# Bonita Camp

Antoine Mottier

# CHECKLIST

- Download Bonita Camp resources from GitHub: https://github.com/Bonitasoft-Community/bonita-camp/releases
- Make sure that latest version of Studio is correctly installed (see exercises instructions)
- Install latest Oracle JDK 8
- Start the Studio and click on Portal button
- We recommend to use Chrome or Firefox as your default web browser for development

# AGENDA

- Bonita solution
- Bonita components
- BPMN 2.0
- Data & Contracts
- Forms & Pages
- Actors
- Connectors
- Bonita applications
- Deployment

# BONITA SOLUTION

- Lower the volume of code required to build an application
- Rely on models (e.g. BPMN diagrams) to define the application
- Models improve collaboration during the definition of the application
- Process oriented approach ease collaboration for process execution
- Track process execution for monitoring and optimization

# BONITA COMPONENTS

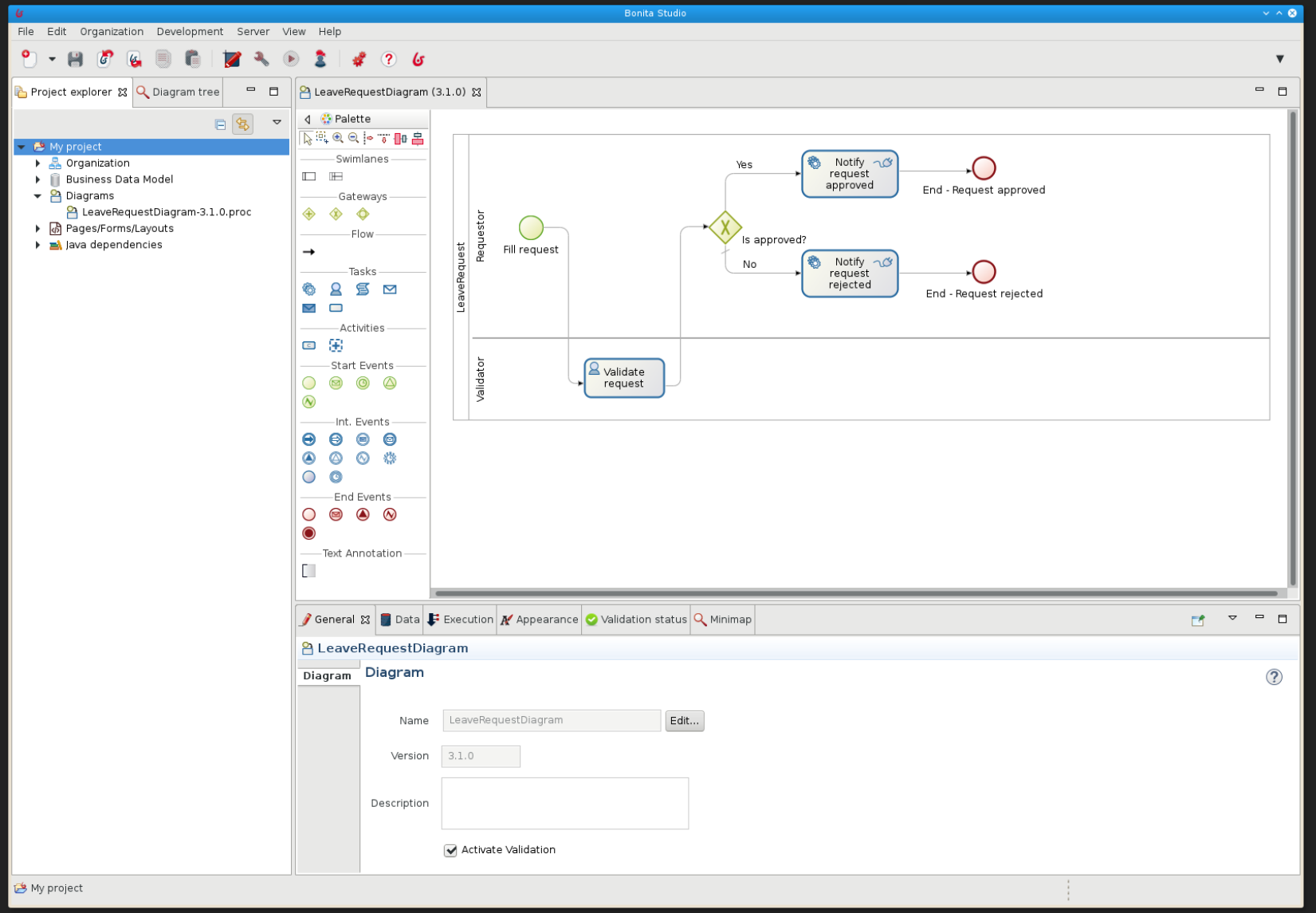
Four main components:

- Bonita Studio
- Bonita UI Designer
- Bonita Engine
- Bonita Portal

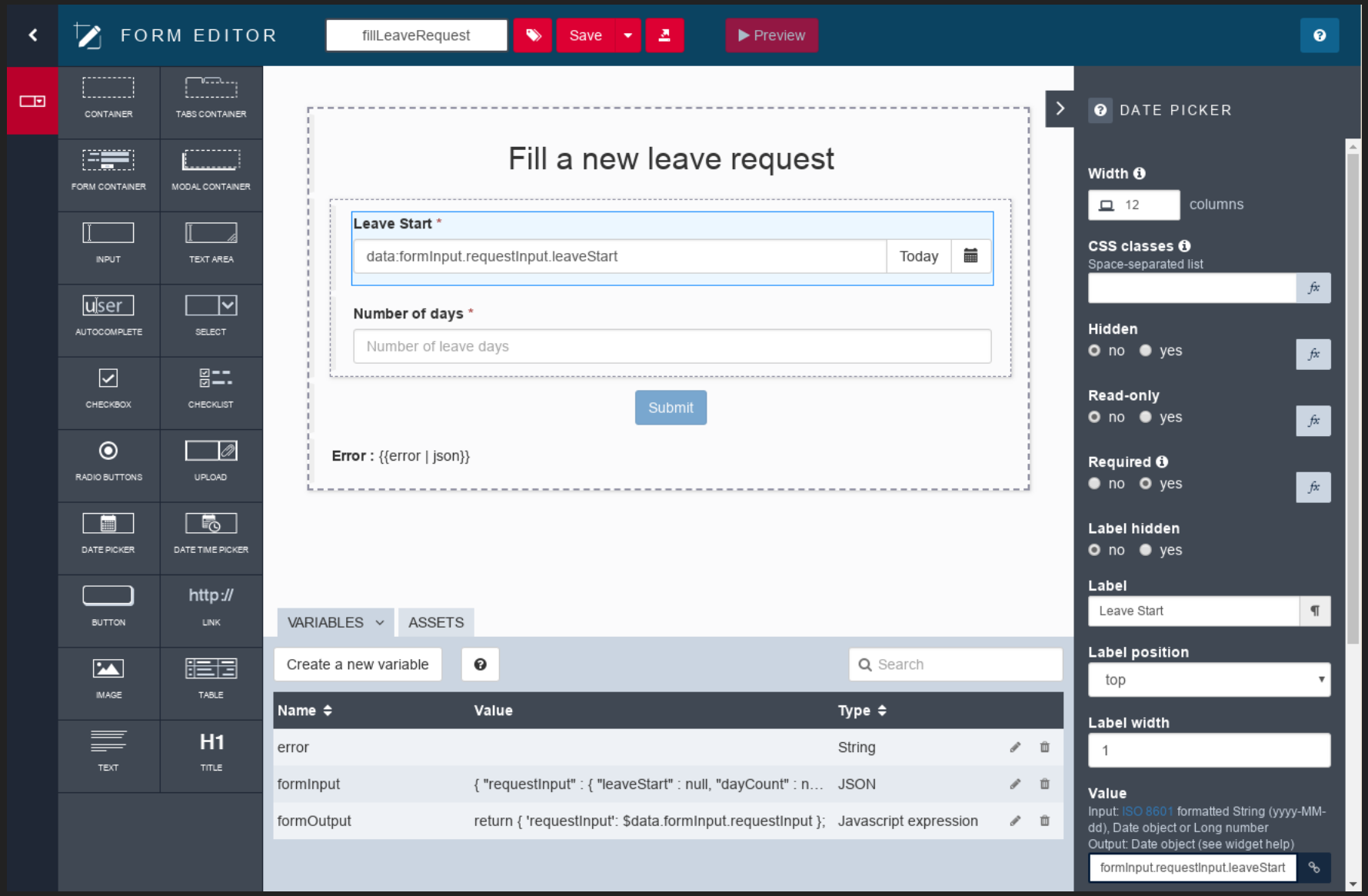
Engine + Portal: packaged in a single JEE web application (war)

Studio embeds a test environment with Tomcat + web application

# BONITA STUDIO: PROCESS EDITOR



# BONITA STUDIO: UI DESIGNER



# BONITA ENGINE

In charge of process execution (no built-in UI)

Expose REST APIs

Based on Java SE 8

Packaged with the Bonita Portal in a Java web application

Requires:

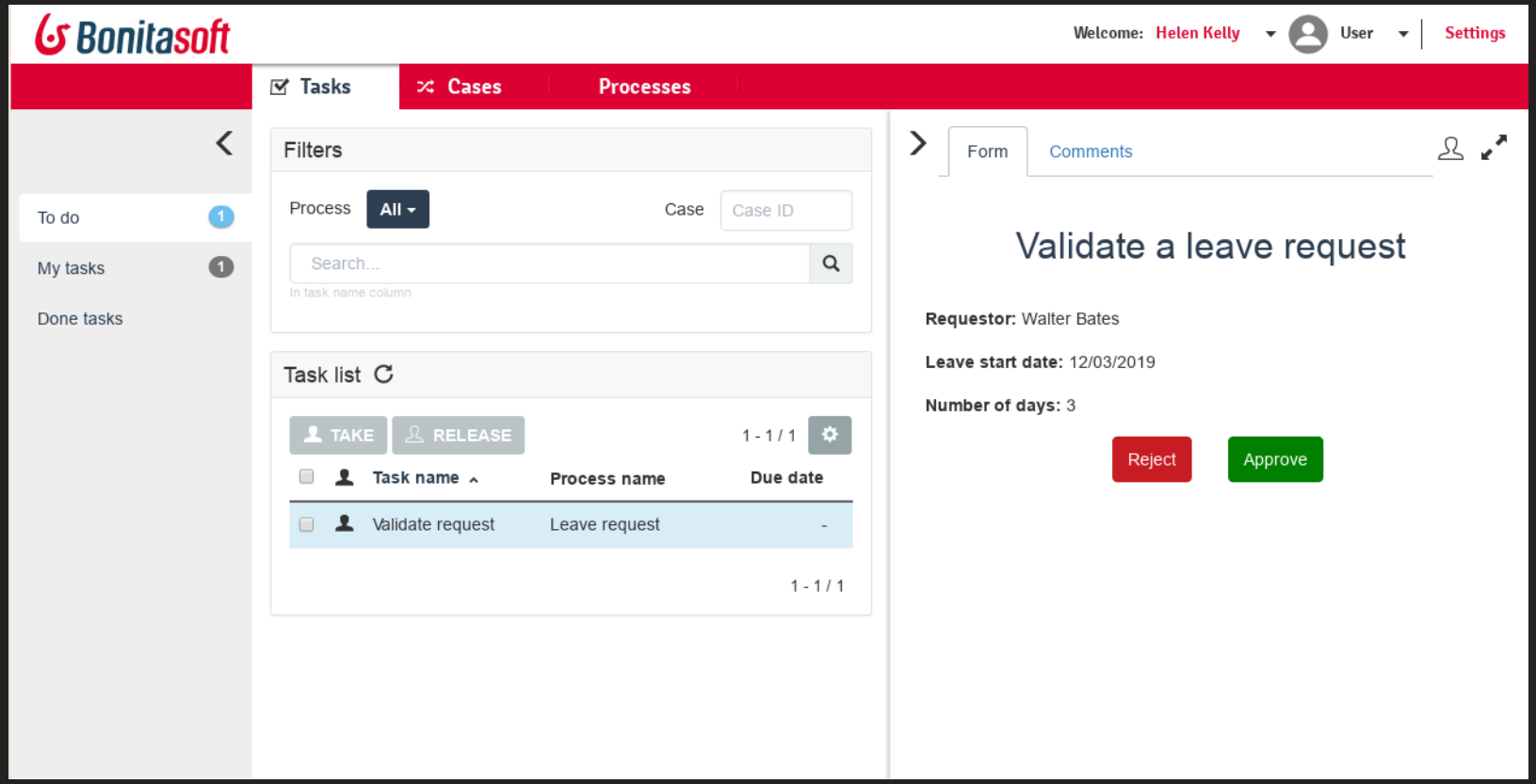
- A Java EE application server
- A JDBC compliant transactional database

# BONITA PORTAL

End user and administration web interface

Displays the list of available tasks

Allows to deploy and configure processes

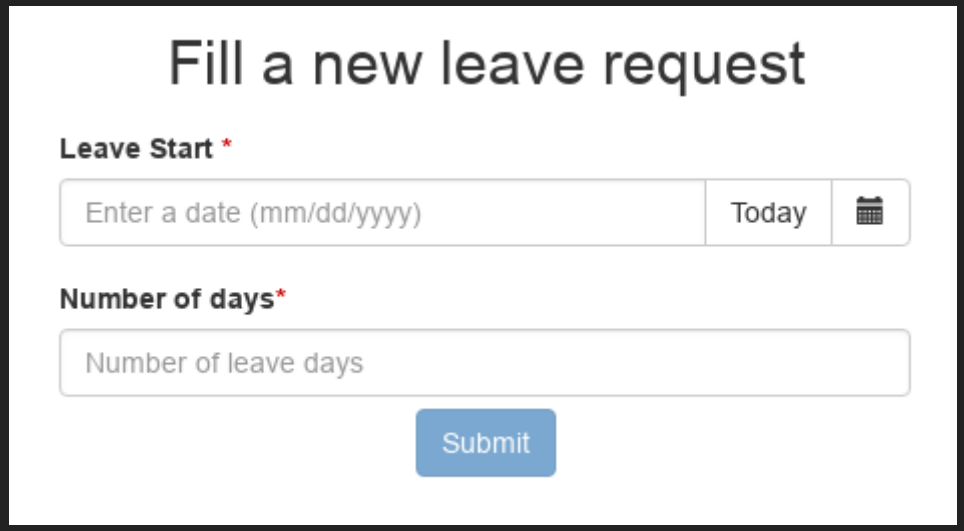


# BONITA PORTAL: FORMS

Renders forms created with UI Designer

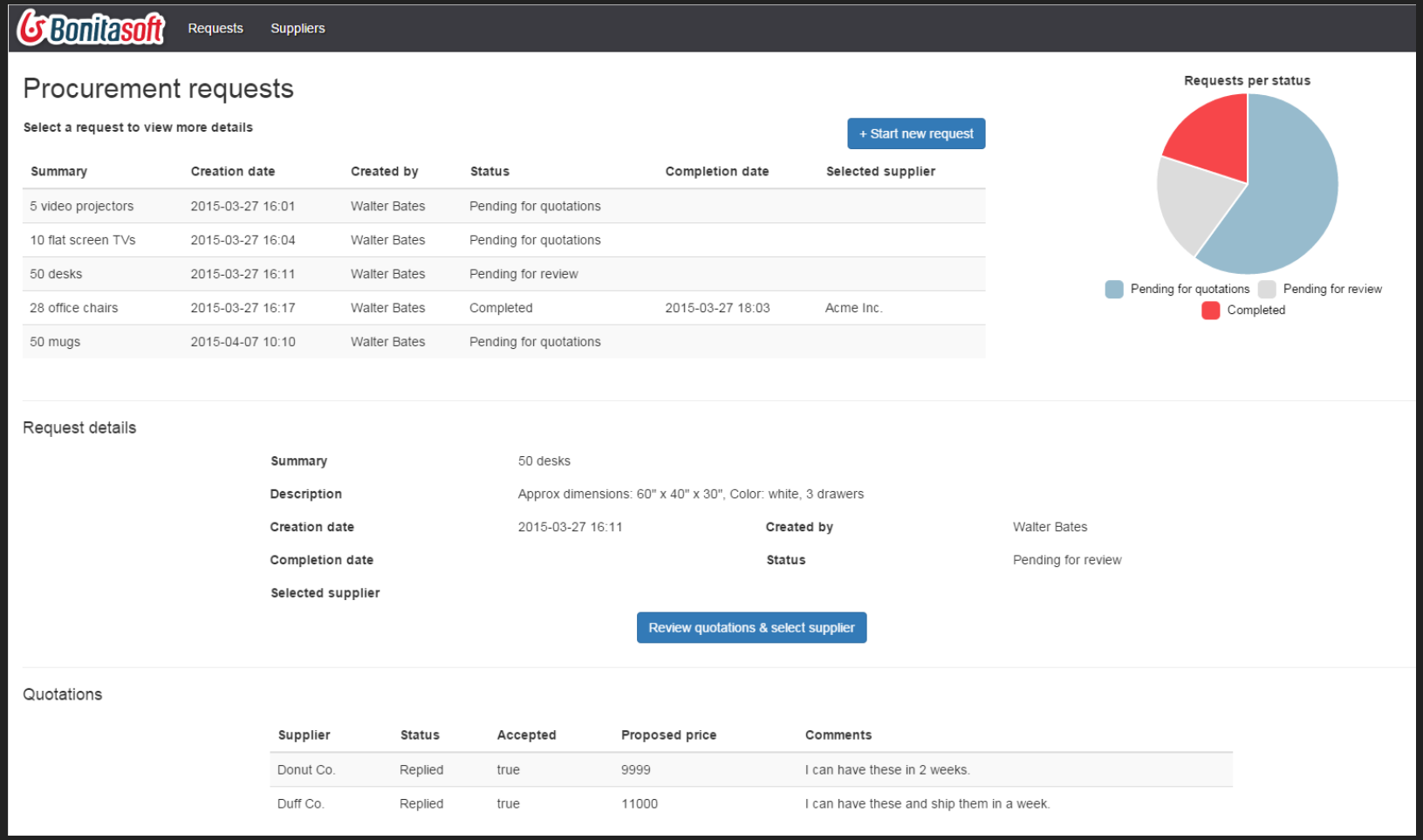
Forms are associated with process (instantiation) or human tasks

Served by Bonita Portal



# BONITA PORTAL: APPLICATIONS

Allows to deploy Business Applications composed of pages implemented with the UI Designer



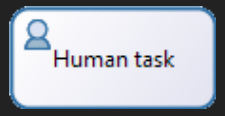
# BPMN 2.0

Standard notation (OMG) for process modeling

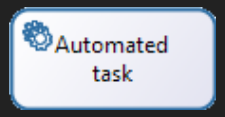
Key elements:


Pool: one pool per process

Start event: the begging of a process




Human task: when user is involved




Service task: automatically run by Engine

End event: trigger process archive

→ Transitions: used to link elements together

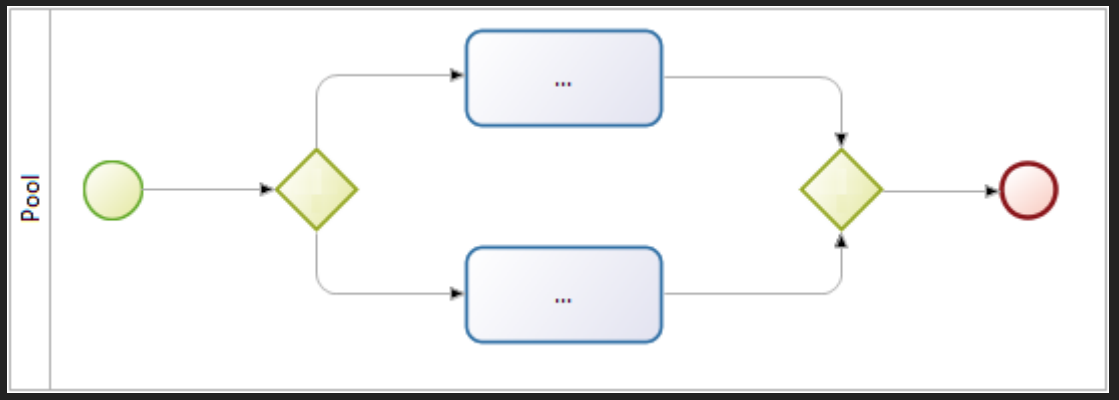
# GATEWAYS

3 types of gateways:

Exclusive

Parallel

Inclusive




- Conditions are defined on gateway outgoing transitions. No condition for parallel gateways
- Gateways should be placed in a symmetrical manner when possible to improve readability

# EXERCISE 1

Modeling a basic process

If you don't already did it get instruction from GitHub:
https://github.com/Bonitasoft-Community/bonita-camp/releases

# BDM: BUSINESS DATA MODEL

- Stores business data related to processes instances and/or applications
- Model defines objects to store business data
- Model is defined in the Studio
- From model Bonita generates Java classes (POJO) and creates database tables
- Persistence is managed by Engine

# BDM: USAGE

- Declared in process definition business variables
- Create/Read/Update/Delete using: default values, operations, connectors...
- Can be shared among different process instances
- Lifecycle control by the developer
- APIs for read operations

# FORM DATA

- Store and process data displayed and updated in UIs
- Accessible in the user web browser
- Can be any type of JavaScript objects
- Initialized and updated by: user inputs, REST API calls, JavaScript code
- Never persisted

# DOCUMENTS

- Bonita provides a lightweight content management system
- References to documents are part of the process definition
- At runtime documents can be initialized, updated...
- Unlike BDM, documents are associated with a given process instance

# CONTRACTS

Set of required inputs + validation rules

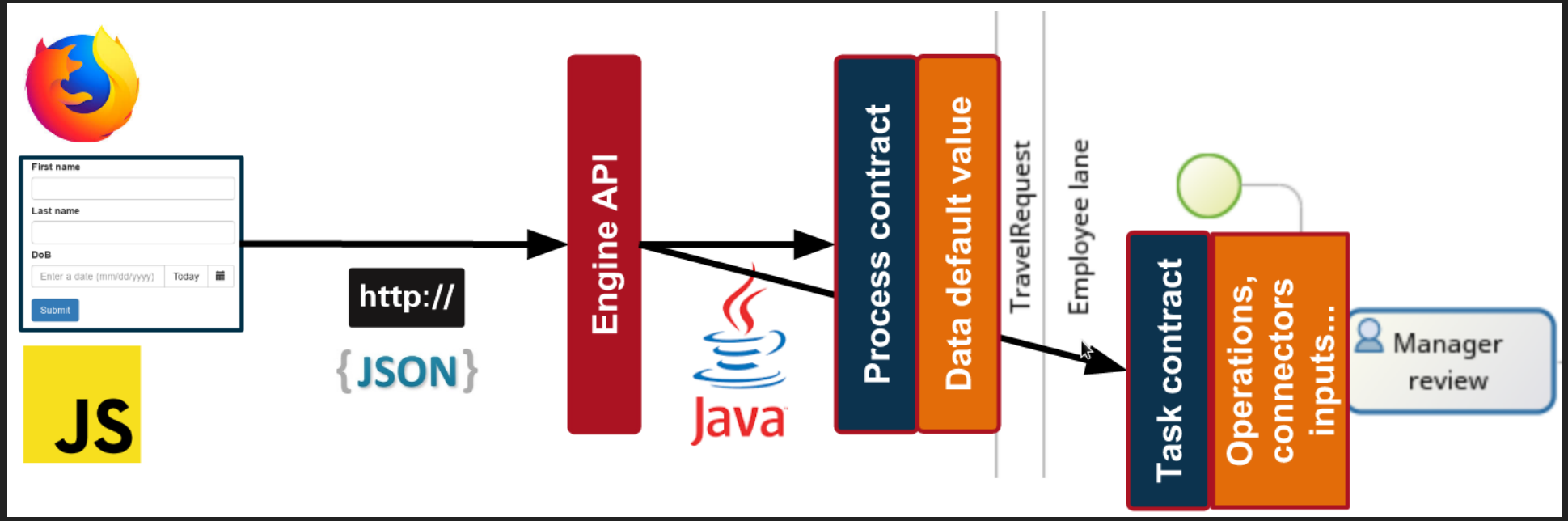
Data sent by forms must satisfy a contract in order to:

- Start a process
- Execute a human task

Contributes to the decoupling of process logic and user interface

# ARCHITECTURE OVERVIEW

Sending data from the form to the business variable



# EXERCISE 2

Adding data and specifying contracts

# FORMS

Created with UI Designer with containers and widgets

Based on AngularJS and Bootstrap

Associated with tasks or process

2 types of forms:

- **Editable forms** that must satisfy a contract
  - Process instantiation defined at pool level
  - Human tasks (no forms on service tasks)
- **Overview form** (read-only) defined at pool level

# PAGES

Build in the same way as a form

Main difference is the execution context:

- Forms have information about process definition or process instance and task instance
- Pages have no context information

Pages can get BDM and process information or trigger actions using REST APIs

Group several pages together with a menu in order to build an application

# WIDGETS

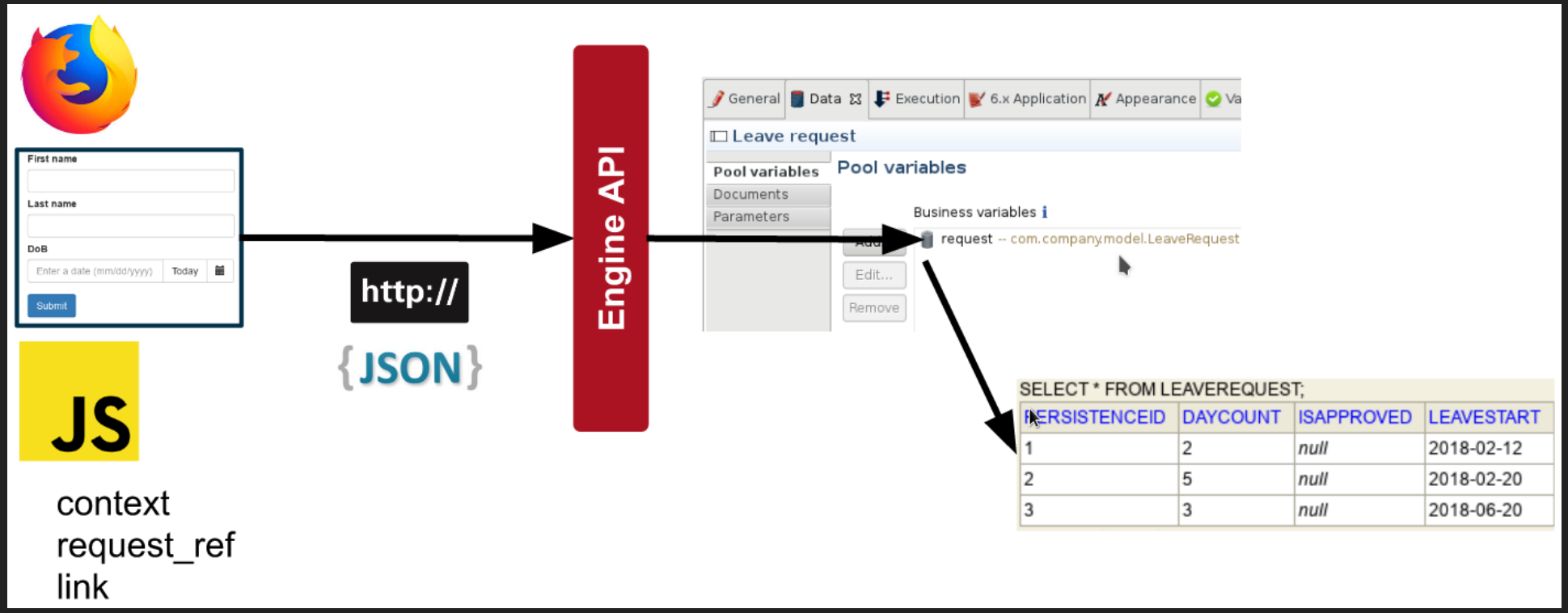
A set of widgets is available out of the box (text field, select, date picker...)

Custom widgets can be created with the UI Designer

You can also reuse contributions shared on community website

# ARCHITECTURE OVERVIEW

Display process variable in a form



# EXERCISE 3

Creating forms

# WHO CAN DO A TASK?

A set of users but protection against concurrent access is provided

2 options available for defining who can do a task:

- Actors
- Actors filters

Both options requires the user to be authenticated

User need to be known in Bonita user database

Support for LDAP, Active Directory, SAML V2, CAS, Kerberos includes in Enterprise Edition

# ACTORS

- Declared one or several actors in process definition (just a name)
- Configured to associate an actor with users:
  - Directly to a set of specific users
  - Indirectly using groups and roles of users
- Map actor directly to the task or using a lane
- Actors cannot use "live data"

# ACTOR FILTERS

Name does not reflect what actor filter does

Completely override the actor configuration

Produce a list of users based on input information and an internal logic

Executed when process reach a step to which the filter is mapped

Can be "refresh" using Engine API

# ACTORS VS ACTOR FILTERS

| | Benefits | Limitations |
|---|---|---|
| Actors | Always up to date with organization | Mapping that can not rely on business/process data |
| Actor filters | Mapping that can be based on business/process data | Executed only once at task instantiation |

# WHO CAN START A PROCESS INSTANCE?

All users associated with the actor with "initiator" flag

Only one actor on a given process definition has this flag

For a given process instance the user who start it will be registered as the initiator of the instance

# EXERCISE 4

Configuring actors

# CONNECTORS

Allows processes to interact with IS (opposite of the Bonita APIs)

A set of standard connectors is shipped in the Studio

Custom connectors can be:

- Implemented using the Studio or other Java IDEs
- Imported from the Community website

# EXERCISE 5

Using a connector to send an email

# BPM-BASED APPLICATIONS

Combine processes together to handle a real business use case

Respect the MVC design pattern:

- Model is the business data model
- View is a combination of pages and process forms
- Controller is a set of processes

Applications are deployed and hosted in Bonita Portal

# EXERCISE 6

Create a leave request application

# PROCESS DEPLOYMENT

Main steps for process deployment:

| | |
|---|---|
| 1. Process configuration | Studio |
| 2. Bar file generation | |
| 3. Bar deployment | Portal |
| 4. Process configuration | |
| 5. Process activation | |

# APPLICATION DEPLOYMENT

Main steps for application deployment:

1. Export pages from UI Designer — Studio
2. Export the application

---

3. Import pages as Resource — Portal
4. Import the application

# THANK YOU FOR YOUR ATTENTION

Any questions? Please post them on Bonitasoft community website.

Extra Resources:

- Documentation - documentation.bonitasoft.com
- Corporate Website - bonitasoft.com
- Community - community.bonitasoft.com
- GitHub - github.com/bonitasoft