



Progress Report

RubberDuckys

Tawfic Jobah,
Dylan Gonzalez,
Aldair Martinez

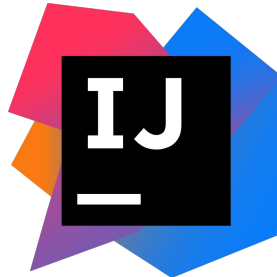
Programs required to set up

VS Code: For the frontend UI

IntelliJ: For the backend logic

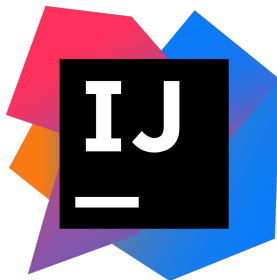
Java: Downloaded for program logic

Github: Cloned repository of the previous group's code



Backend (IntelliJ)

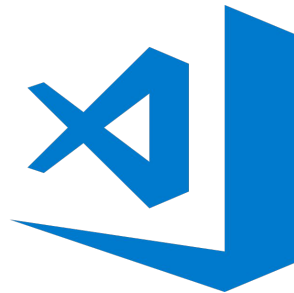
- IntelliJ was our IDE of choice
- Used to run the backend portion of the program
 - Sets up Corda
- We ran these commands:
 - `./gradlew deployNodes`: used to set up the nodes
 - `.\build\nodes\runnodes.bat`: used to run the nodes



Frontend (VS code)

- Used to run the frontend UI portion of the program
- We ran these commands:
 - npm install: installs necessary packages
 - npm start: executes scripts

- Once the packages are downloaded and start command is executed, a webpage is generated.



The results should look like....

The screenshot displays a dashboard for 'Smart Injection'. At the top, a purple header bar contains a hamburger menu icon, the text 'Smart Injection', and three utility icons: an envelope, a question mark, and a user profile. Below the header, there are two main management cards. The left card, 'Projects Management', features a pencil icon and a description: 'Choose an option to manage your projects.' It contains two buttons: a yellow 'CREATE NEW PROJECT' button and a purple 'PENDING PROJECTS' button. The right card, 'Wells Management', features a star icon and a description: 'Choose an option to manage your wells.' It contains two buttons: a yellow 'CREATE NEW WELL' button and a purple 'MY WELLS' button. Below these cards is a section titled 'My UIC Projects' which contains a table with three columns: 'Project Name', 'Project Status', and 'UIC Project Number'.

Smart Injection

Projects Management
Choose an option to manage your projects.

CREATE NEW PROJECT

PENDING PROJECTS

Wells Management
Choose an option to manage your wells.

CREATE NEW WELL

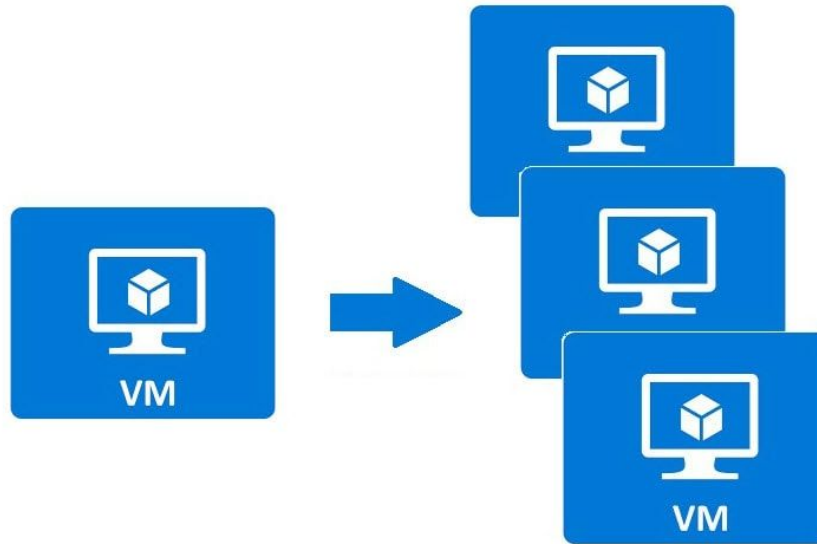
MY WELLS

My UIC Projects

Project Name	Project Status	UIC Project Number
--------------	----------------	--------------------

The next step...

Now that we have the program working on our local machine, the next step is to get it working on a virtual machine.



Azure

- Platform to host virtual machine
 - Currently backend and front end are operating on this virtual machine.
- Specs of VM
 - 2 Virtual Processors @ 2.29 GHz and 8gb of RAM



Results

The image displays a development environment with two main components: a terminal window showing Corda node startup logs and a web browser displaying a 'Smart Injection' dashboard.

Terminal Window (Top Left): Shows the startup of a Corda node. The logs include the following information:

- Listening for transport dt_socket at address: 5007
- There was an earthquake in California, a local bank went into default.
- Corda Open Source 4.5 (461cf07)
- Dolokia: Agent started with URL: http://127.0.0.1:7006/dolokia/
- Log location: C:\Users\Vanah\Documents\Projects\smartinjection\build\nodes\PartyAllog
- Advertised P2P messaging addresses: localhost:10005
- RPC connection address: localhost:10006
- RPC admin connection address: localhost:10046
- Loaded 2 CorDapp(s): Contract CorDapp: Template CorDapp version 1 by vendor Corda Open Source with license Apache License, Version 2.0; Workflow CorDapp: Template Flow version 1 by vendor Corda Open Source with license Apache License, Version 2.0
- Node for "PartyA" started up and registered in 44.41 sec
- Running P2P messaging loop
- Welcome to the Corda interactive shell. You can see the available commands by typing 'help'.
- Fri Oct 29 07:57:02 UTC 2021

Terminal Window (Bottom): Shows the compilation and execution of the web application:

```
clients > src > main > java > com > template > webserver > Controller.java
44 http://localhost:4200/req/project-review"
45 }
46 }
47 }
48 @RequestMapping("/") // The paths for HTTP requests are relative to this base path.
49 public class Controller {
50     private final CordaRPCOps proxy;
51     private final static Logger logger = LoggerFactory.getLogger(Controller.class);
52 }
53 public Controller(NodeRPCConnection rpc) {
54     this.proxy = rpc.proxy;
55 }
56 }
57 @Configuration
58 class Plugin {
59     // ...
60 }
```

Web Browser (Right): Shows the 'Smart Injection' dashboard. The 'My Wells' table contains the following data:

Well Name	Lease	Location Type	Location	Project Name
Name	1	NAD27	3.0.1	N/A

© All rights reserved to Vanah Vo & Angela D. Tante - 2021

Next phase

- setting up the program on AWS instances
- Test run of instances using Heroku, UI is hosted, but Corda nodes require more setup

