# **Software Specification**
for
Rental Service in ServiceNow

Prepared by:

MEBNY-D, FH Aachen, Germany
ServiceNow

Aachen, 29. September 2022

Contents

Chapter 1

Introduction

1.1 Disclaimer

This fictional product exists for educational grounds and has no affiliations with similar brands nor is intended for commercial usage.
While MEBNY-D assures to deliver high quality products, we can not guarantee any absence of bugs or defects in the product.
Under no circumstance is MEBNY-D liable for any direct, indirect, specific or consequent damages that is inflicted by the use, misuse, abuse or inability to use the software.
This documentation does not contain all details or variations of the software nor does it provice guidance or maintenance for any possible contingency.

1.2 About this Document and its Readers

This documentation describes what the service is to do and how the service will perform each function.
This document is written as a Software Requirements Specification (SRS) up until chapter 4 so that we can clarify the scope of this product therefore ease the development process. The audiences for this document include the service developers and the service users.
This version provides general descriptions of the service.
The SRS-part of this document is written according to the standards for SRS explained in "IEEE Recommended Practice for Software Requirements Specifications".
The Chapter 4 "The Product as is" is an additional chapter which describes the final product with its main functionalities.
Chapter 5 includes appendices that have an insight into the implementation of this application.

1.3 Definitions, Acronyms and Abbreviations

Term/ Acronym / Abbreviation	Expansion / Description
GUI	Graphical User Interface
SP	Service Portal
SRS	Software Requirements Specification
DB	Database
aPaaS	Application Platform as a Service
GS	GlideSystem
GR	GlideRecord
GA	GlideAjax
GDT	GlideDateTime

1.4 Business Problem

Service Requests for Rental-Systems have been managed by email, spreadsheets and personal contact making them harder to track and report on.
Rental-Users have have gone untracked and not notified properly, so that rented items have been displaced and therefore be of a large cost for companies.

1.5 Desired Outcomes / Scope

- A single location where users can rent and submit items for various types of products.
- Accurate tracking of rental completion and location
- Fewer lost items due to unorganized spreadsheets or similar types of organization
- Automation of as many steps as possible
- Use the Now Application Platform as a Service (aPaaS) to develop a web based Renting platform

Notice that our Rental Service will be a paid service contrary to other rental services without a payment model.

Chapter 2

Overall Description and Application Design

2.1 Product Perspective

2.1.1 User Characteristics / Roles

The users of this application are assigned to one of the following roles:
System Admin, App Admin, App Co-Worker, App Student, App Guest.
Roles with the Prefix "App" are to be defined newly by the application within the application scope.
Users will be assigned groups with those roles to ensure following best practices in role-management within ServiceNow. Every role on the end of the list contains less permission than the one before.
The "App Student" role is referred to as the "base" role.

2.1.2 System Interface

The system runs on a ServiceNow-personal-developer-instance in the next-to-last version "San Diego".

2.1.3 User Interfaces

The application GUI provides menus, toolbars, button, panels, containers implemented with the ServiceNow Service Portal (SP), allowing for easy control and better overview for the user.

2.2 Expected Product functions

The service performs the following functions. The functions depend on the user's roles and permission, as explained in the user characteristics.

2.2.1 Login / Request Approval

This function allows users with inappropriate permissions (guest) to be approved by an administrator to receive access to other functionalities.

2.2.2 Create/Read/Update/Delete Rental Items

This function allows the user with appropriate permissions (App Co-Worker or App Admin) to create/read-/update/delete rental items. Other users which have not created an item can only perform the read-operation on them.

2.2.3 Rent Items

This functions allows the users with appropriate permission (base) to request to rent items that have been submitted. The user can only rent products that are not his own.

2.2.4 View Items/Rentals

This function allows all users to view list of all items rented from him/her and items rented by him/her.

2.2.5 Adminpanel

This function allows the administrator to generate reports, view all records and approve new users.

2.2.6 Notifications

This function allows all users to be notified upon various occasions as in completion of renting or rental acknowledgement.

2.3 Assumptions and Dependencies

The personal information - ID, title, username, password is previously determined for all the users of the application. Access to the system is restricted to the pre-assigned users.

2.4 Data model

The Application data model

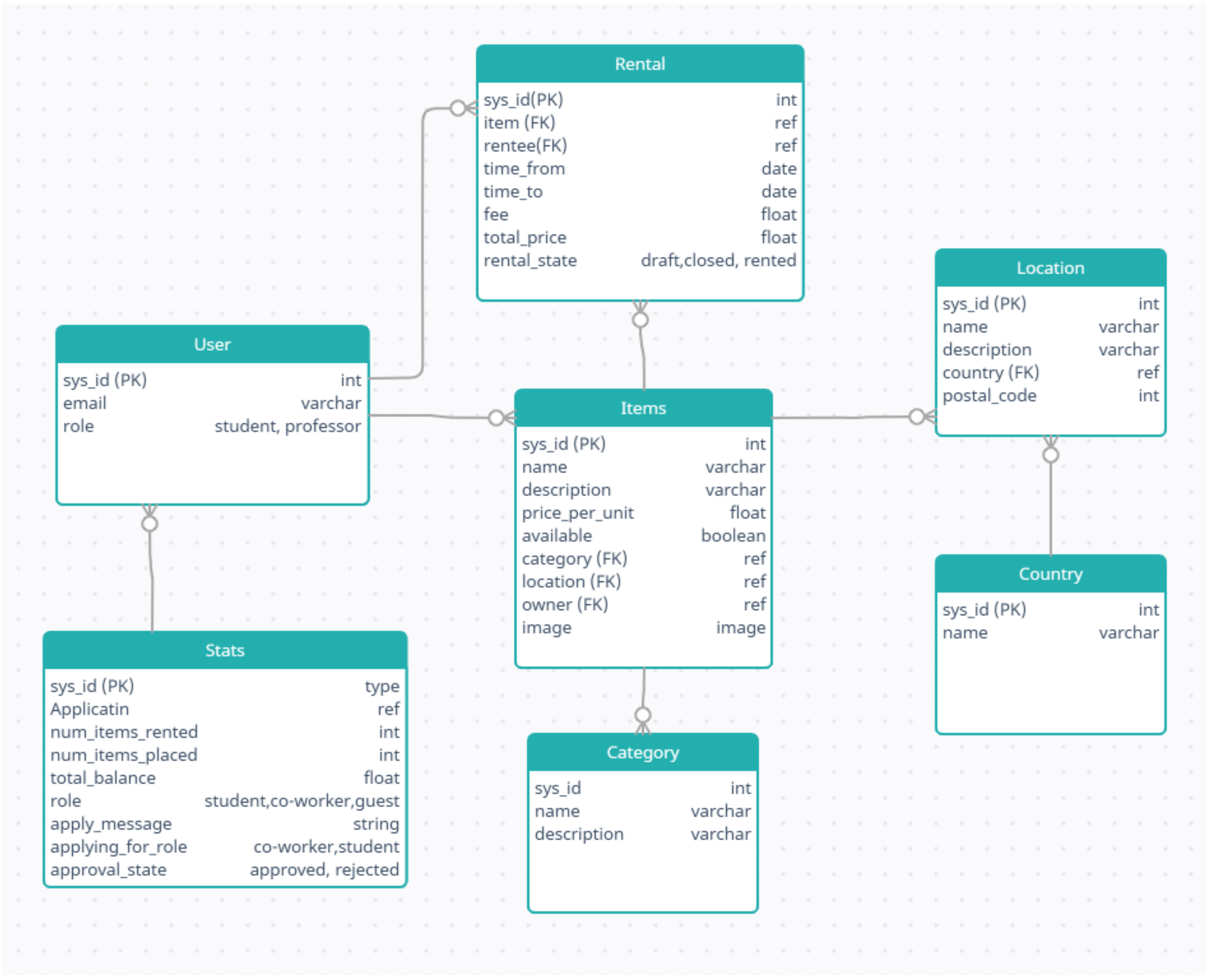


Figure 2.1: Data model

2.5 Processes

Application processes

Process	User	Description
Obtain Approval for Registration	Administrator	After request submitted, look up request Administrator. Administrator approves or rejects a proposal. On rejection request is closed. On acceptance requester's role is elevate to desired one
Obtain Approval for Rental	Administrator	After request submitted, look up Rental Approver. Approver approves or rejects a proposal. On rejection Rental is closed. On acceptance Applicant's Rented Items include new Rental
Notify Overdue Rentals (email)	Rental owner	Check daily for a list of almost overdue Rentals (1 day). Send reminder email to each Rental owner. Continue to send an email until the Rental record ist deleted
Update User Balance on delayed return	Rental owner	Check daily fo a list of overdue Rentals. Send reminder email to each Rental owner. Subtract Rental-Item Fee from Rental owner account balance

2.6 Service Blueprint

Application Service Blueprint

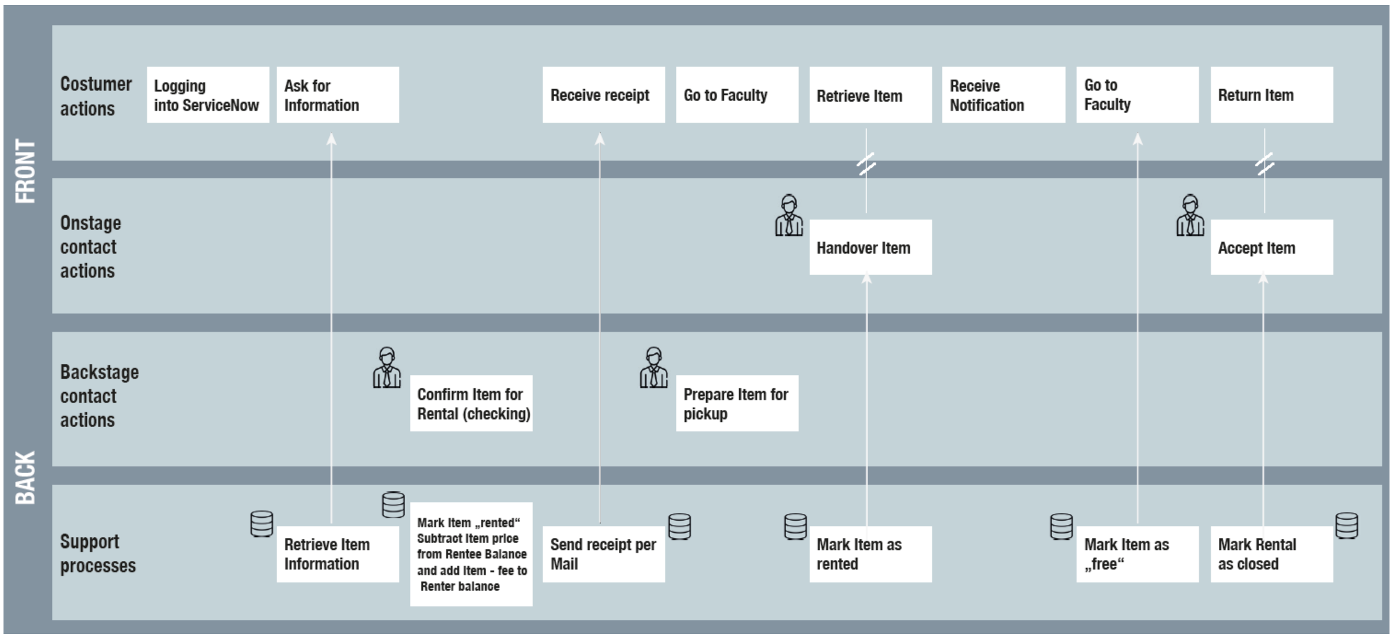


Figure 2.2: Service Blueprint for renting Items

Chapter 3

Specific Requirements

3.1 Functional Requirements

These specific functional requirements will be presented as User story and Use case to guide the developers during their implementation.

3.1.1 Add new Items

Title: Add Item to rent	**Priority: high**
As a general user I want to submit a new item so that other users can rent it	

Actors: Users with appropriate permissions
Preconditions:

1. Authenticated session
2. App Co-Worker or App Admin role

Trigger: Clicking the menu "Add new Item"
Procedure:

1. User on the page clicks on menu "Add new Item" and system transfers in to page "Add new Item"
2. System opens form
3. User provides following information: name, description, price, categories, location
4. User presses "Submit"
5. System validates User Input

Postconditions:

1. New Item is added to DB

Exceptions:

1. Database error
2. Required fields are empty

3.1.2 Rent Items

Title: Rent Items	**Priority: high**
As a general user I want to rent a new item so that i can use it for a limited time for a small fee	

Actors: Users with appropriate permissions
Preconditions:

1. Authenticated session
2. Navigated to the item that is to be rented
3. Base role

Trigger: Clicking the menu "Rent Item"
Procedure:

1. User one the page clicks on menu "Rent Item" and system transfers in to page "Rent Item"
2. System opens form
3. User provides following information: End-date,
4. User presses "Submit"
5. System calculates renting price
6. System validates User Credit is higher or equal to renting fee
7. System waits for Administrator or Renter to approve Rentee

Postconditions (if approved):

1. New Rental Record is added to DB
2. Process for approval has started
3. User Credit has subtracted renting price

Exceptions:

1. Database error
2. Required date field is empty
3. Date is before or equal to request date

3.1.3 Receive a notification when rented item is due

Title: Receive notification a day before item is due	**Priority: high**
As a general user I want to receive a notification a day before the item is due so that i can return it	

Actors: Users with appropriate permissions
Preconditions:

1. Authenticated session
2. User has rented Item
3. Base role

Trigger: Due date of item is one day before today's date
Procedure:

1. System sends outbound email notification to user

Postconditions:

1. New Email Record in DB

Exceptions:

1. Database error

3.1.4 Receive a receipt when Item is rented

Title: Receive a receipt when Item is rented	**Priority: high**
As a general user I want to receive a notification when my items gets rented or when i rent an item so that i can keep tabs on my rented items / the items i have rented	

Actors: Users with appropriate permissions
Preconditions:

1. Authenticated session
2. User has rented Item OR User's Items get rented
3. Base role

Trigger:
Procedure:

1. System sends outbound Notification two renter and rentee

Postconditions:

1. New Email Record in DB

Exceptions:

1. Database error

3.1.5 Register as a guest / Elevate role

Title: Register as a guest / Elevate role	**Priority: medium**
As a guest user I want to verify my account so that i can do things other users with other roles can (i.e. renting, submitting new items)	

Actors: Users with appropriate permissions
Preconditions:

1. Authenticated session
2. Guest role

Trigger: Clicking the menu "Verify Guest"
Procedure:

1. User one the page clicks on menu "Verify Guest" and system transfers in to page "Verify Guest"
2. System opens form
3. User provides following required information: Appealing for role, Appeal message
4. System validates input
5. System waits for condition: approved or rejected
6. If approved: elevate role of user to desired one, create new Stat Record, notify user
7. If rejected: notify user

Postconditions:

1. If approved: new Stat Record for user

Exceptions:

1. Database error
2. Required fields are empty

3.1.6 Return rented Item

Title: Return rented Item	**Priority: medium**
As a general user who rented an item I want to be able to return it so that other users can rent these items and so that i do not get a fee	

Actors: Users with appropriate permissions
Preconditions:

1. Authenticated session
2. User has rented an Item
3. base role

Trigger: Clicking the menu "Return Items"
Procedure:

1. User one the page clicks on menu "Return Items" and system transfers in to page "Return Items"
2. System opens form
3. User provides following information: Items to rent (multiple)
4. System marks selected items as "not rented" and Rentals as "closed"
5. System sends receipt to user

Postconditions:

1. Rental Records changed "rental state"

Exceptions:

1. Database error

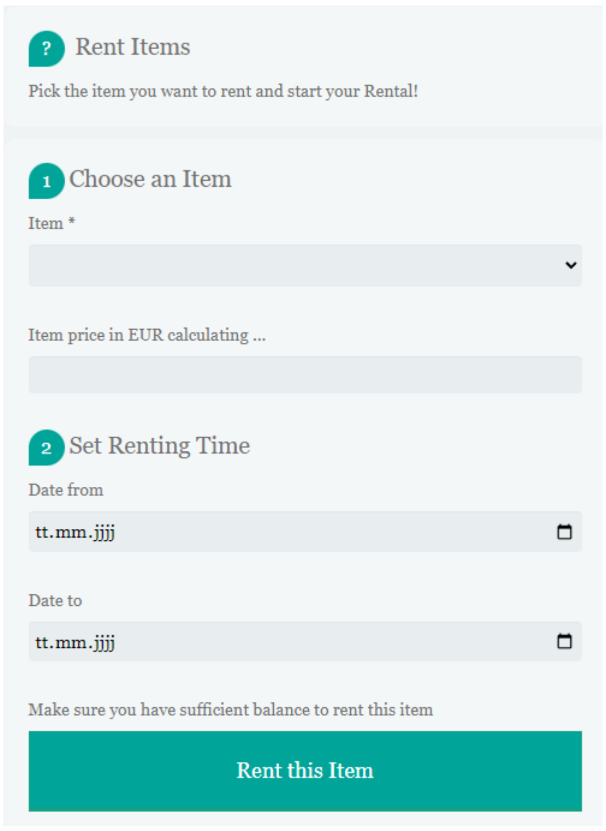
Chapter 4

The Product as is

4.1 Functionalities and Special Cases

4.1.1 Renting Items

A user with a the base role can rent available items if their account balance suffices. The renting time is validated to be a future date. The users account balance needs to match the Rental price.
Items that are rented will be marked visually. Before the User receives the rented Item, another User (App Co Worker, App Admin or Item owner) needs to approve the rental (see following functionality 4.1.2).


? Rent Items

Pick the item you want to rent and start your Rental!

1 Choose an Item

Item *

Item price in EUR calculating ...

2 Set Renting Time

Date from

tt.mm.jjjj

Date to

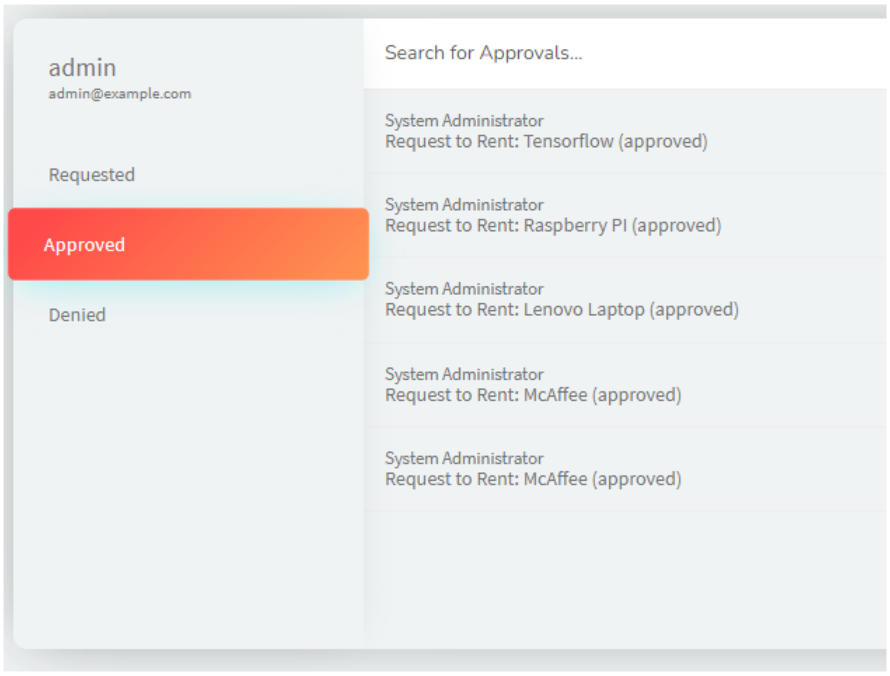
tt.mm.jjjj

Make sure you have sufficient balance to rent this item

Figure 4.1: Renting Form

4.1.2 Approve Rentals

Under Navigation "Item Management" -> "Approve Rentals" an Item owner can manage their Rental proposals.
The Item Owner can approve or reject a proposal.



admin
admin@example.com

Requested
Approved
Denied

Search for Approvals...

System Administrator
Request to Rent: Tensorflow (approved)

System Administrator
Request to Rent: Raspberry PI (approved)

System Administrator
Request to Rent: Lenovo Laptop (approved)

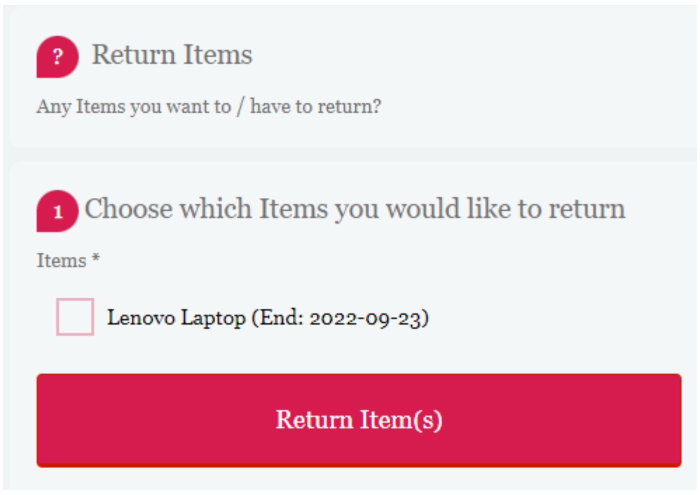
System Administrator
Request to Rent: McAffee (approved)

System Administrator
Request to Rent: McAffee (approved)

Figure 4.2: Rental Request Inbox

4.1.3 Return rented Items

If the user wants to or has to return rented Items the "Return Items-Form allows users to do so.



Return Items
Any Items you want to / have to return?

1 Choose which Items you would like to return
Items *
☐ Lenovo Laptop (End: 2022-09-23)

Figure 4.3: Return Items Form

4.1.4 Recharge Account Balance (Special Case 1)

If a user receives a warning that there is not sufficient balance in his account while trying to rent an item, the user can recharge their balance under Navigation "Account Balance". The user needs to input his payment credentials and then the amount to recharge their account.
For the purpose of this demonstration Payment Credentials can not be validated nor is it saved; just filling in the amount to recharge is enough.

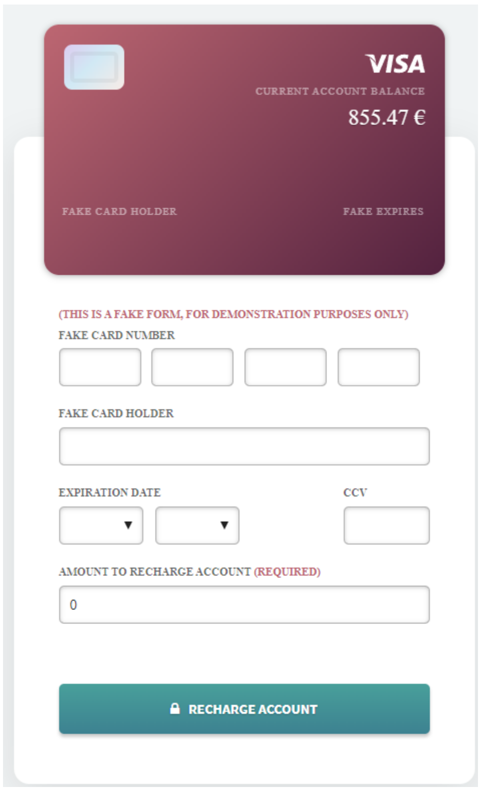
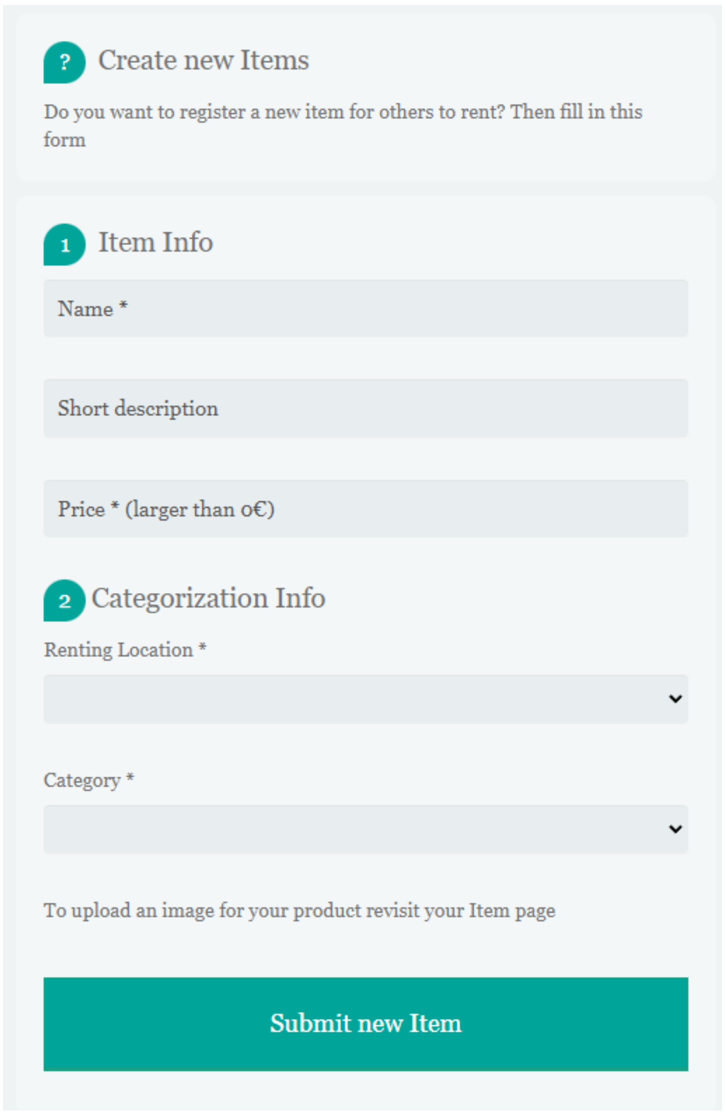


Figure 4.4: Recharge Account Balance Form

4.1.5 Create Item

Under Navigation "Item Management" -> Create new Item a user with the role App Admin or App Co Worker can submit new Items for Rental.
Notice that there is no option to upload an image for the product in this form. Images need to be uploaded once they have been created by this form under Navigation "Item Management"-> "My Items" in the Items section (see section 4.1.6).


Create new Items

Do you want to register a new item for others to rent? Then fill in this form

1 Item Info

Name *

Short description

Price * (larger than 0€)

2 Categorization Info

Renting Location *

Category *

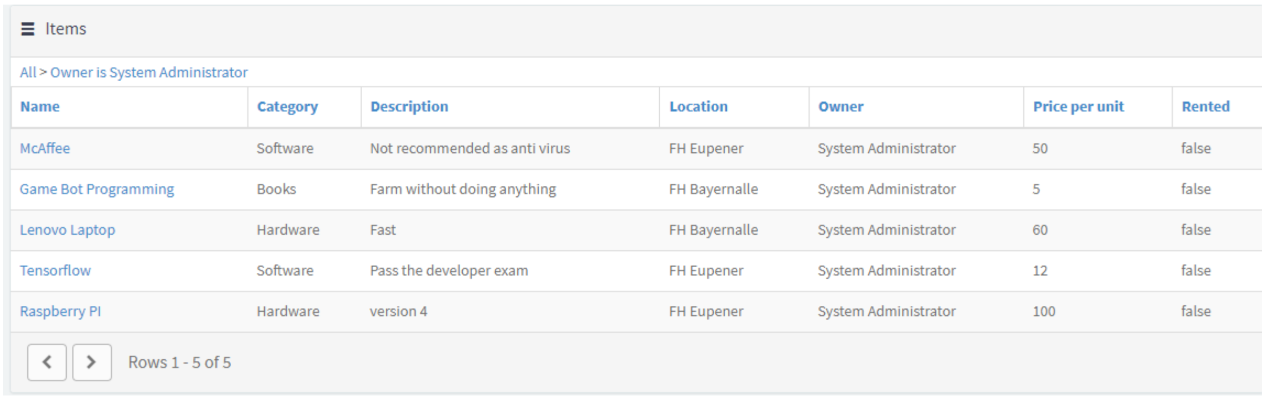
To upload an image for your product revisit your Item page

Submit new Item

Figure 4.5: Create new Items Form

4.1.6 View my Items

Under Navigation "Item Management" -> "My Items" a user with the role App Admin or App Co Worker can view a list of submitted items, a report and a form to delete Items. By clicking on any Record of the Item list the user can edit appropriate fields and upload an image for the product.



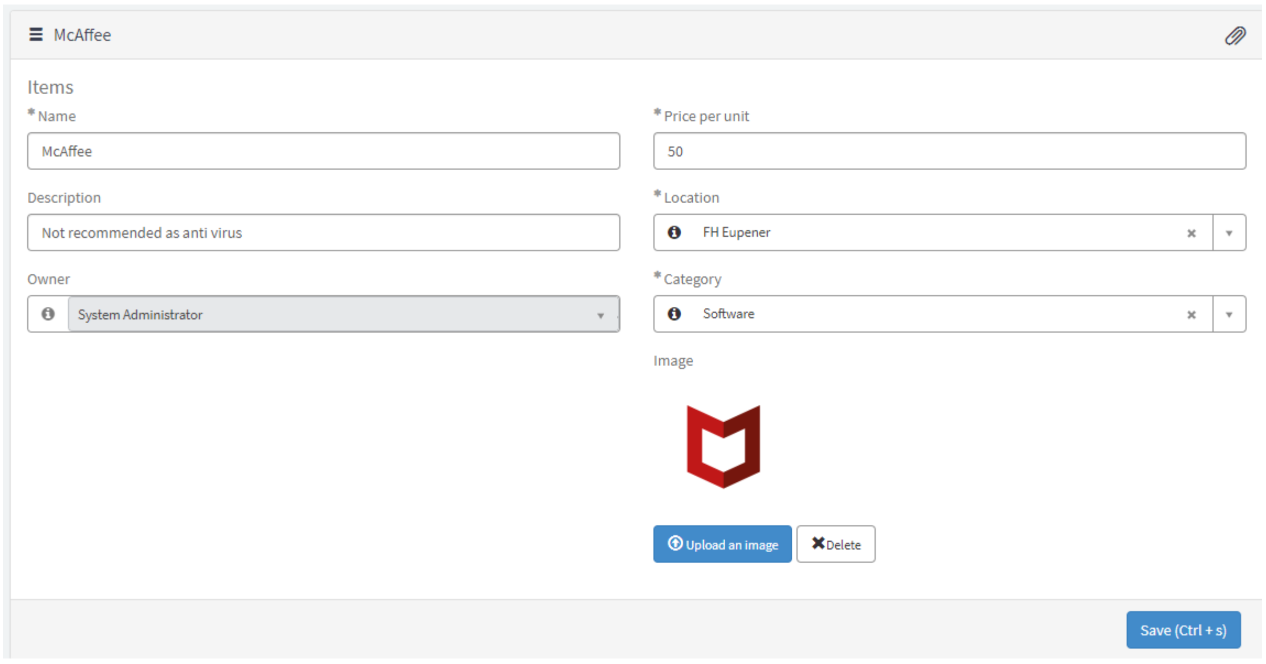
Items

All > Owner is System Administrator

Name	Category	Description	Location	Owner	Price per unit	Rented
McAffee	Software	Not recommended as anti virus	FH Eupener	System Administrator	50	false
Game Bot Programming	Books	Farm without doing anything	FH Bayernalle	System Administrator	5	false
Lenovo Laptop	Hardware	Fast	FH Bayernalle	System Administrator	60	false
Tensorflow	Software	Pass the developer exam	FH Eupener	System Administrator	12	false
Raspberry PI	Hardware	version 4	FH Eupener	System Administrator	100	false

Rows 1 - 5 of 5

Figure 4.6: List of User Items

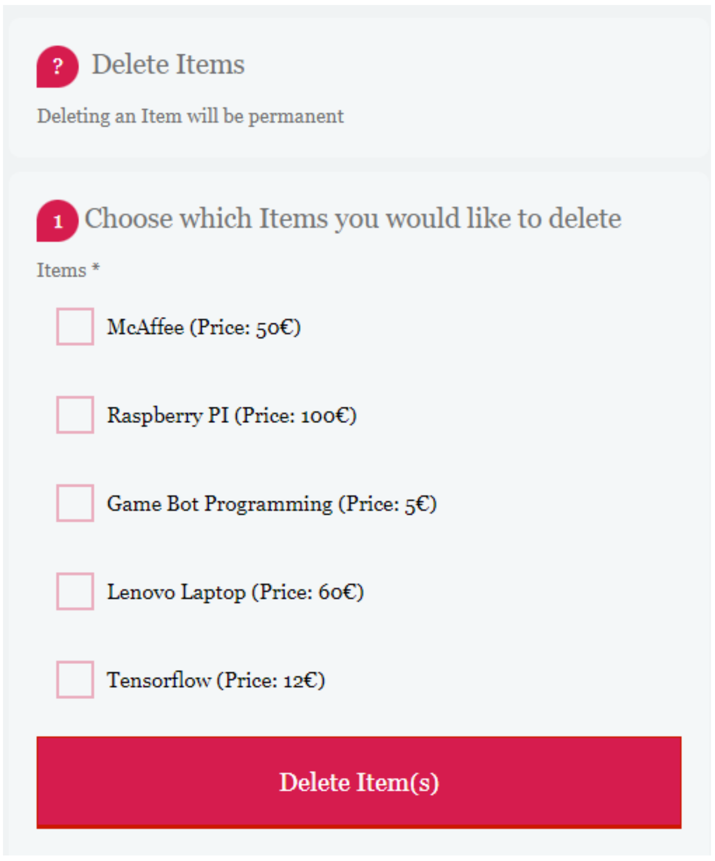


McAffee

Items

* Name: McAffee	* Price per unit: 50
Description: Not recommended as anti virus	* Location: FH Eupener
Owner: System Administrator	* Category: Software
	Image

Figure 4.7: Item Record



? Delete Items

Deleting an Item will be permanent

1 Choose which Items you would like to delete

Items *

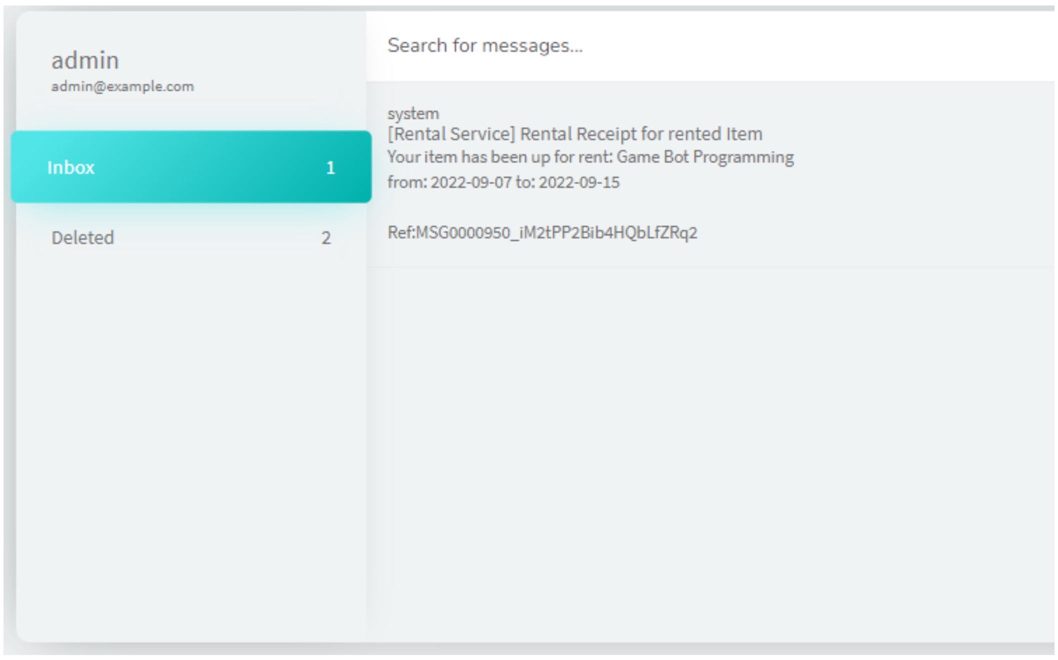
- [ ] McAffee (Price: 50€)
- [ ] Raspberry PI (Price: 100€)
- [ ] Game Bot Programming (Price: 5€)
- [ ] Lenovo Laptop (Price: 60€)
- [ ] Tensorflow (Price: 12€)

Figure 4.8: Delete Items Form

4.1.7 Notification Inbox

Under Navigation -> Bell Symbol the user can view their notifications. The inbox is divided into "Inbox" and "Deleted". By clicking on a notification in the "Inbox" the user can delete the notification and by clicking on a notification in "Deleted" the user can restore that notification.
Users will receive notifications for:

- Item has been rented (Receipt for both parties)
- Reminder for almost expired Rental
- Notice for expired Rental
- Item has been returned (Receipt for both parties)
- Guest Registration Approval



system
[Rental Service] Rental Receipt for rented Item
Your item has been up for rent: Game Bot Programming
from: 2022-09-07 to: 2022-09-15

Ref:MSG0000950_iM2tPP2Bib4HQbLfZRq2

Figure 4.9: Notification Inbox

4.1.8 Admin Admin Dashboard

Under Navigation -> "Admin Management"-> "Admin Dashboard" an App Admin can view several reports and lists.

4.1.9 Admin Pending Approvals

Under Navigation -> "Admin Management"-> "Pending Approvals" an App Admin can approve or reject registrations (see Section 4.1.10).

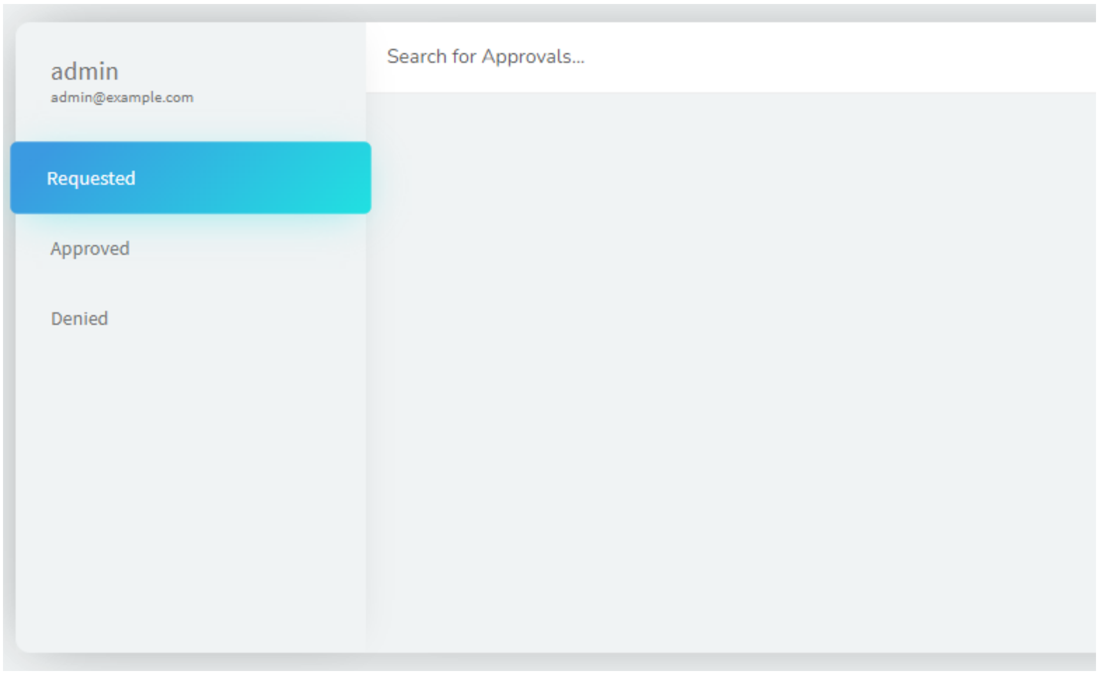


Figure 4.10: Pending Approvals Inbox

4.1.10 Register Guests (Special Case 2)

Users below the base role (guest) can only register in this Application under Navigation -> "Register for Renting". The user has to fill in (fictional) Credentials and the role that they are applying for.
An Admin can approve or reject this Application.

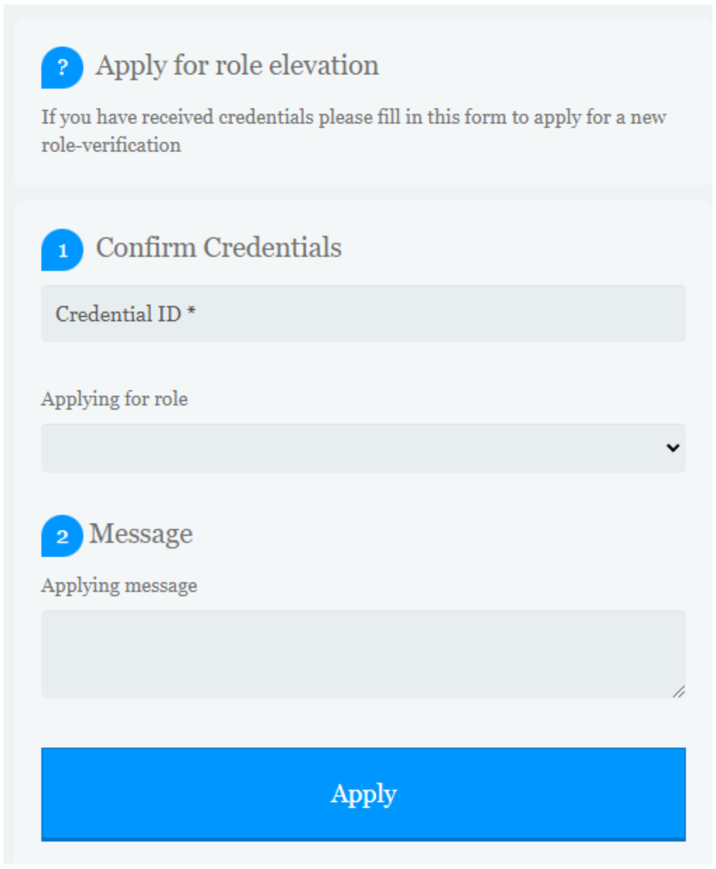
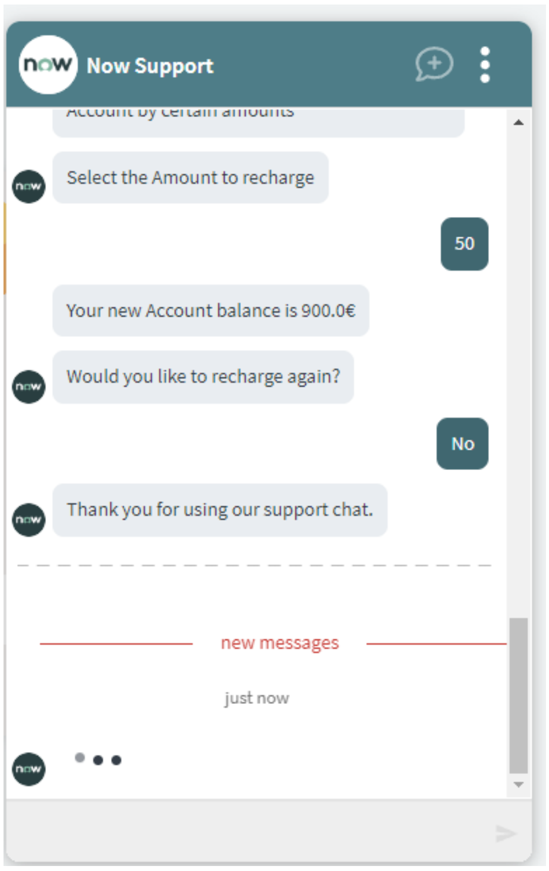


Figure 4.11: Register Guests

4.1.11 Bonus: Virtual Agent

When users need help with fundamental functionalities of the application, the virtual agent can support them. More detailed Information about Virtual agent Topic-Flows can bee seen in Appendix 5.1.18. We use the Virtual agent to support:

- Renting Items
- Returning Items
- Recharging Account Balance


Now Support

Account by certain amounts

Select the Amount to recharge

50

Your new Account balance is 900.0€

Would you like to recharge again?

No

Thank you for using our support chat.

new messages

just now

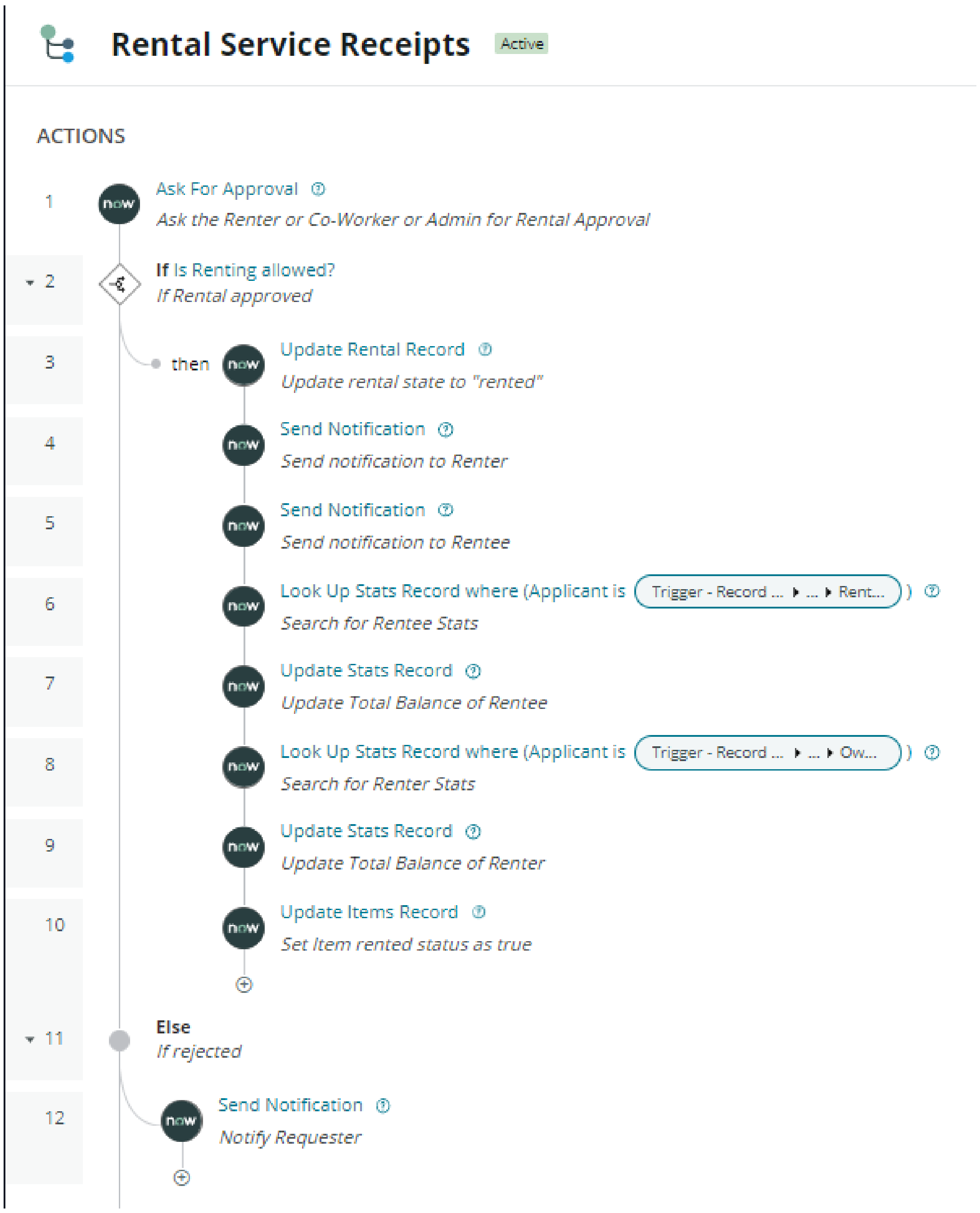
Figure 4.12: Virtual agent

4.2 Workflows

Application Workflows

4.2.1 Workflow for Rental Receipts

This workflows includes, Asking for Approval, updating total balances and sending Notifications.



Rental Service Receipts Active

ACTIONS

1 Ask For Approval
Ask the Renter or Co-Worker or Admin for Rental Approval

2 **If** Is Renting allowed?
If Rental approved

3 then Update Rental Record
Update rental state to "rented"

4 Send Notification
Send notification to Renter

5 Send Notification
Send notification to Rentee

6 Look Up Stats Record where (Applicant is Trigger - Record ... ▸ ... ▸ Rent...)
Search for Rentee Stats

7 Update Stats Record
Update Total Balance of Rentee

8 Look Up Stats Record where (Applicant is Trigger - Record ... ▸ ... ▸ Ow...)
Search for Renter Stats

9 Update Stats Record
Update Total Balance of Renter

10 Update Items Record
Set Item rented status as true

11 **Else**
If rejected

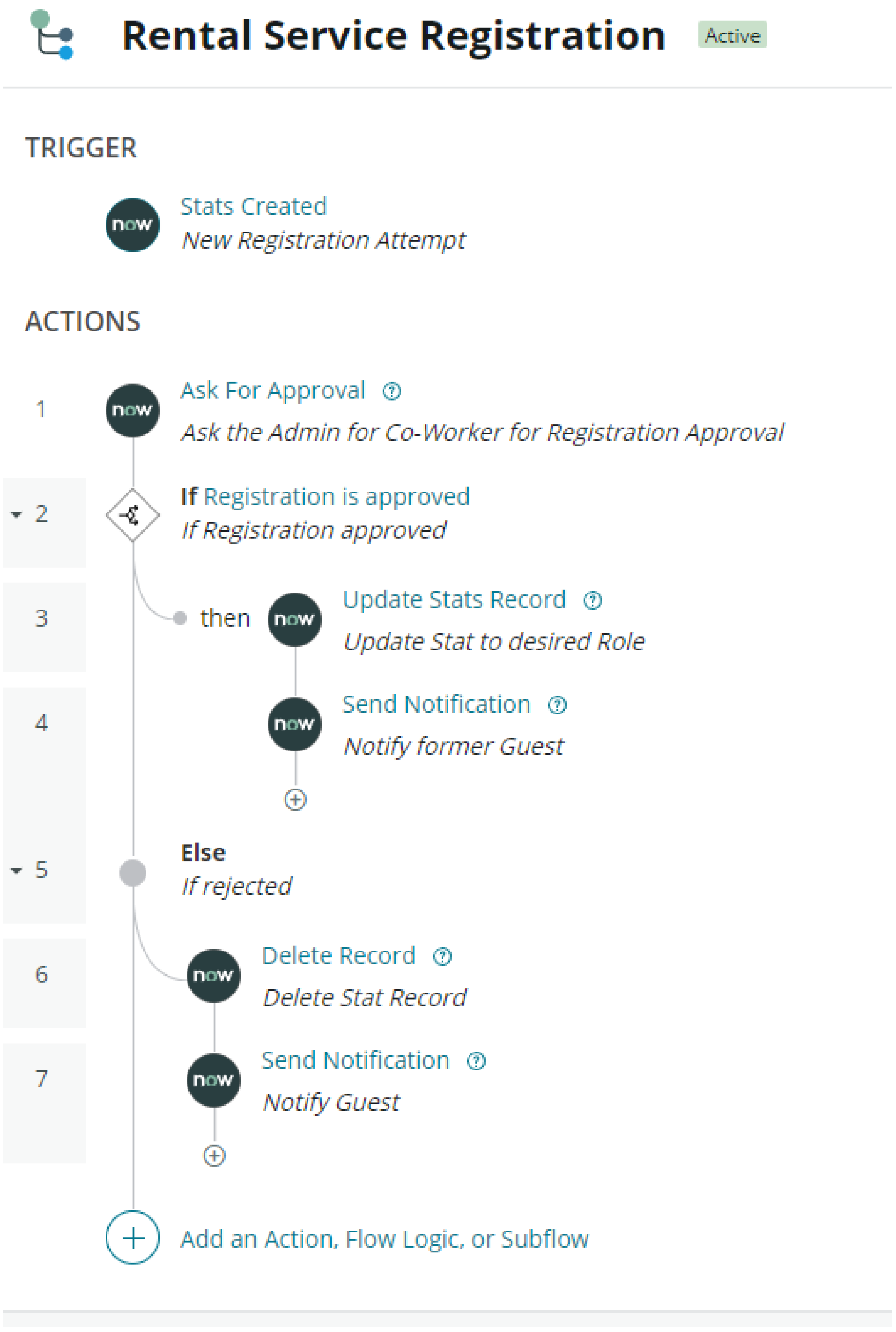
12 Send Notification
Notify Requester

Figure 4.13: Workflow Rental Service Receipts

Quick Notes about Balance Calculations:
When an Item is created the field "Fee" is set to a 19% value of the original price. This fee is important for further calculations.
When a user rents an Item they become a Rentee.
The total price of the Item will be subtracted from the Rentees balance.
The total price of the Item minus the Fee will be added to the Renters balance.
That way the 19th percentile of the price will go to corporate or the organization that offers this application.
If the Rentee does not return the Item on the agreed upon date the Item's Fee is subtracted from their account additionally every day the Rentee exceeds the due date.
This Payment System aims to encourage correct behaviour.

4.2.2 Workflow for Rental Registration

This workflows includes, Asking for Approval, updating stats and sending Notifications.



Rental Service Registration Active

TRIGGER

Stats Created
New Registration Attempt

ACTIONS

1 Ask For Approval
Ask the Admin for Co-Worker for Registration Approval

2 **If** Registration is approved
If Registration approved

3 then Update Stats Record
Update Stat to desired Role

4 Send Notification
Notify former Guest

5 **Else**
If rejected

6 Delete Record
Delete Stat Record

7 Send Notification
Notify Guest

Add an Action, Flow Logic, or Subflow

Figure 4.14: Workflow Rental Registration

Appendices

Chapter 5

Appendices

5.1 ServiceNow Functionalities in our application

5.1.1 UI Policies and UI Policy Actions

When submitting a new Item certain form fields (total price, renter) are made read-only and mandatory.

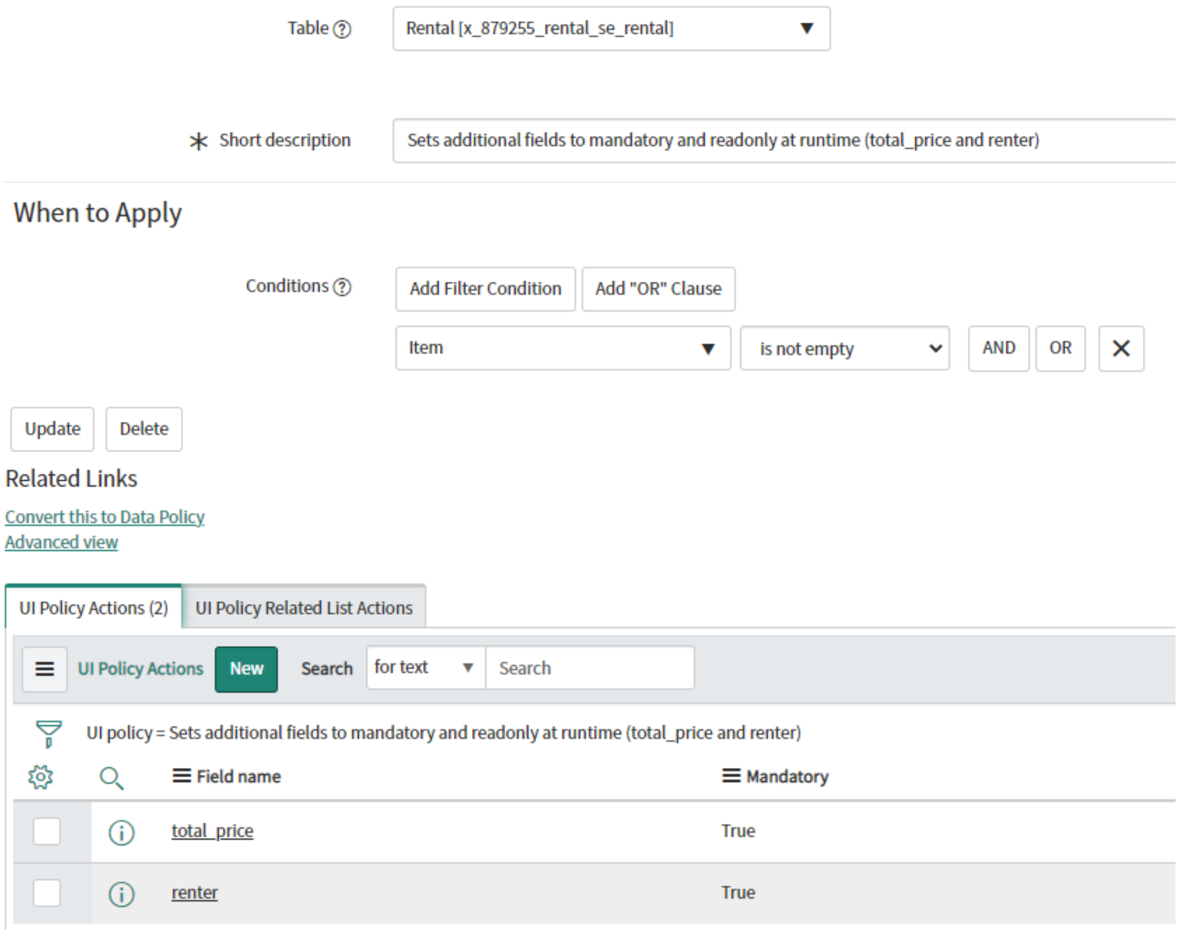


Table	Rental [x_879255_rental_se_rental]
* Short description	Sets additional fields to mandatory and readonly at runtime (total_price and renter)

When to Apply

Conditions	Item is not empty

Related Links

Convert this to Data Policy
Advanced view

UI Policy Actions (2)

UI policy = Sets additional fields to mandatory and readonly at runtime (total_price and renter)

Field name	Mandatory
total_price	True
renter	True

Figure 5.1: Item UI Policy

5.1.2 Data Policies

Since Data Policies enforce data consistency by setting mandatory and read-only field attributes, unlike UI Policies, Data Policies execute server-side. UI Policy logic only applies to data entered in a form. Data Policy logic executes regardless of how a record changes.
Therefore we integrated several Data policies constraints into the application to enforce consistency on the server-side.

5.1.3 Modules

Modules are a way of navigation within ServiceNow. We used modules to test functionalities for the SP.

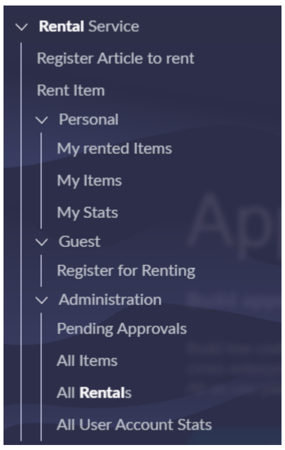
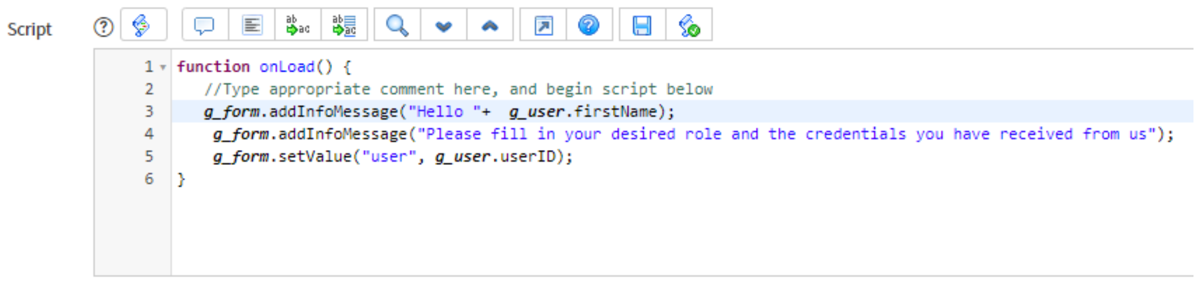


Figure 5.2: Modules

5.1.4 Client-side Scripts

Client-side scripts execute within a user's browser and are used to manage forms and form fields.
In this application client scripts were used for:

- Dynamically populating user fields
- Info-messages on forms



```
function onLoad() {
   //Type appropriate comment here, and begin script below
   g_form.addInfoMessage("Hello "+  g_user.firstName);
    g_form.addInfoMessage("Please fill in your desired role and the credentials you have received from us");
    g_form.setValue("user", g_user.userID);
}
```

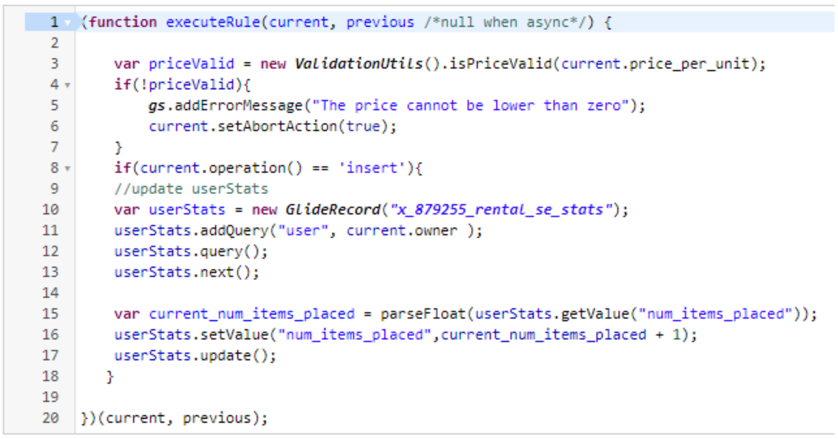
Figure 5.3: Example Client Script

5.1.5 GlideForm, GlideUser

As depicted in Figure 5.3 GlideForm and GlideUser are two client-side API classes mainly used in client scripts.

5.1.6 Business Rules

Business Rules are server-side logic that execute when database records are queried, updated, inserted, or deleted.
We mainly use Business Rules to process form input before a new record is inserted.
For instance before processing a new item we validate its price and update the user's stats.



```
(function executeRule(current, previous /*null when async*/) {

    var priceValid = new ValidationUtils().isPriceValid(current.price_per_unit);
    if(!priceValid){
        gs.addErrorMessage("The price cannot be lower than zero");
        current.setAbortAction(true);
    }
    if(current.operation() == 'insert'){
    //update userStats
    var userStats = new GlideRecord("x_879255_rental_se_stats");
    userStats.addQuery("user", current.owner );
    userStats.query();
    userStats.next();

    var current_num_items_placed = parseFloat(userStats.getValue("num_items_placed"));
    userStats.setValue("num_items_placed",current_num_items_placed + 1);
    userStats.update();
   }

})(current, previous);
```

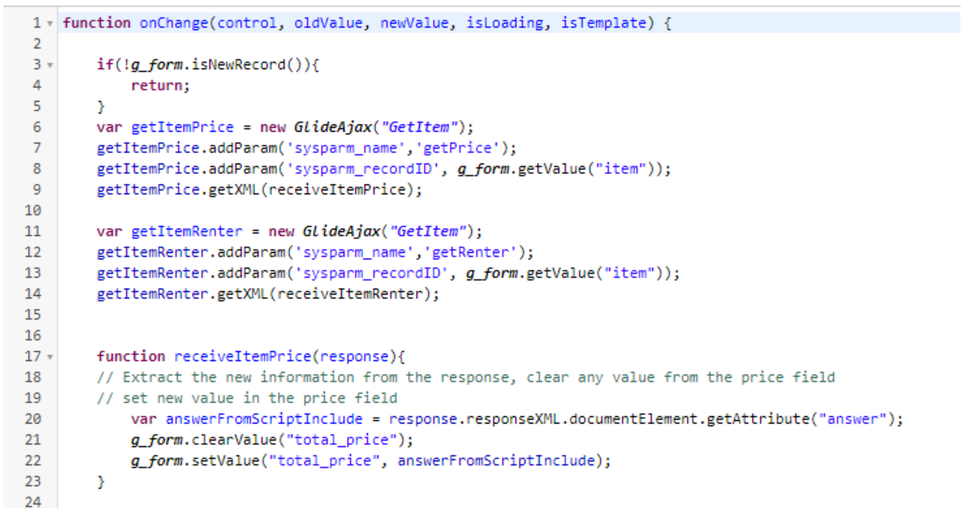
Figure 5.4: Business Rule to process new Items before insert and update

5.1.7 GlideSystem, GlideRecord, GlideDateTime, GlideAjax

Server-side scripts execute on the ServiceNow server or database. Examples of things our server-side scripts can do include:

- Update record fields when a database query runs
- Generate and respond to events
- Compare two dates to determine which comes first chronologically
- Make client callable Utility functions

Server-side API classes like GlideSystem (GS), GlideRecord (GR) and GlideDateTime (GDT) are utilized for those functions.



```
function onChange(control, oldValue, newValue, isLoading, isTemplate) {

    if(!g_form.isNewRecord()){
        return;
    }
    var getItemPrice = new GlideAjax("GetItem");
    getItemPrice.addParam('sysparm_name','getPrice');
    getItemPrice.addParam('sysparm_recordID', g_form.getValue("item"));
    getItemPrice.getXML(receiveItemPrice);

    var getItemRenter = new GlideAjax("GetItem");
    getItemRenter.addParam('sysparm_name','getRenter');
    getItemRenter.addParam('sysparm_recordID', g_form.getValue("item"));
    getItemRenter.getXML(receiveItemRenter);


    function receiveItemPrice(response){
    // Extract the new information from the response, clear any value from the price field
    // set new value in the price field
        var answerFromScriptInclude = response.responseXML.documentElement.getAttribute("answer");
        g_form.clearValue("total_price");
        g_form.setValue("total_price", answerFromScriptInclude);
    }

```

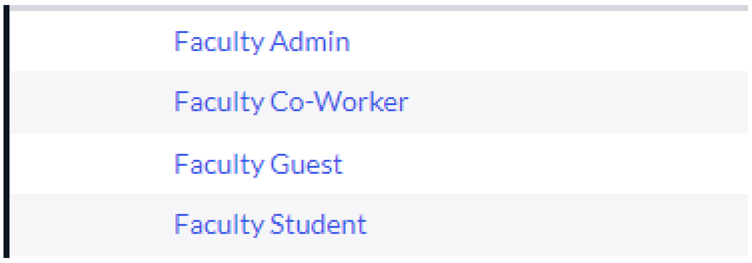
Figure 5.5: GlideAjax (GA) call in a client script

5.1.8 Access Control Lists

Rules for access control lists (ACLs) restrict access to data by requiring users to pass a set of requirements before they can interact with it.
There are several ACLs in this Application that restricts access to tables Item, Rental, Stat, Location and Country.

5.1.9 Role and Groups

This application has four different groups with four roles assigned to them.
That way users just need to be added to groups instead getting a new role which can change over time.

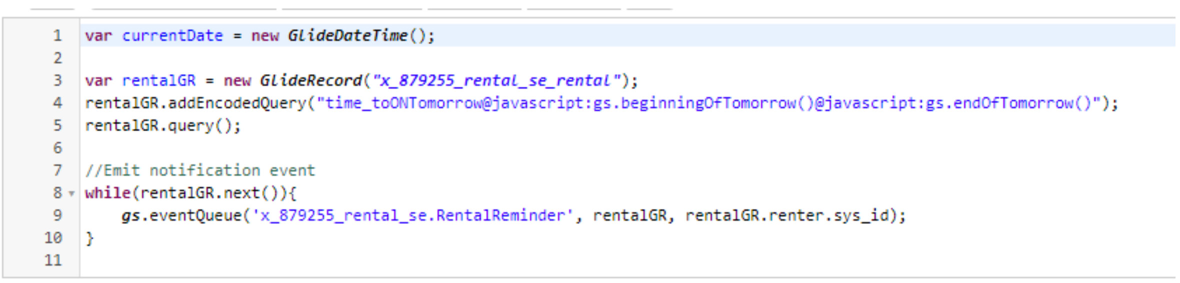


Faculty Admin
Faculty Co-Worker
Faculty Guest
Faculty Student

Figure 5.6: Groups

5.1.10 Scheduled Scripts

Scheduled Script Executions are automated server-side script logic that executes at a specific time or on a recurring basis. This application uses a scheduled script to check if a rental expires so that an outbound email notification can be send.



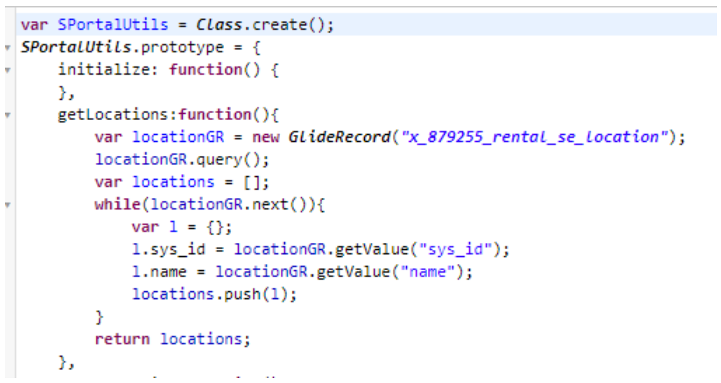
```
var currentDate = new GlideDateTime();

var rentalGR = new GlideRecord("x_879255_rental_se_rental");
rentalGR.addEncodedQuery("time_toONTomorrow@javascript:gs.beginningOfTomorrow()@javascript:gs.endOfTomorrow()");
rentalGR.query();

//Emit notification event
while(rentalGR.next()){
    gs.eventQueue('x_879255_rental_se.RentalReminder', rentalGR, rentalGR.renter.sys_id);
}

```

Figure 5.7: Daily Script to check for Rental expiry

5.1.11 Script Includes

Script Includes are a way to reuse utility functions in other scripts.
There are three different Script Includes in this application.

- SP utilities
- Validation utilities
- Fetching Item-data for GA call



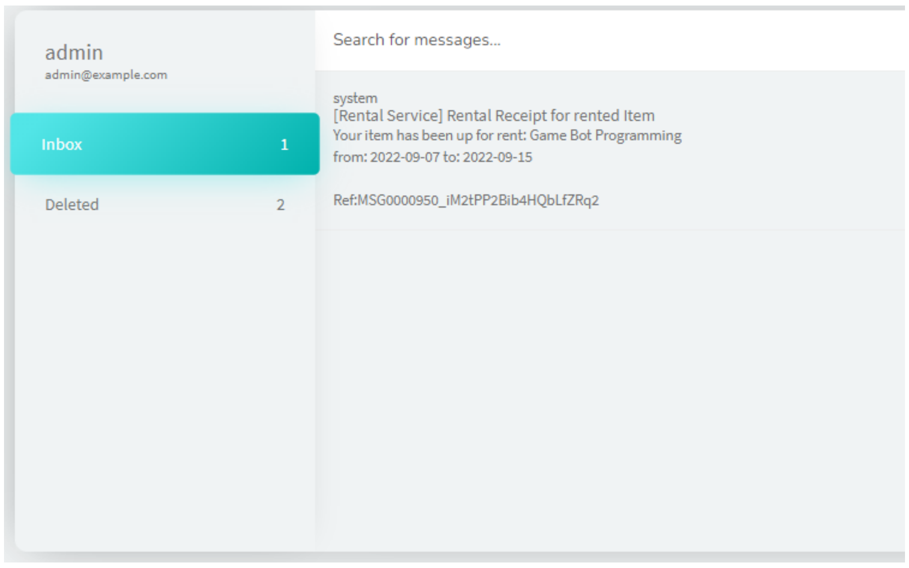
```
var SPortalUtils = Class.create();
SPortalUtils.prototype = {
    initialize: function() {
    },
    getLocations:function(){
        var locationGR = new GlideRecord("x_879255_rental_se_location");
        locationGR.query();
        var locations = [];
        while(locationGR.next()){
            var l = {};
            l.sys_id = locationGR.getValue("sys_id");
            l.name = locationGR.getValue("name");
            locations.push(l);
        }
        return locations;
    },
```

Figure 5.8: SP Portal Script Include

5.1.12 Notifications

Notifications send outbound email to one or more recipients in response to specific activities in ServiceNow.
Notifications in this application is sent when:

- Approval status when registering for a role elevation
- Reminder about Rental
- Notice about Rental expiry
- Rental receipts for Renter and Rentee



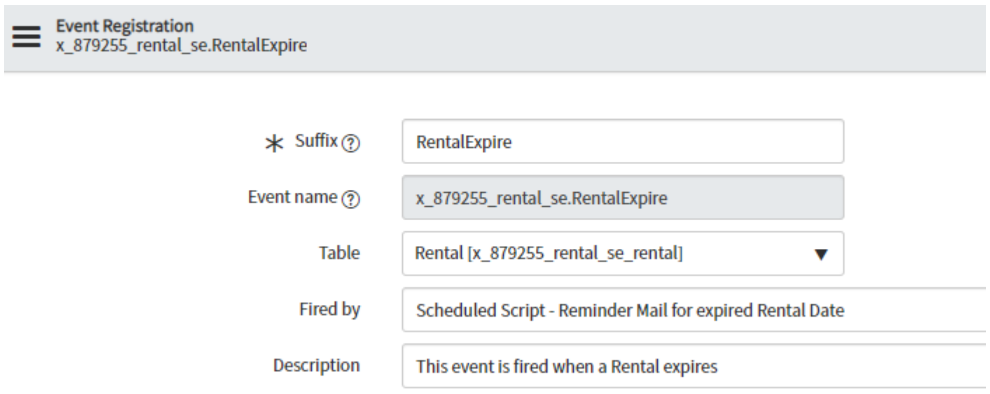
system
[Rental Service] Rental Receipt for rented Item
Your item has been up for rent: Game Bot Programming
from: 2022-09-07 to: 2022-09-15

Ref:MSG0000950_iM2tPP2Bib4HQbLfZRq2

Figure 5.9: Notifications in the Portal

5.1.13 Event Registration

Registered Events in this application trigger certain notifications.



Event Registration
x_879255_rental_se.RentalExpire

* Suffix	RentalExpire
Event name	x_879255_rental_se.RentalExpire
Table	Rental [x_879255_rental_se_rental]
Fired by	Scheduled Script - Reminder Mail for expired Rental Date
Description	This event is fired when a Rental expires

Figure 5.10: Event for Rental expiry

5.1.14 Reports

Reports allow Administrators to have a visual view of data. This applications features reports with the tables:

- Items
- Stats
- Rentals

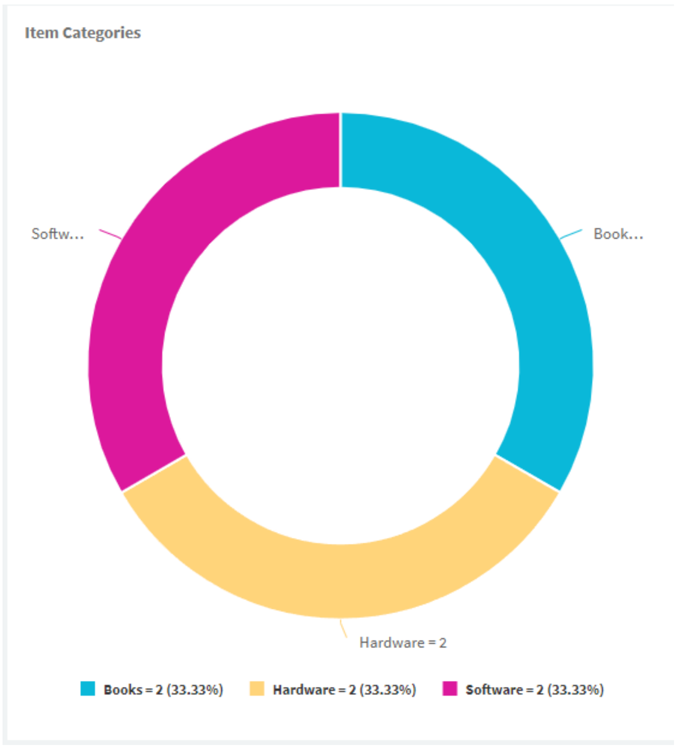


Figure 5.11: Report about Item Categories

5.1.15 Custom Widgets

As seen in figure 5.9 custom widgets enable a higher level of flexibility in style and functionality in the SP. This applications uses Widgets for:

- Several Forms (Item, Rental etc.)
- Displaying Items
- Mail Inbox
- Approval Inbox

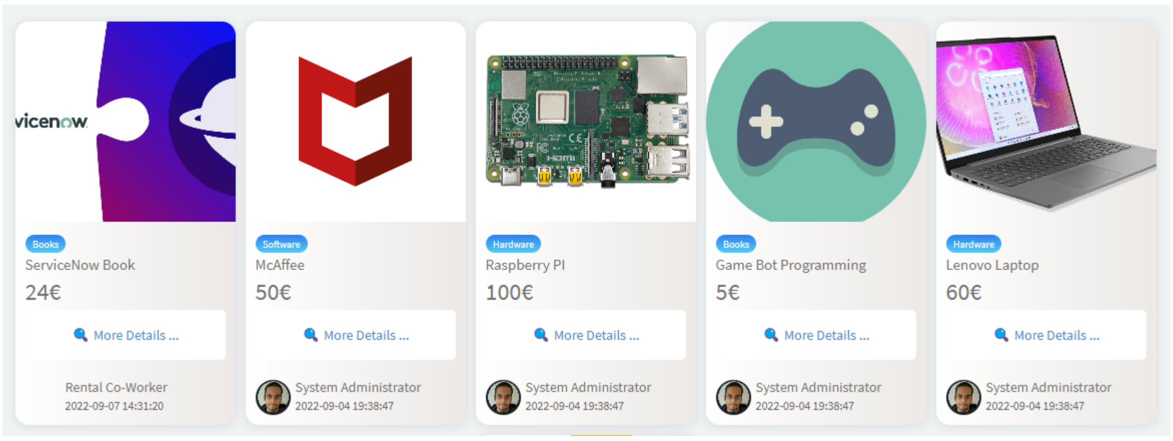
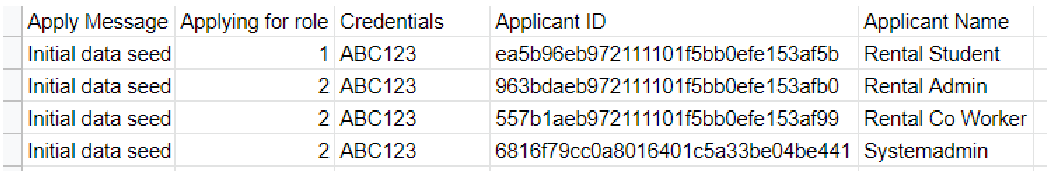


Figure 5.12: Custom Widget for displaying Products

5.1.16 Data Import

Seeding application in ServiceNow is made easy by the data import feature using transform maps. We used several transform maps to import .csv-data for several types of data:

- Items
- Stats
- Locations
- Categories



Apply Message	Applying for role	Credentials	Applicant ID	Applicant Name
Initial data seed	1	ABC123	ea5b96eb972111101f5bb0efe153af5b	Rental Student
Initial data seed	2	ABC123	963bdaeb972111101f5bb0efe153afb0	Rental Admin
Initial data seed	2	ABC123	557b1aeb972111101f5bb0efe153af99	Rental Co Worker
Initial data seed	2	ABC123	6816f79cc0a8016401c5a33be04be441	Systemadmin

Figure 5.13: Seed for Stats

5.1.17 Guided Tours

Guided Tours can be used to demonstrate how to use a feature.
We use guided Tours in the SP.

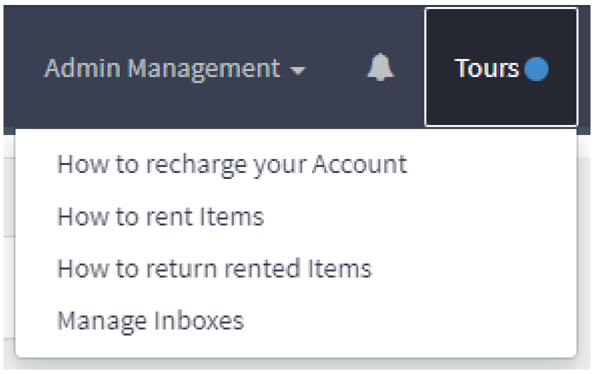


Figure 5.14: Guided Tours of MEBNY-D

5.1.18 Virtual Agent Topic Flows

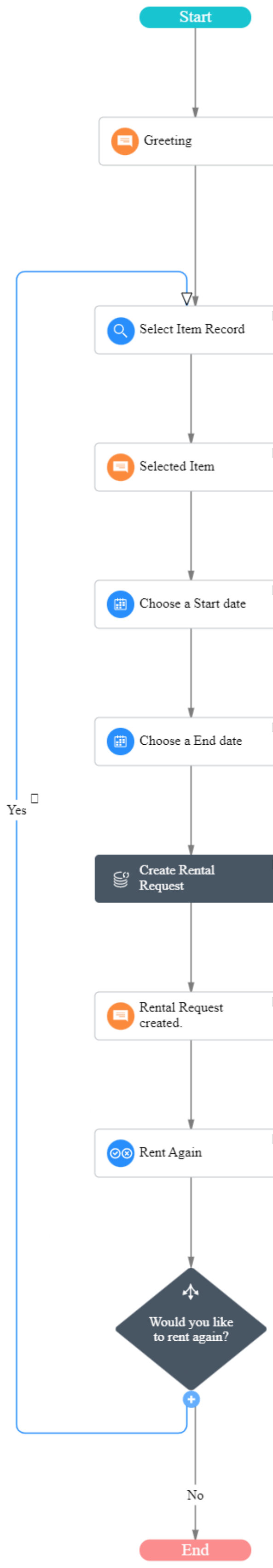


Figure 5.15: Topic: Rent Item

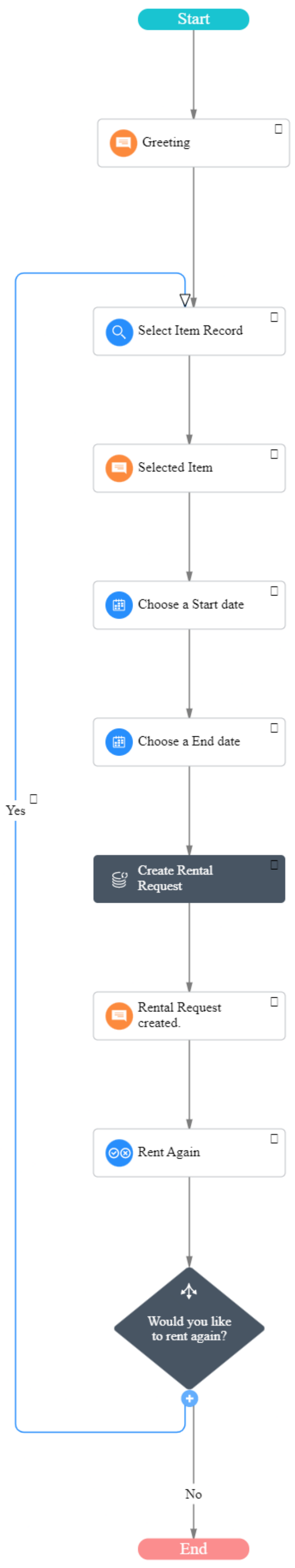


Figure 5.16: Topic: Return Item

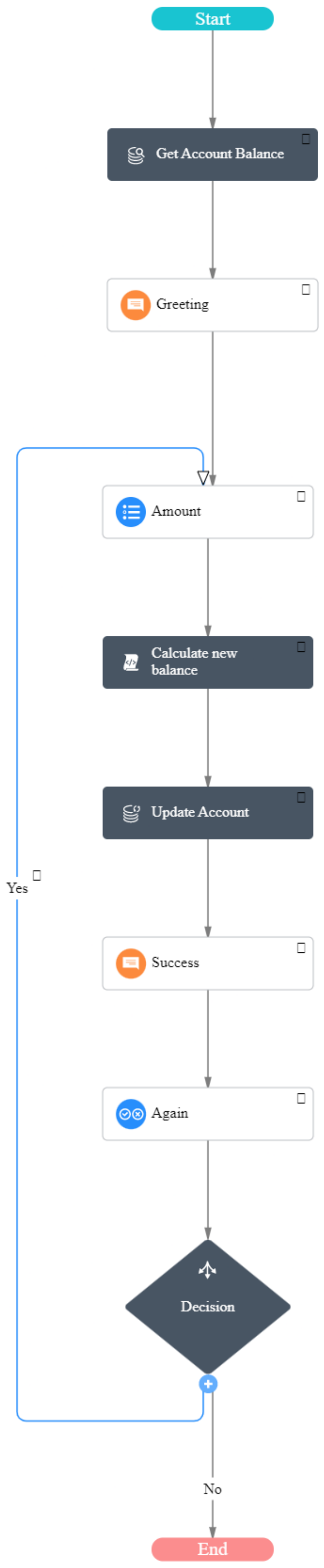


Figure 5.17: Topic: Recharge Account