

Project title

Name:

Group:

Table of Contents

[Deliverable 1 3](#_Toc64843130)

[Project Specification 3](#_Toc64843131)

[Functional Requirements 3](#_Toc64843132)

[Use Case Model 3](#_Toc64843133)

[Use Cases Identification 3](#_Toc64843134)

[UML Use Case Diagrams 3](#_Toc64843135)

[Supplementary Specification 3](#_Toc64843136)

[Non-functional Requirements 3](#_Toc64843137)

[Design Constraints 3](#_Toc64843138)

[Glossary 3](#_Toc64843139)

[Deliverable 2 3](#_Toc64843140)

[Domain Model 3](#_Toc64843141)

[Architectural Design 4](#_Toc64843142)

[Conceptual Architecture 4](#_Toc64843143)

[Package Design 4](#_Toc64843144)

[Component and Deployment Diagram 4](#_Toc64843145)

[Deliverable 3 4](#_Toc64843146)

[Design Model 4](#_Toc64843147)

[Dynamic Behavior 4](#_Toc64843148)

[Class Diagram 4](#_Toc64843149)

[Data Model 4](#_Toc64843150)

[System Testing 4](#_Toc64843151)

[Future Improvements 4](#_Toc64843152)

[Conclusion 4](#_Toc64843153)

[Bibliography 4](#_Toc64843154)

# Deliverable 1

## Project Specification

[Present the project specification.]

The project represents a pet shop application with additional functionalities such as adopting or giving for adoption a pet. The actors which interact with the system have different roles: Administrator, Sellers (of pet products), Customers, Pet fosterers . A user can have one or more of the roles enumerated.

**Functional requirements**

The application is provided with an authentication/authorization system for register or login, based on Spring Security framework. To note that passwords are stored in database using a hashing algorithm, increasing the security level.

Once an user is register, it has the default role of **Customer**. Customers are able to see all products available, add them to a virtual cart or a wish-list and buy them; they can filter the products by category, also having the possibility to see all categories available in the store. In addition, they can see pets given for adoption by types (a.k.a. breeds) and adopt them.

The **Seller** is also able to see all products in the store, create a product (put it up for sale), update product (for example change product description, quantity, category etc.) or delete its products (remove it from sale). It has the permission to see what categories exist in the store.

The **Pet-Foster** can give a pet for adoption (create it virtually), update the pet object, delete it (equivalent to giving it for adoption to another user of the application) and see all pets registered in application.

The **Administrator** is responsible with the creation, modification, deletion of product Categories and Pet-Types. Also, it has the all the permissions a Seller, Customer or Pet-Foster has (for testing purpose).

* **Future improvements** (as features):
  + all modifications made to a product (that implies all CRUD operations) must be approved by the **Administrator.**
  + a user has to provide a certificate approved by **Administrator** in order for it to be an foster / seller

## Use Case Model 1

### Use Cases Identification

Use-Case: Admin logs in, search for specific or all application users, select one and perform actions on it

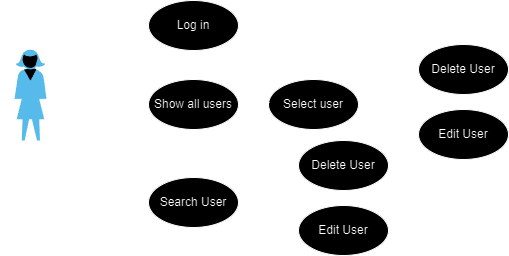
Level: Admin Goal

Primary Actor: Admin

Main success scenario:

* The administrator logs in using its credentials.
* After authorization, it is redirected to main page, where he can select from the top bar the actions on other users
* If he chooses to show all the users registered to application, it will be redirected to a page with a table with all information about those and a search box which improve the efficiency of getting a specific user.
  + He can select the row with specific users and perform CRUD operations on it.
* If it chooses to create a new user, it’s going to see a pop up with specific fields needed for registration.
  + It will be notified if the password is not strong (too short, no special characters, etc.), the email already exists in database, a mandatory field was not filled.
* If it chooses to remove a user, it will be able to do it eighter by select it from the table presended above, or it can choose **Remove User** from the tool bar and search it by name, username, email address.
* It it chooses to modify information of a user, it will be able to do it eighter by select it from the table presended above, or it can choose **Modify User** from the top bar and search it by name, username, email address.

Extensions:

* After log in, the administrator will be redirected to a Start page which consist in displaying personal information about the user,
* All pages have a top bar with all operations permitted for administrator.
* If the operations performed by the Admin on users are incomplete or wrong data is provided, a notify will pop up, letting to know what it did wrong.

Use-Case: Admin logs in, display all application user-roles and create new ones

Level: Admin Goal

Primary Actor: Admin

Main success scenario:

* The administrator logs in using its credentials.
* After authorization, it is redirected to main page, where he can select from the top bar the actions on other user-roles
* If he chooses to show all the users registered to application, it will be redirected to a page with a table with all information
* If it chooses to create a new user, it’s going to see a pop up with specific fields needed for registration.
  + It will be notified if the password is not strong (too short, no special characters, etc.), the email already exists in database, a mandatory field was not filled.

Extensions:

* After log in, the administrator will be redirected to a Start page which consist in displaying personal information about the user,
* All pages have a top bar with all operations permitted for administrator.
* If the operations performed by the Admin on users are incomplete or wrong data is provided, a notify will pop up, letting to know what it did wrong

A diagram of a company

Description automatically generated with medium confidence

Use-Case: Admin logs in, review product, pet creation/modification/deletion or Seller/Adopter Priviledge Request and approve/deny it

Level: Admin Goal

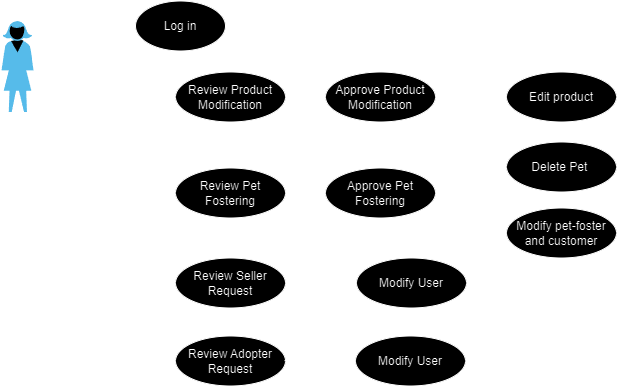
Primary Actor: Admin

Main success scenario:

* The administrator logs in using its credentials.
* After authorization, it is redirected to main page, where he can select from the top bar the actions on another user-roles
* In the tool bar a notification button will pe displayed. By pressing it, the Admin will be redirected to a page where it can filter request ( product or pet adoption )
* Product and/or Pet modifications/creations/deletions seller/adopter priviledge requestrequest will be shown.
* By selecting one of those requests, a pop-up page will apear with all details about the product, along with 2 buttons, Deny and Approve.
  + When one of those is selected, the pop-up will expand with a text field where the admin can specify the motivation of its decision.

Extensions:

* The notification button has a number on it, indicating the number of requests created.
* When the administrator is connected and another user is creating a new request in that moment, besides the fact that the number from notification button will be increased, a tiny pop-up will apear in the corner of the page, containing a summary of the request ( type of request , product or pet adoption, and the creator)



Use-Case: Customer logs in or register, buys products

Level: Customer Goal

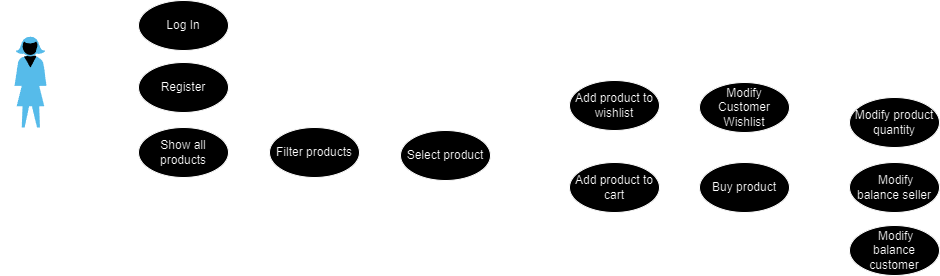
Primary Actor: Customer

Main success scenario:

* New customers hava to create an account by selecting “Register” Button from the top bar
* If the customer already has an account, it can log in application via “Log In” button from the top bar
* After authorization, it is redirected to the main page, where it can press a button “Shop Pet Products” to see all products available.
* If it wants a particular category of products, it can select desired Categories from the “Filter” button
  + Here multiple filters can be added , including a search box with specification of products
* Products will be displyed in a table-like manner; by clicking on a product, the Customer is able to see details of it, including 2 buttons: “Add to Cart“ and “Add to wishlist”
* In the top bar will be 2 new buttons, unlike the Administrator page, designed to redirect the user to a new page with all Favorite Products or The Shopping Cart Page
  + The shopping cart page , besides the list with all the products added to the cart, will have a “Buy” button, that will redirect the Customer to a new page where he has to provide information about the Order

Extensions:

* The top bar can have “Frequent Questions”, “Forum” buttons.
* Besides information about the product, it can have a “Reviews” section, where other users can express their opinions about the product.
* When a product is added to Cart or Favorite list, a tiny notification will pop-up, letting the user know that the operation has succeded.



Use-Case: Customer logs in or register, adopts a pet

Level: Customer Goal

Primary Actor: Customer

Main success scenario:

* New customers hava to create an account by selecting “Register” Button from the top bar
* If the customer already has an account, it can log in application via “Log In” button from the top bar
* After authorization, it is redirected to the main page, where it can press a button “Adopt Pets” to see all pets given for adoption.
* If it wants a particular type of pet, it can select desired Types from the “Filter” button
  + Here multiple filters can be added , including a search box with specification of pets (e.g. color, weight.)
* Pets will be displyed in a table-like manner; by clicking on a product, the Customer is able to see details of it, including 2 buttons: “Request Adoption”
* In the top bar, besides the new buttons previously mentioned in the first use-case, there will be a third button indicating the list with adoptions
  + The user is able to check the adoption status

Extensions:

* The “Customer – Adopter” is able to chat with the Pet-Foster

A screen shot of a computer screen

Description automatically generated

Use-Case: Seller logs in or register, create/update a product

Level: Seller Goal

Primary Actor: Seller

Main success scenario:

* In order to get priviledges of Seller, a simple customer must request Seller Roles from Administrator.
* After the prermissions are given to the customer, after log in , the user must choose as what he wants to continue the login process: customer or seller
* As seller, it is redirected to the main page, where it can select the action wanted to perform
* If he chooses to show all the products selled by him, it will be redirected to a page listing all the products.
* If it chooses to create / update a new user, it’s going to see a pop up with specific fields needed.

Extensions:

A diagram of a product

Description automatically generated

Use-Case: Pet Foster logs in or register, create/update a product

Level: Seller Goal

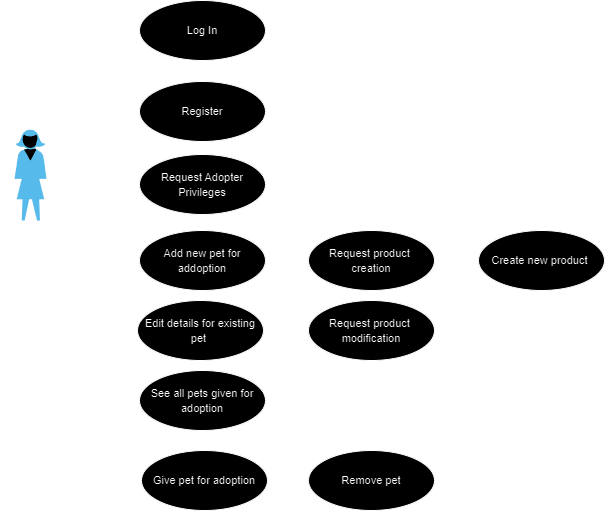
Primary Actor: Seller

Main success scenario:

* In order to get priviledges of Pet Foster, a simple customer must request Pet-Foster Roles from Administrator.
* After the prermissions are given to the customer, after log in , the user must choose as what he wants to continue the login process: customer or seller
* As pet-foster, it is redirected to the main page, where it can select the action wanted to perform
* If he chooses to show all the pets giving for adobption by him, it will be redirected to a page listing all the pets.
* If it chooses to create / update a new pet, it’s going to see a pop up with specific fields needed.

Extensions:

* The “Customer – Adopter” is able to chat with the Pet-Foster



### UML Use Case Diagrams

## Supplementary Specification

### Non-functional Requirements

[Choose 4 NF for your system, describe them and explain why these NF are suitable for your implementation. ]

### **Security** – The login/registration system within the application via the Spring framework ensures a high level of security. Moreover, by passing passwords through a hashing function instead of storing them in plain text, the application is protected against attackers.

### **Maintainability** – Through implementations that adhere to software design principles (OO – SOLID – DPs – architecture), stability and adaptability to changes are ensured, making the application easy to maintain.

### **Usability** – We want our system to be easy to use. That means making sure everything is clear and simple, like how you move around and what you need to do. We've made sure that each type of user has a clear role and knows what they can do. And when you use our system, it'll feel friendly and easy, no matter how much you know about technology.

### **Reliability** – We want our application to work smoothly without any big problems. So, we've checked it a lot to make sure it won't suddenly stop working. We keep an eye on it all the time to catch any issues quickly. And if something unexpected happens, we've set it up to handle it smoothly, so you won't even notice. This way, you can trust our application to work well every time you use it.

### Design Constraints

[This section needs to indicate 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.]

## Glossary

[Present the noteworthy terms and their definition, format and validation rules if appropriate.]

# Deliverable 2

## Domain Model

[Define the domain model and create the conceptual class diagrams]

## Architectural Design

### Conceptual Architecture

[Define the system’s conceptual architecture; use an architectural style and pattern - highlight its use and motivate your choice.]

### Package Design

[Create a package diagram]

### Component and Deployment Diagram

[Create the component and deployment diagrams.]

# Deliverable 3

## Design Model

### Dynamic Behavior

[Create the interaction diagrams (1 sequence, 1 communication diagrams) for 2 relevant scenarios]

### Class Diagram

[Create the UML class diagram; apply GoF patterns and motivate your choice]

## Data Model

[Create the data model for the system.]

# System Testing

[Describe the testing methides and some test cases.]

# Future Improvements

[Present some features that apply to the application scope.]

# Conclusion

# Bibliography