SOFTWARE REQUIREMENTS SPECIFICATION for

Android application "Fluber"

Version 1.0 approved

Prepared by
Babich Kirill
Beryukhov Andrey
Klochkiv Lev
Repina Anastasia

September 20, 2018

Contents

1 Introduction

1.1 Purpose

The purpose of this document is to present a detailed description of the “Fluber” application for Android. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli. This document is intended for the developers of the application and end-users.

1.2 Scope of the product

People who buy flowers do not have an opportunity to compare the prices for the bouquets in the nearest shops of the district when they are already in the shop, as this data is not agregated and availiable. To let users choose the best option and buy the best bouquets for the cheapest price correct their timetable we give them an opportunity to make an order inside the application. Flower suppliers will be able to add their shops to the application base, so both: buyer and seller will be in positive territory. It should increase flower shops revenue and help people to make the choise.

The application contains a relational database containing a list of Shops, Users, Flowers, Orders.

1.3 Definitions, acronyms and abbreviations

Term	**Definition**
API	A set of subroutine definitions, protocols, and tools for building software and applications.
Database	Collection of all the information monitored by this system.
Field	A cell within a form.
Software Requirements Specification	A document that completely describes all of the functions of a proposed system and the constraints under which it must operate. For example, this document.
Buyer	A person, who buy flowers and make an order in the application.
Seller	A person, who sell flowers and get orders in the application.
User	See Buyer and Seller

1.4 References

IEEE. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications. IEEE Computer Society, 1998.

1.5 Overview of the document

The next chapter, the Overall Description section, of this document gives an overview of the functionality of the product. It describes the informal requirements and is used to establish a context for the technical requirements specification in the next chapter. The third chapter, Functional Requirements section, of this document is written primarily for the developers and describes in technical terms the details of the functionality of the product. The fourth chapter, Non-functional Requirements section, of this document is about specify criteria that can be used to judge the operation of a system, rather than specific behaviors. The last chapter is Other Requirements, which contains requirements not covered elsewhere in the SRS. All sections of the document describe the same software product in its entirety, but are intended for different audiences and thus use different language.

2 Overall Description

This section will give an overview of the whole system. The system will be explained in its context to show how the system interacts with other systems and introduce the basic functionality of it. It will also describe what type of stakeholders that will use the system and what functionality is available for each type. At last, the constraints and assumptions for the system will be presented.

2.1 Product Perspective

This system is an Android application, which will be used to make and fullfill orders for flowers, view information about them. The mobile application will need an access to the Internet in order to use Firebase API to work with database and Google Maps API, which allows to find out all shops locations. Diagram below shows, how different modules interacts, see Figure 1.

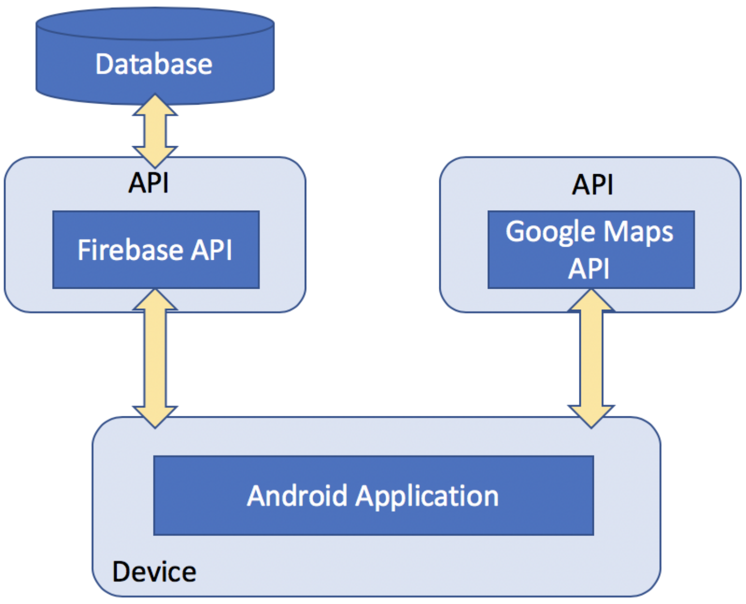


Figure 1. Blocks diagram

Since this is a data-centric product, it will need somewhere to store the data. For that, a database will be used. A mobile application will communicate with the database via Firebase API. In database it will send, get and modify data.

The mobile application has no restrictions about the resource allocation.

2.2 Product Functions

- Flowers sale (for gardener - sellers)
- Flowers purchase (for clients - buyers)
- Filtration the results (to find the best variant)
- Geotracking the sellers around and showing on Google map
- Tracking the orders using notifications
- Ability to pay for order inside application

All changes will be saved and stored in database.

2.3 User Classes and Characteristics

There are two types of users that interact with the system: buyers and sellers. They have different interaction plans with the application. Users main goal is to post an order and get the flowers, whereas sellers after adding their shop to the shops database can fullfill the customer orders.

2.4 Constraints

The mobile application is constrained by the Internet, because of filling database.

2.5 User Documentation

No user documentation is planned.

2.6 Assumptions and Dependencies

One assumption about the product is that it will always be used on mobile phones that have enough performance. If the phone does not have enough hardware resources available for the application, for example, the users might have allocated them with other applications; there may be scenarios where the application does not work as intended or even at all.

3 Specific requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

A first-time user of the mobile application should see the Login page when he/she opens the application. There he/she is able to sign up into existing account, see Figure 1, or go to Register page, see Figure 2.

After registration or login or if user has already been logged in after launch of the application the Main page with the list of abailable bouquets near the user's location will be visible, see Figure 3. Bouquet images are scrollable together with the changing of the Seller's name and his location.

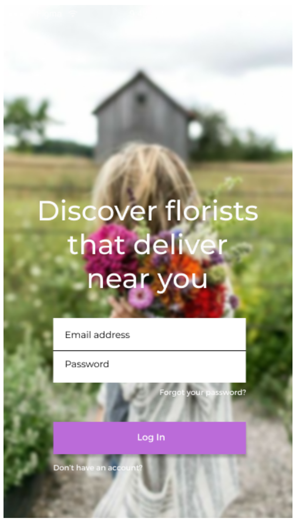


Figure1. Login page

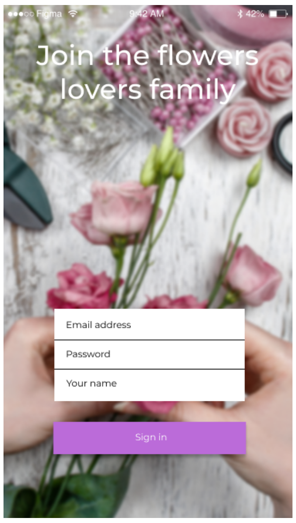


Figure 2. Register page

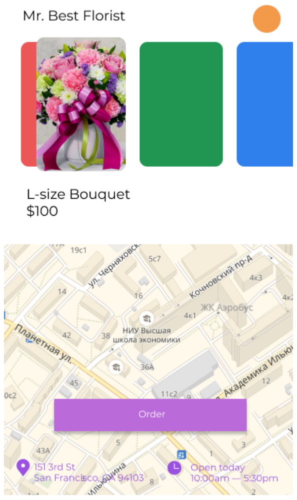


Figure 3. Main page with the list of avaliable bouquets near the user's location

Navigation drawer is an implementation of a menu navigation in the application, which appears from the left side of the screen, see Figure 4. There is a name of user together with the list of other options: go to Main screen or to Seller's page, to Chat or to Info page.

Seller page, see Figure 5, is a grid of bouquets suggested by the Seller to purchase together with the info about booking requests and cancelations, and the button for new bouquets addition.

After pushing it sellers sees the screen as in Figure 6. Here he can change the photo of good, the description together with the size, the price and available amount.

After pushing the button Order buy Buyer on main page, see Figure 3, he sees the product page. Here is a detailed information about bouquet and a button for ordering, see Figure 7.

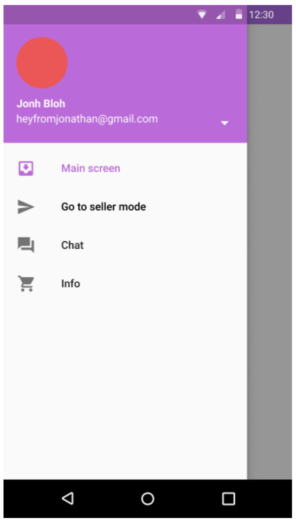


Figure 4. Menu page implementation in Navigation drawer

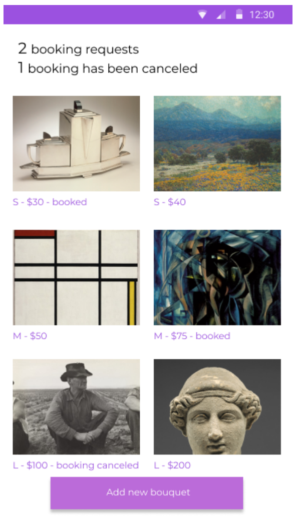
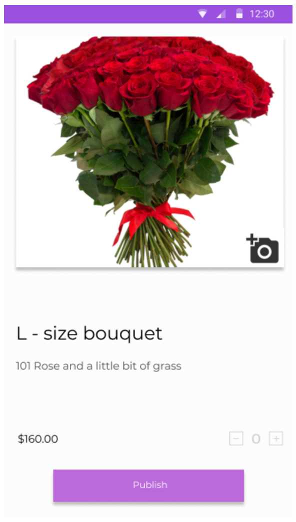

2 booking requests
1 booking has been canceled

Figure 5. Seller page with bookings information

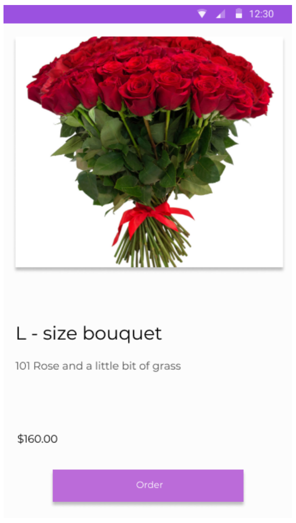

L - size bouquet

101 Rose and a little bit of grass

Figure 6. Product page visible by Seller

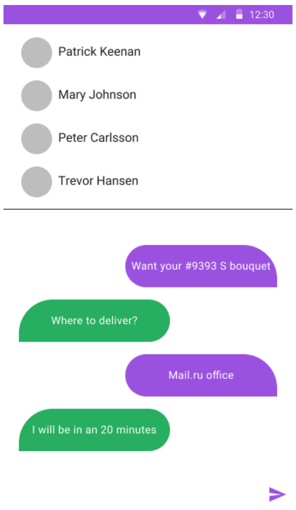
Chat is available for Buyers and Sellers to communicate about purchase and delivery details, see Figure 8.

Info page, see Figure 9, contains the information aboiut the application.


L - size bouquet

101 Rose and a little bit of grass

Figure 7. Product page visible by Buyer



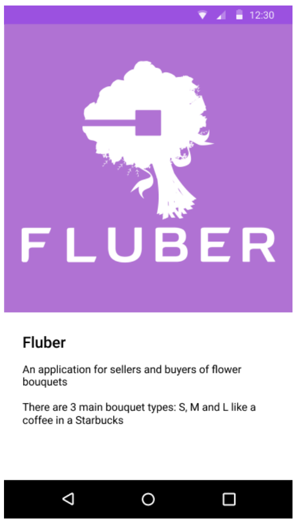
Want your #9393 S bouquet

Where to deliver?

Mail.ru office

I will be in an 20 minutes

Figure 8. Chat between buyers and sellers page



Fluber

An application for sellers and buyers of flower bouquets

There are 3 main bouquet types: S, M and L like a coffee in a Starbucks

Figure 9. Info page

3.1.2 Hardware Interfaces

Since the mobile application doesn't have any designated hardware, it does not have any direct hardware interfaces. The physical GPS is managed by the GPS application in the mobile phone and the hardware connection to the database server is managed by the underlying operating system on the mobile phone.

3.1.3 Software Interfaces

The mobile application communicates with the GPS application in order to get geographical information about where the user is located and the visual representation of it, with Firebase system to sync orders, bookings, chats and user info.

3.1.4 Communications Interfaces

The communication between the different parts of the system is important since they depend on each other. However, in what way the communication is achieved is not important for the system and is therefore handled by the underlying operating systems for both the mobile application and the backend service.

3.2 Functional requirements

This section includes the requirements that specify all the fundamental actions of the software system.

3.2.1 Download mobile application

ID: FR1
TITLE: Download mobile application
DESC: A user should be able to download the mobile application through either an application store or similar service on the mobile phone. The application should be free to download.
RAT: In order for a user to download the mobile application.
DEP: None

3.2.2 Download and notify users of new releases

ID: FR2
TITLE: Download and notify users of new releases
DESC: When a new/updated version or release of the software is released, the user should check for these manually. The download of the new release should be done through the mobile phone in the same way as downloading the mobile application.

RAT: In order for a user to download a new/updated release.
DEP: FR1

3.2.3 User sign in with credentials

ID: FR3
TITLE: User enters their e-mail and password
DESC: Given that a the application is installed, the user should be able to sign in providing their e-mail and password, which were provided during sign up and stored safely in the backend. If the e-mail and password pair does not exist in the database, a message should be shown suggesting the user to check correctness of provided credentials.
RAT: In order for a user to sign in to the mobile application.
DEP: FR2

3.2.4 User sign up

ID: FR4
TITLE: User signs up to the mobile application
DESC: Given that a the application is installed, the user should be able to sign up to the application providing e-mail address, password and user's full name. The e-mail address must not be already stored in the backend. If so application should show to the user a message suggesting that an account with this address already exists. The information entered by user should be checked for validity according to the database rules, then saved to the database. If an account was successfully created, a message is shown to the user.
RAT: Creating an account
DEP: FR2

3.2.5 Location acquiring

ID: FR5
TITLE: The application acquires user's location
DESC: Given that the user signed in successfully to the mobile application, the application should be able to acquire user's current location using API provided by the operation system. If needed, the application should ask the user permission to access location data. If the permission is denied, the application should show the user a message explaining, why the application requires location data to work correctly. If the user's location is still unavailable, the application should ask the user to provide their location selecting a point on the map or by passing a query congaing a street address to the map API to determine geographical coordinates. Geographical coordinates received from the operation system or the map, should be decoded to a street address and shown to the user. The app should show resulting user location on the map and ask user to confirm if the location is acquired correctly.

RAT: Showing offers available in user's neighborhood
DEP: FR3

3.2.6 Choose filters

ID: FR6
TITLE: Download offers to show on the main page
DESC: Given that user signed in successfully to the mobile application, the filters for searching offers should be changeable. The user should be able to choose the range of search, the size of the bouquet and the pricing category. After the filters are changed, new values should be saved in the application. By default no filters are applied.
RAT: Search offers.
DEP: FR3

3.2.7 Download the offers

ID: FR7
TITLE: Download offers to show on the main page
DESC: Given that user signed in successfully to the mobile application and user's current location is available, the offers should be downloaded from the backend server. The application forms a request to the backend that contains user's geographical coordinates and chosen filters. The server should response with the entities describing offers available for buying in this neighborhood.
RAT: Showing offers.
DEP: FR5

3.2.8 Display offers on main page

ID: FR8
TITLE: Display offers on the main page
DESC: Given that the offers have been downloaded from server successfully, the offers should be shown on the main page. For each offer the application shows a photo, size and price. This page should be updated each time it is opened if Internet connection is available or by user request (scroll down).
RAT: Display offers view.
DEP: FR7

3.2.9 App menu

ID: FR9
TITLE: App menu
DESC: The user should be able to select different pages: Main screen, Seller mode, Chat, About.

RAT: In order for user to change pages.
DEP: FR3

3.2.10 Offer page

ID: FR10
TITLE: Offer page
DESC:On user clicked on an offer, a detailed information should be shown about the chosen offer. The application requests the detailed information of the offer from the backend server. This information includes photos, detailed description, size and price. User should be able to make a purchase. On seller mode page, if the offer is made from the current account, this information should be alterable, moreover seller can set the number of available for sell bouquets.
RAT: In order to show and edit offer details.
DEP: FR8

3.2.11 Seller mode page

ID: FR11
TITLE: Seller mode page
DESC: The application should request from the backend server information relevant to the user as seller. This includes number of offers sold, number of offers booked, number of offers canceled and offers made by this account. For each offer the application shows a photo, size, price and number of products available. User should be able to add a new offer. This opens an empty offer page.
RAT: In order to show seller's offers. This page should be updated each time it is opened if Internet connection is available or by user request (scroll down).
DEP: FR9

3.2.12 Making a purchase

ID: FR12
TITLE: Making a purchase
DESC: Magic happens here
DEP: FR10

3.2.13 Chat

ID: FR14
TITLE: Chat
DESC: After user makes a purchase, sellers contact is added to the chat page. When seller or user clicks on a contact, previous messages are shown. User and seller should be able to send and receive messages.

RAT: In order to provide means of communication between user and seller.
DEP: FR9

3.2.14 About

ID: FR13
TITLE: About
DESC: The user should be able to read information about app developers.
RAT: In order to user read information about developers.
DEP: FR9

3.3 Perfomance requirements

The requirements in this section provide a detailed specification of the user interaction with the software and measurements placed on the system performance.

3.3.1 Usage of sign in feature

ID: QR1
TITLE: Usage of sign in feature
DESC: Signing in should be easy to understand.
RAT: In order for a user to make purchases easily
DEP: None

3.3.2 Prominent menu feature

ID: QR2
TITLE: Prominent menu feature
DESC: Menu should be prominent and easy to find for user.
RAT: In order to for a user to use the app easily
DEP: None

3.3.3 Usage of the offer page

ID: QR3
TITLE: Usage of the offer page
DESC: The different search options should be evident, simple and easy to understand.
RAT: In order to for a user to perform a search easily.
DEP: None

3.3.4 Response time

ID: QR4
TAG: ResponseTime
GIST: The fastness of the offers page
SCALE: The response time of offers
METER: Measurements obtained from 1000 loadings during testing.
MUST: No more than 2 seconds 100% of the time.
WISH: No more than 1 second 100% of the time.

3.3.5 SystemDependability

ID: QR5
TAG: SystemDependability
GIST: The fault tolerance of the system.
SCALE: If the system loses the connection to the Internet or to the GPS device or the system gets some strange input, the user should be informed.
METER: Measurements obtained from 1000 hours of usage during testing.
MUST: 100% of the time.

4 Prioritization and Release Plan

The main task is to create a working application where buyer can make an order and buy flowers, find information about the nearest flower shops, and a seller can add shop to the shops DB and get the orders inside the application. When the minimal requirements will be done, the additional features will be added via updates, but only if the product will find its market and its users.

Version 1 requirements: make orders for buyer, fullfill customer orders for seller, find shops on the map, Firebase API using, Google Maps API using, Database with described in the preceding paragraphs operations.

4.1 Choice of prioritization method

The initial list was created at the stage of presenting the idea. After analyzing the already published applications and users needs the list of all requirements to both versions was created.

Requirements	**Rating**
Firebase API connections	10
Database	10
Orders making	10
Orders fullfilling	10
Shops map	10
User-friendly interface	9
Google Maps API connections	5
Search for shop	5
Settings	3

All values are normalized.

The main goal is to create a stable working application with the requirements included in version 1, which were mentioned above. In updates additional features will be added.

4.2 Release Plan

The requirements were divided into two releases based on complexity/necessity. First release is an application with minimum extra functions, that do its job. The second release includes additional functions for users. However, these requirements are not vital for a functional application. They are more suited to act as additional features that can contribute to making the software product more attractive.

5 Other Requirements

5.1 Appendix A: Analysis Models

RDD for database Entities

- Flower = [flower_id:Text, name:Text, full_name:Text, color:Text, country:Text, price: Integer, available:Bool]
- Bouquet = [bouquet_id:Text, price:Text, flowers_count:Integer, wrapping:Text, available:Bool]
- Shop = [shop_id:Text, name:Text, location:Text, about:Text, rating:Text, opens:Date, closes:Date]
- User = [email:Text, name:Text, password:Text, date_registered: Date, isSeller:Bool]
- Order = [order_id:Text, created_at:Date, discount:Integer, price:Integer, delivery:Integer, address:Text, shop_id:Text]
- Chat = [chat_id, buyer_id:Text, seller_id:Text]
- Message = [message_id, text:Text, created_at:Date]

Relationships

- bouquets = [flower_id:Text, bouquet_id:Text]
- flowers_at_shop = [flower_id:Text, shop_id:Text]
- bouquets_at_shop = [bouquet_id:Text, shop_id:Text]
- users_orders = [user_id:Text, order_id:Text]
- orders_details = [order_id:Text, bouquet_id:Text]
- users_chats = [user_id:Text, chat_id:Text]
- chats_messages = [chat_id:Text, message_id:Text]