Room Service MVP Report

Team 4:
Valeria Yurinskaya,
Vladimir Solovev,
Marvin Lopez,
Vladimir Semenov

Introduction

To perform the final MVP version of the Room Service our team looked up the use cases in the requirements document, found non-implemented ones and changed some of them for our goals and planned to implement them.

Use cases

The current MVP didn't implement all of the use cases. There is the list of skipped use cases, which we implemented in this iteration:
- Cancel order (Delete Order (User/Manager part))
- Deactivate order
- Accept Order
- Decline Order
- Send notification

Also during this iteration, we decided to modify some use cases. There is a list of modified use cases:
- Create Account - we decided to delete the option of account's confirmation by email.
- Cancel order - we decided that the user can directly delete his order without the manager's approving.
- Send notification - we decided to send an email with notification to the user each time when the manager changes a user's order (status, etc).

There is the list of removed use cases:
- Pay order and Control Payments - these use cases don't connect with developed software, payment process provides directly between manager and user in real life.
- Accept cancellation - we don't need to have this use case because we already modified the «Cancel order» use case, in which the user deletes his order without the manager's approving.
- Assign Cleaning, Cleaning Done, Cancel Cleaning and Perform Cleaning - work with The Personnel was removed.

Below are use cases which were modified:

Create Account

Use Case Name	Create Account
Actors	User
Pre-conditions	• The User is willing to use our services
Flow of events	1. The User opens the platform and selects registration. 2. The User fills the required information like first name, last name, email. 3. The User selects a strong password ensured by the System. 4. The User registered in the system
Post-conditions	The User got an account to login with.

Cancel order -> Delete Order (User part)

Use Case Name	Cancel order
Actors	User
Pre-conditions	• The User has an account registered in the system. • The User has an open order. • The User account has been authenticated.
Flow of events	1. The User clicks on the action button. 2. The User chooses the delete button. 3. The User confirms order deleting.
Post-conditions	Order deleted by the User

Cancel order -> Delete Order (Manager part)

Use Case Name	Cancel order
Actors	Manager
Pre-conditions	● The Manager account has been authenticated.
Flow of events	1. The Manager clicks on the action button. 2. The Manager chooses the delete button. 3. The Manager confirms order deleting.
Post-conditions	Order deleted by the Manager

Send notification

Use Case Name	Send notification
Actors	Manager, User
Pre-conditions	● The User has an account registered in the system. ● The User account has been authenticated. ● The Manager account has been authenticated.
Flow of events	1. The Manager opens one order. 2. The Manager selects the action (Edit/Activate/Approve/Deactivate/ Delete/ Decline/Cancel). 3. The system sends to the User a email with a notification about action. 4. The User receives a notification.
Post-conditions	The User received the notification created by the Manager.

Below are use cases which were implemented:

Accept Order

Use Case Name	Accept Order
Actors	Manager
Pre-conditions	• The Manager account has been authenticated. • The User created an order. • The Manager received the User's order.
Flow of events	1. The Manager checks the order details. 2. The Manager accepts the order.
Post-conditions	The order's status is changed to Accepted.

Decline Order

Use Case Name	Decline Order
Actors	Manager, User
Pre-conditions	• The Manager account has been authenticated. • The User created an order. • The Manager received the User's order.
Flow of events	1. The Manager checks an order details. 2. The Manager initiates declining of the order. 3. The Manager inputs a reason for declining. 4. The Manager sends declining information to User.
Post-conditions	The order's status is changed to Declined.

Send notification

Use Case Name	Send notification
Actors	Manager, User

Pre-conditions	• The User has an account registered in the system. • The User account has been authenticated. • The Manager account has been authenticated.
Flow of events	1. The Manager opens one order. 2. The Manager selects the action (Edit/Activate/Approve/Deactivate/Delete/Decline/Cancel). 3. The system sends to the User a email with a notification about action. 4. The User receives a notification.
Post-conditions	The User received the notification created by the Manager.

Cancel order -> Delete Order (User part)

Use Case Name	Cancel order
Actors	User
Pre-conditions	• The User has an account registered in the system. • The User has an open order. • The User account has been authenticated.
Flow of events	1. The User clicks on the action button. 2. The User chooses the delete button. 3. The User confirms order deleting.
Post-conditions	Order deleted by the User

Cancel order -> Delete Order (Manager part)

Use Case Name	Cancel order
Actors	Manager
Pre-conditions	• The Manager account has been authenticated.
Flow of events	1. The Manager clicks on the action button. 2. The Manager chooses the delete button. 3. The Manager confirms order deleting.
Post-conditions	Order deleted by the Manager

Deactivate Order

Use Case Name	Deactivate Order
Actors	Manager, User
Pre-conditions	• The Manager account has been authenticated. • The User created an order. • The Manager received the User's order.
Flow of events	1. The Manager checks the order details. 2. The Manager accepts the order.
Post-conditions	The order's status is changed to Accepted.

Run project instruction

Prerequisites
- MySQL Server
- Local PHP server (Openserver or Xampp/MAMP(for Mac))
- PHP v5.6
- SMTP (Gmail or Mailtrap)

Instructions
1 Create MySQL database with name «roomservice»
2 Find the "database.sql" file inside database folder and import that file to recently created roomservice database
3 Edit "env.development" file
 3.1 Default user for mysql server is "root", password is empty, if you need to change these, you can edit "env.development" file)
 3.2 Default settings for email are empty, you need to change them, you need to add in "env.development" file: EMAIL_HOST, EMAIL_USER, EMAIL_PASS, EMAIL_PORT, EMAIL_CRYPTO.
4 Install Composer dependencies "composer install"