Test Functionality of Entire Desktop Application and Cloud Based System Working Together	Whole Team	Everyone	4/15/21	4/29/21
Bug Fixes From Final Testing	Whole Team	Everyone	4/20/21	5/7/21

Figure 2.1. Gantt Chart of the Project Development Timeline

2.5 Project Tracking Procedures

The 3 major tools that our group will use to track progress throughout the lifecycle of this project include Slack, Pivotal Tracker, and Gitlab.

In order to communicate effectively, our group will use Slack to communicate. This will allow us to determine when to schedule team meetings, advisor meetings, and meetings with True360 to keep them informed. This will also allow for quick, informal communication to troubleshoot and resolve any problems, instead of waiting until the next meeting to bring the issue up.

In order to stay on track with our project timeline, we will be using Pivotal Tracker, a tracking tool similar to Trello. This will allow us to create stories to work on and keep track of what needs to be done and what is done. In addition, we will be able to brainstorm ideas as well as assign and prioritize certain tasks.

In order to keep the group's software uniform and up to date, we will be using Gitlab. This will allow us easy access to update, manage, and receive the latest revisions of the project's code. Additionally, it will allow for the group to resolve merge conflicts and allow for any changes in the code to be approved by other developers.

2.6 Personnel Effort Requirements

In order to keep track of effort levels for a task, we will be assigning each task with a number of hours to complete the task. These hours are generally assumed to be for the task overall, rather than per team member. Depending on the task, this means that if a task can be completed by one person in 2 hours, but everyone in the group has to do it, it will be listed as 14 (2 hours * 7 team members) hours. These numbers also include the hours put into creating automated tests for the feature, as necessary.

Each member of the team is expected to complete approximately 5-7 hours of work per week.

Table A

Task	Effort Level (hours)	Explanation
Source Control Setup	14	Source Control is a simple process, and we all have experience with this, so we have assigned it 2 hours per member.
Cloud Environment Setup	10	The cloud environment needs to be deployed in a place that is accessible, but is otherwise a simple process that will take approximately 3 hours for 3 members.
Desktop Development and Testing Environment Setup	12	The desktop environment is similar to the cloud environment setup, but slightly longer since it includes the testing environment.

Operating System Registry Development and Testing Setup	12	Editing the registry can be a difficult task, but it is still simple in scope, thus it is given a 12.
Right Click Feature Development	6	Once it is known how to edit the registry, adding a registry key for a right click feature becomes a simple task.
Backend Desktop Application Development	60	The bulk of development for the desktop application is done within this task, and since it contains a large number of features, it will take 60 total hours.
Cloud Web Server Development	60	Similar to the backend desktop application, the cloud web server will also take a similar number of hours, since it is large in scope.
Testing of Desktop Application Linked with Cloud Web Server and Database	8	Since most of the testing is done when different functions are created, this task will have the majority of work already finished, with it needing minor additions.
Social Media API Integration Development	20	This task is largely dependent on the number of platforms, but it is estimated that it will take 5 hours for each of the 4 platforms we will support.
Cloud Database Schema Design for Social Media Response Messages	8	The database we will be using for social media will be simple, but it still requires thought. This should take 2 members about 4 hours.
Mac OS Catalina and Windows 10 Installer Generation	10	Electron supports creating installers, but we will need to edit this slightly by including editing of the registry keys, which will take slightly longer than the custom out of the box Electron installers.
Develop CI/CD Process for Updates to Desktop Application	10	Setting up the CI/CD will need to be done for both the desktop application and cloud server, but it is a straightforward process for

		anyone that has created a CI/CD pipeline before.
Develop Process for Building Social Media Configurations	20	This task is slightly more complex than other tasks because it requires taking into account many different other tasks.
Develop Process for Parsing Social Media Configurations	6	Assuming the configurations are made with a standard notation such as JSON, parsing is simple. It is assigned a few more hours to account for any errors that come up with the config associated with our code.
Develop Process for Updating Social Media Configurations	10	Once we know how to parse a configuration, we will be able to update it, but we will need to figure out what information needs to be updated.
Create UI That is Linked to the Backend Functionality of the Desktop Application	30	The UI requires more care since it is a product that will be given by a company. This requires it to be modern, and have significant research done into ensuring everything is working correctly.
Test Functionality of Entire Desktop Application and Cloud Based System Working Together	15	This task assumes that most functions are already tested via automated testing, but it requires more hours to ensure that we have covered all edge cases, and put the product through QA.

Table A. Tasks and Corresponding Effort Levels

2.7 Other Resource Requirements

Other resources that are required to complete this project include:

- Access to an AWS Web Server
- Access to an AWS Storage Bucket
- Access to a Central Database
- Designated Social Media Accounts for Testing Purposes
- Testing VMs for Mac OS and Windows 10
- Online Material and Information for Programming and Development Purposes

2.8 Financial Requirements

No financial resources will be required to complete the project.

3 Design

3.1 PREVIOUS WORK AND LITERATURE

In order to know that we are not "reinventing the wheel" we have compared our proposed project, the "True360 Publishing System", to other products that perform a similar function.

	Hootsuite	Adobe Premiere Pro	Final Cut Pro	True360 Publishing System
Product Description	Social Media Management platform	Video editing software	Video editing software	Social Media publishing platform
Advantages	Already developed for use, multiple partners and thoroughly tested.	Able to edit videos and upload in the same program.	Able to edit videos and upload in the same program.	Unique to True360, more 1-on-1 support scenarios and can tweak to specific user needs.
Shortcomings	To upload to certain sites (i.e. YouTube) it requires a premium subscription	It is a video editor software that can require a large sum of money to obtain.	It is a video editor software that can require a large sum of money to obtain.	Developed by a group of college students rather than professional engineers, some of whom have little experience developing such a product.
Differences	Global platform used by many businesses for social media management.	Has video editing	Has video editing	Application unique to True360 and their customers.

Table B. Proposed Project and Similar Products Comparison*

*References for Hootsuite, Adobe Premiere Pro, and Final Cut Pro can be seen in section 6.2

3.2 DESIGN THINKING

In order to know how to design this system, we needed to discuss the various social media platforms that users are publishing their content to, the metadata that should be included with the video files and also the different OS platforms our software should support. The zoo users have a lot of influence on the design because they are ultimately the ones who are going to be using it. For example, we want the interface to be easy to use for non-technical zoo users. Since zoos in general are not very technical, we have to accommodate our design to that and therefore build our application in such a way that it is easy for the zoos to understand.

Some of the other design ideas that have come up was making the system only web based with no installer, having users install an Electron app, having no app to install and using just the context menus from the OS. We ultimately settled with using Electron since it is cross platform and allows development of UI and editing registry files for the context menu. Other problems we ran into was how we would handle updating

our app or how the zoos would handle updating apps on their systems because of security reasons. All of this affected our design thinking and process.

Our client also wanted the app to be built a certain way so we had to come up with a way that satisfied what the client was envisioning, but also design our app in a way that was possible within our experience. During our project planning phase, we had to clarify the requirements with our client in order to move onto the design phase of how we think our app would capture what the client wanted. Originally, the client wanted to include video editing and other features to our software, but it was out of scope of our experience and also not particularly relevant to the main design of what our app is supposed to do. Therefore, during our design thinking phase, we settled with what would best fit the requirements.

3.3 Proposed Design

Right Click Feature

- This feature will allow users of both Windows and Mac OS to right click on a video file and publish their WOW! Moment.
- When someone right clicks on a video file, it shows the option "Open With True360" which opens the Desktop Application when clicked on.

Desktop Application

- The True360 Desktop Application will be made using Electron.
- Users will be able to open relevant files using the True360 Desktop application
- We will use NodeJS for this application
- Available on Mac OS and Windows
- This Desktop application will publish to multiple platforms such as Youtube and Vimeo.

Cloud and Database

- We need a place to store these videos' information
- The application will send notifications to the server when a video is published

Benefits of Mentioned Standards

- NodeJS is a simple language to use and the team has adequate knowledge on it.
- Mac OS and Windows will give True360 the ability to benefit from consumers who use both OS's.
- The Database will allow True360 to monitor and record the videos being published in their brand's name.
- Being able to publish on multiple platforms will allow for more exposure for the videos to be seen and published.

Testing

- Right click testing on Mac OS
- Right click testing on Windows
- Audio formatting and meta data research
- AWS experimentation and research
- Electron Application

Team Experience

In order to determine what technologies to use, our team needs to recognize the strengths and weaknesses in our team experience, then see the potential trade-offs between them.

> *Strengths:*
> - Front End development experience
> - IoT development experience
> - Networking experience
>
> *Weaknesses:*
> - Have little experience with registry editing
> - Have little experience with developing/deploying/updating executables
>
> *Trade-offs:*
> - We have enough experience to complete the majority of the software
> - Our weaknesses can be overcome by studying the necessary resources

Knowing this, our team has several possible solutions and design alternatives to remedy these constraints.

Possible Solutions and Design Alternatives:

- Find a software package that can simplify the registry editing process instead of direct modification
- Use an application API to aid in performing automatic updates

Technology

In the process of finding a solution to the clients problem, we came up with two possible solutions. One of them is a web application while the other is a desktop application.

Web Application

> *Strengths:*
> - No need to deliver updates to all clients
> - No need to create an installer
> - No need to create application for different platforms
> - UI will need to be created and tested
>
> *Weaknesses:*
> - No direct access to zoo server filesystem
> - Will need to handle high-volume video compression and streaming
>
> *Trade-offs:*
>
> - Needs to handle video streaming

Desktop Application

> *Strengths:*
> - Direct access to server file system

- UI will be native to the operating system

  Weaknesses:
  - Updating system will be needed
  - Installer will need to be created for each platform
  - Updates will need to be delivered to all clients

  Trade-offs:
  - Needs to be updated and installed on users' machines
  - Needs to work with native OS

A desktop application also led to a follow-up technology discussion, the type of software to use, which eventually reduced down to whether or not to use Electron or UI native to the OS.

Electron

  Strengths:
  - Works with languages our team is familiar with using
  - Will only have to make one application to run on multiple OS's
  - Automatically updates to the latest version

  Weaknesses:
  - Our team has not directly used Electron in the past

  Trade-offs:

  - Need to spend more time designing a UI
  - Not guaranteed for an UI to look good

Native OS

  Strengths:
  - Guaranteed to work with an OS

  Weaknesses:
  - Our team is unfamiliar with creating native applications
  - Have to directly edit more things in the registry

  Trade-offs:
  - Will have to make different applications to run with each native OS
  - More complex to create than an Electron app

3.5 DESIGN ANALYSIS

Overview

The process of designing the systems for the True360 Desktop Application took many different turns during the ideation process. There were concerns of developing within new frameworks and also across different operating systems. These concerns were quite large in the beginning which had the team worried about how we were going to accomplish this goal. In the end, we settled on the Desktop Application due to the research that was conducted for each possible option that our team had brainstormed. Below is a list of possible solutions and reasoning as to why they would or would not work.

Completely Web Based Application

Due to the concerns of developing a desktop application, the first pivot was to create a web based application. There were many pros and cons for this, but there were just as many for the desktop application. The main reason as to why the web application was not chosen was due to the uploading of large video files to the cloud. For this reason, the Web Application idea needed to be scrapped and head back to the drawing board.

Complete Desktop Application with No Outside Internet Contact

Since the Web Application had troubles with uploading the videos, the next approach was to try and figure out how to integrate the desktop application completely based on the network at the zoos. This was a good approach for the videos living in an on premises database but lacked the ability for True360 to monitor usage of their application. This was a large part that True360 would have liked to have so we needed to head back to the drawing board.

Intermingled Desktop Application with Web Based Functionalities

The final idea that was settled upon was to build a desktop application that sent certain information to a cloud based system. This way, the video uploading problem would be solved and True360 could have visibility as to how much their system was being utilized.

Final Design Choices

The final product that will be delivered is a desktop application that allows zoo employees to easily upload their "WOW!" moment clips to their social media platforms. Behind the scenes, there are programs that use Facebook's, Youtube's and Vimeo's API to upload selected videos to the zoo's corresponding social media account. When this is done, certain data is sent to the cloud based system for True360 to stay up to date on what videos were published through their platform and also which zoos are using the system. These metrics will also be displayed in the desktop application for the zoo employees to see as well so that zoos can have a centralized dashboard to keep track of their published videos.

3.6 DEVELOPMENT PROCESS

Our team is following the Agile development process, as all of our team members are most familiar with this practice. Using an Agile-based approach will allow us to compile the requirements into different tasks to complete. Each sprint will be one week long. Section 2.4 illustrates the tentative timeline that our team will follow in order to complete this product on time, broken up into one week sprints.

We will use Pivotal Tracker in order to assign various tasks and to team member(s) and track progress. Section 2.3 outlines the different milestones and tasks we need to complete in order to finish this product. Additionally, we will use the merge requests on GitLab to track each individual's progress and changes they have made. Our milestones represent the main phases needed to complete the project (phase 2 is dependent on phase 1, phase 3 is dependent on phase 2, etc.). Some tasks from some phases overlap, however those tasks are independent with each other and will not bottleneck the project completion.

Following Agile practice, at the end of each sprint we will have a weekly meeting for the team, as well as a meeting with the client True360, in order to effectively communicate. This will allow for any input or concerns along the development process, as well as give individual and project progress updates.

Furthermore, in line with standard Agile practices, our team will create and evaluate various tests for the features that are carried out and throughout the project development. These tests are outlined in section 4 of this document.

Design Plan Overview

Our project is split into 4 different modules, a desktop application, a centralized cloud server, a zoo server, and publishing platforms. Two of these modules are ones that are already created, the publishing platforms, and the zoo server. We will be developing and creating the other two modules, the desktop application, and the cloud server.

Figure 3.1 shows an overview of the architecture of the system, and how they will communicate. The design for the two modules we are creating and all modules will communicate will be explained.
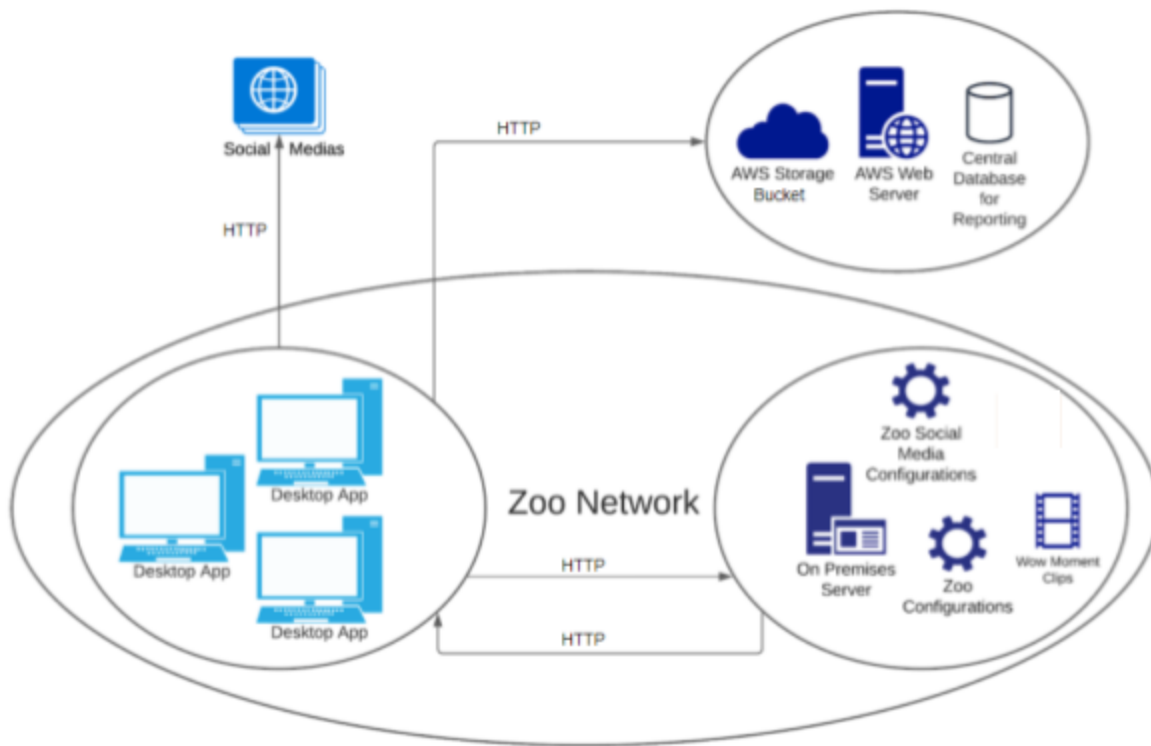
Figure 3.1 Overview of the product's architecture design plans.

Desktop Application

Which use cases will it cover?

- Upload 360 captured camera footage of animals (WOW! moments) to specified media platforms
- Extendability to add additional social media platforms
- Automatically tag uploaded footage with True360 metadata
- Email True360 "Project Edit Requests" with selected videos(s)

Technical Design

The desktop app will be the primary program that users will interact with. It will be created with Electron.js, so that it will be compatible across platforms. The program will be launched when users right click on a video file, and select the correct option. Figures 3.2, and 3.3 show the concepts for right clicking to launch the program, for Windows and Mac, respectively.
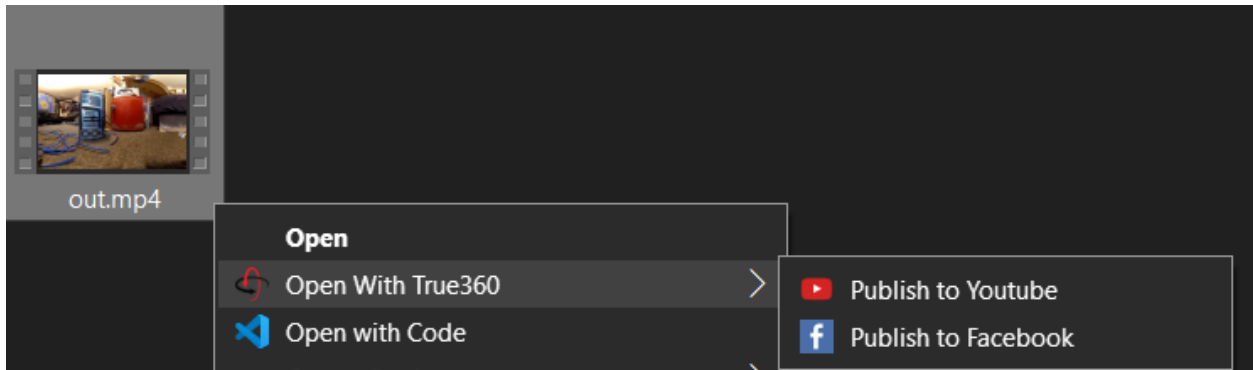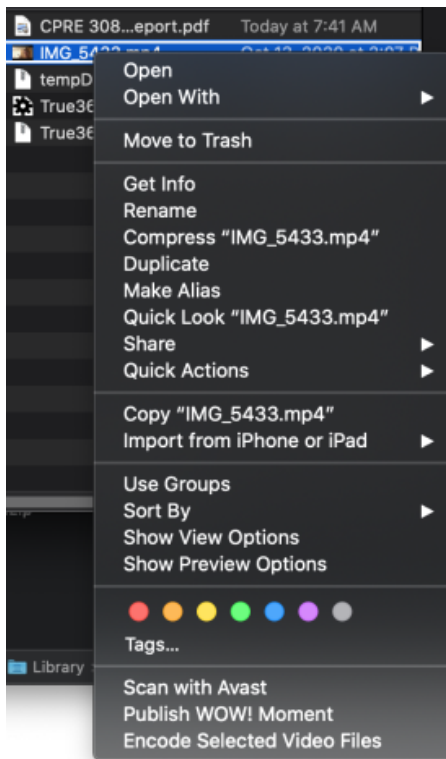
Figure 3.2 Demonstration of the Right-Click Option on Windows

Upon opening the app, the behavior will depend on if it is Mac or Windows. On Windows, since a multi-level context menu is possible, the application will open to the corresponding page for the platform they selected. On Mac, users will be brought to a page where they can select the platform they want to publish to, which will then bring them to the corresponding page.

Wireframes of the user interface are shown in Figure 3.4. Note that this design is only a demonstration of what the final design could look like, and doesn't take into account color. Figure 3.5 also does not show every possible screen within the application, rather the three main screens.

Figure 3.3 Demonstration of the Right-Click Option on Mac
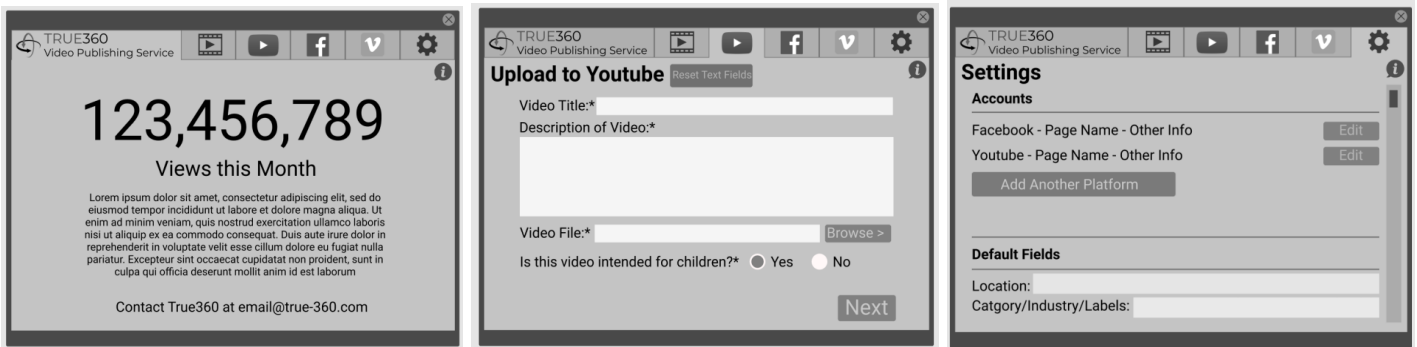
Figure 3.4 (Left-Right) Wireframes of the index page, the upload to YouTube page, and the settings page
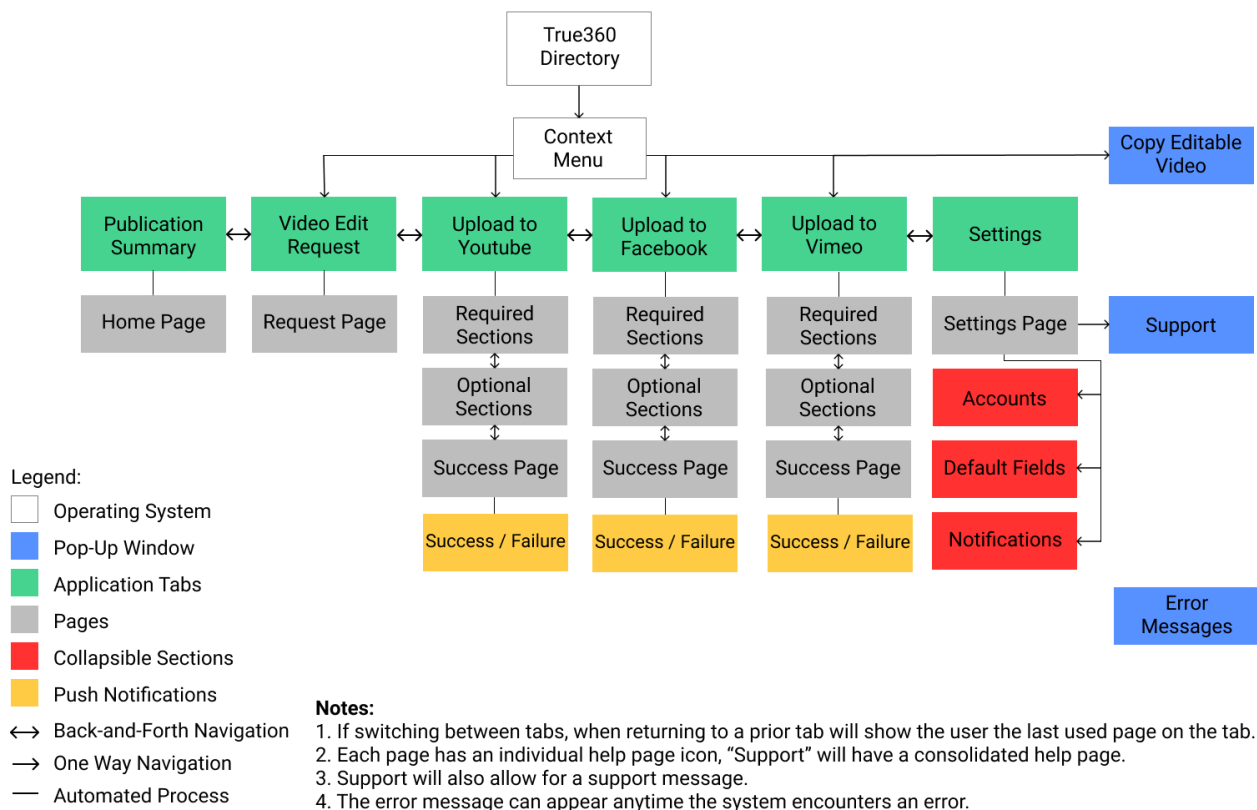

Figure 3.5 Sitemap of the Publishing Application

The page for a publishing platform will contain forms for information about the upload that the user wants to provide, for example, title, description, etc. Once the user has filled in the correct forms, they can click a "Publish" button. A flowchart for what happens when a user clicks "Publish" is seen in figure 3.7.
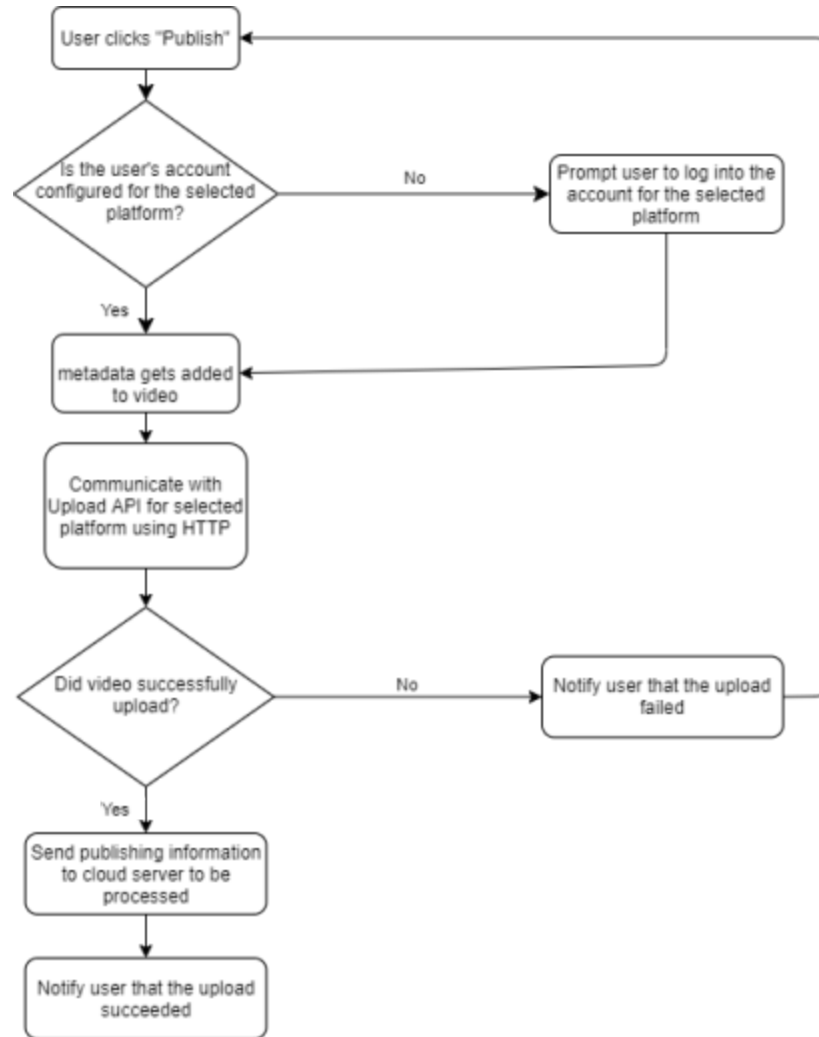
Figure 3.6 User Flowchart for Publishing

In addition to the standard feature of publishing videos, the technical design of other features is outlined as follows:

Bug Report Submissions

- Users will be able to select a button in the application, which will bring them to a form where they can describe their issue.
- Upon clicking submit, the information will be sent to the Zoo Server, where it will be then emailed to support@true360.com

Project Edit Requests

- Users will be able to select one or more videos to be selected for an editing project. If they select an option similar to "Start Edit Project with True360", an email with the selected videos will be sent to editing@true360.com

User Documentation

- Users will be able to select a question mark button that will show them documentation on how to use the product.

<u>Installing and Uninstalling</u>

Although Electron can be used to make installers for both of the platforms we will be supporting, special care needs to be taken in order to add the right click feature.

On Windows

1. The right click feature is added via the Windows Registry
2. Upon installation, a function will be run to add the correct keys to the registry.

On Mac

1. The right click feature is added as an Automator Service.
2. A True360.workflow directory can be created via Automator on MacOS, and then included within the installer, where it will then be run in order to install

During uninstallation, these additional items (the Windows registry keys and the Mac service) will need to be removed.

Centralized Cloud Server

Which use cases will it cover?

*   Document video publication data

Technical Design

The cloud server will consist of an Express.js server that connects to a mySQL database. Both the database, and the server will be hosted with Amazon Web Services. The primary function of the central server is to document video publication data. It will do this by using a database table for each publishing platform. Information stored within the database will consist of general analytical data of a published video, such as likes, and views, as well as video IDs. The specific content included can be seen in Figure 3.7 in the database diagram.
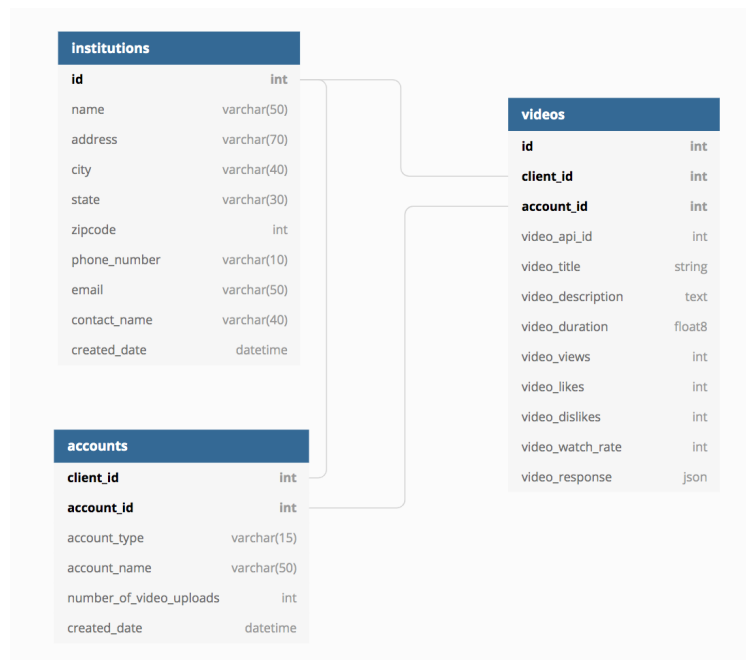

Figure 3.7 - Database Diagram for Video Publication Data

4 Testing

Below are the different units in our design that we will need to test throughout development.

- Test proper retrieval of social media API keys
- Test that API keys can establish connection
- Test that users can log in to all social media accounts
- Test that accounts can be added/deleted
- Test that account configurations are synced between computers
- Test that all API calls are submitted with proper parameters
- Test that database connections are properly configured
- Tests that database upload parameters are correct
- Tests that database calls return the right information.
- Tests that client/user data is stored securely.

4.2 Interface Testing

Since there is no hardware involved, the testing will be mainly software driven. We have broken the project implementation into a front-end app and back-end system which will need to be tested. The interface testing will include the front-end desktop app and back-end system which consists of the desktop server and cloud server.

For the front-end application that is being developed with Electron, we are using Spectron. Spectron is an Electron testing framework for writing test cases. This will make it easy to tell if the UI is functioning correctly as the core functionality of the front-end application is to have users configure social media accounts to and navigate through a UI consisting of tabs. Spectron can test if these tabs are visible, displaying properly and log errors. We will have to set up dummy accounts on each social media platform for publishing videos to test if they are being uploaded properly to the account.

For the back-end server systems, we are using SinonJS and MochaJS. We can send requests to a web server and get the response back to make sure we are getting the correct data. Further testing can be done with mocking server responses from the APIs we are going to use.

4.3 Acceptance Testing

Every task we complete will be assigned to a ticket. In the tickets description it will contain acceptance criteria. These criteria will outline exactly what is expected for the ticket when it is turned in. When a developer is showing the result of a ticket, they must go through each bullet point and show that the functional and non-functional requirements are completed.

Throughout our project development process, we will be involving our client in the acceptance testing. Our group will be doing this testing weekly in accordance with our sprints. Our client will be present as our team develops and specifies our acceptance criteria for each task. By doing it like this, we will not have acceptance criteria until right before we start working on a feature, but that allows us to be able to adjust acceptance criteria based on current needs of the project. We will also be able to know the exact requirements needed, which will avoid any ambiguity. This will help ensure that everyone is on the same page and allow the project to be completed efficiently.

4.4 RESULTS

Throughout this initial design phase, we've needed to complete certain tasks that allow us to know if future project development is possible. We called these tasks "chores" and they enable us to complete further features within the project. The following is a list of chores that we have completed thus far, and the results from them:

- Right Click for Windows Proof of Concept
  - Changing the context menu is available by editing the registry
  - Submenus are possible
- Right Click for Mac Proof of Concept
  - Changing the context menu is available by creating an Automator Service
  - Submenus are not possible
  - File path is able to be passed into an Electron application
- Rapid WireFrame Mock-up for Organization of Desktop Application Pages
  - A "tab" based system will work better than a multi-window application
  - The simpler the system looks will be better for the end user
  - The required video detail fields that an end-user would want
- Research Response Messages After Publishing
  - The APIs for what we are developing for return the correct video statistics we need
  - The YouTube upload API defaults to uploading videos private
  - There are various API limits that may stop us from developing the features as intended
- Amazon Web Services Tests
  - There are free tiers for AWS that will be able to host our cloud side
  - Node JS web server running on an Amazon EC2 Instance
  - MySQL Server running on an Amazon RDS instance
- Video File Types/Compression/Metadata
  - Identifying what sort of standards metadata is stored as and what sort of metadata will be relevant for publishing and tracking video analytics
  - The types of video files we are expected to work with to publish
  - Publishing uncompressed raw video data vs compressed data

To see a list of what each chore enables us to accomplish, please see Table C

Chore	What completing it enables
Right Click for Windows Proof of Concept	Develop ways to create/open Windows Electron apps
Right Click for Mac Proof of Concept	Develop ways to create/open Mac Electron apps
Rapid WireFrame Mock-up for Organization of Desktop Application Pages	Create a user-friendly interface
Research Response Messages After Publishing	Configure the publication database to hold information we can receive
Amazon Web Services Tests	Create a centralized cloud server/database
Video File Types/Compression/Metadata	Optimizing videos per platform

Table C - Why We Needed to Complete Each Chore

5 Implementation

As brought up in section 4.4, we have implemented some basic proof of concept applications to showcase that our design will be possible. The main application is a basic electron app that allows a user to get information about a file.

A user can right click on a file in their file directory, and select "Publish WOW! Moment", which will then open the app with the selected file, as shown in Figures 5.1, and 5.2.
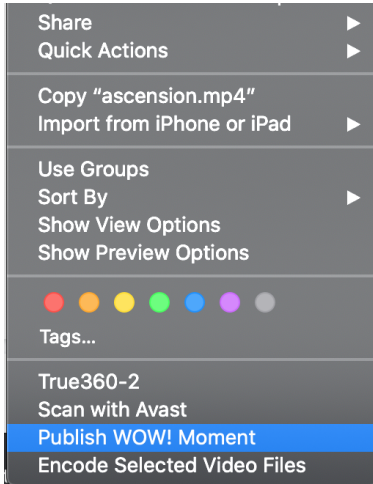

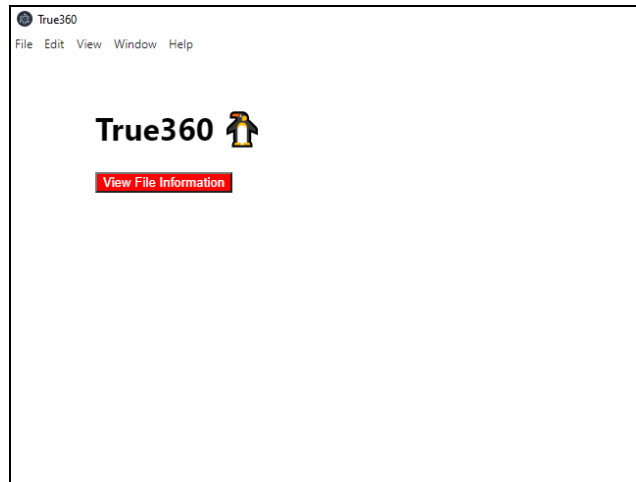Figure 5.1 - Right Clicking to Open a File

Figure 5.2 - Proof of Concept Electron App

Upon clicking "View File Information", the app will update with information about the selected file as shown in Figure 5.3. This showcases that a file path can be passed into the electron app, and the file can be accessed. Once this is possible, it proves that we are able to access the file in order to publish it.
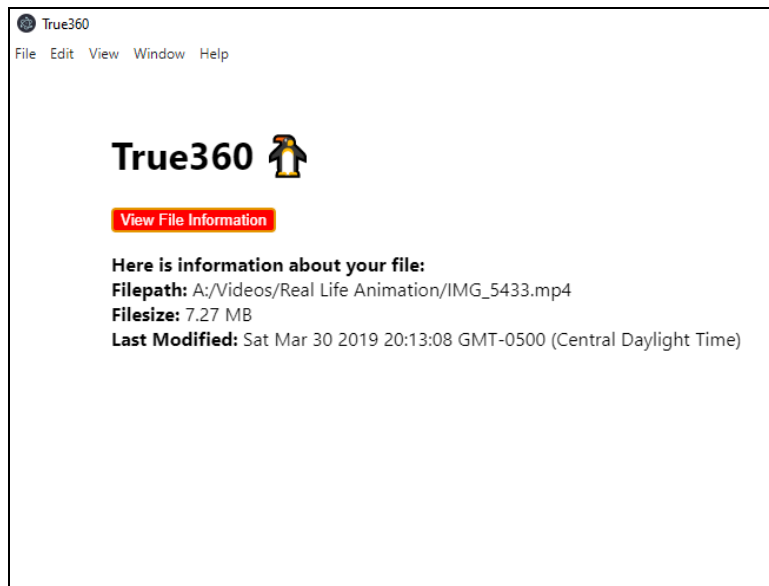

Figure 5.3 - File Information in Electron Proof of Concept

Another large part of the True360 application is the cloud based web server and database that will centralize the data from every institution that is a part of True360. We have implemented a basic proof of concept that will be our approach for the final application.

The web server will be made using nodeJS utilizing the express framework. The POC for the web server is running on an AWS EC2 instance. This web server is connected to the AWS RDS instance that is a mysql database. This allows for http messages to be sent to the web server and ingest/retrieve data from within the database.

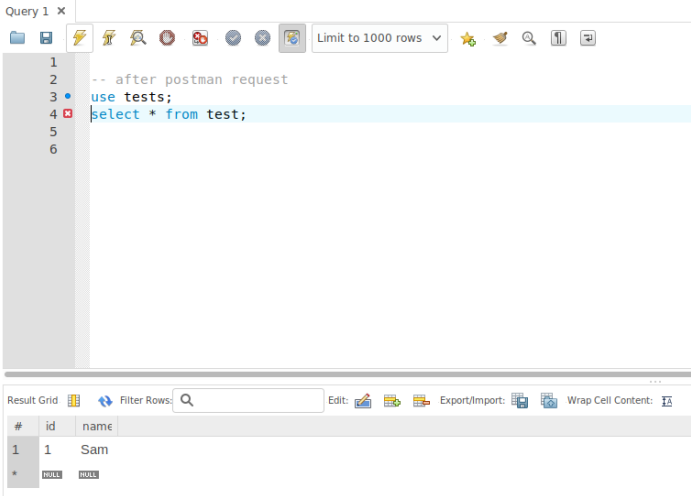An example of sending and receiving requests can be found in Figures 5.4, and 5.5.

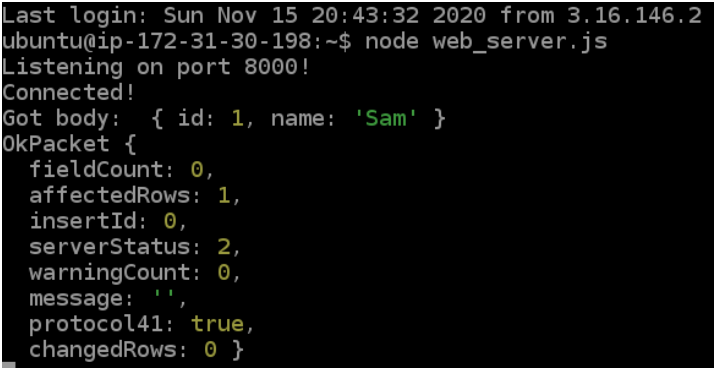Figure 5.4 - Cloud Database after Request is Sent to Add a User

Figure 5.5 - Output on EC2 Instance Once Request is Received

6 Closing Material

6.1 CONCLUSION

In closing, the end product for True360 will be a cross platform desktop application that streamlines the process for publishing videos to different social media platforms. It will allow for True360 to monitor their applications usage and also to manage institutions within their system through a centralized database. Due to the large size of the video files that the zoos and aquariums will be managing, a desktop application with the on premises server is the best possible solution for capturing and managing their 4k 360 degree videos. And finally, this application lets the user focus on which content they would like to share and does the heavy lifting of configuring the best possible settings for their 4k 360 degree video so the zoos and aquariums can spend more time choosing the best possible clips to share.

6.2 REFERENCES

H. Inc., "About Us," *Hootsuite*. [Online]. Available: https://hootsuite.com/about. [Accessed: 26-Oct-2020].

"Final Cut Pro X," *Apple*. [Online]. Available: https://www.apple.com/final-cut-pro/. [Accessed: 26-Oct-2020].

"Adobe Premiere Pro," *Adobe Premiere Pro User Guide*. [Online]. Available: https://helpx.adobe.com/premiere-pro/user-guide.html/premiere-pro/using/whats-new.ug.html. [Accessed: 26-Oct-2020].

6.3 APPENDICES

Appendix A: This appendix contains links to documentation relating to the APIs we will be using in the project. This includes things such as API quota limitations from various social media platforms.

YouTube Quota: https://developers.google.com/youtube/v3/determine_quota_cost

Vimeo Quota: https://developer.vimeo.com/guidelines/rate-limiting

Facebook Quota: https://developers.facebook.com/docs/video-api/guides/publishing/

Appendix B: This appendix contains links to different technology we will be using for development.

Electron: https://www.electronjs.org/

NodeJS: https://nodejs.org/en/about/

Amazon Web Server (AWS): https://aws.amazon.com/about-aws/

Postman: https://www.postman.com/

Spectron: https://www.electronjs.org/spectron

SinonJS: https://sinonjs.org/

MochaJS: https://mochajs.org/

MacOS Automator Service: https://developer.apple.com/library/archive/documentation/LanguagesUtilities/Conceptual/MacAutomationScriptingGuide/MakeaSystem-WideService.html