Software Requirements Specification

for

Crew Connect

Version 1.2 approved

Prepared by Shanice Rapp

WASUP Airlines

<date created>

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Shanice Rapp	02/14/23	Added information in 1.1-1.5	1.1
Shanice Rapp	03/08/23	Added information in 2.1-2.7	1.2
Shanice Rapp	03/09/23	Created content for Appendix B	1.3

1. Introduction

1.1 Purpose

The product presented is the Crew Connect application which is being developed for WASUP Airlines only. After several revisions, the application will be used with version XX which is the latest software version. This document will go over the application's purpose, the product scope, functional requirements, limitations, external interface requirements, system features, nonfunctional requirements, an overall description of the product, and comes with a glossary and reading suggestions for the intended audience. Throughout the document, the whole system is being described.

1.2 Document Conventions

All text in the application is written in black color, Arial font. However, if there is an emergency or important notification, the text will appear in **bold red**. *Italic* text in the calendar stands for the *vacation* time. The transportation information will pop up with a taxi icon next to the information.

1.3 Intended Audience and Reading Suggestions

User or Reader	Suggested Areas to Read
Stakeholders	2
IT, Developers, System workers, PM	3,4,5
End-users	1, 2

1.4 Product Scope

The Crew Connect application is being developed with the idea that the crew has a single place in which all communication can take place. The most helpful features of this application will be the schedule, the weather, routes, hotel information, pick up information, and a vacation request feature. Faster and more effective communication are the main objectives of this new application. If the communication within and to the crew is being improved, emergencies can be dealt with faster and flight delays due to miscommunications can be prevented. All this helps to reach WASUP Airlines' main goal of maximizing customer satisfaction.

1.5 References

Go to www.wasupairlines.com/crewconnect for more information and an introduction to the application. There you will also find references and versions of the application and other documents related to the Crew Connect app.

2. Overall Description

2.1 Product Perspective

The Crew Connect application is a new self-contained product with the intended use being only for the Crew members inside of WASUP airlines. Prior to Crew Connect, there were several misunderstandings due to simultaneous communication on several platforms and important information was often read too late due to the lack of an efficient system for delivering this information. To prevent these issues, the purpose of developing the application is to bring several different systems into one application. For efficient and effective usage, the application is dependent on timely and correct input of data about the flight schedules, vacation time, and needs a third party cooperation that supplies the application with the current weather information of the crews current location as well its destination location.
Figure 2.1 in Appendix B shows the main functions and how they will be present on the navigation page as well as a first idea of the interface in WASUP airlines' colors.

2.2 Product Functions

The most important functions of the Crew Connect application are listed here:

- Chat amongst the crew (a new channel is opened for each flight with the corresponding crew)
- Calendar (contains scheduled flights and vacation times)
- Transport and Hotel Information
- Weather alerts

For further understanding refer to the WASUP Flow Diagram in Appendix B.

2.3 User Classes and Characteristics

The end-user of this product is the whole crew. Regardless of the crew members' position, both pilots and flight attendants have the same product functions. There are no admin roles, it is an only user-tool. The information will be inputted by the backend team.

2.4 Operating Environment

Crew Connect will operate on a Linux-based mobile operating system and will not function properly on macOS. Therefore, each crew member will receive an Android smartphone. Furthermore, the compatibility with weatherbug.com/api as the supplier for all weather data is assured. The data will be stored on SQL servers and Oracle. After several revisions, version 2.0 will be the running version on the crew members' smartphones.

2.5 Design and Implementation Constraints

There are several limitations that come with Crew Connect. First, Crew Connect is only to be used by the crew, not by other employees of WASUP airlines. Second, the application will only work on Android devices. Third, to access all information through Crew Connect, the device must be connected to the internet. The basic information will be stored on the phone and the schedule for a specific event can be downloaded to access it without internet connection. Since WASUP airlines is an American airline Crew Connect will be offered in an English and a Spanish version. The chat is to be solely used for work purposes, not to have private conversations. Therefore, the memory of 64GB for each crew member should be sufficient. To improve security, each crew member will receive their own smartphone with the app preinstalled and the application itself is not open for public use. Besides, the design of Crew Connect follows the lightskyblue and navy blue company colors and the font to use is Lato. Furthermore, the weather updates will be accessed through weatherbug.com/api and the database used for vacation request is provided by Oracle.
The development must fall into the $500,000.00 budget.

2.6 User Documentation

To access introduction videos and tutorials go to WASUP airlines' website and click on Crew Connect under the "For Flight Crew" button. You can also use this link to access it: www.wasupairlines.com/crewconnect. For further questions while using Crew Connect, click on the question mark symbol in the lower right corner and look for your answer there.

2.7 Assumptions and Dependencies

Crew Connect uses several servers to store the data. The PeopleSoft server (Oracle) is responsible for the vacation requests, the Travel server (SQL server) contains travel adjustments, and the flight and crew scheduling comes from the AIMS server (Oracle). Failing of one of the servers could lead to a disruption of the whole system and mistakes in scheduling and general flight delays. Furthermore, if the Weatherbug channel is down or if there are problems in transferring the data to Crew Connect, the application's weather page will not show data. Another dependency of Crew Connect comes with hotel and especially the shuttle service. The application can only inform the crew about the car that picks them up and the driver and arrival time if the data has been put in by the transfer party. As well as the application is dependent on third parties that supply data, Crew Connect is also dependent on the proper usage of its crew members.
It is assumed that accurate budgeting and planning has been conducted to stay within the $500,000.00 budget.

3. External Interface Requirements

3.1 User Interfaces

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

3.2 Hardware Interfaces

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>

3.3 Software Interfaces

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>

3.4 Communications Interfaces

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>

4. System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

4.1 System Feature 1

<Don't really say "System Feature 1." State the feature name in just a few words.>

4.1.1 Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

4.1.2 Stimulus/Response Sequences

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

4.1.3 Functional Requirements

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use "TBD" as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-1:
REQ-2:

4.2 System Feature 2 (and so on)

5. Other Nonfunctional Requirements

5.1 Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

5.2 Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product's design or use. Define any safety certifications that must be satisfied.>

5.3 Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

5.4 Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

5.5 Business Rules

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>

6. Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>
Leg

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>