Colby

Smart Mirror

Project Management Plan

Version 1.3

12/12/2017

Version History

Version #	Implemented By	Revision Date	Reason
1.0	Austin Nantkes	10/6/17	Initial Document
1.1	Austin Nantkes	10/22/17	Fixing Errors in Document, no design changes
1.2	Zena Abulhab	12/4/17	Reorganized table of figures, made glossary table, modified table of contents
1.3	Zena Abulhab	12/12/17	Final adjustments

Table of contents

List of Figures

Glossary of Terms

Term	Definition
Rpi	Raspberry Pi mini computer
Milestone	Successfully integrating a new feature into the overall system
SRS	System Requirements Specification
SDD	Software Design Document

1. Overview

1.1. Project Summary

Purpose, Scope and Objectives:

Purpose: This project aims to provide a comprehensive interactive mirror application for users' convenience. The mirror will recognize a user and display useful information for that user's viewing.

Scope: The software will recognize a user from their face using a camera and classifying system. The software will display a user's emails and upcoming events. It will give live weather reports and the time. An alarm will be added. Relevant news stories to a user will be pulled from a news website to be displayed. After a short period of time the screen will go into standby mode where a user face will awake it.

Objectives: Develop a realistic prototype of the mirror by using a Raspberry Pi and a monitor.

Assumptions and constraints

The software is due December 13th for the class presentation. The plan is to use a raspberry pi to run a node.js web server on to connect to the google API services through javascript and use RSS to pull news events. We plan to use a camera along with a cascade classifier to detect user facial features.

Project deliverables

Smart mirror software, but not an actual smart mirror, delivered on December 13th. The "mirror" will be simulated using a monitor.

Schedule and budget summary

Schedule: Figure 1 shows milestones in a table format, figure 2 represents them in a gantt chart.

SRS deadline is the 19th of October

SDD deadline is the 2nd of November

Final deadline is December 13th

Task Name	Start	End	Duration (days)
Get web server working	11/12/2017	11/20/2017	8
Pull RSS feeds from news sites	11/20/2017	11/22/2017	2
Facial recognition	11/12/2017	11/30/2017	18
Local Weather Forecast	11/20/2017	11/22/2017	2
Display the time	11/20/2017	11/22/2017	2
Link google account	11/22/2017	11/26/2017	4
Display Emails	11/27/2017	11/30/2017	3
Display Events	11/27/2017	11/30/2017	3
Alarm	11/30/2017	12/3/2017	3
Screen auto shutoff	11/22/2017	11/25/2017	3
Voice Recognition	11/30/2017	12/10/2017	10
Presentation	12/13/2017	12/14/2017	1

Figure 1. Task list with start and end dates

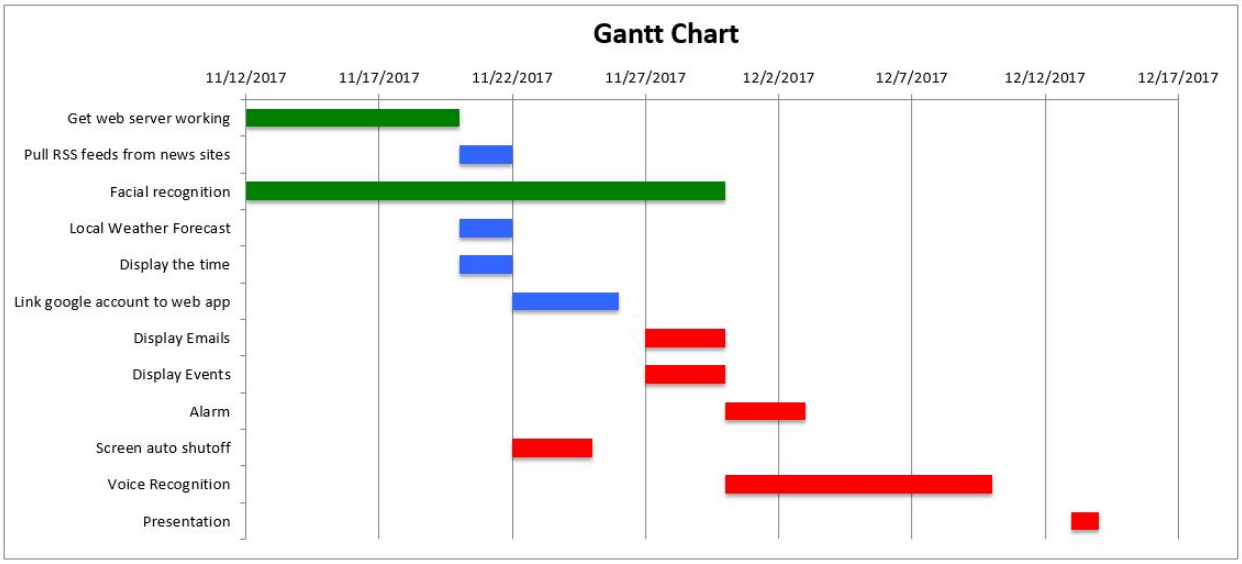

Figure 2. Gantt chart made from figure 1 information. Gives a visual representation of the tasks where color indicates what sequence they are started in. Green and Red both have one outlier, but otherwise the previous colors must be completed before the other tasks are complete.

2. Project organization

2.1. External interfaces

No external organizations

2.2. Internal structure

Quality assurance will be done by the whole team before the product demo

Verification and validation will be based on the original design specifications

2.3. Roles and responsibilities

Austin

Role: Team Leader

Responsibilities: Bringing resources for project, coordinating group work, communicating with client

Gerardo

Role: Team Member and Computer Vision Specialist

Responsibilities: Implementation of facial recognition feature using OpenCV and a cascade classifier. Report progress to the rest of the team and communicate constantly to aid in the completion of late milestone deliverables. Implementation of other features according to the schedule created through team meetings.

Makoto

Role: Team Member and NodeJS Specialist

Responsibilities: Implementation of features according to the schedule created through team meetings. Report progress to the team leader, and make sure that we can finish the project by the deadline as much as possible.

Zena

Role: Team Member and Documentation/QA Specialist

Responsibilities: Implementation of features according to the schedule created through team meetings. Report progress to the team leader, and make sure that we can finish the project by the deadline as much as possible.

3. Managerial Process Plans

3.1. Start-up plan

Estimation Plan

Resources needed: Rpi

Computer Monitor

USB Camera

Low watt speakers

Staffing Plan

We cannot acquire any new help

All 4 personnel will be needed for the entire project for proper QA and

verification/validation

Resource acquisition plan

All resources are being brought in by team members as they are already owned. The one resource being borrow will be a USB camera, from the Colby College CS department lab.

3.2. Work Plan

Work activities

Get website server working: Write the server side program that hosts website using Node.js. Initially, we do not expect it to update database, so it only needs to listen GET request. However, it is likely that we will add a news feature that can update database thorough voice interactions. In that case, we also have to make it to listen POST request. The estimated time for this part is a week and half.

Facial recognition: Implement the visual recognition system using Python and the OpenCV library. The visual recognition system will detect that a user is in front of the mirror using a trained cascade classification system (also implemented in the OpenCV library). The development of the dataset and training of the classifier will take place in the computer cluster of the Davis Robotics Lab and then loaded onto the Raspberry Pi for use with the input from a webcam. The estimated time for this task is 18 days due to the scope of the task. Subtasks will be broken down if necessary immediately after starting work on the project.

Pull RSS feeds from news sites: Write a function that periodically sends HTTP requests to certain news websites which has RSS feeds such as NY Times, and use existing npm package for RSS

feed parsing. This part has to be done after we finish hosting web server. The estimated time for this part is 3 days.

Local Weather Forecast: It uses OpenWeatherMap API, and whenever the server receives GET request, it accesses the weather data from the API and show it on the web page. Including the design of the webpage, it is expected to take 3 days.

Display the time: It will show the current time on the webpage. It is possible to implement this feature by putting a Javascript script tag on HTML code. It is expected to take 2 days to finish this part.

Link Google Account to Web page: Since Google provides API so that we can access google account information easily, we will use Google API. We do have to configure a certain setting on google API console to make the user information accessible. It is expected to take 4 days.

Display Email: We do have to design the web site to show the email address. Designing the website part is very similar to displaying local weather forecast and time part, so it will not take that long time to finish this part. It is expected to take 3 days.

Display Events: Similar to displaying the email, we have to show the the google calendar information on the website. However, it is less certain that how the google calendar information is available and can take longer than displaying email part. It is expected to take 3 days.

Alarm: We will implement alarm feature using chrome.alarms API to make alarm go off at a certain time. We do not have any experience using chrome.alarms API, but it seems to be straightforward to implement this feature. It is expected to take 3 days.

Screen auto shutoff: We want to shut off the screen if a camera does not recognize a human face for a certain amount of time. Since it requires to communicate with OS, it is unsure how we could implement this feature. However, once we figure it out how to interact with OS, it will not take that long to implement this feature. It is expected to take 3 days.

Voice Recognition: Voice recognition feature makes it possible for a user to interact using his/her voice. There are multiple APIs including Google Cloud API. We do not have any experience using speech recognition API, so this part includes the highest risk in this project. It is expected to take 10 days, but it can easily fluctuate.

Schedule allocation

On the next page is our PERT diagram, which provides a detailed look at schedule milestone deadlines and slack.

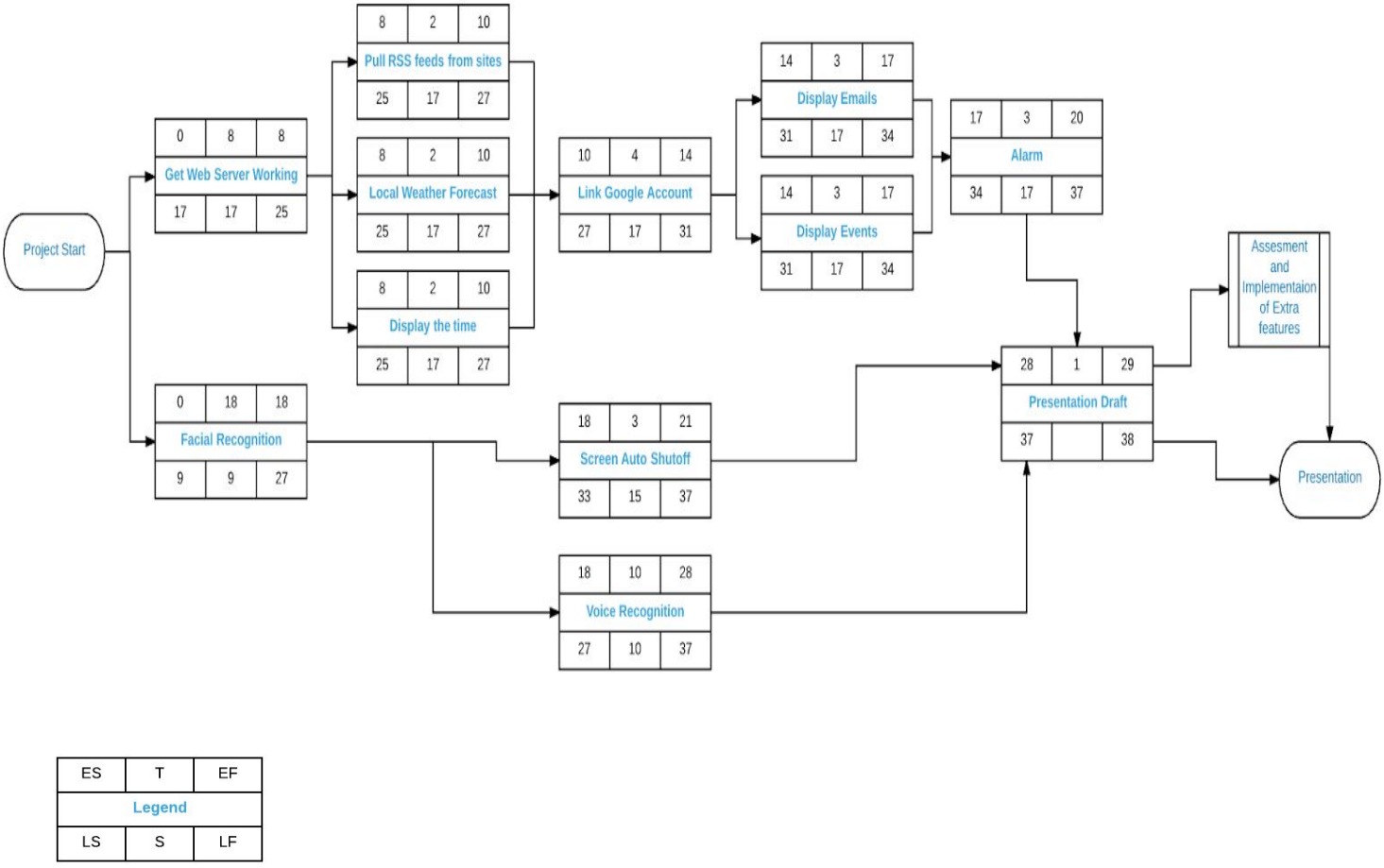

Figure 3. PERT Diagram from Figure 1 information gives a representation of the task milestone timelines.

The top row (from left to right): Early Start, Task Completion Time, Early Finish.

The bottom row (from left to right): Late Start, Slack Time, Late Finish.

Resource allocation

The tasks that require special resources (hardware, special software libraries):

Facial Recognition: OpenCV, Computer Cluster in Davis Lab, Webcam from Davis Lab.

Web Server: Raspberry Pi

Display weather, news, google account, events, etc: Various APIs to be determined (based on availability and suitability)

Budget allocation

N/A

3.3. Control Plan

Requirements control plan

Changes to requirements are measured by seeing what is needed further into the project. Changes will be reported to everyone in the group as well as the client. To control changes, we will analyze the impact of each first and determine if they are absolutely necessary or better than the current requirements.

Schedule Control plan

To measure progress, we will evaluate our status (which tasks are complete, if any are taking longer than planned) at every milestone. When actual progress does not meet planned progress, we can place more effort into tasks that are behind. One form of corrective action is having other members review that task's code and design of the solution.

Quality control plan

For quality control, we will have regular testing, proper commenting/documentation, and code reviewing.

Reporting plan

We will regularly meet the client to discuss progress. We can use powerpoint, diagrams, or short reports. Emails may also be in order to have continual communication.

3.4. Risk management plan

List of main risks in order of probability and gravity with planned responses if the risk were to materialize.

Late milestone completion: The main risk to the project. Late milestone completion can be exacerbated by the fact that the team is not exclusively working on this project, so a buildup of other academic and extracurricular work coupled with a belated milestone completion date could prove very challenging. The planned response to this risk is more preventive than reactive. The PERT diagram in the schedule allocation section outlines the target deliverable dates and provides ample slack time for most tasks, which will be our safeguard against belated milestones.

Unexpected compatibility issues: The project's scope encompasses processes in a handful of different languages and applications. An unforeseen problem with compatibility could arise when trying to achieve internal communication between the different parts of the mirror. The planned response is twofold: Preventive through ample slack time, and reactive through possible expert consultation (e.g. A solution to a problem with the linux installation could be reached more quickly by making an appointment to consult with Professor Li or Randy Downer).

Infeasibility of planned features: The feature at the biggest risk of infeasibility is voice recognition, but others could potentially pose a similar problem. Planned responses are: changes in the design of the features, scrapping non-critical features, and ample slack time.

3.5. Closeout plan

Our closeout plan is the end of semester presentation where we describe which lessons we learned and an analysis of the project objectives that we achieved. This will be given as an oral demonstration to the class on December 13th.

4. Technical Process Plans

4.1. Process model

In figure 2, the Gantt chart shows the milestones to be completed and at what time. In figure 3 the PERT chart shows how much slack each process has along with which milestones are pre-requirements for each milestone.There are no approvals for the project timeline, and no formal reviews schedules. The final deliverable is a working raspberry pi software bundle that can be paired with a mirror to create a Smart Mirror.

4.2. Methods, tools, and techniques

Tools (Utilized Languages)

Javascript

HTML

Python

Integration Technique

Github

Documentation Techniques

READMEs

Comments when uploading to Github

4.3. Infrastructure plan

Hardware, OS, network, and software vary per person

4.4. Product acceptance plan

The client is satisfied with the completed product.

5. Supporting process plans

5.1. Configuration management plan

If any changes to the project occur, or any timeline changes happen due to the aforementioned risks, the client and all team members will be informed and will create a new time plan to catch back up.

5.2. Verification and validation plan

Use of Milestone tracking, peer reviews and prototyping will keep the project on track.

5.3. Documentation plan

All software will have documentation throughout it to help readability of the code. Requirements and other important work documents will be created and reviewed by all team members. Work documents will follow the templates provided by IEEE on the Colby College Moodle Webpage for CS397.

SRS deadline is the 22th of October

SDD deadline is the 2nd of November

5.4. Quality assurance plan

Constant review and analysis by other team members will occur on each milestone.

5.5. Process improvement plan

If the team has extra time to go back and redo certain features to improve the process it must be approved as to not disrupt the current work on the project. The next phase of the project will probably involve a large improvement section.