

UNIVERSITY OF LATVIA
FACULTY OF COMPUTING

**PLANTO360 – PLANT SHARING, SELLING AND
BUYING APPLICATION**

SOFTWARE ENGINEERING PROJECT

Authors: **Jekaterina Aleksejeva**, jm10071

Margarita Danilenko, md20075

Andrievs Jakovels, aj21001

Sandra Jegina, sj08075

Laura Riekstina, lr12097

Ieva Slisane, is21001

Scientific advisor: Mg.sc.comp. Guntis V. Strazds

RIGA 2021

ABSTRACT

This project work describes a mobile application named Planto360 for sharing, selling or buying plants, as well as providing a platform for communication on related topics. The mobile applications allows users to connect based on their interest and needs. Project document consists of overall description, application requirements and application design description. The document is developed according to the ISO/IEC/IEEE 29148:2018 standard.

Keywords: application, plants, plant swapping, sharing.

ANOTĀCIJA

Šis projekta darbs apraksta mobilo lietotni, kuras nosaukums ir Planto360, kas ir paredzēta augu apmaiņai, pārdošanai un pirkšanai, kā arī nodrošinot platformu komunikācijai par saistītām tēmām. Mobilā lietotne ļauj lietotājiem savienoties atbilstoši viņu interesei un vajadzībām. Projekta darbs sastāv no vispārēja apraksta, lietotnes prasībām un lietotnes projektējuma apraksta. Dokuments ir izstrādāts saskaņā ar ISO/IEC/IEEE 29148:2018 standartu.

Atslēgas vārdi: lietotne, augi, augu apmaiņa, dalīšanās.

TABLE OF CONTENTS

List of figures	6
Glossary	7
Introduction	8
1. Overall description	9
1.1. Current Situation	9
1.2. Client	9
1.3. Product vision	9
1.4. Business requirements	10
1.5. System users	10
1.6. General restrictions	11
1.7. Assumptions and dependencies	11
2. Software requirements specification	12
2.1. Conceptual DB Model	12
2.2. Functional Requirements	13
2.2.1. General description of functions	13
2.2.2. Functional breakdown by modules	15
2.2.3. Module - User management	17
2.2.4. Module - Plant request	20
2.2.5. Module - Plant offer	23
2.2.6. Module - Plant catalogue	26
2.2.7. Module - User feedback	28
2.3. Non Functional requirements	31
3. Software design description	32
3.1. Database design	32
3.1.1. Entity – Users	33
3.1.2. Entity - Plant request	34
3.1.3. Entity - Plant offer, Plant, Species	35
3.1.4. Entity - User feedback	36
3.2. Partial Functional Design	37
3.2.1. User interface	37
3.2.2. User interface (authorized users)	38

3.2.3. User profile registration	39
3.2.4. Plant request creation	40
3.2.5. Plant request status change.....	41
3.2.6. Plant offer post	42
3.3. Partial Design of User Interface.....	43
3.3.1. General user interface.....	43
3.3.2. User interface wireframes	44
References and sources.....	49

LIST OF FIGURES

Number	Title	Page
1.1.	Planto360 system - data flow diagram (level 0)	10
2.1.	Conceptual Database Model (Chen Syntax)	12
2.2.	Division of Planto360 into modules	13
2.3.	Function and user group use according to module	14
2.4.	Data flow diagram (level 1)	16
2.5.	User management module - data flow diagram (level 2)	17
2.6.	Plant request module - data flow diagram (level 2)	20
2.7.	Plant offer module - data flow diagram (level 2)	23
2.8.	Plant catalogue module - data flow diagram (level 2)	26
2.9.	User feedback module - data flow diagram (level 2)	28
3.1.	Logical database model (Crowfoot syntax)	32
3.2.	Physical database model	33
3.3.	User interface (UML use case diagram)	37
3.4.	User interface (authorized users) (UML use case diagram)	38
3.5.	User profile registration (UML sequence diagram)	39
3.6.	Plant request creation (UML activity diagram)	40
3.7.	Plant request status change (UML state diagram)	41
3.8.	Plant offer post (UML timing diagram)	42
3.9.	General user interface	43
3.10.	Guest user - user interface	44
3.11.	Login screen wireframe	44
3.12.	User registration screen wireframe menu	45
3.13.	Menu screen wireframe	45
3.14.	Search screen wireframe	46
3.15.	Search results screen wireframe	46
3.16.	Plant offer screen wireframe	47
3.17.	User contact screen wireframe	47
3.18.	Create request screen wireframe	48
3.19.	Create offer screen wireframe	48

GLOSSARY

Acronym	Definition
CRUD	Create, read, update, delete- the four functions that are considered necessary to implement a persistent storage application

INTRODUCTION

Purpose

The purpose of this document is to describe the idea, requirements, functionality and design of an application that enables people to exchange with plants and businesses to sell plants and supplies.

Scope

The mobile application Planto360 allows people to easily search for plants by using different filters. Plant owners and shops will be able register in the system and create listings for available plants. The system will store information about each seller, people that have signed up for searching plants, information about different types of plants and their care. Users will be able to rate others based on their experience and quality of the plant.

Relations

This document is based on the standards specified in ISO/IEC/IEEE 29148:2018.

General Overview

This document consists of three main sections: “Overall description”, “Software requirements specification” and “Software design description”

1. OVERALL DESCRIPTION

1.1. Current Situation

When researching similar offers online, it appears that there is poor global offer of online applications for users that would be interested in sharing, swapping, selling or buying plants, that at the same time would include possibility to access plant care database, plant variety database, give access to communication forum, as well as give opportunity for commercial users to participate as well.

At the moment in the global market (on play.google.com, apps.apple.com) there can be found similar applications, such as Blossm or PlantSwapp, but still those applications are not covering all the features that Planto360 is providing. Blossm is providing possibility to swap, buy or get for free some plants; there is a forum to communicate with other plant owners to find a plant, identify plants; through this application it is possible to access database on plant care; map feature of users in area; scanning QR codes from participants of plant events or farmer markets; rating system. PlantSwapp is even more narrow and provides only the possibility to offer or take from somebody a plant that is no longer needed and chat with the owner.

At the moment in Latvian market there can't be found similar applications as Planto360. Only options to get a plant online is to buy them in an online shop or use an application for selling/buying personal items, including plants, such as Andele or ss.lv.

1.2. Client

The system is developed by the group of students as an assignment for the course DatZ2072: Software Engineering. The hypothetical client is a startup company who wants to provide services for gardening and plant-oriented enthusiasts and organizations.

1.3. Product vision

With people starting to live in more urban spaces and having less nature around there's a greater chance of getting anxiety and depression. It has been proven that having houseplants help with these conditions. We as a company want to make sure our product improves people's mental health in a very convenient and effective way. Green spaces with less effort and less

money spent. Considering there are no applications similar to this in the local market, it could be a way for people to get connected based on the same interests.

1.4. Business requirements

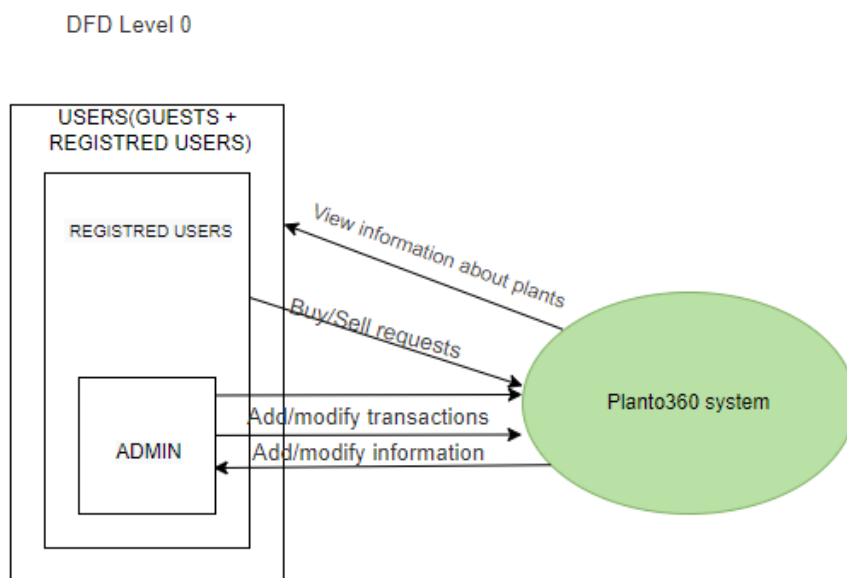
Planto360 will be a standalone application available on mobile devices with IOs and Android, also with different type touch screen devices with internet connection.

The application will provide opportunities for Plants Growers to sell online and reach big audency and for individuals to find and buy the needed item faster than physically attending many markets and gardens.

The application is meant to earn money by advertisement placement as well as gardening companies would have to pay a fee to be presented in the app. In addition users would be able to rent plants for personal use, e.g., photoshoots and events.

1.5. System users

We have three types of users. One is registered users that includes individuals who are passionate about different kinds of plants, and are willing to sell, buy, give away and/or exchange their plants with others and vendors who offer vast amounts of different greenery and provide it to their customers when demand for specific plants is higher than supply. Second is administrators with CRUD rights of the content and full access to user management. Third type is guest users who have limited access to the system, only to view information.



1.1. figure Planto360 system - data flow diagram (level 0)

1.6. General restrictions

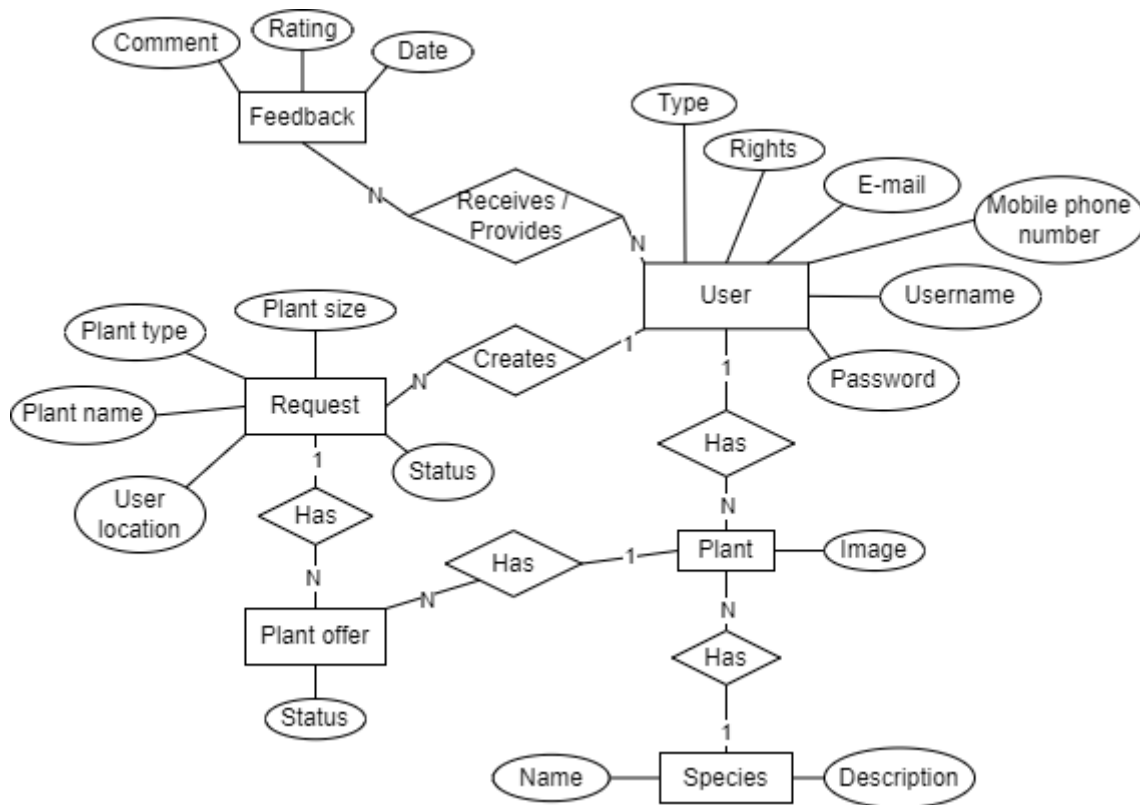
1. To add content and interact with other users within the application a user must be authorized;
2. Users have to be at least 16 years old to buy and sell plants;
3. Plants have to be local flora neutral (<https://www.vaad.gov.lv/en>).

1.7. Assumptions and dependencies

It is assumed that for the application to function there is needed internet connection. It is assumed that administrators will add the species to the catalogue and maintain the database. Registered users populate the plant requests and offers database.

2. SOFTWARE REQUIREMENTS SPECIFICATION

2.1. Conceptual DB Model



2.1. figure Conceptual Database Model (Chen Syntax)

- Entity “User” contains the following attributes- type (guest user, registered user (individuals and vendors) and administrator), rights, username, password, e-mail address and mobile phone number. E-mail and mobile phone number is used for password reset, notifications and announcements, username and password for access to application.
- Entity “Feedback” contains information about ratings and comments a registered user provides/receives to/from another registered user. Ratings reflect the quality of interaction between users on a scale of 1 to 5 where 1 means poor and 5- excellent. Users can add a comment to the ratings they give. To follow up the frequency of

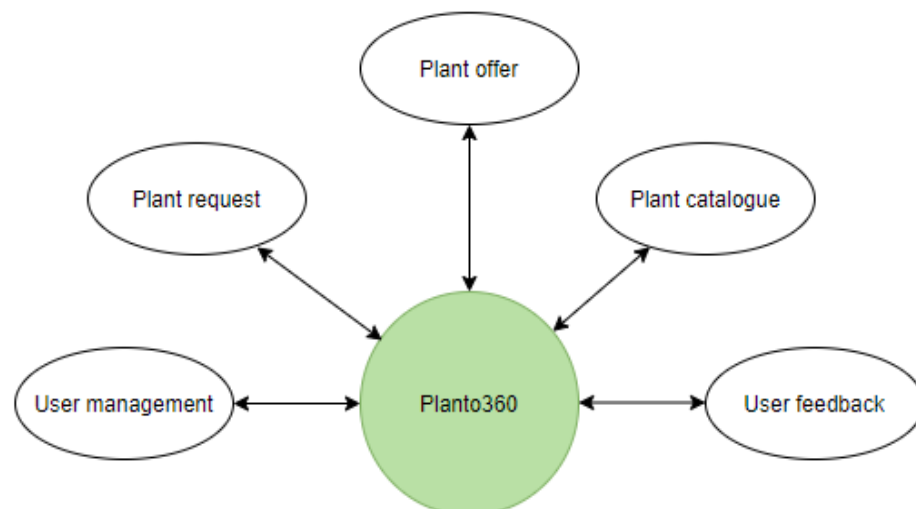
interactions each feedback has the date when the rating was provided. Entity “Feedback” has a “many to many” relationship with “User”.

- Entity “Plant” contains the plant's image. It has a “many to one” relationship with “User” entity as many users can have many plants to offer.
- Entity “Species” contains name and description of a particular plant. Entity has “one to many” relationship with “Plant” as a species can have many plants, but one plant cannot have many species.
- Entity “Request” is created by a user who is looking for a particular plant. It contains the following attributes- plant name, plant type (indoor/outdoor), plant size, location of the user who creates the query and status of the request (active/inactive). “Request” has “many to one” relationship with “User” as one user can create many requests.
- Entity “Plant offer” is created by another user which has the required plant to offer and has the following attribute- status (available/not available). Offer contains the link to one plant available with corresponding image and description which is sent to the request author privately. “Plant offer” has a “many to one” relationship with “Plant”.

2.2. Functional Requirements

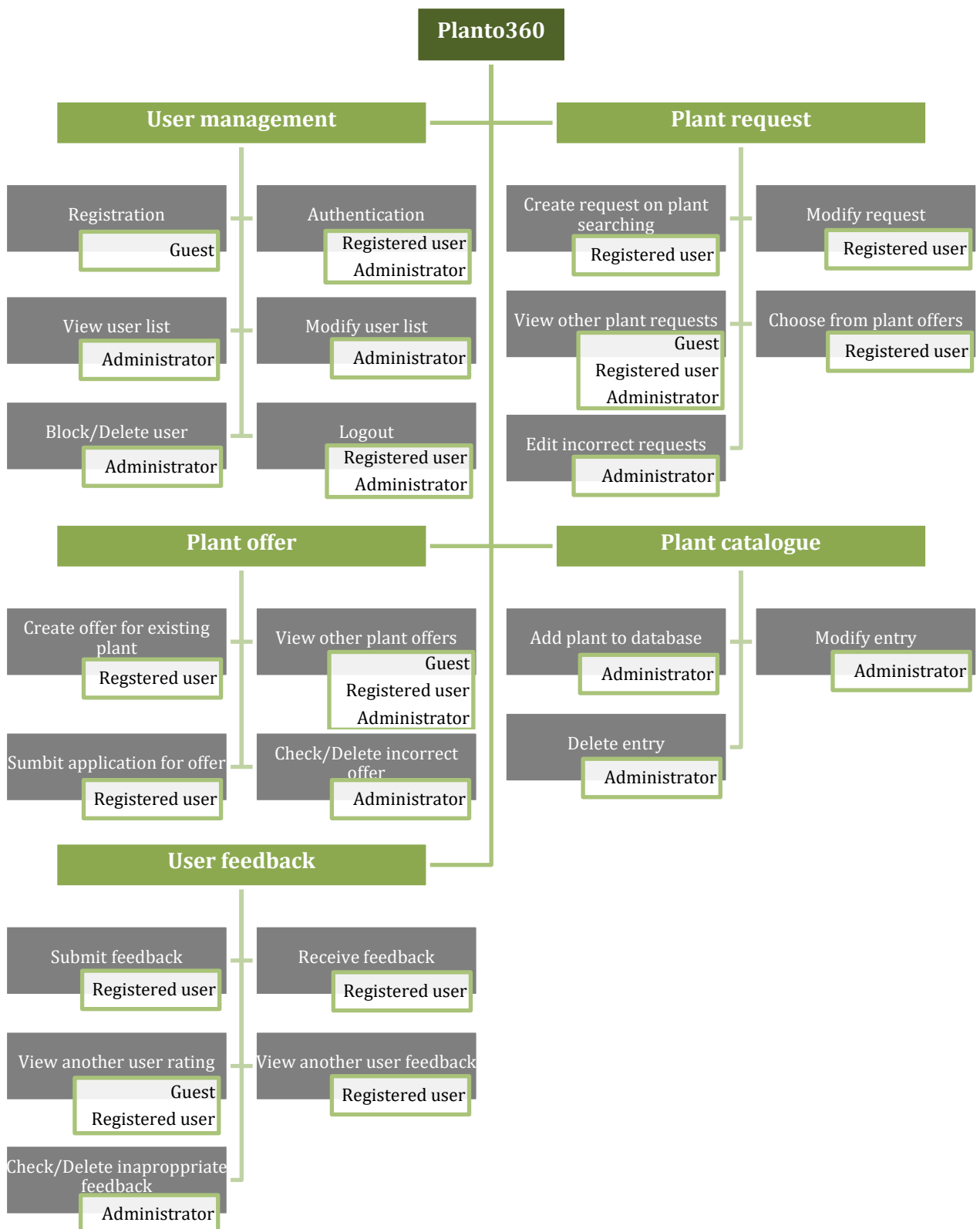
2.2.1. General description of functions

The Planto360 system consists of 5 modules. The *figure 2.2.* visualize those modules, which are - User management module, Plant request module, Plant offer module, Plant catalogue module and User feedback module.



2.2. *figure* Division of Planto360 into modules

The *figure 2.3.* below describes the modules by giving an overview of the functions which are applicable to the corresponding module and the user groups that are applicable for each function.



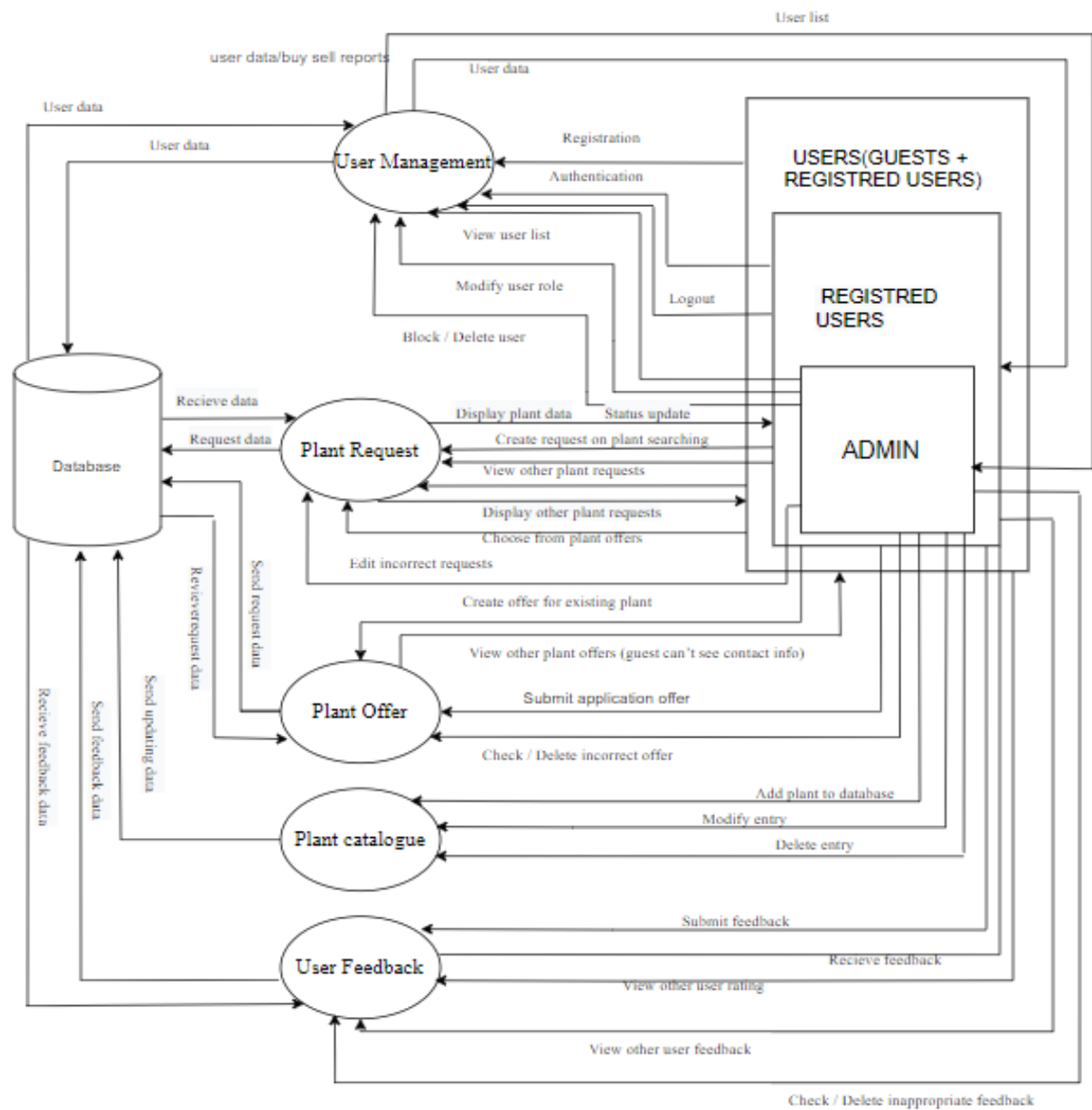
2.3. figure **Function and user group use according to module**

2.2.2. Functional breakdown by modules

All five modules are interconnected between themselves. They are also interacting with the user groups that are involved in the Planto360 life - guests, registered users and admin. There is also action with the database, e.g. data that is applicable for the module is moving from user to database through specific function.

The *figure 2.4.* - data flow diagram shows the five modules of the Planto360 system and their interaction within the user groups and database involved in the processes. The overall description of the modules is the following:

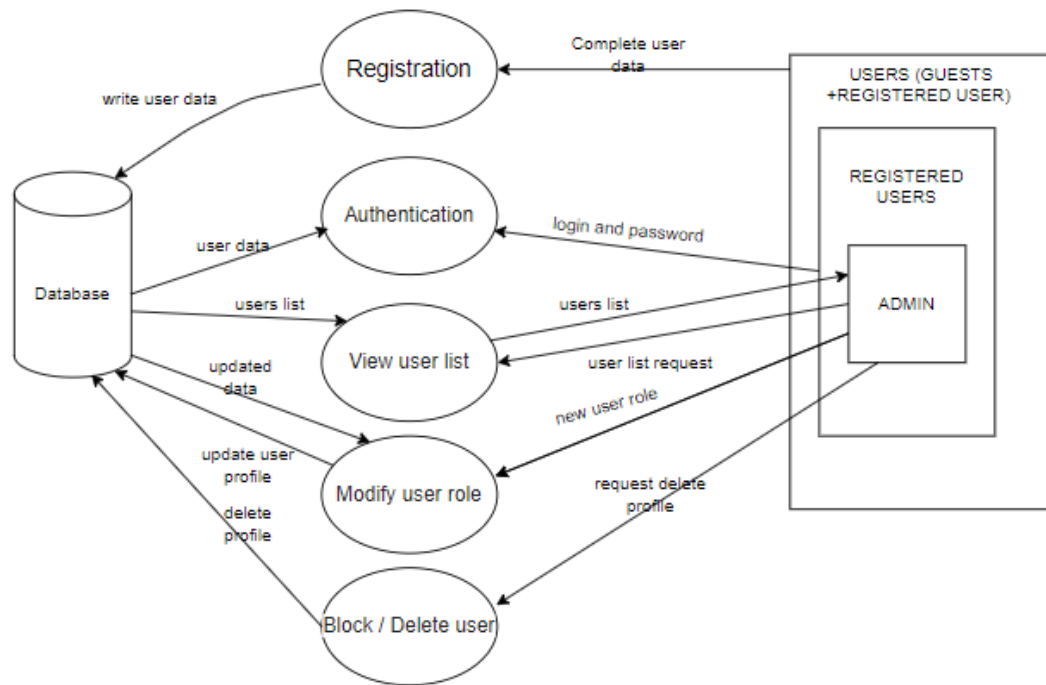
- user management module - consists of the functions that are related to user registration, their profiles and managing users from administrator side;
- plant request module - consists of the functions of placing the plant search request in the system, as well as other editing or deleting options;
- plant offer module - includes functions related to placing and maintaining records of plants that user has and is willing to share;
- plant catalogue module - includes the actions done by administrator to maintain the database of plants;
- user feedback module - consists of functions that allows users to leave/receive feedback, as well as maintain the user rating in the system.



2.4. figure Data flow diagram (level 1)

2.2.3. Module - User management

The *figure 2.5.* is the data flow diagram (level 2) that visualizes the user management function and how they are connected with the users involved and the database that stores the data.



2.5. figure User management module - data flow diagram (level 2)

More detailed description of each function for the user management module is described in the *table 2.1.* below.

2.1. table User management module - registration function

Registration
<i>Overview</i>
Registering new user by a guest
<i>Input</i>
A valid email address - not registered in system before Display email Password Password confirmation

<i>Process</i>
<ol style="list-style-type: none"> 1. The system receives from user input: desired username, password, email address, mobile phone number and CAPTCHA 2. Verifies if the username is not exist in database 3. Checks if password matches with security policy 4. Checks CAPTCHA is valid 5. Sends to the email address mail with unique link to verify if email address is valid 6. User receives email , opens links, the link open website to continue registration process 7. System send the sms to user to mobile phone number with random validation code 8. Wait users input to enter this code 9. After all data verified the system creates the users in databases with all data
<i>Output</i>
If user is successfully added to the system, he will be automatically logged in the system

2.2. table User management module - authentication function

Authentication
<i>Overview</i>
Login to the system
<i>Input</i>
A user enters their username and password to login in the system
<i>Process</i>
<ol style="list-style-type: none"> 1. user input username and password 2. system verifies its in database - if success logs in 3. waits 5 sec(to exclude brute force attacks) than returns to login page
<i>Output</i>
The user receives a message of requested is approved or rejected

2.3. table User management module - view user list function

View user list
<i>Overview</i>
Proceed and shows the list of all users of the system with corresponding user role and actions to be taken with users

<i>Input</i>
Button “show users”
<i>Process</i>
1. Asks database to return the list of all usernames, emails and phone numbers , and roles
<i>Output</i>
Returns a formatted list of all users of the system, consisting of username, email. At least there always will be 1 user admin itself.

2.4..table User management module - modify user list function

Modify user role
<i>Overview</i>
Changes the role of the selected user to different to change the user’s right
<i>Input</i>
Logged to the system as Admin, User existed and has a role, need to modify it to other role
<i>Process</i>
<ol style="list-style-type: none"> 1. Logged to the system as Admin 2. Opened list of all users by View user list function 3. Select function “change user role” for a specific user 4. Updates record field “user role” of selected user in the database based on the new role chosen from the drop-down list.
<i>Output</i>
Returns a formatted list of all users of the system with changed user role.

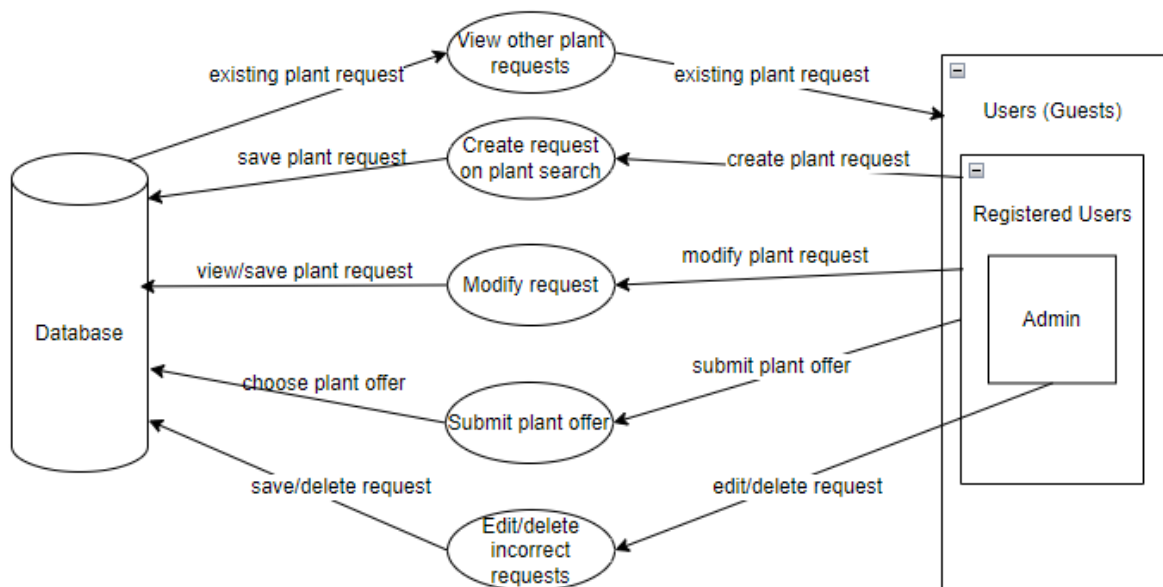
2.5. table User management module - block/delete user function

Block / Delete user
<i>Overview</i>
Admin can block/delete some users

<i>Input</i>
User to block/delete
<i>Process</i>
<p>User blocking :</p> <ol style="list-style-type: none"> 1. The user's request is received by system 2. The system checks whether the user is already blocked 3. If the user is already blocked, then user unblocked 4. Otherwise the user is blocked <p>User deleting:</p> <ol style="list-style-type: none"> 1. Checks if the user exist 2. Delete the user and all his data from database
<i>Output</i>
A message that user's blocked/delete

2.2.4. Module - Plant request

The *figure 2.6.* is the data flow diagram (level 2) that visualizes plant request function and how it's connected with the users involved and the database that stores the data.



2.6. *figure* Plant request module - data flow diagram (level 2)

2.6. table **Plant request module - create request on plant searching**

Create request on plant searching
<i>Overview</i>
Allow registered users to create plant request
<i>Input</i>
Logged to the system as Registered user
<i>Process</i>
<ol style="list-style-type: none"> 1. Logged to the system as Registered user 2. Opened list of plant requests 3. Select function “create request” 4. Updates record fields
<i>Output</i>
Returns a formatted data with all plant requests

2.7. table **Plant request module - modify request**

Modify request
<i>Overview</i>
Allow registered users to modify their requests
<i>Input</i>
Logged to the system as Registered user
<i>Process</i>
<ol style="list-style-type: none"> 1. Logged to the system as Registered user 2. Opened list of plant requests 3. Opened list of your plant requests 4. Selected function “modify/edit” your request <ol style="list-style-type: none"> 1. After edited, selected function “save” to save changes 2. Updates edited fields
<i>Output</i>
Returns a formatted data with all your plant requests

2.8. table **Plant request module - view other plant requests**

View other plant requests
<i>Overview</i>
Allow all users to view all plant requests
<i>Input</i>
App opened. User can be either registered, guest or admin
<i>Process</i>
<ol style="list-style-type: none"> 1. App opened 2. Opened list of plant requests
<i>Output</i>
Returns a formatted data with all plant requests

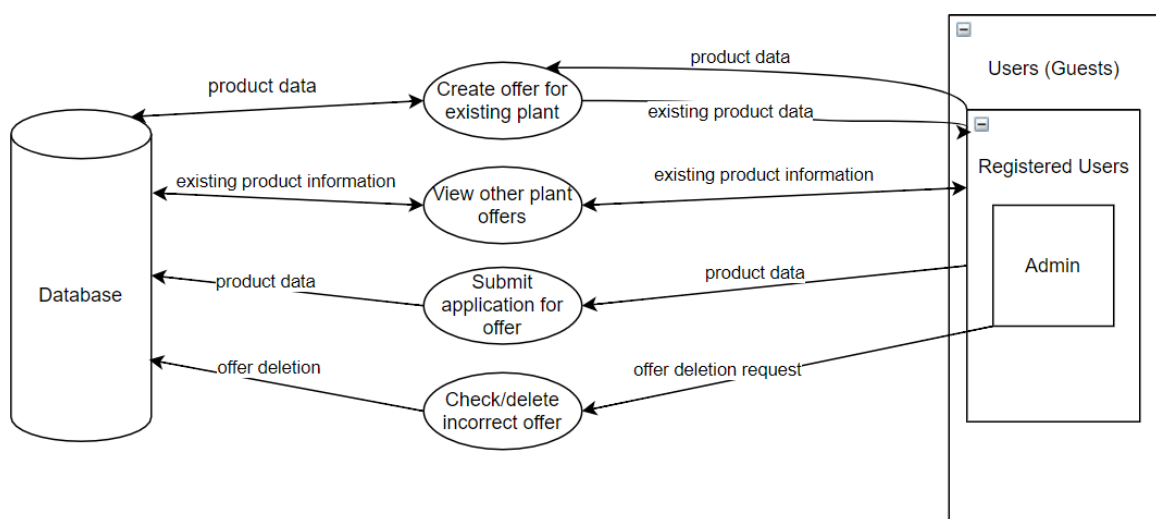
2.9. table **Plant request module - submit plant offer**

Submit plant offer
<i>Overview</i>
Allow registered users to submit plant offer
<i>Input</i>
Logged to the system as Registered user
<i>Process</i>
<ol style="list-style-type: none"> 1. Logged to the system as Registered user 2. Opened list of plant requests 3. Select function “make an offer” for a specific request
<i>Output</i>
<p>Message (pop-up window): “Thank you! You have made an offer! Wait for the approval and more details from the listing owner”</p> <p>Returns list of plant requests.</p>

Edit incorrect requests	
<i>Overview</i>	
Allow admin to edit and delete incorrect requests	
<i>Input</i>	
Logged to the system as admin	
<i>Process</i>	
<ol style="list-style-type: none"> 4. Logged to the system as admin 5. Opened list of plant requests 6. Select function “edit” for any request 7. After edited, selected function “save” to save changes 8. Updates edited fields 	
<i>Output</i>	
Returns a formatted data with all plant requests	

2.2.5. Module - Plant offer

The *figure 2.7.* is the data flow diagram (level 2) that visualizes the plant offer function and how they are connected with the users involved and the database that stores the data.



2.7. figure **Plant offer module - data flow diagram (level 2)**

2.11. table **Plant offer module - Create offer for existing plant**

Create offer for existing plant
<i>Overview</i>
Allow registered users to create posts for existing plants
<i>Input</i>
Offer product information
<i>Process</i>
<ol style="list-style-type: none"> 1. Logged to the system as Registered user 2. Select function “Create New Offer” 3. Provide offer information 4. After created, selected function “save” to save changes 5. Creates new offer
<i>Output</i>
Returns a formatted data with new plant offer

2.12. table **Plant offer module - View other plant offers**

View other plant offers
<i>Overview</i>
Allow registered users to view posts for existing plant offers
<i>Input</i>
Request to view offers
<i>Process</i>
<ol style="list-style-type: none"> 1. Logged to the system as Registered user 2. Open list of plant offers
<i>Output</i>
Returns a formatted data with existing plant offers

2.13. table **Plant offer module - Submit application for offer**

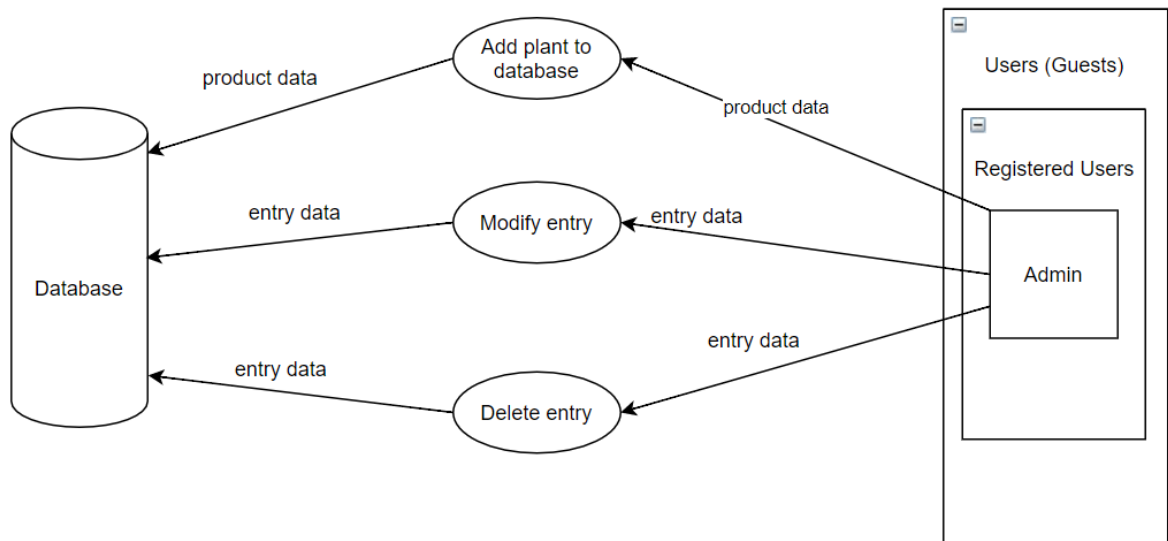
Submit application for offer
<i>Overview</i>
Allow registered users to apply for plant offer
<i>Input</i>
Information about plant offer user is applying to as well as delivery details
<i>Process</i>
<ol style="list-style-type: none"> 1. Logged to the system as Registered user 2. Open plant offers 3. Submit application for offer
<i>Output</i>
Returns a formatted data with new application for the plant offers

2.14. table **Plant offer module - Check/delete incorrect offer**

Check/delete incorrect offer
<i>Overview</i>
Allow admin to delete incorrect offer
<i>Input</i>
Information about plant offer
<i>Process</i>
<ol style="list-style-type: none"> 1. Logged to the system Admin 2. Open list of plant offers 3. Request plant offer deletion
<i>Output</i>
Returns a formatted data with deleted plant offer

2.2.6. Module - Plant catalogue

The *figure 2.8.* is the data flow diagram (level 2) that visualizes the plant catalogue function and how they are connected with the users involved and the database that stores the data.



2.8. *figure* Plant catalogue module - data flow diagram (level 2)

2.15. *table* Plant offer module - Add plant to database

Add plant to database
<i>Overview</i>
Allow admin to add plant to database
<i>Input</i>
Information about plant
<i>Process</i>
1. Logged to the system Admin 2. Open plant submission form 3. Submit plant information
<i>Output</i>
Returns a formatted data with new plant information

2.16. table **Plant offer module - Modify entry**

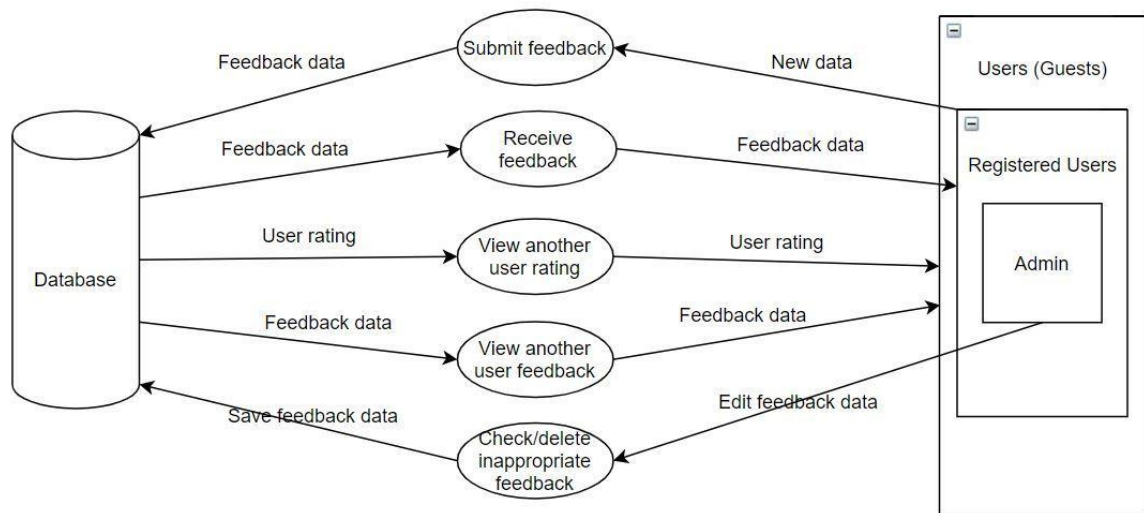
Modify entry
<i>Overview</i>
Allow admin to modify plant information
<i>Input</i>
Modified information about plant
<i>Process</i>
<ol style="list-style-type: none"> 1. Logged to the system Admin 2. Open particular plant information page 3. Request plant offer modification
<i>Output</i>
Returns a formatted data with modified plant information

2.17. table **Plant offer module - Delete entry**

Delete entry
<i>Overview</i>
Allow admin to delete plant information
<i>Input</i>
Information about plant
<i>Process</i>
<ol style="list-style-type: none"> 1. Logged to the system Admin 2. Open particular plant information page 3. Request plant offer deletion
<i>Output</i>
Returns a formatted data with deleted plant information

2.2.7. Module - User feedback

The *figure 2.9.* is the data flow diagram (level 2) that visualizes user feedback function and how it's connected with the users involved and the database that stores the data.



2.9. figure User feedback module - data flow diagram (level 2)

2.18. table User feedback module - submit feedback

Submit feedback
<i>Overview</i>
Allow registered users who have interacted with others to submit feedback
<i>Input</i>
Logged to the system as registered user
<i>Process</i>
<ol style="list-style-type: none"> 1. Logged to the system as registered user 2. Opened profile information 3. Opened history of listed actions (requests/offers) 4. Select function "leave feedback" for any action performed in the last 100 days. 5. Select rating from 1-5 and leave optional comments. 6. After select function "save" to save feedback 7. Updates edited fields

<i>Output</i>
Returns a formatted data with all your plant requests and offers

2.19. table User feedback module - receive feedback

Receive feedback
<i>Overview</i>
Allow registered users who have interacted with others to receive feedback
<i>Input</i>
Logged to the system as registered user
<i>Process</i>
<ol style="list-style-type: none"> 1. Logged to the system as registered user 2. Opened profile information 3. Opened “Feedback” section
<i>Output</i>
Returns a formatted data with all your feedback

2.20. table User feedback module - view another user rating

View another user rating
<i>Overview</i>
Allow all users to view another user rating
<i>Input</i>
App opened. User can be either registered, guest or admin
<i>Process</i>
<ol style="list-style-type: none"> 1. App opened 2. Opened list of plant requests of plant offers 3. Opened any request/offer
<i>Output</i>
Returns a plant request/offer with user rating. To see a full other user's profile, the user must be registered.

2.21. table User feedback module - view another user feedback

View another user feedback
<i>Overview</i>
Allow all users to view another user feedback
<i>Input</i>
App opened. User can be either registered, guest or admin
<i>Process</i>
<ol style="list-style-type: none"> 1. App opened 2. Opened list of plant requests of plant offers 3. Opened any request/offer
<i>Output</i>
Returns a plant request/offer with latest user feedback. To see a full other user's profile, the user must be registered.

2.22. table User feedback module - check/delete inappropriate feedback

Check/delete inappropriate feedback
<i>Overview</i>
Allow admin to check and delete inappropriate feedback
<i>Input</i>
Logged to the system as admin
<i>Process</i>
<ol style="list-style-type: none"> 1. Logged to the system as admin 2. Opened list of users feedback 3. Select function "delete" for any request 4. Updates edited fields
<i>Output</i>
Returns a formatted data with all users feedback

2.3. Non Functional requirements

Performance

- The system must provide access to 100'000 registered users in total - simultaneous at least 15'000 active users. The system runtime must be up to 4 seconds. Also there should be a graphical loading display if there are some delays in system performance.

Security

- User data is protected by encryption and not disclosed to public parties.
- Access to users personal data is only available to the same user and administrators. All systems users are familiar and agree to the privacy policy and the processing of their data within the system.

Scalability

- The system must be able to increase the number of registered users while not losing performance speed and the service at the same time.

Availability

- The servers hosting the system and its database are available continuously.

Usability

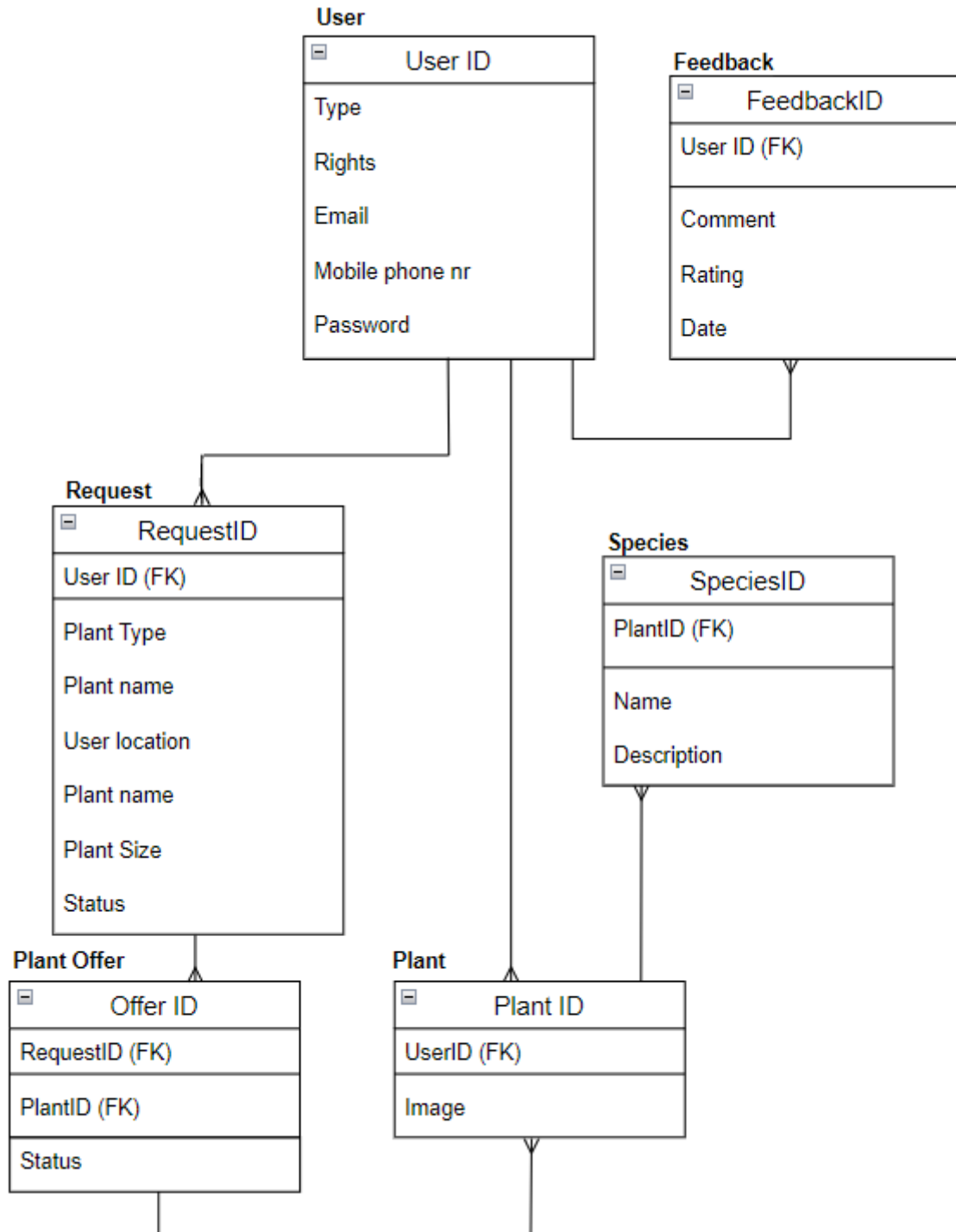
- The system must be available to users 24/7 hours a day, 7 days a week. Except for system maintenance if it can't be performed at the same time.
- If a system error is detected that causes the system the operation must be stopped, fixed error in 5 days, and set back on track.
- The language used in the interface must be intuitive to all user groups and must be easy for the user to understand.

Maintainability

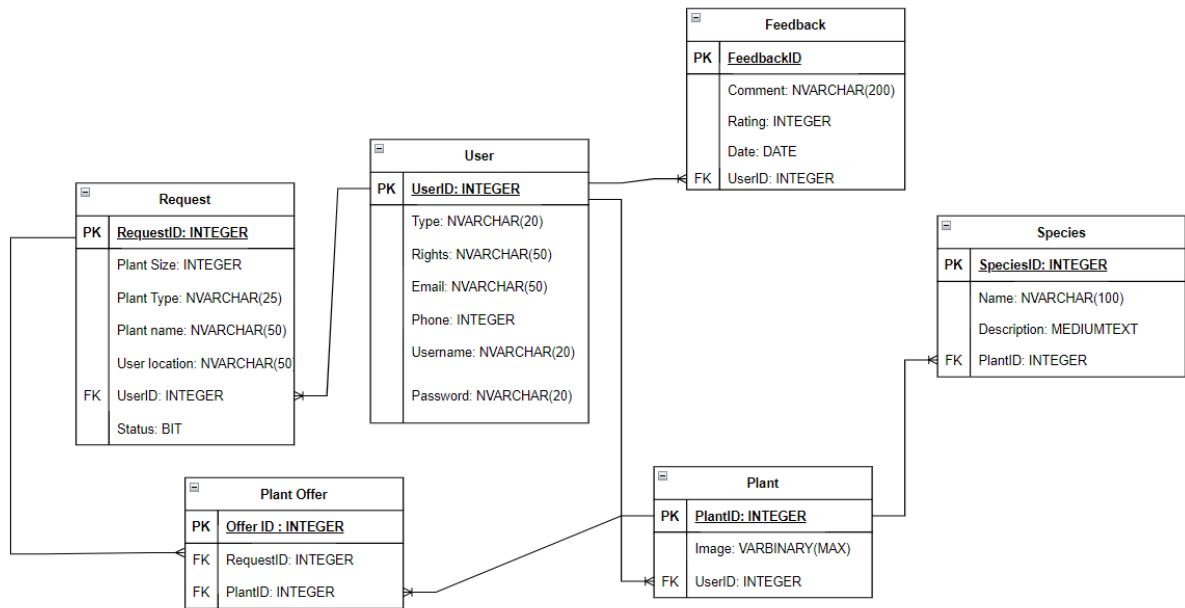
- The system is maintained and updated as needed and time for it is scheduled by the system administrator. The system must be operational 99% of the time. The system failure rate limit is not more than 3h per week.

3. SOFTWARE DESIGN DESCRIPTION

3.1. Database design



3.1. figure Logical database model (Crowfoot syntax)



3.2. figure Physical database model

3.1.1. Entity – Users

This subsection contains detailed information about table design of User entity (*table 3.1.*).

3.1. table User entity - description

Key	Column name	Data Type	Null / Not Null	Other parameters	Description
PK	UserID	int	Not Null	IDENTITY (1, 1)	Artificial id for user identification.
	Type	nvarchar(20)	Not Null		Type of user - individuals, vendors and administrators.
	Rights	nvarchar(50)	Not Null		Type of rights - create, read, update, delete.
	Email	nvarchar(50)	Not Null	UNIQUE	Unique email of the user.
	Phone	int	Not Null	UNIQUE	Mobile phone number of the user.

	Username	nvarchar(20)	Not Null	UNIQUE	Specific name created by user for login.
	Password	nvarchar(20)	Not Null		Specific sequence of characters for login.

3.1.2. Entity - Plant request

This subsection contains detailed information about table design of Request (*table 3.2.*).

3.2. table **Request** entity – description

Key	Column name	Data Type	Null / Not Null	Other parameters	Description
PK	RequestID	int	Not Null	IDENTITY (1, 1)	Artificial id for request identification.
	Plant Size	int	Null		Size of plant in centimeters for particular request.
	Plant Type	nvarchar(25)	Not Null		Type of plant - indoor/outdoor for particular request.
	Plant name	nvarchar(50)	Not Null		Name of plant for particular request.
	User Location	nvarchar(50)	Null		Location of the user of a particular request.
	Status	bit	Not Null		Request status
FK	UserID	int	Not Null	UNIQUE, FK referenced from entity Users (table 3.1., UserID attribute)	Connection between user's account and plant request.

3.1.3. Entity - Plant offer, Plant, Species

This subsection contains detailed information about table design of Plant offer (*table 3.3.*).

3.3. *table* **Plant offer entity – description**

Key	Column name	Data Type	Null / Not Null	Other parameters	Description
PK	OfferID	int	Not Null	IDENTITY (1, 1)	Artificial id for the plant offer placed by user.
FK	RequestID	int	Not Null		Reference to the plant request.
FK	PlantID	int	Not Null		Reference to the plant offered by the user.
	Status	bit	Not Null		Plant offer status

This subsection contains detailed information about table design of Plant entity (*table 3.4.*).

3.4. *table* **Plant entity - description**

Key	Column name	Data Type	Null / Not Null	Other parameters	Description
PK	PlantID	int	Not Null	IDENTITY (1, 1)	Artificial id for plant offered by user.
	Image	varbinary(max)	Not Null		Image of the plant offered.
FK	UserID	int	Not Null		Reference to the plant owner.

This subsection contains detailed information about table design of Species entity (*table 3.5.*).

3.5. table Species entity - description

Key	Column name	Data Type	Null / Not Null	Other parameters	Description
PK	SpeciesID	int	Not Null	IDENTITY (1, 1)	Artificial id for plant species.
	Name	nvarchar(100)	Not Null		Name of the plant species.
	Description	mediumtext	Not Null		Description of the plant species.
FK	PlantID	int	Not Null		Reference to the plant.

3.1.4. Entity - User feedback

This subsection contains detailed information about table design of User Feedback (*table 3.6.*)

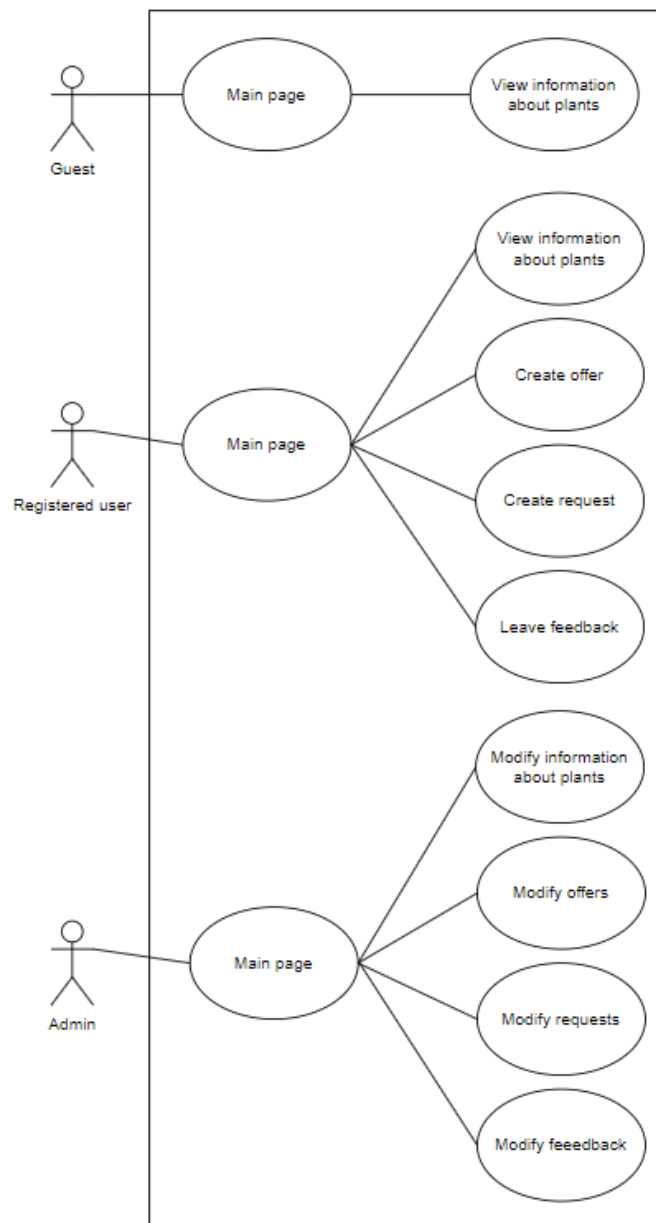
3.6. table Feedback entity - description

Key	Column name	Data Type	Null / Not Null	Other parameters	Description
PK	FeedbackID	int	Not Null	IDENTITY (1, 1)	Artificial id for feedback identification.
	Comment	nvarchar (200)	Null		Comment of particular feedback.
	Rating	int	Not Null		Evaluation of a user in a range from 1 to 5.
	Date	date	Not Null		Date of particular feedback.
FK	UserID	int	Not Null	UNIQUE, FK referenced from entity Users (table 3.1., UserID attribute)	Reference to user.

3.2. Partial Functional Design

3.2.1. User interface

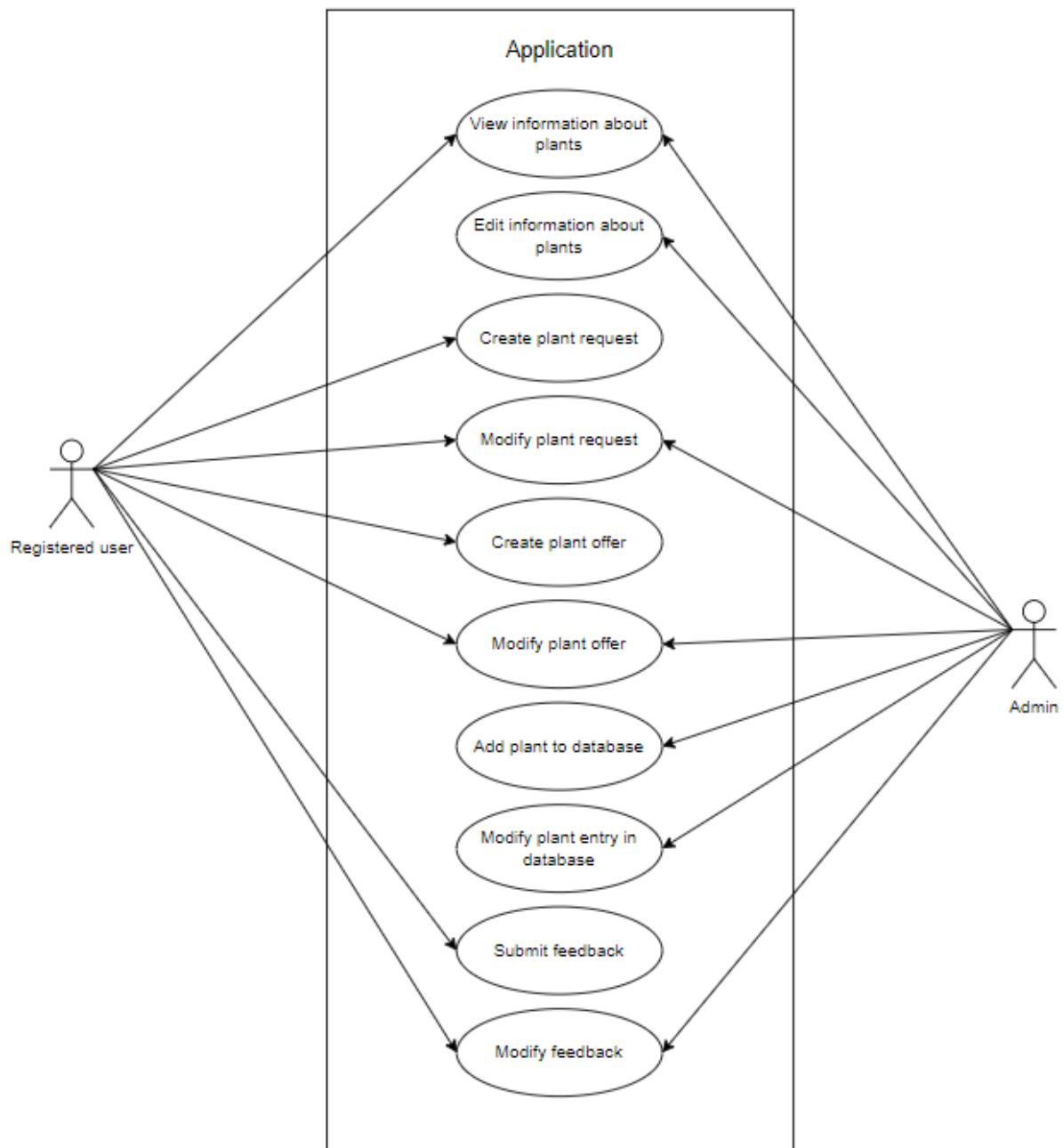
In the *figure 3.3*. UML use case diagram “User interface” describes the high-level functions of the application. It identifies the interactions between the application and its users - guest user, registered user and admin.



3.3. *figure* User interface (UML use case diagram)

3.2.2. User interface (authorized users)

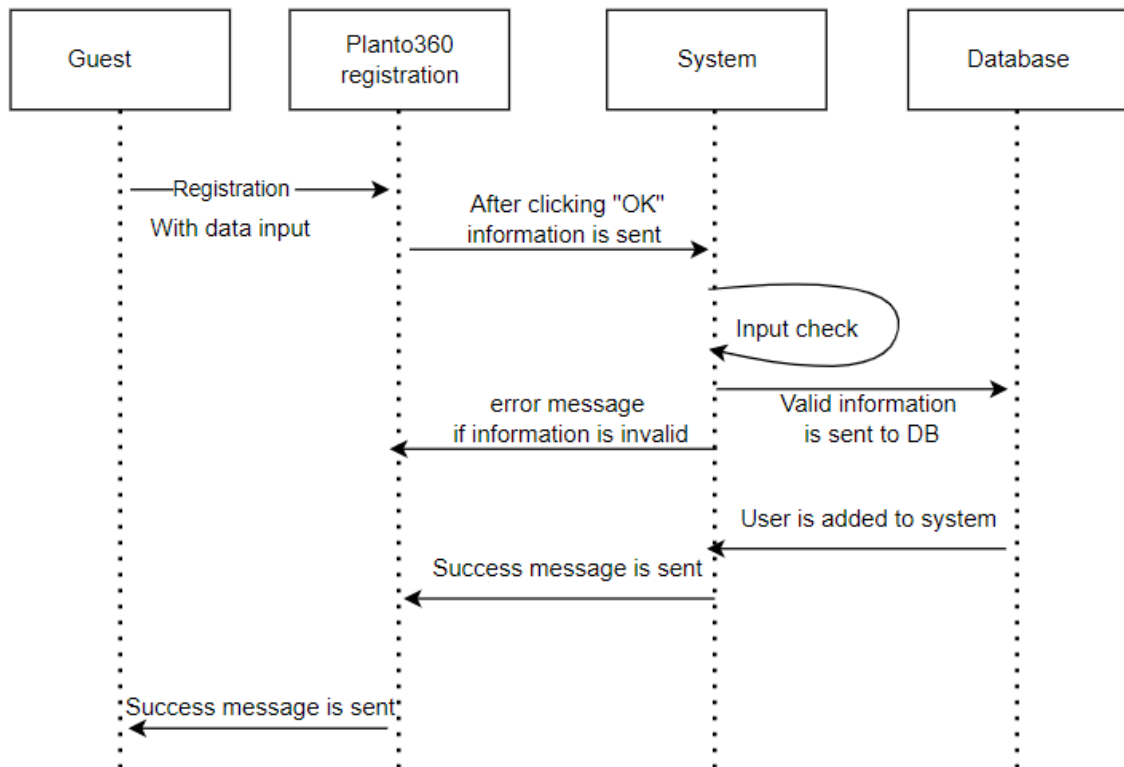
In the *figure 3.4*. UML use case diagram “User interface (authorized user)” identifies the interactions between the application and authorized users only. It’s similar to the previous diagram, but is more detailed and it’s easier to see what’s common for authorized users and what’s not different.



3.4. *figure* User interface (authorized users) (UML use case diagram)

3.2.3. User profile registration

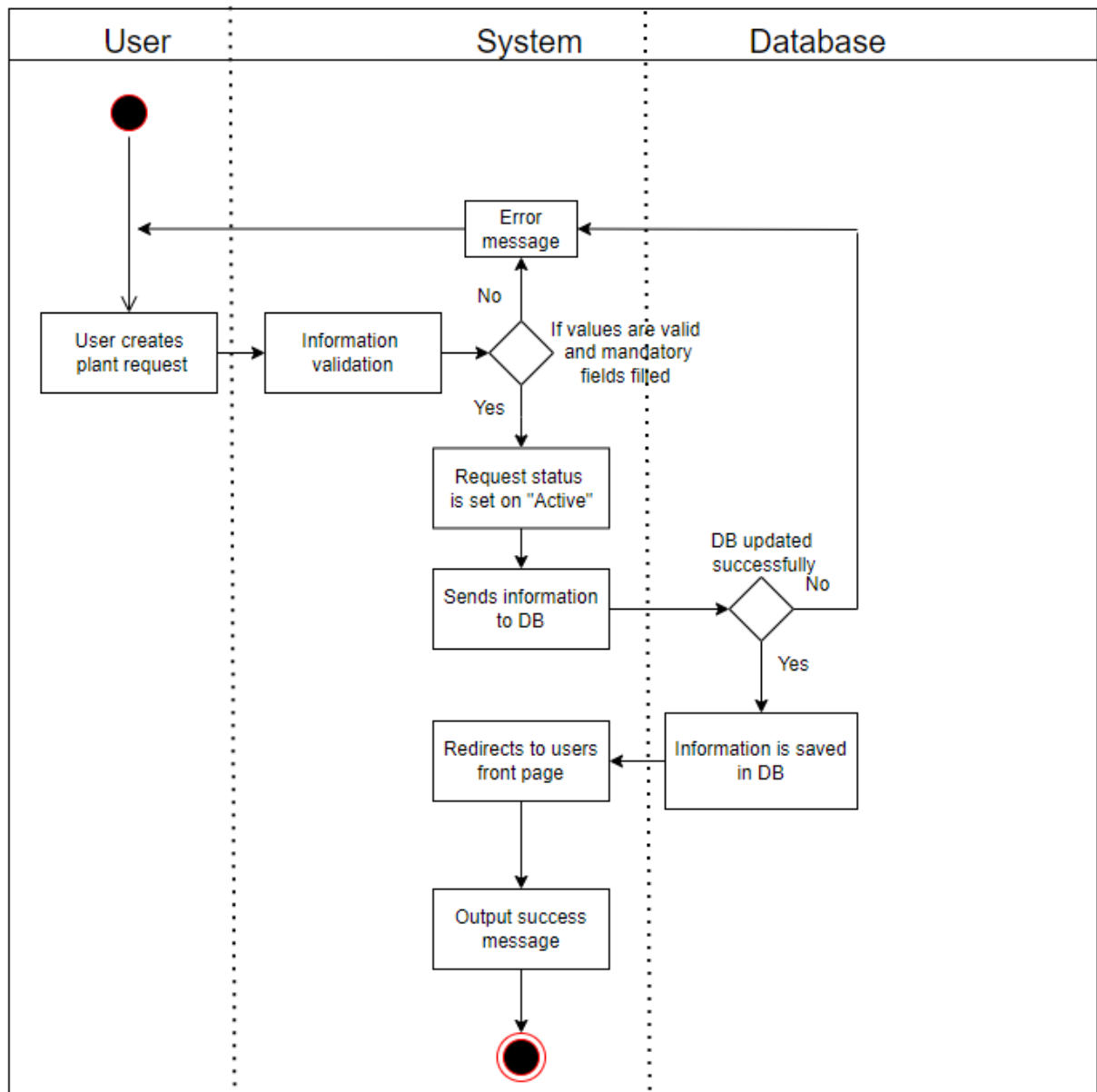
This *figure 3.5.* is sequence diagram that describes how user registration is performed. Guest/Unregistered user uses Planto360 registration function. After filling all mandatory information and clicking “OK”, System checks if data is valid, if it's not, then Error message is displayed. If data is valid, information is sent to Database and Success message is sent to user.



3.5. *figure* User profile registration (UML sequence diagram)

3.2.4. Plant request creation

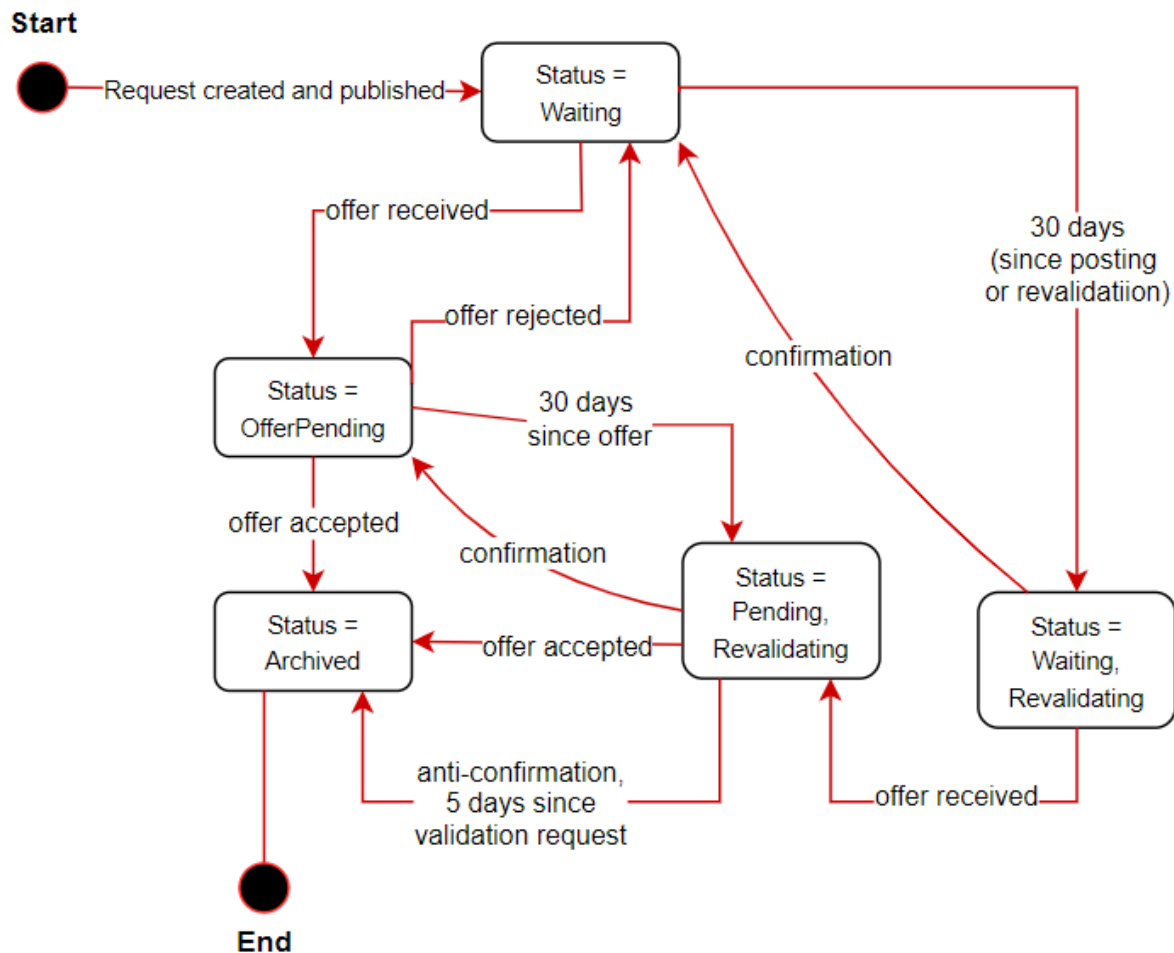
User creates a plant request by filling all mandatory fields in the request form and submits it to the system. If input data is not valid, an error message is returned. If data is valid, Request status is changed to active and information is sent to the database. If the Database is not updated successfully error message is returned, if it's successful user is redirected to the front page and success message is displayed.



3.6. figure Plant request creation (UML activity diagram)

3.2.5. Plant request status change

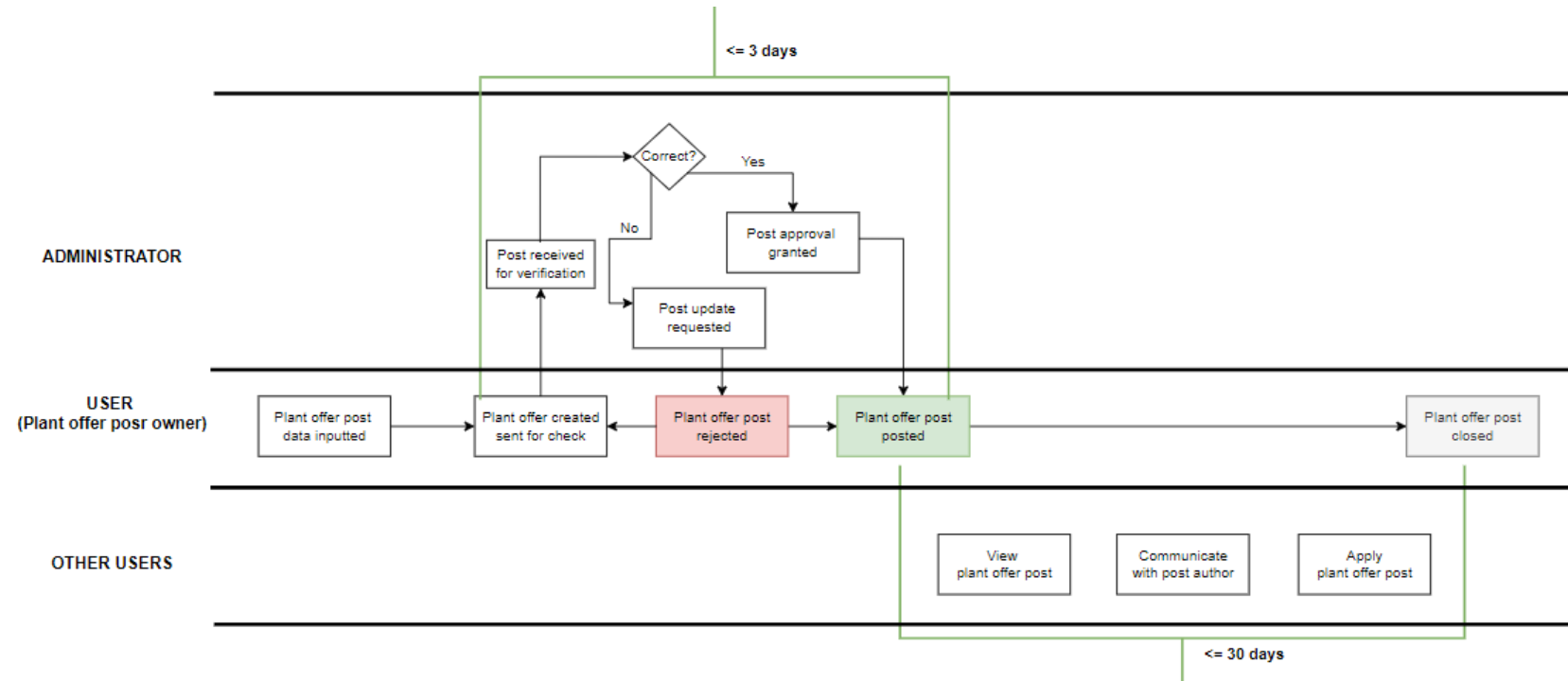
The *figure 3.7.* plant request status change. When a plant request is successfully created and published, its status in the system is active. When the requesting user receives a plant offer and accepts it, the request status automatically changes to inactive and the request is archived. If the user receives offers but does not accept any, the request's status stays active for 30 days. After 30 days the user is asked to validate whether the request has to be kept as active. If confirmation is received then the status is prolonged for 30 more days and other users can post plant offers in the thread. If no response from the user is received after validation request, the request's status is automatically changed to inactive and the request is archived.



3.7. figure Plant request status change (UML state diagram)

3.2.6. Plant offer post

The *figure 3.8.* shows the plant offer post lifecycle on the UML timing diagram base - starting at the point where the post owner creates and submits the post until the post closure. The timing diagram includes the actions of 3 parties - user, administrator, other users -, in regards to plant offer. The user is the owner of the offer post, the user submits the post, and then it is submitted to administrator review to either be posted publicly or rejected, which means that the owner of the post needs to do updates; this process should not take more than 3 days. After the post has been posted it would stay by default up till 30 days or less, e.g. if there are no applications for the offer, the post is taken off after 30 days, or it can be taken off if there is application for it or the user decides to delete it based on personal reasons.

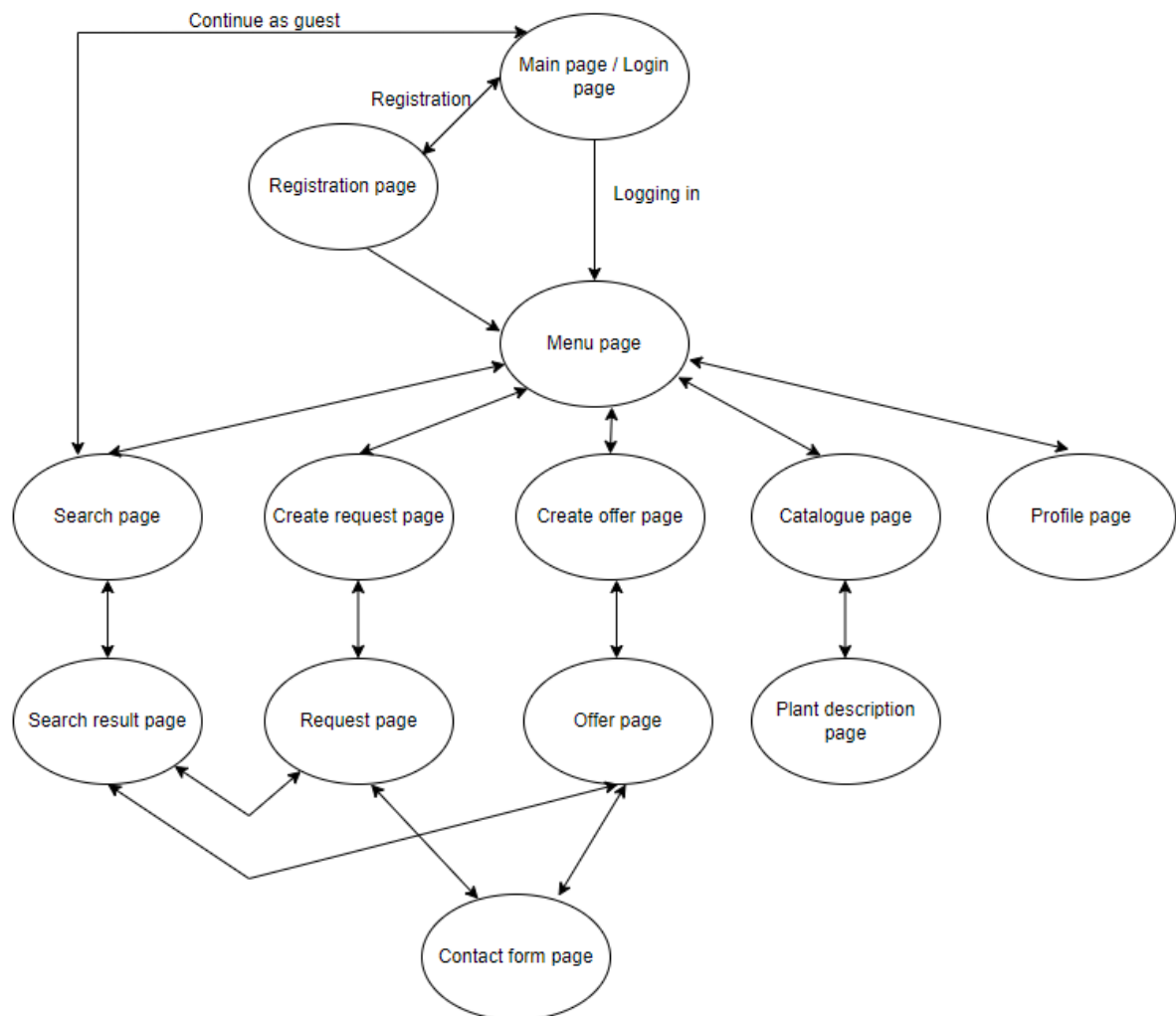


3.8. figure Plant offer post (UML timing diagram)

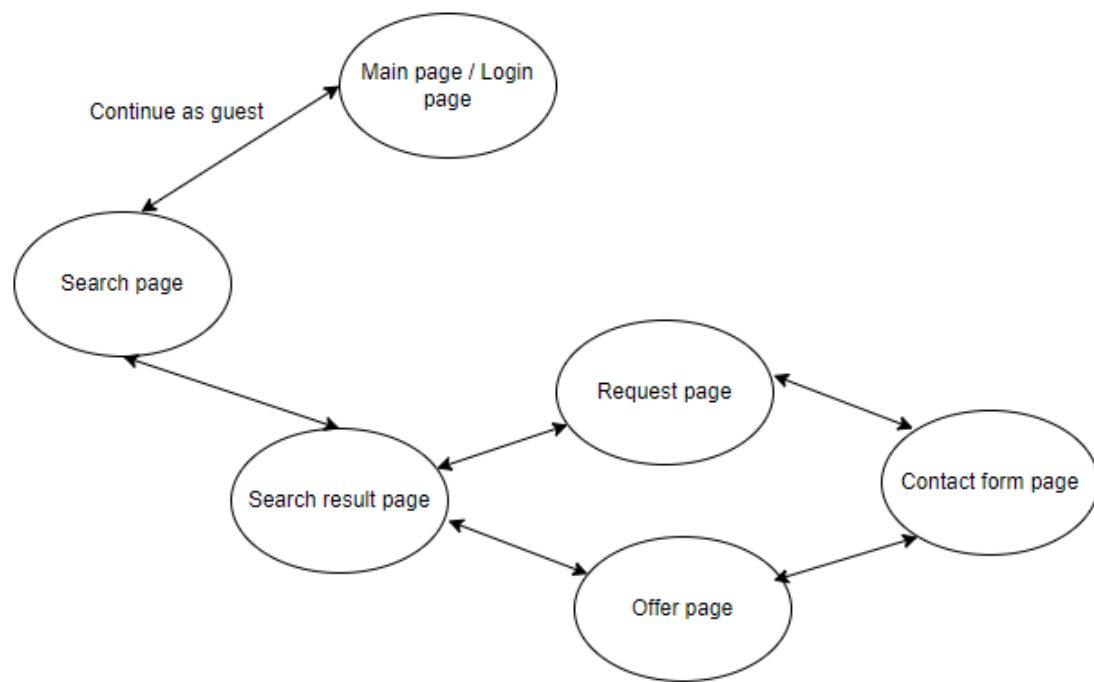
3.3. Partial Design of User Interface

3.3.1. General user interface

Planto360 has a user-friendly user interface. The overall User interface is presented in *figure 3.9.*, where all users start from the Main Page which is the same as the Login page, there user has 3 options to login, register or continue as guest. If user decides to login or register, he is redirected to Menu page, from where there are 5 options - Search page, Create request page, Create offer page, Catalogue page, Profile page. In case user continues as guest, see *figure 3.10.*, the only option is to get redirected to Search page, view the results and go to Contact form page. The register after also has options to create request or offer, see the plant descriptions and edit profile. Planto360 provides also the Back button, for user convenience, which allows to get to the previous page, where user was.

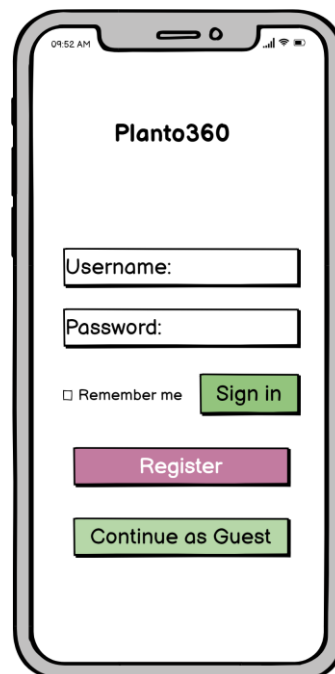


3.9. figure General user interface

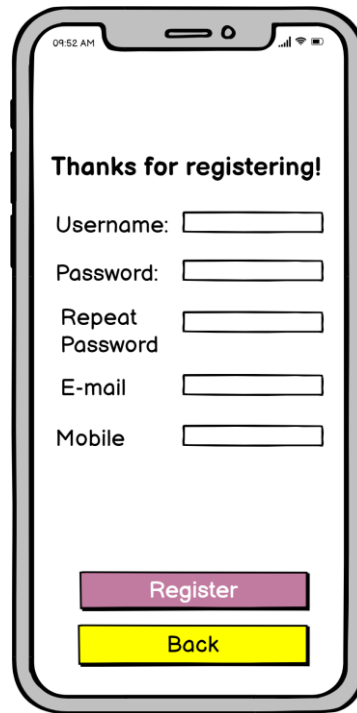


3.10. figure Guest user - user interface

3.3.2. User interface wireframes



3.11. figure Login screen wireframe



09:52 AM

Thanks for registering!

Username:

Password:

Repeat Password

E-mail

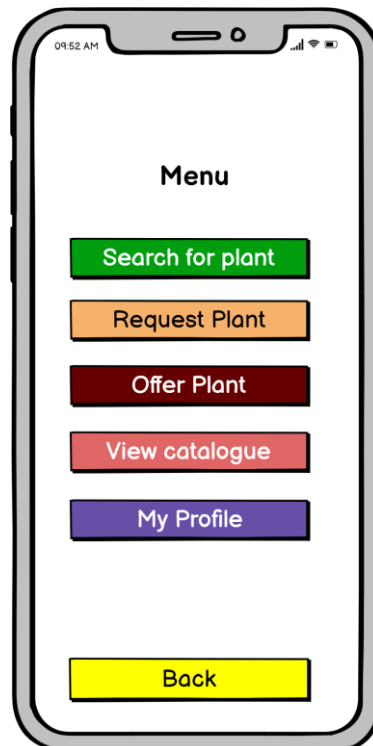
Mobile

Register

Back

This wireframe shows a mobile app screen for user registration. At the top, the status bar displays '09:52 AM'. Below it, a title 'Thanks for registering!' is centered. The form contains five input fields: 'Username', 'Password', 'Repeat Password', 'E-mail', and 'Mobile'. At the bottom, there are two buttons: a purple 'Register' button and a yellow 'Back' button.

3.12. figure User registration screen wireframe menu



09:52 AM

Menu

Search for plant

Request Plant

Offer Plant

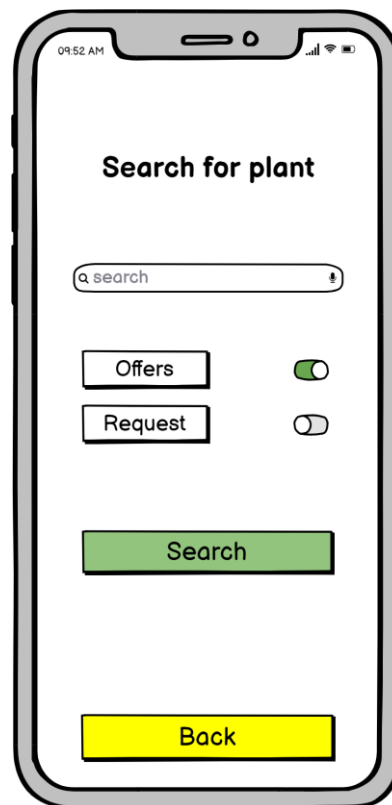
View catalogue

My Profile

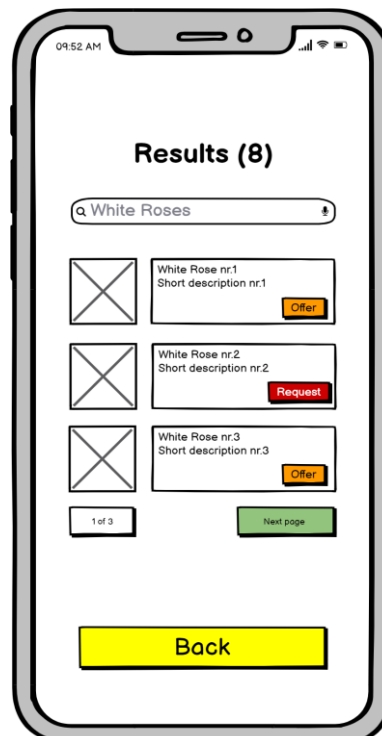
Back

This wireframe shows a mobile app screen for a menu. At the top, the status bar displays '09:52 AM'. Below it, a title 'Menu' is centered. The screen features six buttons stacked vertically: 'Search for plant' (green), 'Request Plant' (orange), 'Offer Plant' (dark red), 'View catalogue' (red), 'My Profile' (purple), and 'Back' (yellow).

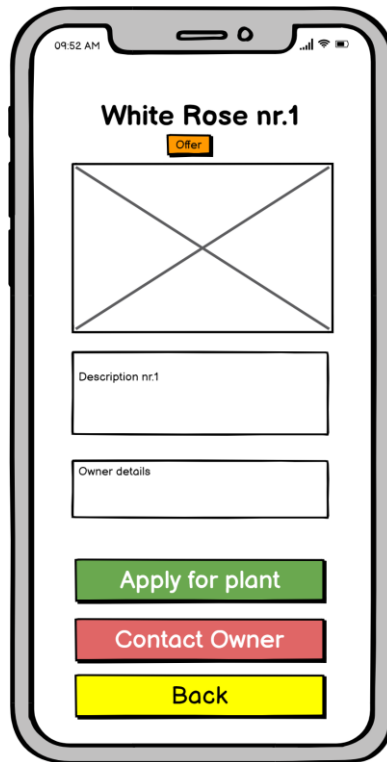
3.13. figure Menu screen wireframe



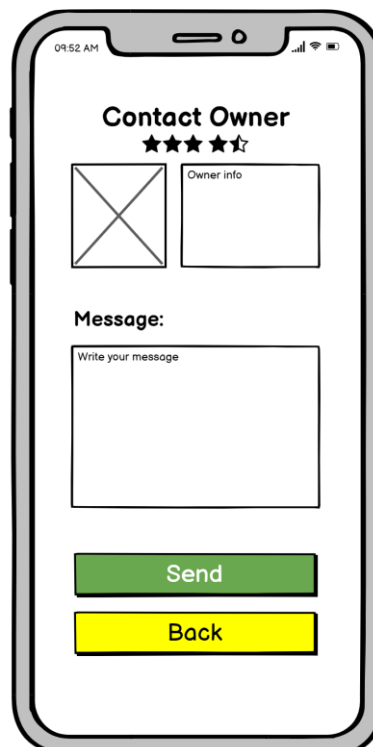
3.14. figure Search screen wireframe



3.15. figure Search results screen wireframe



3.16. figure Plant offer screen wireframe



3.17. figure User contact screen wireframe

09:52 AM

Request Plant

Plant type ☐ Indoor ☐ Outdoor

Plant name

User
location

Plant size

Comment

Write your comment here

Create

Back

3.18. figure Create request screen wireframe

09:52 AM

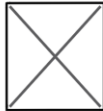
Offer Plant

Plant type ☐ Indoor ☐ Outdoor

Plant name

User
location

Plant size

Upload picture 

Comment

Write your comment here

Create

Back

3.19. figure Create offer screen wireframe

REFERENCES AND SOURCES

Books

Sommerville I. *Software Engineering, 9th ed.* Boston: Addison-Wesley, 2010.

Pressman R.S. *Software Engineering: A Practitioner's Approach, 7th ed.* New York: McGraw-Hill, 2010.

Bourque P., Fairley R.E. *SWEBOK Guide v 3.0. Guide to the Software Engineering Body of Knowledge.* // <https://ieeecs-media.computer.org/media/education/swebok/swebok-v3.pdf>
(Used in period from 02.11.2021 to 23.12.2021)

Internet resources

Blossm - Social Plant Market. // <https://apps.apple.com/us/app/blossm-social-plant-market/id1503863325> (Used on 04.10.2021); <https://play.google.com/store/apps/details?id=garden.blossm&hl=en&gl=US> (Used on 04.10.2021)

Plantswapp: Plant Marketplace. // <https://play.google.com/store/apps/details?id=app.plantsw&hl=en&gl=US> (Used on 05.10.2021); <https://apps.apple.com/pl/app/plantswapp/id1508020545> (Used on 05.10.2021)

Systems and software engineering — Life cycle processes — Requirements engineering // ISO/IEC/IEEE 29148:2018, 30 November 2018. (Used in period from 11.10.2021 to 23.12.2021)