Ryan Wiesenberg

rwies@umich.edu

Abstract

Software Requirements Specification (SRS) based on Bruegge & Dutoit for a smart mirror.  
The goal of the smart mirror is to aggregate and present daily and upcoming tasks, aggregated for the user approaching the mirror based on supported, linked accounts.

Smart Mirror

CIS 553 Term Project

Table of Contents

[1 Introduction 2](#_Toc87115113)

[2 Current System 2](#_Toc87115114)

[3 Proposed System 2](#_Toc87115115)

[3.1 Overview 2](#_Toc87115116)

[3.2 Functional Requirements 2](#_Toc87115117)

[3.3 Nonfunctional Requirements 2](#_Toc87115118)

[3.4 Constraints 2](#_Toc87115119)

[3.5 System Models 2](#_Toc87115120)

[3.5.1 Scenarios 2](#_Toc87115121)

[3.5.2 Use Case Model 2](#_Toc87115122)

[3.5.3 Object Model 2](#_Toc87115123)

[3.5.4 Dynamic Models 2](#_Toc87115124)

[3.5.5 User Interface 2](#_Toc87115125)

[4 Glossary 2](#_Toc87115126)

# Introduction

The purpose of this document is to outline the software requirements and resultant system models for the smart mirror project. The goal of this project is to provide a centralized system for multiple users to interact with to see upcoming tasks for themselves and the group. This system is expected to contain a separate list of tasks for individual users and a list of tasks for the group. Additionally, the system should be able to differentiate users and present tasks and other information relevant to them.

The next section, Current System, details the current system state, based on user interviews. The Proposed System section then seeks to outline how the proposed system will seek to address the gaps in the current system, the system requirements, and any constraints on the system design. Finally, the Proposed System section will outline the resulting system models from the requirements specification and the goal user interface.

# Current System

Currently this system does not exist in an aggregated format and users must independently, manually view their upcoming tasks and mentally compare their deadlines and requirements. This creates undue stress for the user and detracts from making progress on the tasks themselves. Additionally, it is easy to forget one of the many systems used to track upcoming tasks and events and miss a task that is due or has been in the queue for an extended period without action.

# Proposed System

## Overview

The primary goal of the smart mirror system will be to minimize the obstructions to the users as rephrased from the above:

* Multiple task tracking systems or applications
* Maintaining a mental model of task priority
* Obscured visibility of tasks due to infrequent access

For ease of user access, the system should also be able to provide the following:

* configuration for different users
* method to authenticate and validate which user is accessing the system

This system is expected to run on low-end static hardware, affixed to a wall as a touch screen behind a mirror-like device. This document will not detail the physical element construction process but will aim to minimize computation overhead and design all UI elements and user interactions for a touch-screen device for ease of integration into the hardware.

## Functional Requirements

This section will organize itself now based on the goals listed above:

1. The system must be able to connect to multiple task tracking systems
   1. The systems should at least include access to a user’s Trello, Google Calendar, and Tody.
   2. All task systems linked should be shown adjacent in the same calendar and task list
   3. A user should be able to complete tasks by accessing the mirror
2. Mental model maintenance
   1. All tasks should optionally be able to store a due date or no due date
   2. All tasks should be able to signal the user if
      1. They have not been worked on in some time
      2. The due date is coming up
   3. A user should be able to check current individual tasks and group tasks
      1. Sort by due date, name, last accessed, and people involved
      2. There should be a calendar view to see the soon to be due tasks
3. Obscured visibility of tasks due to infrequent access
   1. A user must be able to view tasks that have not been accessed or worked on in some time
   2. A user can request a random task with infrequent access to complete
4. Configuration for different users
   1. Multiple users, and the ability to add users
   2. Users must be able to access and configure multiple linked task systems
5. Method to authenticate and validate which user is accessing the system
   1. The user must choose a user from the menu and then sign in with a passcode
   2. If possible, the system should be able to authenticate the user with a camera

## Nonfunctional Requirements

* The data storing the passcode does not have to be very secure, but the system should not be open to access from the internet.
* The system should minimally be supported by a RaspberryPi Zero or similar device that can support a touch screen monitor.
* A user needs access at any hour of the day
* The user must have a convenient way to manage the system as an administrator
* Uptime is not important this a non-critical component
* The product must be usable through a touchscreen and have a black background with primarily white text to promote the ability to use it as a smart mirror.
* Standby screen should be primarily black

## Constraints

## System Models

### Scenarios

### Use Case Model

### Object Model

### Dynamic Models

### User Interface

# Glossary