Vaadin Bluemix Boilerplate Introduction

IBM Bluemix Hands-on Workshop

Please enjoy this step-by-step workshop exercise for getting familiar with the Vaadin Boilerplate inside Bluemix. This exercise will guide you through:

1. Creating your own instance of the SimpleCRM example application inside Bluemix
2. Creating your own fork of the codebase
3. Making small modifications to the codebase on-line at hub.jazz.net
4. Setting up a local development environment for Eclipse

After this all the fun is up to you. Happy coding!

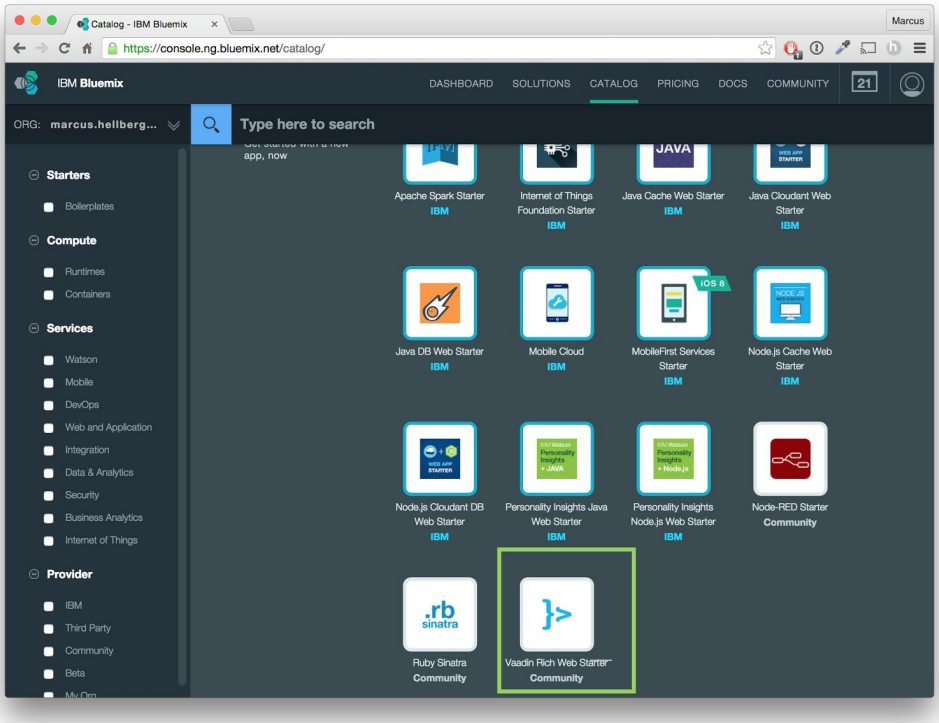
Preparations

1. Make sure you have an account to bluemix.net. If you don't please create one before continuing.
2. Make sure you have an account to hub.jazz.net. If you don't please create one before continuing.

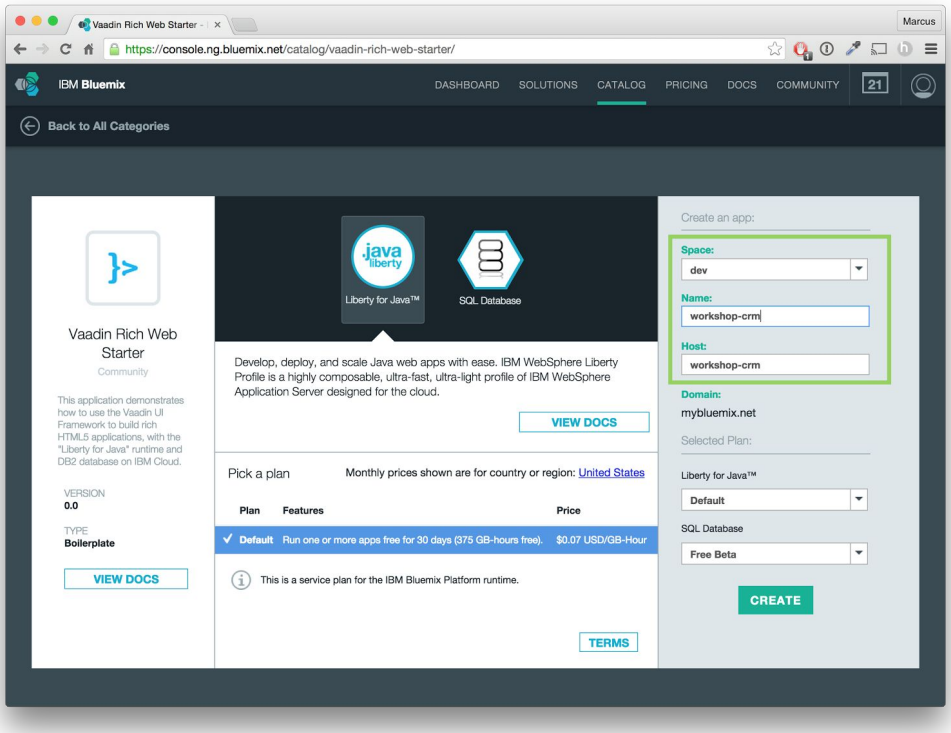
Part #1: using the on-line tooling

Create your Bluemix instance from the boilerplate

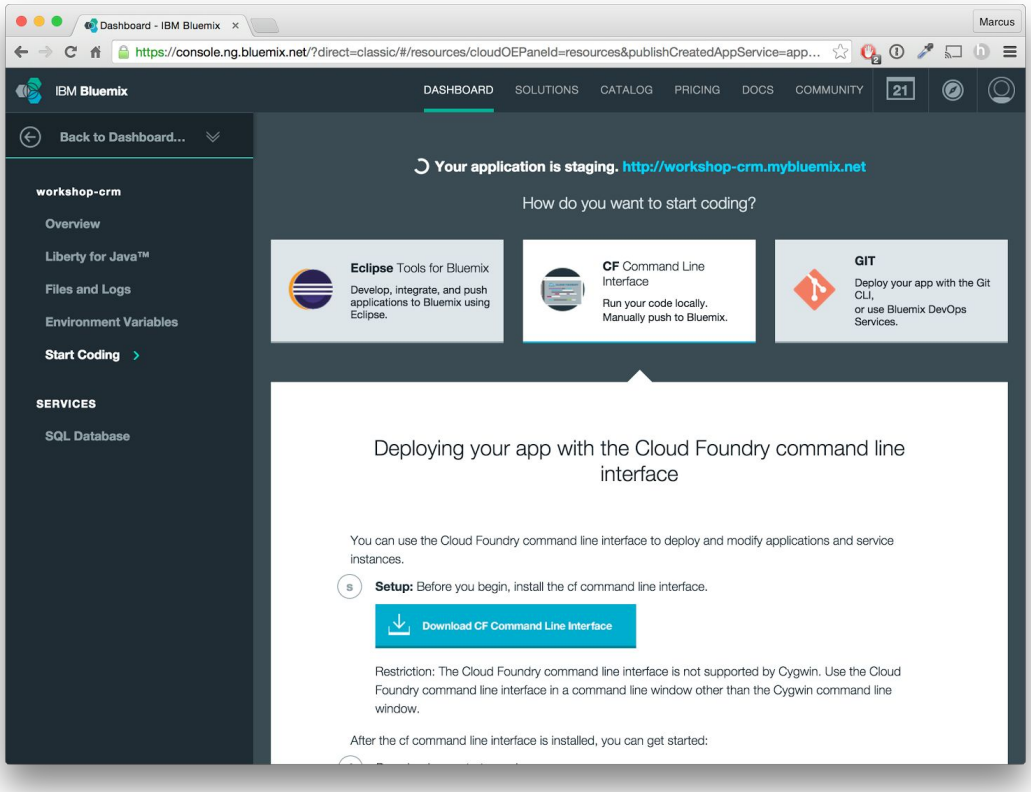
1. Login to bluemix.net with your IBM Id
2. Go to Bluemix catalog page and click the Vaadin Boilerplate, see below



3. Enter a unique ‘name’ and ‘host’ for your application, I’m using ‘workshop-crm’. Avoid spaces and special characters. Once ready click ‘create’.

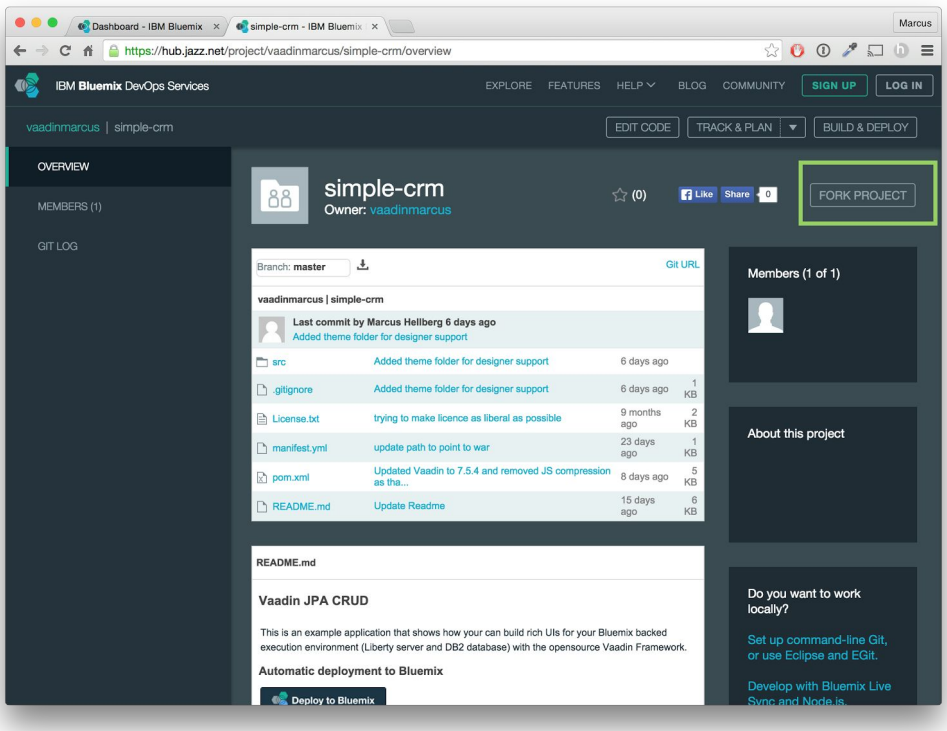


4. Now your application instance is being created so sit back and wait for a while. When everything is ready you should see a similar page as below. Now you have an empty JavaEE server and a database ready.

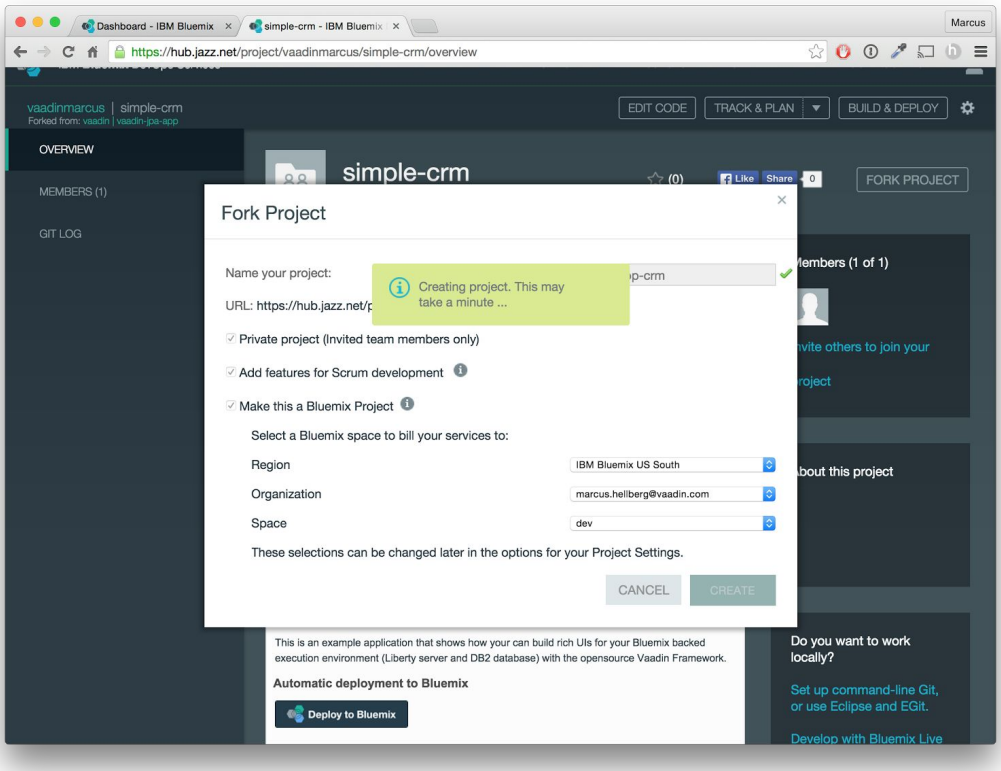


Setting up Git and deploying to Bluemix

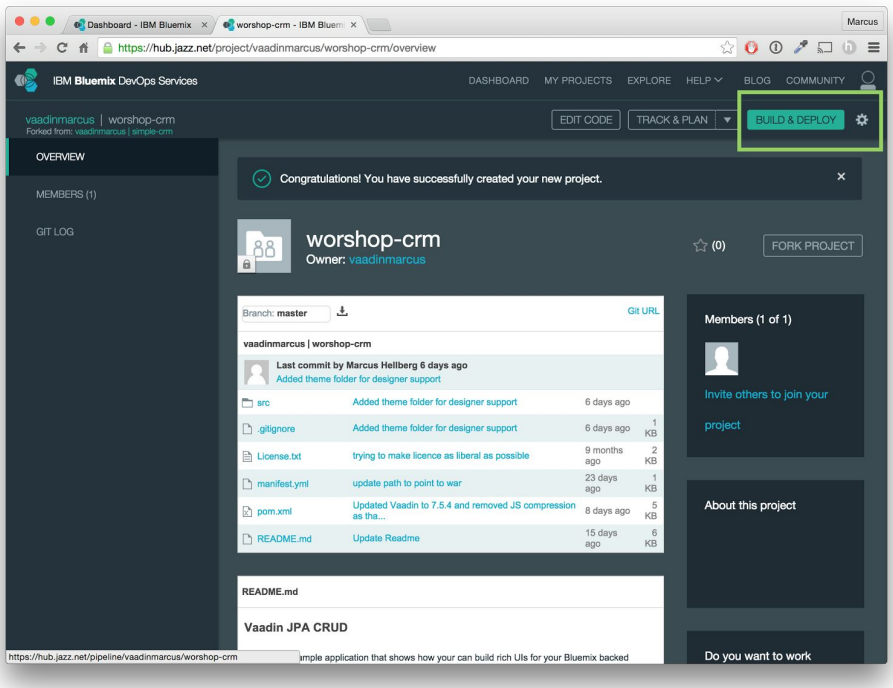
1. Browse to the boilerplate source code repository at https://hub.jazz.net/project/vaadinmarcus/simple-crm and click ‘fork project’.



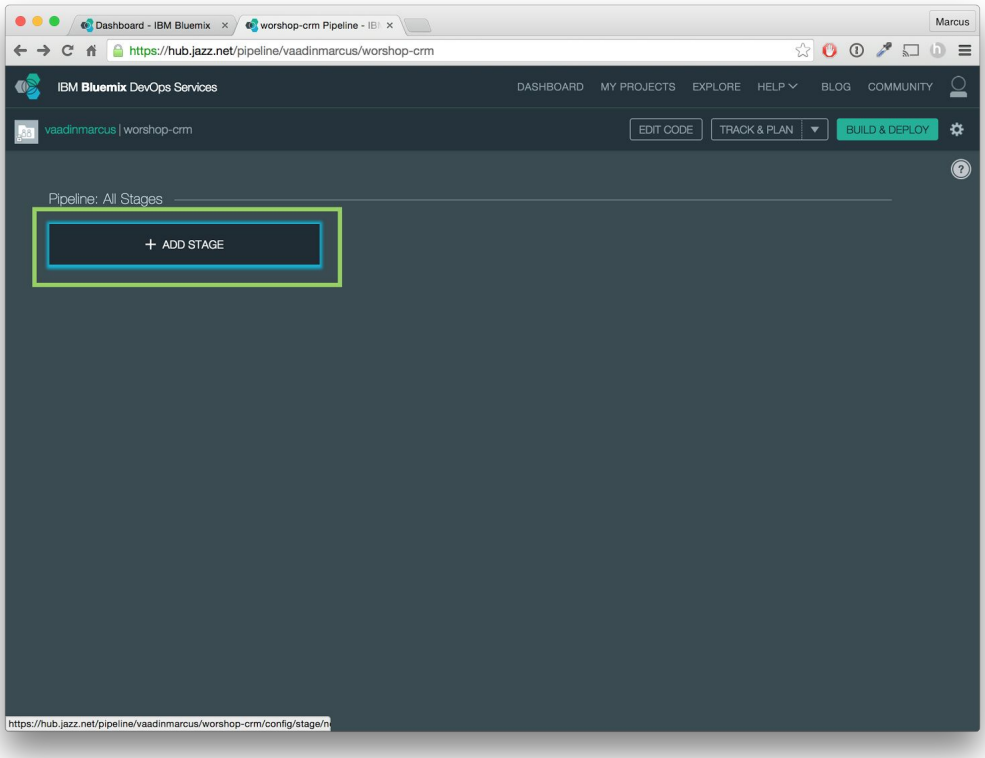
2. Name your project with the same name you used for the bluemix instance and check the ‘deploy to Bluemix’ option as shown below.

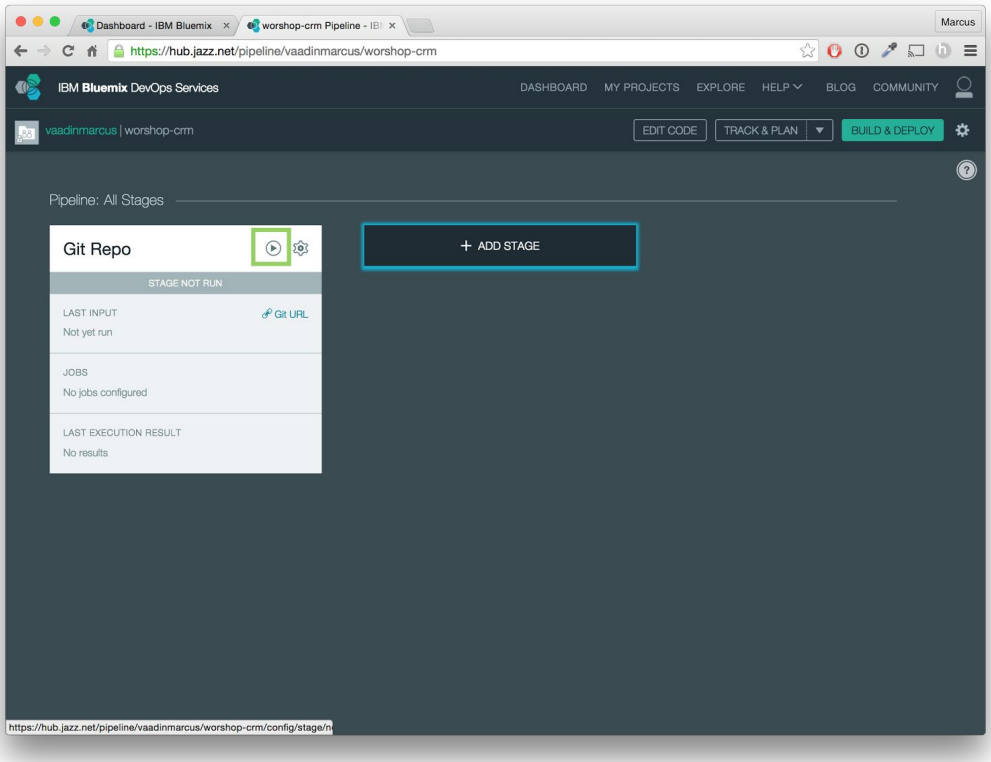


3. Click create and wait while everything is being created for you.
4. When all is good you should see a similar page as below. Go on and click ‘build & deploy’.

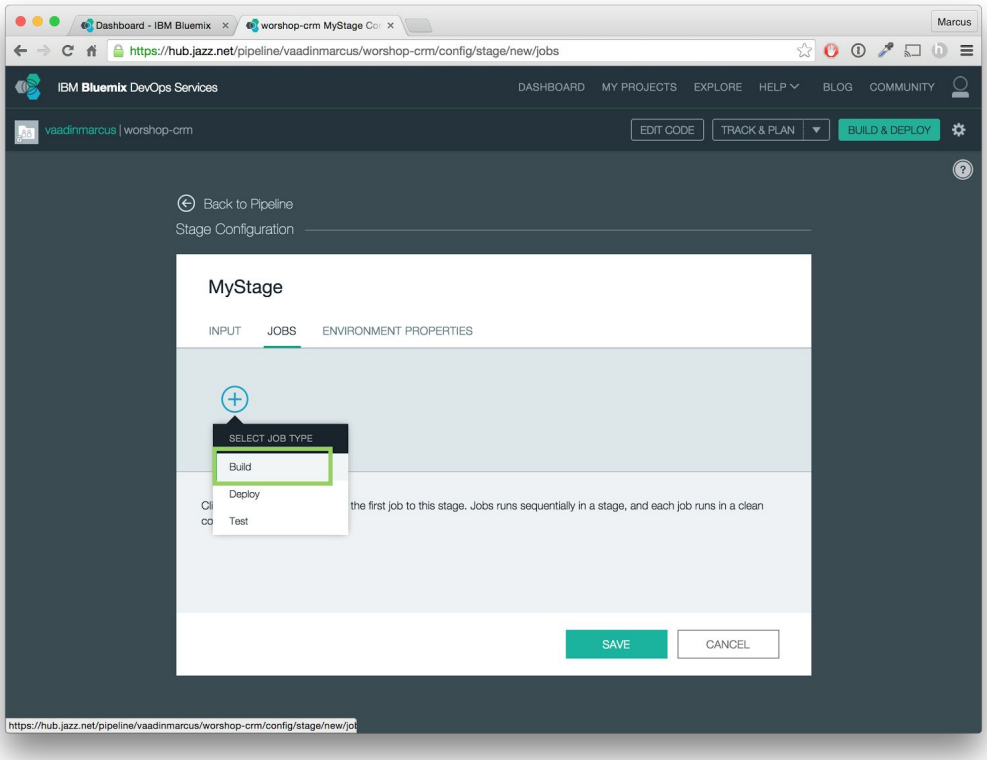


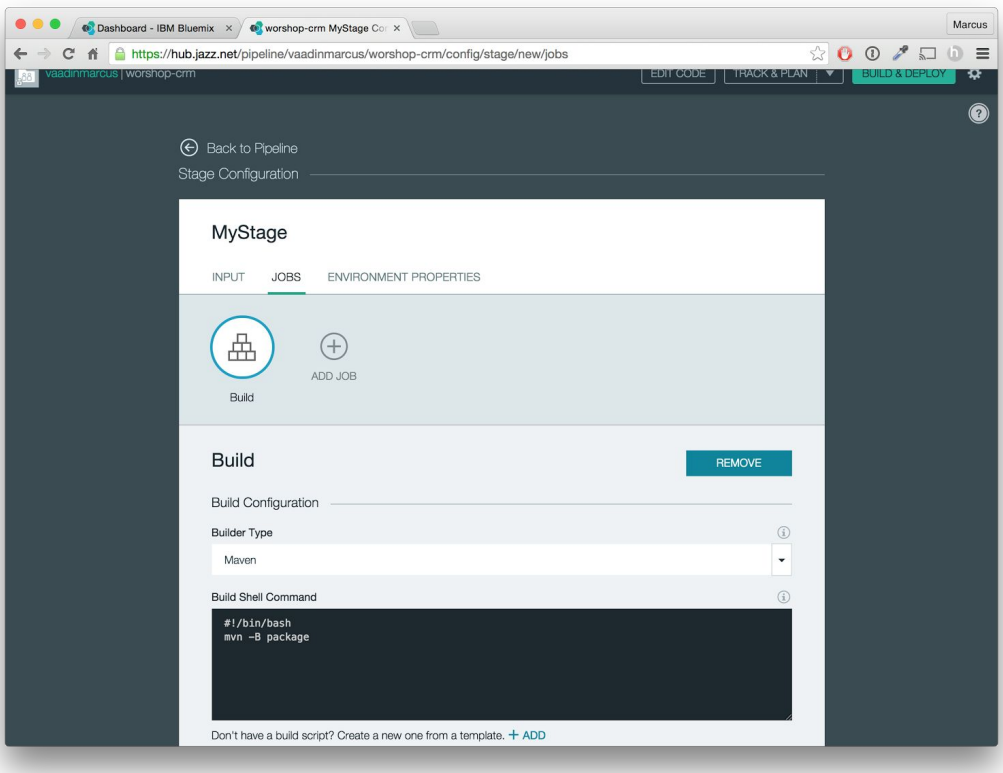
5. Select the ‘advanced’ option. Now click ‘add builder’.



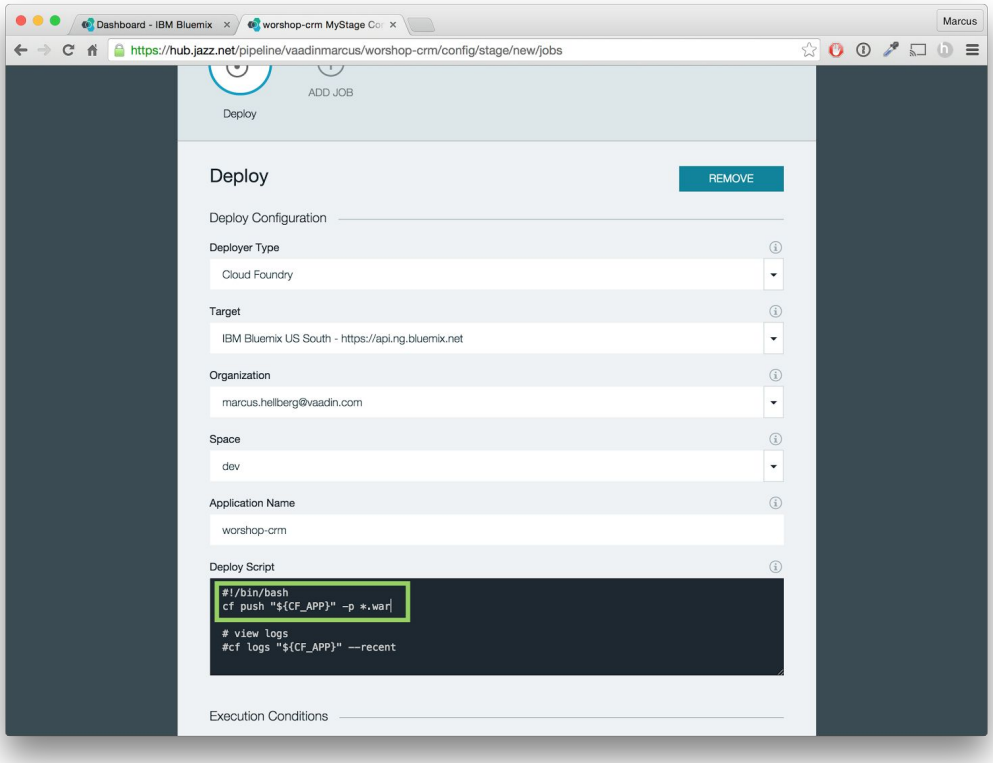


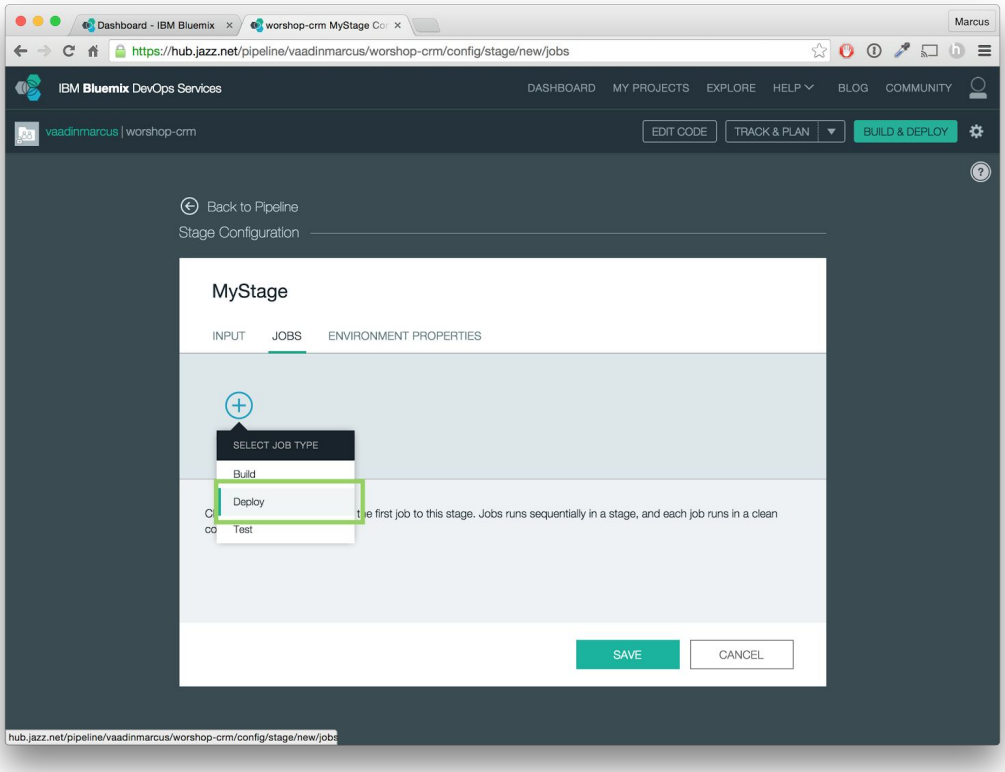
6. Select Maven for builder and click save.



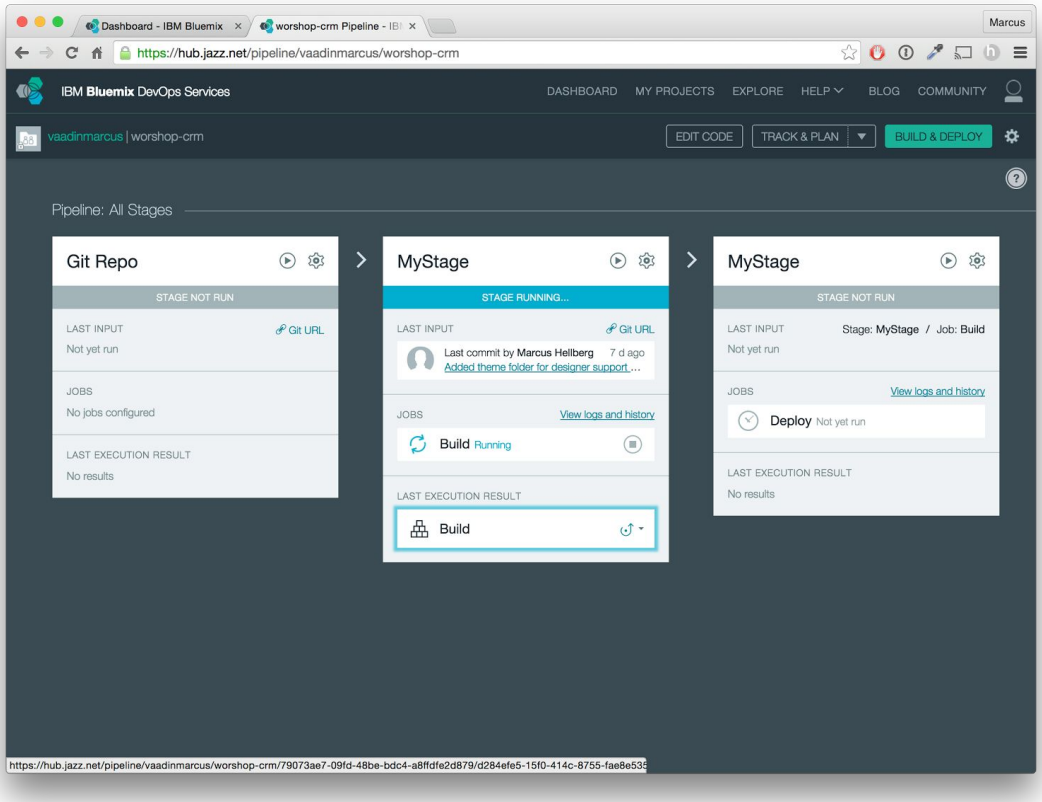


7. Now click ‘request build’ and wait until it finishes. You can also click the build in progress to see a console output of the progressing build process. This first build will take quite a while as all maven dependencies are downloaded etc.
8. After a successful build click the ‘add a stage’ option.
9. With different stages you could easily maintain e.g. nightly-build deployments or testing deployments. In this exercise we just want a direct deployment. Please edit the script to match the screen below and click save.

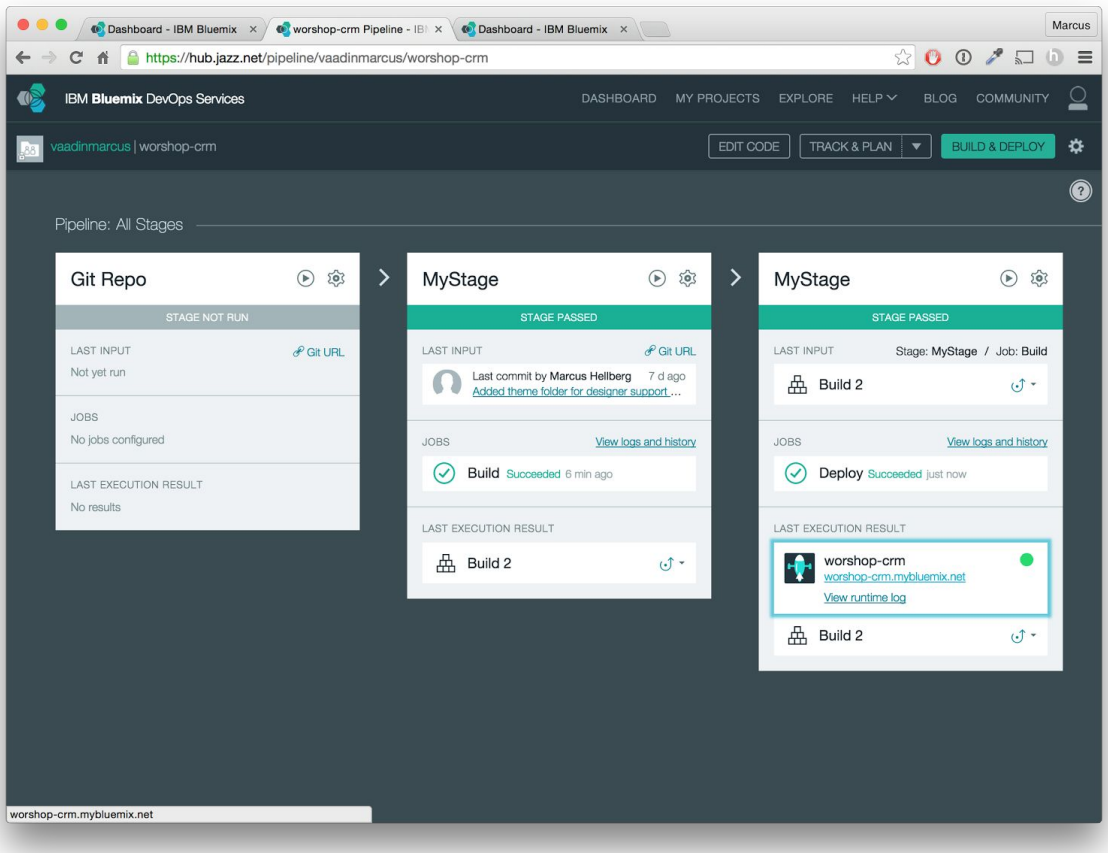

Dashboard - IBM Bluemix
worshop-crm MyStage Co
Marcus
https://hub.jazz.net/pipeline/vaadinmarcus/worshop-crm/config/stage/new/jobs
Deploy
ADD JOB
Deploy
REMOVE
Deploy Configuration
Deployer Type
Cloud Foundry
Target
IBM Bluemix US South - https://api.ng.bluemix.net
Organization
marcus.hellberg@vaadin.com
Space
dev
Application Name
worshop-crm
Deploy Script
#!/bin/bash
cf push "${CF_APP}" -p *.war
view logs
#cf logs "${CF_APP}" --recent
Execution Conditions


Dashboard - IBM Bluemix
worshop-crm MyStage Co
Marcus
https://hub.jazz.net/pipeline/vaadinmarcus/worshop-crm/config/stage/new/jobs
IBM Bluemix DevOps Services
DASHBOARD
MY PROJECTS
EXPLORE
HELP
BLOG
COMMUNITY
vaadinmarcus | worshop-crm
EDIT CODE
TRACK & PLAN
BUILD & DEPLOY
Back to Pipeline
Stage Configuration
MyStage
INPUT
JOBS
ENVIRONMENT PROPERTIES
SELECT JOB TYPE
Build
Deploy
Test
SAVE
CANCEL
hub.jazz.net/pipeline/vaadinmarcus/worshop-crm/config/stage/new/jobs

10. Now drag the latest build on top of the stage you just created.

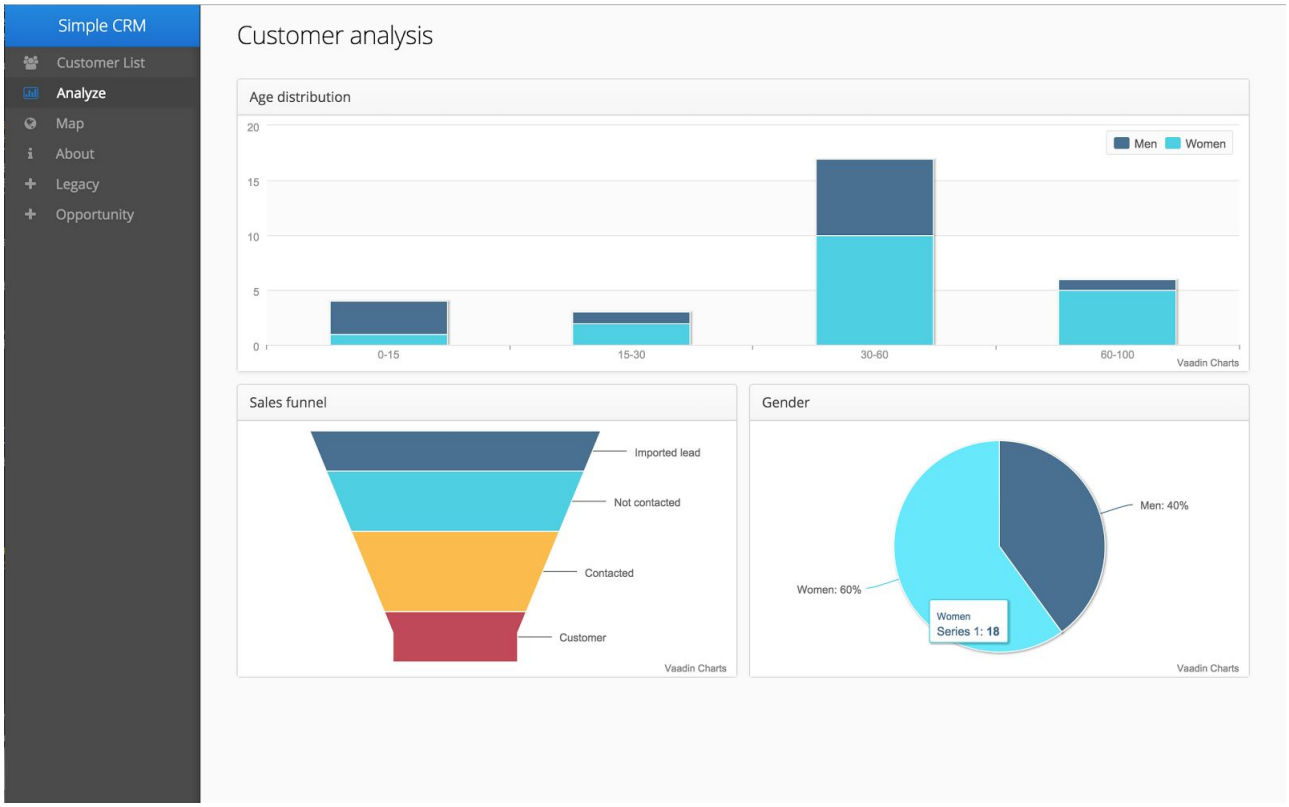


11. After a successful deployment you simply click the link to your application.
12. Now you have your new Vaadin application running in Bluemix cloud! Feel free to



explore

your application, fill in some test data, try it with your mobile phone etc etc.



Creating a local development environment

In this this step you'll set up the project for further development in your local Eclipse using a Derby database (you can of course use other databases if you prefer to do so). With this approach you will be pushing changes to your Git repository and your Bluemix instance will get updated directly from Git.

1. Open the command line and navigate to a location where you wish to have your project folder.
2. Check what is your repository Git URL and execute git clone. For my project the command is: `git clone https://hub.jazz.net/project/vaadinmarcus/simple-crm`. This will create a folder 'simple-crm' and download all resources inside it.
3. Open your Eclipse and choose Import > Existing Maven Project.
4. In the import dialog navigate to your newly created project folder. Eclipse should find the pom-file for your project. Select it and click finish.
5. If you get errors at this point about the Maven configuration you can safely ignore them. Maven integration for Eclipse isn't actually that smart and in some cases it doesn't recognize all features included in your pom-files.
6. Make sure that you have a JDK installed on your machine. Otherwise the maven build won't work.
7. Now run Maven install for your project. In my case right-click simple-crm > Run as > Maven Install.
8. Check that you have a local Liberty Profile server inside your Eclipse. If not, find the 'Servers' view from Eclipse and select 'new'. Follow the instructions on screen.
9. Download and place a derby jar file from http://db.apache.org/derby/ to usr/shared/resources/derby/derby.jar into your Liberty server directory.
10. Edit the server.xml of your Liberty Profile server by adding the following snippet. most likely usr/servers/defaultServer/server.xml in your Liberty server directory. This will create the derby database for your project. The snippet is mentioned also in your project's README.md file so you can safely copy-paste it from there.

```
<!--  JDBC Driver configuration -->
<jdbcDriver id="DerbyEmbedded" libraryRef="DerbyLib" />
<library id="DerbyLib" filesetRef="DerbyFileset" />
<fileset id="DerbyFileset" dir="${shared.resource.dir}/derby" includes="derby.jar" />
<!-- Configure an in-memory db for the vaadin app configuration -->
<dataSource id="jdbc/vaadindb" jndiName="jdbc/vaadindb" jdbcDriverRef="DerbyEmbedded" transactional="true">
  <properties databaseName="memory:jpasampledatabase" createDatabase="create" />
</dataSource>
```

11. Now you are ready to test your environment. Right-click your project: Debug as > Debug on Server. Choose the Liberty Profile server.
12. After the server has started your application should be available at http://localhost:9080/vaadin-jpa-application/
13. If you wish you can change the context from ‘vaadin-jpa-application’ to something else in your server.xml configuration.
14. Your local development environment is now all set up and ready!

Verifying the development cycle from local Eclipse to Bluemix

1. Let’s do a small modification. Find again the CustomerForm class and edit the notification message to something else.
2. Right-click your project and Team > Synchronize Workspace
3. Select the CustomerForm.java from changes and select ‘commit...’
4. Give a commit message and select ‘Commit and push’
5. Give your hub.jazz.net credentials to Eclipse if requested
6. Now you can go again to hub.jazz.net and monitor how your automatic build & deploy works.
7. Sit back and wait for changes to come on-line. Enjoy.