### **Automatic Time Management System**

(ATOMS)

# **CS 3337 Software Engineering**

# **Application Requirements Specification Document**

## Prepared By:

LaFrance, Montague

Chan, Micky

Gomez, Carlos

Vargason, Austin

# **T**able of Contents

|  |  |  |
| --- | --- | --- |
| Section Number | Section Name | Page Number |
| 0.0 | Document Revision History | [3](#_Document_Revision_History_1) |
| 1.0 | Introduction | [4](#_Introduction) |
| 1.1 | Purpose | [4](#_1.1_Purpose) |
| 1.2 | Intended Audience and Reading Suggestions | [4](#_1.2_High_Level) |
| 1.3 | Product Scope | [4](#_1.3_Product_Scope) |
| 1.4 | Definitions, Acronyms, and Abbreviations | [4](#_1.4_Definitions_and) |
| 1.5 | References | [5](#_1.5_References) |
| 2.0 | Overall Description | [6](#_Overall_Description) |
| 2.1 | Product Perspective | [6](#_2.1_Product_Perspective) |
| 2.2 | Product Functions | [6](#_2.2_Product_Functions) |
| 2.3 | User Classes and Characteristics | [7](#_2.3_User_Classes) |
| 2.4 | Operating Environment | [7](#_2.4_Operating_Environment) |
| 2.5 | Design and Implementation Restraints | [7](#_2.5_Design_and) |
| 2.6 | User Documentation | [7](#_2.6_User_Documentation) |
| 2.7 | Assumptions and Dependencies | [7](#_2.7_Assumptions_and) |
| 2.8 | Apportioning of Requirements | [8](#_2.8_Apportioning_of) |
| 3.0 | External Interface Requirements | [8](#_External_Interface_Requirements) |
| 3.1 | User Interfaces | [8](#_3.1_User_Interfaces) |
| 3.2 | Hardware Interfaces | [9](#_3.2_Hardware_Interfaces) |
| 3.3 | Software Interfaces | [9](#_3.3_Software_Interfaces) |
| 3.4 | Communications Interfaces | [9](#_3.4_Communication_Interfaces) |
| 4.0 | Requirements Specification |  |
| 4.1 | Functional Requirements |  |
| 4.2 | External Interface Requirements |  |
| 4.3 | Logical Database Requirements |  |
| 4.4 | Design Constraints |  |
| 5.0 | Other Nonfunctional Requirements |  |
| 5.1 | Performance Requirements |  |
| 5.2 | Safety Requirements |  |
| 5.3 | Security Requirements |  |
| 5.4 | Software Quality Attributes |  |
| 5.5 | Business Rules |  |
| 6.0 | Other Requirements |  |
|  |  |  |

# Document Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Version # | Revision Author | Revision Summary | Revision Date |
| 0.1 | Austin Vargason | Initial Draft | 2/23/19 |
| 0.2 | Austin Vargason | Meeting Template Requirements | 3/9/19 |
| 0.3 | Carlos Gomez | Added Performance and Safety Requirements | 3/13/19 |
| 0.4 | Austin Vargason | Completed Section 2 and 3 | 3/14/19 |

# Introduction

This document seeks to outline the software requirements for the Automatic Time Management System (ATOMS) android application. ATOMS is an Android Application that helps the user automatically manage their time. The ATOMS application allows users to schedule projects, assignments, goals, or other activities around their current schedule. ATOMS allow users to integrate their existing Google Calendar data, which creates an intuitive scheduling experience. The goal of the ATOMS development team is to improve time management and reduce stress for our users.

## 1.1 Purpose

This SRS document version 0.1 contains specifics for the ATOMS application for Android. In this document, the overall description of the product, as well as the product’s functional, interface, design, and, security, and safety requirements will be outlined. The architects involved in the completion of the ATOMS project shall use this document to create an overall Solution Design Document (SDD).

## 1.2 Intended Audience and Reading Suggestion

This document is intended to be viewed by developers and architects involved with the ATOMS application to fully understand the requirements of the application and the interfaces it presents. This document may also be viewed by third-parties under the scope of product presentations.

## 1.3 Product Scope

The scope of the ATOMS application is Android Users on operating systems at or above: Android version 7.1.

## 1.4 Definitions and Acronyms

Android Studio: Development environment to develop Android Applications

Java: Object Oriented Programming Language

GUI: Graphical User Interface

ATOMS: Automatic Time Management System

API: Application Programming Interface

OS: Operating System

XML: extensible markup language

Strictly Scheduled Event: A calendar event that has a pre-defined start and end time

Automatically Scheduled Event: A calendar event scheduled by the defined ATOMS algorithm

## 1.5 References

1.5.1 Google Sign-In and Google Play Services Authentication

Author: Google

Version Number: 16.0.1

Source: <https://developers.google.com/identity/sign-in/android/start-integrating>

1.5.2 Google Calendar API

Author: Google

Version Number: v3-rev305-1.23.0

Source: <https://developers.google.com/calendar/>

1.5.3 Google OAuth Client

Author: Google

Version: 1.23.0

Source: <https://developers.google.com/api-client-library/java/google-oauth-java-client/>

1.5.4 Google Code Labs for Android Development

Author: Google

Source: <https://codelabs.developers.google.com/android-training/>

1.5.5 Android Studio Documentation

Author: Google

Source: <https://developer.android.com/studio/intro>

# Overall Description

The ATOMS application has been conceived as an Android application for use on Android phones at or above OS level 7.1. The ATOMS application establishes a connection to the user’s Google account through an initial sign in screen the first time the app is launched. Once logged in, the user will be taken to a screen showing the user’s events for today’s date with a floating action button in the bottom corner. The floating action button gives the user to add auto-scheduled events to their calendar, or strict events (those events with a defined start and end time). The main screen will also have an options button that will allow the user to change the theme of the app as well as sign out, switch user accounts, or change the calendar view to a daily, weekly or monthly view. The user may also use gesture controls (swipe left, or right) to navigate throughout their calendar.

## 2.1 Product Perspective

Many applications have designed for the task of time Management, but many of them feel unintuitive and require too much effort to justify consistent use. The ATOMS development team has the perspective that the ATOMS application can be a powerful time management tool that is also easy to use.

## 2.2 Product Functions

* Login: User is able to log in through their Google Account
* Calendar Views: Presents the choice of Daily, Weekly, or Monthly Calendar View
* Adding Events: The user may add a strictly scheduled event or an automatically scheduled event
* Theme Options: The user shall have the ability to change the color theme of the app
* Gesture View Switching: The user shall have the ability to use a swipe-left or swipe-right gesture to switch to the next day, week, or month’s events depending on the current view provided
* Account Switching: The user may have the ability to switch the user account through an option in the options menu
* Data Backup: All calendar changes shall be applied to the user’s Google calendar data. If the user chooses to undo their changes, the user may have the ability to undo the previous added event or go back to their original calendar import.

## 2.3 User Classes and Characteristics

* General Assumptions:
  + Technical Expertise: Basic Smartphone Usage
* Business Professionals:
  + Frequency of Use: High
  + Characteristics: Uses ATOMS with business meetings present in Google Calendar and uses ATOMS scheduling abilities to schedule project workloads or deadlines
* Students:
  + Frequency of Use: High
  + Characteristics: Uses ATOMS with classes present in Google Calendar and Uses ATOMS scheduling abilities to schedule workloads for assignments and studying.
* General Consumer:
  + Frequency of Use: Medium
  + Characteristics: Uses ATOMS with general meetings or events present in Google Calendar and uses ATOMS scheduling abilities to schedule out chores, work, or general day to day tasks.

## 2.4 Operating Environment

The ATOMS application requires Android version 7.1 and above.

## 2.5 Design and Implementation Restraints

* Hardware Limitations:
  + Limited to Android devices
* Memory Constraints
* Capabilities of Google APIs

## 2.6 User Documentation

* Help documentation present in ATOMS application

## 2.7 Assumptions and Dependencies

* API Dependencies: Google Calendar and Google Sign In
* This application uses a minimum API version of Android in order to support as many devices as possible as presented in [2.4 Operating Environment](#_2.4_Operating_Environment)

## 2.8 Apportioning of Requirements

* User Theme Creation: Would allow users to define a custom theme for the application instead of choosing a preselected theme.
* Account Switching: Allows Users to switch accounts in the application if using multiple email accounts
* Undo: Ability to undo a previously submitted change to the User’s calendar in scheduling.

# External Interface Requirements

## 3.1 User Interfaces

|  |  |
| --- | --- |
| Requirement Number | Requirement Description |
| 3.1.1 | Login screen shall be presented to the user on the first time the app is opened |
| 3.1.2 | The login screen shall present the ATOMS logo at the top of the layout |
| 3.1.3 | The login screen shall present a button to login with Google Sign In, this will redirect the user to the standard google sign in screen as Implemented in the API |
| 3.1.4 | A smooth transition from login screen to the main app screen shall be created |
| 3.1.5 | The center of the main app screen shall consist of card views, each representing a calendar event |
| 3.1.6 | The main app screen shall have an easily accessible floating action button in the right corner of the screen to add a new event to the current view (dynamically scheduled, or statically scheduled) |
| 3.1.7 | The main app screen shall present a settings button in the upper right-hand corner of the app, which presents a dropdown menu listing the options: “change theme”, “change view”, “log out” |
| 3.1.8 | The change theme screen shall present a list of themes in a option box format to update the theme of the application |
| 3.1.9 | The change view screen shall present a list of view types (daily, weekly, monthly) in a option box format to change the event view on the main app screen. |
| 3.1.10 | The add event screen as accessed from the floating action button shall present an event type option (static, dynamic) in a select box format. |
| 3.1.11 | The add event screen shall change its user input fields based on event type selected |
| 3.1.12 | Both statically scheduled events and dynamically scheduled events shall populate the event screen with a textbox representing the title of the event. |
| 3.1.13 | Statically scheduled events shall populate the event screen with a date chooser representing a start datetime and a date chooser representing an end time. |
| 3.1.14 | Dynamically scheduled events shall populate the event screen with a number chooser representing expected work in hours needed and a date chooser representing a due date |

## 3.2 Hardware Interfaces

The supported hardware are devices that run Android version 7.1 and later.

## 3.3 Software Interfaces

3.3.1 Google Sign-In and Google Play Services Authentication

Author: Google

Version Number: 16.0.1

Source: <https://developers.google.com/identity/sign-in/android/start-integrating>

3.3.2 Google Calendar API

Author: Google

Version Number: v3-rev305-1.23.0

Source: <https://developers.google.com/calendar/>

3.3.3 Google OAuth Client

Author: Google

Version: 1.23.0

Source: <https://developers.google.com/api-client-library/java/google-oauth-java-client/>

## 3.4 Communication Interfaces

* Google Sign In is handled through Google Play Services API
* Google Calendar and Oath work through Google’s REST API communication methods

# Requirements Specification

|  |  |
| --- | --- |
| Requirement Number | Requirement Description |
| 4.0.1 | The login screen shall be presented to the user when previous authentication is not present |
| 4.0.2 | The login screen shall only redirect to the main app upon proper retrieval of a Google Authentication |
| 4.0.3 | The main app screen shall present the user with the day’s current calendar event (see section 4.1 for functional specifics) |
| 4.0.4 | The main app screen shall be easily navigable and scroll when necessary. |
| 4.0.5 | Google Calendar events shall be loaded upon load of the main app screen and upon submission of a new calendar event |
| 4.0.6 | The user shall be notified that changes in the app to calendar events will affect their underlying Google Calendar through the Calendar API |
| 4.0.7 | The floating action button specified shall present a fragment to the user that allows the user to create a new statically scheduled or dynamically scheduled event (see section 4.1 for inputs and outputs) |
| 4.0.8 | Transitions between layouts / fragments shall be defined to keep the app from being in a frozen state during loading |
| 4.0.9 | Theme options provided from accessing the fragment provided when selecting the theme settings option shall update the app theme accordingly |
| 4.0.10 | Event views shall update the main screen to a new view type (weekly, monthly, daily) |
| 4.0.11 | Notifications for upcoming events shall be delivered via Android’s notification system by the ATOMS app |
| 4.0.12 | The application shall notify a user when an event is past due and prompt for completion status. |
| 4.0.13 | The app shall not rearrange the User’s first time input calendar events unless authorized by the user |

## 4.1 Functional Requirements

|  |  |
| --- | --- |
| Requirement Number | Requirement Description |
| 4.1.1 | The application shall receive the User’s google Calendar data upon authentication through an API call |
| 4.1.2 | The application shall handle the Calendar events as list input and output dynamically generated card views on the main screen |
| 4.1.3 | Upon submission of a new statically scheduled event from the event screen, the application shall add and event to the Google Calendar data through a push and then add the new event to the main screen’s output |
| 4.1.4 | Upon submission of a new dynamically scheduled event from the event screen, the application shall auto schedule the event, then add and event to the Google Calendar data through a push and then add the new event to the main screen’s output |
| 4.1.5 | If the input of calendar events exceeds the length of the main app screen, the output shall become scrollable |
| 4.1.6 | A statically scheduled event shall accept an input of an Event Name, Start Time, and End Time. |
| 4.1.7 | A dynamically scheduled event shall accept the input of a priority, number of hours estimated, and a due date. |
| 4.1.8 | Error handling shall be present when analyzing new event inputs, specific GUI input types shall be used to restrict what the user may enter in |
| 4.1.9 | The app shall handle the users request to create a new statically scheduled event and add it into the user’s calendar |
| 4.1.10 | Upon selection of a settings menu option, a fragment shall be loaded updating the display of the main screen, taking the id of the element selected as input. |
| 4.1.11 | The theme menu shall take a predefined theme selection as input and output the color theme to the app accordingly |
| 4.1.12 | Event types and priority shall be used as input to color scheme of the card views on the main app |
| 4.1.13 | The view change fragment shall take a predefined view type as input and update the output of the main screen accordingly. |
| 4.1.14 | A log out setting item selection shall unauthorize the current user and update the app interface output to the login screen. |

## 4.2 External Interface Requirements

|  |  |
| --- | --- |
| Requirement Number | Requirement Description |
| 4.2.1 | Google Login: The application shall implement a Google Login for authentication using OAuth 2.0 for connecting to the Google Calendar API |
| 4.2.2 | Event Creation: The app shall accept the input designated above in the Add Event fragment and construct a new Calendar Event interfaceable with the Google Calendar API |
| 4.1.3 | REST API calls shall be established for communication with the Google Calendar API |

## 4.3 Logical Database Requirements

Logical database is not required, data will be handled through push and pull to Google Cloud.

## 4.4 Design Constraints

* **OS Level Constraints**: Operating system must be a supported Android Version
* **Google Calendar Access**: The system may be limited on the access level of private Calendar events depending on previous OAuth
* **Hardware Level Constraints**: Hardware was support newer level Android operating systems, OS updates may depend on carrier / phone manufacturer.
* **Invalid Event Requests**: The system may be unable to create dynamically scheduled events if the user’s calendar is already full to beyond reason.