Smart Planner

Version 1.0

[Note: The following template is provided for use with the Rational Unified Process. Text enclosed in square brackets and displayed in blue italics (style=InfoBlue) is included to provide guidance to the author and should be deleted before publishing the document. A paragraph entered following this style will automatically be set to normal (style=Body Text).]

[To customize automatic fields in Microsoft Word (which display a gray background when selected), select File>Properties and replace the Title, Subject and Company fields with the appropriate information for this document. After closing the dialog, automatic fields may be updated throughout the document by selecting Edit>Select All (or Ctrl-A) and pressing F9, or simply click on the field and press F9. This must be done separately for Headers and Footers. Alt-F9 will toggle between displaying the field names and the field contents. See Word help for more information on working with fields.]

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <dd/mmm/yy> | <x.x> | <details> | <name> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Introduction 3

1.1 Purpose 3

1.2 Scope 3

1.3 Definitions, Acronyms, and Abbreviations 3

1.4 References 3

1.5 Overview 3

2. Overall Description 3

3. Specific Requirements 3

3.1 Functionality 3

3.1.1 <Functional Requirement One> 3

3.2 Usability 3

3.2.1 <Usability Requirement One> 3

3.3 Reliability 3

3.3.1 <Reliability Requirement One> 3

3.4 Performance and Security 3

3.4.1 <Performance Requirement One> 3

3.5 Supportability 3

3.5.1 <Supportability Requirement One> 3

3.6 Design Constraints 3

3.6.1 <Design Constraint One> 3

3.7 On-line User Documentation and Help System Requirements 3

3.8 Purchased Components 3

3.9 Interfaces 3

3.9.1 User Interfaces 3

3.9.2 Hardware Interfaces 3

3.9.3 Software Interfaces 3

3.9.4 Communications Interfaces 3

3.10 Database Requirements 3

3.11 Licensing, Legal, Copyright, and Other Notices 3

3.12 Applicable Standards 3

4. Supporting Information 3

# Introduction

This Software Requirement Specification document captures the complete software requirements for the “Smart Planner” application. This includes the purpose, scope, definitions, acronyms, abbreviations, references, and overview of the SRS

## Purpose

The SRS fully describes the external behavior of the “Smart Planner” application. It also describes nonfunctional requirements, design constraints, and other factors necessary to provide a complete and comprehensive description of the requirements for the software. This document will be used as the guideline to develop this system.

## Scope

The “Smart Planner” is an Android application together with a web server which provide the functionality on planning the daily routine. It ranges from about scheduling a message to someone to scheduling a set of meetings in business life. The users can schedule a task based on time or on location or on both. When the user arrives at the scheduled location or the time arrives, then a simple notification will be visible on the phone. User can either choose “Complete” or “Forget” in the notification. Further the user can save their favorite locations within the application.

The main contrasting feature of the application is the artificial intelligence which will be integrated to analyze the data and provide an optimized schedule. When the system identifies the user is late for some place he/she has scheduled an event, the system will inform the user about the delay and will provide a route on a map which he/she can use to arrive at the location. At the end of every day the data about the user’s daily tasks are sent to a web server. Then the server analyze the data and creates an predicted schedule for user for future days. The daily and monthly reports will be sent via emails by the server. The Android application provides an option to generate a predicted schedule via server which will make it easy for users.

## Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| Term | Definition |
| User | Person who use the Android application to schedule tasks |
| Server | Web server coded using PHP |
| SRS | Software Requirements Specification |

## References

[This subsection provides a complete list of all documents referenced elsewhere in the SRS. Identify each document by title, report number if applicable, date, and publishing organization. Specify the sources from which the references can be obtained. This information may be provided by reference to an appendix or to another document.]

## Overview

The rest of the SRS contains three sections. The first section will give an overall description of the system functionality and the system interaction with other systems. And also it will describe the system constraints and assumptions about the product.

The second section will give a detail description of functional and non-functional requirements of the system. The section will describe each functional requirements and regarding non-functional requirements how to meet them.

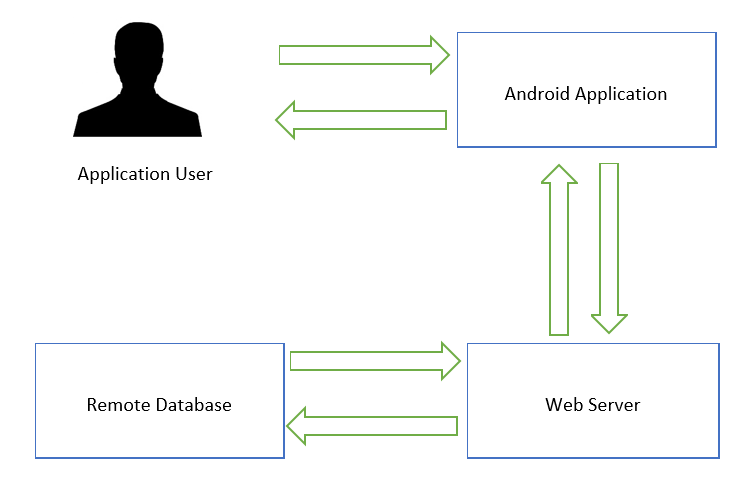
The final section gives the details on supporting information about SRS. It includes the table of contents, appendix and the indices.

# Overall Description

This section of the SRS describes the general factors that affect the product and its requirements. This section does not state specific requirements. Instead, it provides a background for those requirements, which are defined in detail in Section 3.

## Product perspective

The system consists of an Android application and a web server to interact with it. The web server will use a separate remote database to store the data. Application users can only interact through the Android application. The following block diagram will show the communication between components.



The web server analyzes the data sent by the Android application using algorithms and it sends the reports to the user via emails. User can request for predicted schedules via Android application, then the application sends a request to the web server and the server process the data and return the schedule to the Android application.

## Product functions

The functionalities of the user are,

* Register to the system
* Schedule a message based on time
* Schedule a task based on time or location
* Check for the predicted schedule
* Check for a route for a certain location

Tasks of the system are,

* Notifying the task to be done according to the alert time
* Notifying the tasks to be completed within a certain location when the user arrives at that location
* Asking the user whether he completed the tasks when he leaves a certain location
* Analyze the daily gathered data and update the predicted schedules (by the web server)
* Sending daily, weekly and monthly reports by analyzing the data (by the web server)

## User characteristics

There is only one type of user, the application user who uses the Android application. He/she needs to register to the system and according to the tasks he/schedule the web server will automatically respond. An admin user is not available as he/she does not have a specific task to do with the system.

## Constraints

* The Android application will only support the operating systems which have a version 5.0 (Lolipop) or higher.
* Location based tasks will work accurately within the places where the GPS or Network signals are available.
* The capacity of the remote database will limit the space allocated for a user. Therefore, the data stored about a user will need to be cleared after a certain period.

## Assumptions and dependencies

The main assumption is that the GPS of the user’s Android phone function properly. The phone is assumed to be connected to the internet when the application needs to connect with the web server. The user is assumed to have a google account before registering to the system.

## Requirements subsets

# Specific Requirements

This section of the SRS contains all functional and non-functional requirements in detail. This also include the qualitative requirements of the system.

## Functionality

This section describes the functional requirements of the system.

### User functions

#### User registration

[Describe each of the function of the system/ user activity in detail.]

When the user installs the Android application and runs it for the first time, then the user will be asked to “sign in with google”. Once the google sign in is complete, the application will send the email address of the user to the web server. The web server will create an account for the user based on the email address of the user and will send the confirmation message to the application.

#### Change user email address

In the settings page of the application, an option to change the user email will be displayed. When the user select that option, a “google sign in” option will be displayed. If the google sign in is completed, the new email address and the previous one will be sent to the web server and the account will be updated.

#### Schedule a message

User can select a contact (or a valid phone number) to send the send the message and a date and time to send it. The details about the scheduled message will be stored locally within phone.

#### Schedule a task

User can schedule a task entering a short description of it. He/she can select the date of the task. Three types of tasks will be available. (Location based, Time based and based on both). If the task is based on time only then the time of the task and the alert period must need to be entered. If the task is only based on location only then the location and range (the distance from the target location the alert need to appear) must need to be entered. If the task is based on both then all the above details must be entered. An alarm will be set for each task.

#### Edit a scheduled task

All the scheduled tasks for the selected date will be visible in main screen which can be edited. When editing the location, range, time, alert time and task description will be able to changed.

#### Cancel a scheduled task

When a scheduled task is deleted, all the record related to it will be deleted from the database.

#### Check for a predicted schedule

When user select the option of predicted schedule, the Android application will send a request including user email address (as the unique user ID) and date to the web server. Then the web server will check the past tasks of user in the similar days in the remote database. Then the server will process the data and send back a predicted schedule to the Android application. User can either agree with some or all the predicted tasks or cancel the schedule.

#### Check for a route

The application will provide an interface with a google map where the user can enter a location name and check for the route from the current position.

#### Enter a “My Place”

User can select his/her favorite places using a map interface and can enter a specific name. Then those places will be saved locally within the phone.

#### Edit/Delete a “My Place”

User can edit the name of a saved “My Place” or delete it completely.

#### View the reports

User can view the reports which describes how well the user has completed the scheduled tasks. User can select the time period for the report details such as daily, weekly or monthly. The reports will be generated by the web server and send back to the Android application.

### System functions

#### Notify a task

If the time scheduled for a specific task arrives or if the user arrives within the range of the location of the scheduled task, then a notification will be appeared in the phone showing the task description. Two options will appear in the notification as “Complete” and “Forget”. If the user select the “Complete” option, then the task will be marked as a completed one. Or else if the user select the other option then the task will be marked as a incomplete one.

#### Notify location tasks

When there are multiple tasks to completed within a certain location and when the user arrives at that location, those tasks will be notified in the phone. When user leaves the location, those tasks will be notified again asking whether they are completed or not.

#### Send daily data

At the end of each day, the Android application will send the details about the completed and incomplete tasks to the web server.

#### Analyze the daily data

Once the data of the daily tasks are sent by the Android application to the web server, the web server will add the data to the remote database. Then those data will be compared with the past data stored in the database and the predicted schedules of the user will be updated in the remote database.

#### Send reports

At the end of each day, week and month, the web server will process the data of task of each user. Then a report mentioning the completion percentages of tasks, mostly visited places etc. will be emailed to the users.

## Usability

[This section includes all those requirements that affect usability. For example,

* specify the required training time for a normal users and a power user to become productive at particular operations
* specify measurable task times for typical tasks or base the new system’s usability requirements on other systems that the users know and like
* specify requirement to conform to common usability standards, such as IBM’s CUA standards Microsoft’s GUI standards]

### Accessibility

System should support English language only. All the system operations and instructions should be in English language only. User should only be able to enter the details of tasks in English language only. Therefore, the users with basic English knowledge will be able to use the application easily.

### Graphical User Interface

The Android application should provide simple and attractive graphical user interfaces. The buttons and icons used should clearly emphasize the functions related to them. The navigation between different views should be simple.

### Ease of Use

When the Android application connects with the web server to do some processing of data, it should not take much time more than some seconds. Therefore, the calculations done within the web server need to be optimized.

## Reliability

### Availability

The Android application would be 100% available. Because, no huge processing done within the application. At any time, the user would be able to use the application to schedule a task. But the web server would be 95% available. Since the web server is stored in a remote server, the server failures will occur. Further, at the end of every day the server does a large amount of processing of daily gathered data from different users. Therefore, the server would not respond to application request within this period. These reasons contribute to the 5% unavailability of the server.

### Mean Time Between Failures

The server failures occur due to the issues of remote host at 1 per 1000 days.

### Mean Time to Repair

According to the service provider, the server will be fixed within a hour.

### Accuracy

The time-based tasks will 100% accurately fire to notify the user. But since the location-based tasks use the GPS service of the phone which is not accurate in some situations, the location-based tasks will not fire 100% accurately. By considering GPS accuracy, those tasks will provide a 90% accuracy. The accuracy of the predicted schedules generated by the web server will be varying. The web server only considers the past data to come to a conclusion. Therefore, a 100% accuracy of the prediction would not be expected.

### Maximum Bugs or Defect Rate

The maximum bug rate will be 1 bug/KLOC.

### Bugs or Defect Rate

* Minor

Minor bug will be the inability to fire a time-based task at exact time. Inability to send reports due to server failure will also be a bug. The server failure rate is equal to the defect rate at this moment.

* Significant

Significant bug will be inability to fire a location-based task within the relevant location. The GPS unavailability will cause this issue.

* Critical

The complete loss of data in the remote database will be a critical bug. Because, then the web server will not be able to accurately predict the schedule. Defect rate will be very less because the remote database will be backed up.

## Performance and Security

[The system’s performance characteristics are outlined in this section. Include specific response times. Where applicable, reference related Use Cases by name.

* Response time for a transaction (average, maximum)
* Throughput, for example, transactions per second
* Capacity, for example, the number of customers or transactions the system can accommodate
* Degradation modes (what is the acceptable mode of operation when the system has been degraded in some manner)
* Resource utilization, such as memory, disk, communications, and so forth.

### Response time for a transaction

Maximum response time for user registration would be 5 seconds. Because the Android application must “Sign in” with google and then crate a user account in the remote web server. The average time would be 3 seconds.

Maximum response time for scheduling a time-based or location-based task would be 500 milliseconds. Average time would be 300 milliseconds. Only time taking procedure is the saving of data to the local database.

Maximum response time to check the predicted schedules would be 5 seconds. Average time would be 4 seconds. Because the server must connect with the remote web server and do some processing of data before getting a response.

### Throughput

### Capacity

The remote database would be extendable. Therefore, as much as users would be able to register to the system.

### Degradation modes

When the web server does a large amount of processing and analyzing of daily gathered data from users, it will not respond to the requests of the Android application. In this situation, the application will only unable to get predicted schedule from the web server. All the other functionalities of the Android application will function as before.

### Resource utilization

## Supportability

[This section indicates any requirements that will enhance the supportability or maintainability of the system being built, including coding standards, naming conventions, class libraries, maintenance access, and maintenance utilities.]

### Coding standards

### Naming conventions

### Class libraries

### Maintenance access

### Maintenance utilities

## Design Constraints

[This section indicates any design constraints on the system being built. Design constraints represent design decisions that have been mandated and must be adhered to. Examples include software languages, software process requirements, prescribed use of developmental tools, architectural and design constraints, purchased components, class libraries, and so on.]

- Standards Compliance

- Hardware Limitations

### Software languages

The Android application will only support English language. All the operations and instructions will be provided in English language

### Software process requirements

### Development tools

Android application will be developed using “Android Studio IDE”. The web server will be developed using “PHP Storm IDE”. “Xampp” will be used to host the web server locally.

### Architectural and design constraints

### Purchased components

### Class libraries

“Google Services” libraries will be used for sign in and map services. “Volley” library will be used for server request handling in the Android application.

## On-line User Documentation and Help System Requirements

[Describes the requirements, if any, for o-line user documentation, help systems, help about notices, and so forth.]

## Purchased Components

[This section describes any purchased components to be used with the system, any applicable licensing or usage restrictions, and any associated compatibility and interoperability or interface standards.]

## Interfaces

[This section defines the interfaces that must be supported by the application. It should contain adequate specificity, protocols, ports and logical addresses, and the like, so that the software can be developed and verified against the interface requirements.]

**You do not have to include the screen shots of user interfaces itself ( as at this point those things are not implemented yet).**

**You may include (not all):**

**1. Explain the things that should be display in / consider for the interfaces**

**2. Describe**

**- For user interfaces : functionalities and the required menu items/ panels/ text boxes/ option buttons/ drop down lists that should be in the interfaces (eg. login page/ data entry page/ view pages/ analysis pages)**

**- For software interfaces: required interfaces to connect with the server, interfaces for access web services/ plugins**

**- For hardware interfaces: required client side pre-requisites( Disk space/ RAM/ Processor)**

**- For communication interfaces: eg. Asynchronous HTTP protocol requests over internet/ FTP file transfer**

**3. A draft diagram (block diagram) showing some main interfaces required by the user**

### User Interfaces

* “Sign in” view

When the user installs the Android application and runs it for the first time, then this view should be visible. There should be only one button indicating “Sign in with google”.

* Main view

After signed for the system the Android application should direct to this view. It should have a menu bar with a toggle button to view a “Side navigation panel”. In the top a label should be visible showing the selected date. Below the label, a “Calendar view” should be visible in which the date can be selected. Below the calendar view, a “list view” showing the list of scheduled tasks for the selected date should be visible. On the bottom and as a floating panel, a group of buttons should be visible to create a new task, view predicted list and view reports.

* Side navigation panel

The log of the application should be visible on the top. Below that the email address of the logged user should be visible. By touching the email address, the user should be direct to sign in with google. Below that, an option to direct to “My Places” view should be visible. Then options to direct to “Repeating tasks” and “Route finder” should be visible. In the bottom of the panel, an option to exit the app should be visible.

* “My Places” view

At the top of the view, a text input box to enter the name of the “My Place” should be visible. Below that, an option to select the location of “My Place” should be visible. Below that a button indicating “save” should be visible. The button should save the newly added “My Place”. Below the button, a list view including all the added “My places” should be visible. The list items within the list should have an edit button and a delete button.

* New task view

At the top, two buttons to save or cancel the task should be visible. Below that, a check box to schedule the task for every week should be visible. Then, a text input box to enter the description of the task should be visible. A panel with two check boxes to select the time-based task or a location-based task should be visible below that. Then an option to select the date of the task will be visible. By clicking that option a floating window should appear with a “calendar view” to pick the date. Then an option to select the time which should direct to a floating window with a clock to pick the time should be visible. Below that, an option to select location and an input box to enter the range of the location in meters should be visible.

* “Route Finder” view

At the top, an input box with autocomplete will be visible to enter the name of the location to travel should be visible. Below that, a map to view the route should be visible.

* Reports view

At the top, 3 options should be visible to select “daily”, “weekly” or “monthly”. Below that view with progress bars showing the completion of tasks should be visible.

### Hardware Interfaces

[This section defines any hardware interfaces that are to be supported by the software, including logical structure, physical addresses, expected behavior, and so on.]

### Software Interfaces

[This section describes software interfaces to other components of the software system. These may be purchased components, components reused from another application or components being developed for subsystems outside of the scope of this SRS but with which this software application must interact.]

The Android application reads and writes data to the web server. The web server reads and writes data to the remote database. These are the two main software interfaces.

### Communications Interfaces

[Describe any communications interfaces to other systems or devices such as local area networks, remote serial devices, and so forth.]

The Android application communicates with the web server by sending HTTP requests. But both the web server and the remote database will be hosted in the same remote host. Therefore, the communication between the web server and the remote database will be handled by the underlying hardware.

## Database Requirements

[Defines any database requirements for the system.]

Lot of daily gathered data will be stored in the remote database. Therefore, the capacity of the database need to be high. To generate predicted schedules a lot of data will be accessed at a time. So, the data should be stored in a way that they are easily accessible. Suitable integrity constraints should be applied to secure the data. Use of a NoSQL database in the remote database will ensure the scalability, adaptability and manageability of the database.

## Licensing, Legal, Copyright, and Other Notices

[Defines any licensing enforcement requirements or other usage restriction requirements that are to be exhibited by the software.]

[This section describes any necessary legal disclaimers, warranties, copyright notices, patent notices, wordmark, trademark, or logo compliance issues for the software.]

## Applicable Standards

[This section describes by reference any applicable standard and the specific sections of any such standards which apply to the system being described. For example, this could include legal, quality and regulatory standards, industry standards for usability, interoperability, internationalization, operating system compliance, and so forth.]

# Supporting Information

[The supporting information makes the SRS easier to use. It includes:

* Table of contents
* Index
* Appendices

These may include use-case storyboards or user-interface prototypes. When appendices are included, the SRS should explicitly state whether or not the appendices are to be considered part of the requirements.]

**Refer any data/ information in a standard format (eg. IEEE referencing style)**

**For different algorithms/ techniques/ theories you can refer text books.**

**For tools you can refer web pages.**

**For similar work you can refer research paper articles that describe the work.**

**You may include white paper articles for the description of technologies; web URL for the tool references. When you refer such a web page, you have to indicate the (Accessed on <<date>>)**