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.

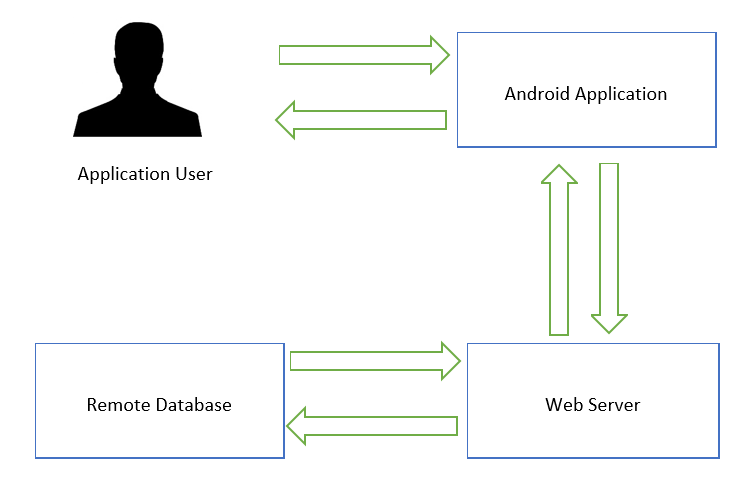
The final section gives the details on supporting information about SRS. It includes the table of contents, appendix and the indices.

# 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 next task to be done before some calculated 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.

## Requirements subsets

# Specific Requirements j

[This section of the SRS contains all software requirements to a level of detail sufficient to enable designers to design a system to satisfy those requirements, and testers to test that the system satisfies those requirements. When using use-case modeling, these requirements are captured in the Use Cases and the applicable supplementary specifications. If use-case modeling is not used, the outline for supplementary specifications may be inserted directly into this section, as shown below.]

## Functionality

[This section describes the functional requirements of the system for those requirements that are expressed in the natural language style. For many applications, this may constitute the bulk of the SRS package and thought should be given to the organization of this section. This section is typically organized by feature, but alternative organization methods may also be appropriate; for example, organization by user or organization by subsystem. Functional requirements may include feature sets, capabilities, and security.

Where application development tools, such as requirements tools, modeling tools, and the like, are employed to capture the functionality, this section of the document would refer to the availability of that data, indicating the location and name of the tool used to capture the data.]

Do not use usecase diagram; Describe each of the function of the system/ user activity in detail. i.e. elaborate on each functionality of the system.

### <Functional Requirement One>

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

## 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]

### <Usability Requirement One>

[The requirement description goes here.]

## Reliability

[Requirements for reliability of the system should be specified here. Some suggestions follow:

* Availability—specify the percentage of time available ( xx.xx%), hours of use, maintenance access, degraded mode operations, and so on.
* Mean Time Between Failures (MTBF) — this is usually specified in hours, but it could also be specified in terms of days, months or years.
* Mean Time To Repair (MTTR)—how long is the system allowed to be out of operation after it has failed?
* Accuracy—specifies precision (resolution) and accuracy (by some known standard) that is required in the system’s output.
* Maximum Bugs or Defect Rate—usually expressed in terms of bugs per thousand lines of code (bugs/KLOC) or bugs per function-point( bugs/function-point).
* Bugs or Defect Rate—categorized in terms of minor, significant, and critical bugs: the requirement(s) must define what is meant by a “critical” bug; for example, complete loss of data or a complete inability to use certain parts of the system’s functionality.]

### <Reliability Requirement One>

[The requirement description.]

## 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.

### <Performance Requirement One>

[The requirement description goes here.]

## 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.]

### <Supportability Requirement One>

[The requirement description goes here.]

## 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

### <Design Constraint One>

[The requirement description goes here.]

## 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

[Describe the user interfaces that are to be implemented by the software.]

### 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.]

### Communications Interfaces

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

## Database Requirements

[Defines any database requirements for the system.]

## 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>>)**