In this project, we will extend the application we developed in Homework #3 in order to implement the Plant Hire process at RentIT and the corresponding order-to-cash process at BuildIT. Below is a description of these processes. Do not hesitate to ask questions to the lecturer or lab assistants about ambiguities in this description, or about design choices you make.

**Context: Plant Hire**

*BuildIT* is a construction company specialized in public works (roads, bridges, pipelines, tunnels, railroads, etc.). Within BuildIT, it often occurs that engineers working at a construction site (called “site engineers”) need a special type of equipment, such as a truck, an excavator, a bulldozer, a water pump, etc. A piece of heavy equipment is called a “plant” in the construction jargon.

*Figure 1. Plant for hire (image taken from Holden Plant Hire Ltd., UK)*

BuildIT owns very little equipment and instead it hires most of its equipment from specialized heavy equipment suppliers. One of BuildIT’s preferred plant hire supplier is RentIT.

**BuildIT’s Plant Hire Process**

The business process for hiring a plant (seen from BuildIT’s perspective) is described below.

When a *site engineer* needs to hire a plant, he/she consults the catalogue of a plant supplier in order to identify a plant that fulfils the requirements. Once the site engineer has identified the required plant, he/she checks the plant’s availability during the required period of time as well as the price. If the plant is available, the site engineer creates a *Plant Hire Request*, which includes the identifier of the construction site where the plant is needed, the supplier of the plant, the plant’s identifier, the expected start and end date of the hire period and the cost of hiring the plant for this period of time. This cost is calculated based on the plant’s price per day of the selected plant and the number of days the plant is hired. Naturally, the plant hire request also includes the name of the site engineer making the request.

Every plant hire request has to be approved by a *works engineer* at BuildIT. The purpose of this
approval is to avoid excessive or unnecessary plant hiring and, more generally, to ensure that plant hiring costs are minimized and that they are within budget. In some rare cases, the works engineer rejects the plant hire request or makes modifications to the plant hire request before approving it. If the works engineer rejects or changes the plant hire request, they normally write a short explanation in a “comments” field in the plant hire request. In such cases, it is usually the case that the works engineer talks with the site engineer prior to rejecting or modifying their request in order to avoid misunderstandings.

Once the works engineer has approved the plant hire request, BuildIT’s information system automatically generates a Purchase Order (PO) for hiring the plant and sends this PO to the plant supplier. The supplier may accept or reject the PO. One reason why a PO might be rejected is that the plant that is being requested is no longer available during the requested period of engagement (e.g. it has been hired by someone else).

When the plant is hired, the supplier delivers it to the construction site at the required date. The site engineer inspects the plant and if everything is in order, he/she accepts the engagement. In some cases, the plant is sent back because the plant does not comply with the original specifications of the site engineer.

When the period of engagement is concluded, the supplier comes to pick up the plant. Sometimes, the site engineer requests an extension of the period of engagement (e.g. in order to keep the plant for an additional week). To this end, the site engineer requests the extension via BuildIT’s information system, which forwards it to RentIT’s.

A few days after the plant is picked up, the plant’s supplier sends an invoice to BuildIT by e-mail. The invoice is automatically matched with a PO by BuildIT’s information system. The invoice should then be approved by the site engineer who hired the plant. The reason why this check is needed is to ensure that the supplier is not invoicing for a plant that was rejected or for an incorrect time period. If the invoice is accepted, payment is scheduled and a remittance advice is sent to the supplier. If the invoice is rejected, the payment is not scheduled and the site engineer is responsible for communicating with the plant supplier in order to resolve the issue.

Plant suppliers may send reminders of unpaid invoices (every invoice has a due date). When BuildIT’s system receives a payment reminder, it checks if the corresponding payment has been scheduled or not. If it has been made, BuildIT re-sends the corresponding remittance advice. If on the other hand the payment has not been made yet because the site engineer still needs to approve the corresponding invoice, the site engineer is notified (in one way or another) that a payment reminder has been received for an unapproved invoice. This is achieved for example by marking the unapproved invoice with a “late payment” note.

RentIT’s Order-to-Cash Process

The order-to-cash process at RentIT starts when a new purchase order (PO) for a plant hire is received. The PO refers to a specific plant (also called a Plant Inventory Item, e.g. a specific jackhammer with serial number 234136) of a specific type (also called a Plant Inventory Entry, e.g. Hilti Jackhammer model TE 905). Prior to placing an order, potential customers can query RentIT’s catalogue to check which plant inventory items are available during any period of time in the future.

When a PO is received, RentIT’s information system automatically checks if the requested plant is available during the requested period of time. If it is not, the PO is rejected. If the request plan is not available, an employee at RentIT needs to check the PO and either accept it or reject it (there may be other reasons why a PO is rejected even though the item is available). In any case, RentIT’s information system will send a notification to the customer whenever their PO is accepted or
rejected.

Once a PO is accepted and the shipment date arrives, a clerk at RentIT’s depot will dispatch the plant to the customer’s site. They will then mark the PO as “dispatched”. The customer can cancel the PO anytime before it is dispatched, but not after. The customer may also change the rental period until the plant is dispatched.

Once a plant is dispatched, the only change that the customer is allowed to request is to postpone the date when the plant will be returned (i.e. to extend the period of the plant hire). Every change request must be approved by an employee at RentIT and it can only be approved if the plant is available during the extended period.

When a request to extend the engagement period for a plant arrives in RentIT, system automatically checks the plant availability for the requested period of time, and if the plant is available it will automatically adjust the purchase order (i.e., create a new reservation and update the overall rental period and cost) and notify the customer that the extension has been accepted. If the period of the extension overlaps with the engagement with another customer, RentIT’s system automatically rejects the extension and notify the customer accordingly.

When the date of return arrives, the clerk asks the transportation provider to pick up the plant. When the plant arrives back at RentIT’s depot, the clerk marks the corresponding PO as “returned”. Once in the warehouse, the maintenance team checks the plant. If the maintenance team finds a dysfunction in the plant, the maintenance team leader would schedule a corrective repair. If the plant at hand is engaged with other customers in the short term, RentIT’s information would first try to find a replacement and book it to honor the purchase order. If no replacement is found though a RentIT’s information system must notify the customer about the problem. Although some other alternatives to resolve this situation exist (e.g. subcontracting), we will not consider them.

RentIT’s system automatically generates invoices for returned plants. The system monitors when a payment (remittance advice) is received and it then marks the invoice as “paid”. The system also sends reminders to the customer when the payment date of an invoice arrives and the invoice has not yet been paid.

1. Requirements (12 points)

The goal of this project is to implement a service-oriented system in order to support the above business processes (both on BuildIT’s side and on RentIT’s side).

The functional requirements (features) for BuildIT’s system are:

CC1. The system should allow site engineers to create a plant hire request.

CC2. The system should allow site engineers to modify a plant hire request prior to its approval by the works engineer.

CC3. The system should allow site engineers to cancel a plant hire request. Cancellations are allowed until the day before the plant is due to be dispatched. If a cancellation is requested after the PO has been sent, a request for cancellation should be sent to the supplier.

CC4. The system should allow site engineers to check the status of a plant hire request.

CC5. The system should allow works engineers to approve or reject a plant hire request. A works engineer may also modify the plat hire request and approve the modified request. The allowed modifications include changing the delivery site, or reducing the period of engagement (e.g. postponing the start date of the hire or bringing forward the end date).
CC6. The system should produce a PO for every approved plant hire request and forward it to the corresponding supplier. The supplier may respond with an acceptance or a rejection message, asynchronously.

CC7. The system should allow BuildIT employees to view all submitted POs and their statuses.

CC8. The system should allow site engineers to request an extension in order to keep a plant longer than its initial period of engagement. When an extension is requested, the system should produce a modified PO and forward it to RentIT. The RentIT’s system may accept/reject the modified PO, and so the extension.

CC9. The system should allow a supplier to submit invoices for a given plant hire.

CC10. When an invoice is received, the system must check that the PO number in the invoice corresponds to an existing and unpaid Purchase Order. If the PO does not exist or has already been paid, an error message is returned to the supplier.

CC11. The system must allow site engineers to approve an invoice and to retrieve the PO associated to an invoice.

CC12. The system must submit a remittance advice to the supplier after the invoice is approved (normally the remittance advice should only be sent after the payment has been triggered, but in this project we do not deal with sending payment orders to the bank).

NOTE: One should keep in mind that BuildIT’s accounts payable department handles hundreds of invoices per month, not only for plant hiring but for many other types of expenses such as construction material, office supplies, sub-contracting services, etc. In this project we will focus on building a sub-system for handling invoices for plant hiring, but we should keep in mind that the system should be extensible to handle many other types of invoices.

The above features are described mainly from the perspective of the construction company’s system (BuildIT). The plant supplier (RentIT) also needs a system to handle its order-to-cash process (from the moment a purchase order arrives until the payment of the invoice is received). RentIT’s system should also allow BuildIT to query the catalogue. One can derive the features of the plant supplier’s system from the above ones. In particular:

PS1. The system should allow a customer to list the available plants and their prices.

PS2. The system should allow a customer to check the price for a given plant (given the plant identifier)

PS3. The system should allow a customer to check the availability of a given plant during a given time period

PS4. The system should allow a customer to submit a PO for hiring a plant. If the plant item is not available during the period indicated in the PO, the system should automatically reject it. If it is available, the system should show the PO as “pending” to the workers at RentIT. A worker at RentIT can accept or reject a PO (but the system should always prevent accepting a PO when the plant is not available).

PS5. The system should allow a customer to view the status of any PO they have submitted. Possible statuses are: pending, accepted, rejected, plant dispatched, plant returned, invoiced.

PS6. The system should allow a customer to submit modified POs requests and accept/reject a modified PO depending on the plant’s availability. PO modifications can only be used for modifying the period of plant rental. They cannot be used to change the plant inventory item
that has been ordered (i.e. they cannot be used to order a different plant). If the plant item is not available during the whole (modified) rental period, the PO modification request should be automatically rejected. If the plant is available for the requested extension period, the system should accept the request and modify the PO accordingly.

PS7. The system should allow a customer to submit a request to cancel a PO. A cancellation request is accepted if the request arrives prior to the plant being dispatched. If the plant has already been dispatched, the cancellation request is rejected.

PS8. The system should allow employees at the plant depot to mark a plant as “dispatched”. This happens when the plant leaves the depot.

PS9. The system should allow employees at the plant depot to mark the plant as “delivered” or “rejected by customer”. This latter option happens if the customer did not accept the plant because the plant did not meet the specifications in the catalogue.

PS10. The system should allow employees at the plant depot to mark a plant as “returned”, meaning that the plant has been returned in due form and the rental is completed.

PS11. The system should allow employees from the maintenance department to enter the information to schedule a corrective maintenance task for the plant being returned if the plant needs any repair.

PS12. The system should try to book another plant to replace the one that has been scheduled for maintenance (cf. PS11), if the latter was engaged with another customer. The system should send a notification to the customer accordingly.

PS13. The system should submit invoices for “returned” plants.

PS14. The system should submit payment reminders for unpaid invoices, when the due date arrives.

PS15. The system should allow customer to submit remittance advices and it shall mark the corresponding invoice as "paid" once the corresponding remittance advice is received.

NOTE: The plant supplier has two separate sub-systems: one for handling plant hire requests, and the other for invoicing (i.e. accounts receivable).

Non-functional requirements

- The BuildIT system should implement role-based authentication. There are two roles: site engineer and works engineer.
- There should be user interfaces both on the RentIT and the BuildIT side. These user interfaces can be very minimalistic and are only meant to demonstrate fulfilment of the functional requirements and the integration requirements.
- The backend implementation must be done in Java Spring Boot and the frontend using Vue.js. You may use React for the frontend, but we will not answer technical questions about React.

2. Testing (8 Points)

You must implement test cases covering all the requirements. You can use both integration (i.e. tests on rest controllers) and acceptance test (i.e. cucumber scenarios). For each one of the requirements, in a document, indicate where to find the corresponding test and the type of test.
3. **Integration Requirements (5 Points)**

Each team must complete one implementation of BuildIT and one implementation of RentIT. Upon completion of the project, every implementation of BuildIT should be integrated with two implementations of RentIT (the team’s own implementation of RentIT plus the implementation of another team). In this context, integration means the following:

- The BuildIT system provides the ability to search the list of available plants and prices of different RentIT systems, either together (site engineer can do this in a single query) or separately (site engineer has to run separate queries to consult each RentIT system).
- The BuildIT system can fulfill its functional requirements (placing an order, checking order status, etc.) with each of the RentIT systems.
- The BuildIT system can handle invoices from each of the RentIT systems.

**NB:** You must use the Spring Integration framework, e.g., you can check Lab session #10 “Enterprise Integration patterns (Scatter/Gatter + Transformations)” materials as a reference. Teams violating this restriction may be penalized up to 2 points.

4. **Deployment requirements (2 points + up to 2 bonus points)**

Your RentIT and BuildIT systems should be deployed on a server. We recommend you to use Heroku, but you can deploy on any (cloud) service. You can get bonus points by creating Docker containers with your RentIT and BuildIT applications and deploying them on a cloud platform such as DigitalOcean.

You will get points for deployment as follows:

- One point for deploying your RentIT system on a server (e.g., on Heroku). You will get one additional (bonus) point for creating a Docker container of your RentIT system, pushing the image to Docker Hub, and deploying it on a server (e.g. using DigitalOcean).
- One point for deploying your BuildIT system on a server (e.g. on Heroku). You will get one additional (bonus) point for creating a Docker container of your BuildIT system, pushing the container image to Docker Hub, and deploying it on a server (e.g. using DigitalOcean).

5. **Report (3 points)**

You should submit a report with the following structure by **18 May at 14:00**.

1. A title page including the names and student numbers of all team members. Only include team members who contributed to the project in a non-negligible way.
2. Any models to document your system design (both for BuildIT and RentIT). As a minimum you should provide, for both RentIT and BuildIT, a domain model, a state model of the main resources, and a description of the API in a table form (as shown in Lecture 6).
3. A list of the RentIT systems with which you integrated your BuildIT system, and brief explanation of how did you achieve this integration. For example, did you introduce adapters, or do you have multiple separate RentIT clients, or did you simply hack it all by mixing business logic code with integration code in your BuildIT system.
4. Links to the deployed RentIT and BuildIT systems (and the Docker Hub username and repository (ies) if you wish to claim points for Dockerization).
5. Links to the code repository(ies) containing all the code to build, deploy, and run your
BuiltIT and RentIT systems. Please note that we will check the commit history of your project to assess the contributions of each team member.

6. Any other documentation you feel would support your submission, but please keep it short.

**Point Allocation and Checkpoints**

In order to encourage you to start early and to be iterative, the points corresponding to **Requirements** and **Testing** will be allocated through three checkpoint meetings with the teaching staff, as follows:

- The dates of the checkpoints are the following:
  
  CHP #1: Lecture Session 27/04 (14:15 – 16:00) or Lab Session 28/04 (14:15 – 16:00)
  
  CHP #2: Lecture Session 04/05 (14:15 – 16:00) or Lab Session 05/05 (14:15 – 16:00)
  
  CHP #3: Lecture Session 11/05 (14:15 – 16:00) or Lab Session 12/05 (14:15 – 16:00)

- The duration of each checkpoint is 30 minutes.

- At each checkpoint, you or one of your team members will explain and show your team’s progress to the lab assistant. Specifically, you must present the implementation of 8 requirements (i.e., CC1-CC12 and PS1-PS15); also, the selected requirements must be covered by either integration or acceptance tests. Note that, teams are free to choose which requirements they are presenting at each checkpoint, but they must select requirements not implemented in previous checkpoints. Clearly, during CHP #1 teams are free to choose 8 out of the 27 requirements in the project description.

- At each checkpoint the team may receive up to 5 points (3 for implementation and 2 for testing). In order to get the maximum amount of points, all the 8 selected requirements must be implemented and properly tested. Also, the team should be able to illustrate the functions via Postman or Web UI, running all the tests, and they must PASS.

After the 3 first checkpoints, teams could receive up to 15 points (half of the points). The remaining 15 points (plus 2 bonus points) will be received from **Integration, Deployment and Report**, and for the implementation of the missing requirements/tests, as well as the non-functional requirements (i.e., role-based access control, and VueJs front end). To that end we, we will have a 30 minutes’ interview on the Project Final Checkpoint on 18-19/05. Note that, you won’t receive points in this last checkpoint, but instead you will be asked about the completeness of your project. Instead, the remaining 15 points will be granted by the teaching staff after checking the code of the final version.

Each team must create one or several repositories for their projects. Add the teaching staff as collaborators of your repositories. Username: orlenyslp (email: orlenyslp@gmail.com) and mcamargo85 (email: manuel.camargo@ut.ee).

**Deadline for last commit to the project repositories:** 18/05/2020 at 14:00.