

CrowSoft's Processes, Team and Environments (Version 1.1)

Contents

CrowSoft's Processes, Team and Environments (Draft)	0
Introduction	0
Agile Process using Scrum.....	1
Sprint Planning	1
Daily Scrum	1
Sprint Reviews.....	1
Sprint Retrospective.....	1
Agile (Scrum) Key Artifacts.....	2
DevOps Process and Environments	3
DevOps Process.....	3
DevOps Environments.....	3

Introduction

As we are starting as a brand new Scrum team, to successfully deliver the new CrowSoft product for developing an online system for business intelligence creating building costs and analysis according to CrowSoft's business requirements, it will be vitally important to define our Agile and DevOps processes, setting up the team (including roles), how we will work together, what tools we need to set up our environment, to collaborate, and finally the architecture and design for our solution.

In this document, you will find an outline of the proposed Scrum process, roles required per process item, artifacts required and DevOps process and environments required

The outcome of our first spring planning meeting was as follows:

- Define overall processes for CI/CD, including what environments is required for the new CrowSoft Solution
- Define and select tools for Continuous Integration
- Define and select tools for Continuous Deployment
- Branching Structure on Github
- ***NOTE: The team was going to add more stories to this list on Jira over the weekend before the first Sprint Kick off on Monday, 4th March 2019.***

Updated: 02/04/2019

Created by: Charles Aylward

Agile Process using Scrum

Sprint Planning

- Hosted by Scrum Master (Alternate each week)
- Select highest priority items on Product Backlog and team turns item into Sprint Backlog
- Estimate Sprint Backlog in hours
- Work breakdown
- Declare sprint goal

Daily Scrum

- Hosted by Scrum Master (Twice a week)
- Roles required
 - PO – Ruth Lennon
 - Scrum Master (alternate)
 - Development Team (As per Slack List for CrowSoft):
 - UX, Design, Development, QA/Testing, Delivery
- 30 Mins Standup – Monday 9:30pm, Friday 9:30am
- Not for problem solving (Let's use Slack for problem solving)
- 3 Question:
 - What did you do?
 - What will you do?
 - What's in your way?

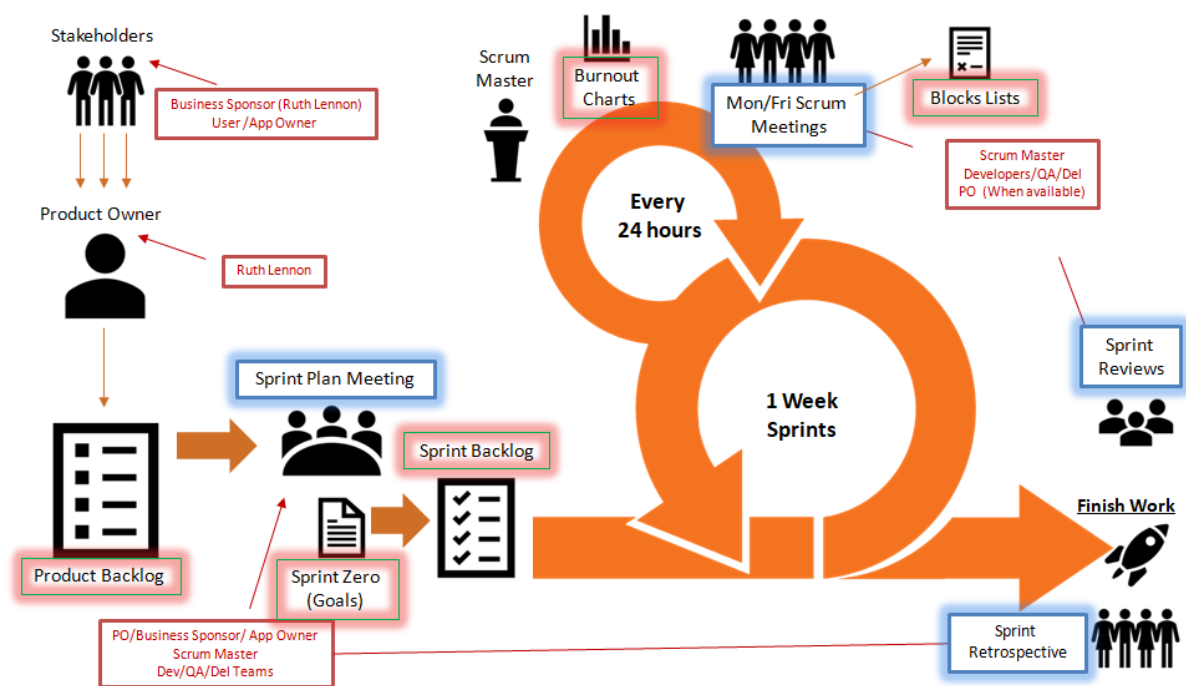
Sprint Reviews

- Hosted by Scrum Master
- After each sprint
- Accomplishments
- Whole Team present
- Demo shippable product/features

Sprint Retrospective

- Hosted by Scrum Master
- Discuss "Start Doing", "Continue Doing", "Stop Doing"

Figure 1 - Scrum Process



Agile (Scrum) Key Artifacts

Product Backlog

- User Stories (Epics in Jira)
- Desired Work
- Prioritized by Product Owner

Sprint Zero (Goal)

- High Level context Diagram (CrowSoft Solution)
- Summary of work in sprint
- Declared by Product Owner
- Accepted by Team

Sprint Backlog

- Team select work to do
- Owned / manage by team
- Estimated work remaining updated daily

Blocks List

- List of blocks /unmade decisions
- Owned by Scrum Master
- Updated daily

Burndown Chart

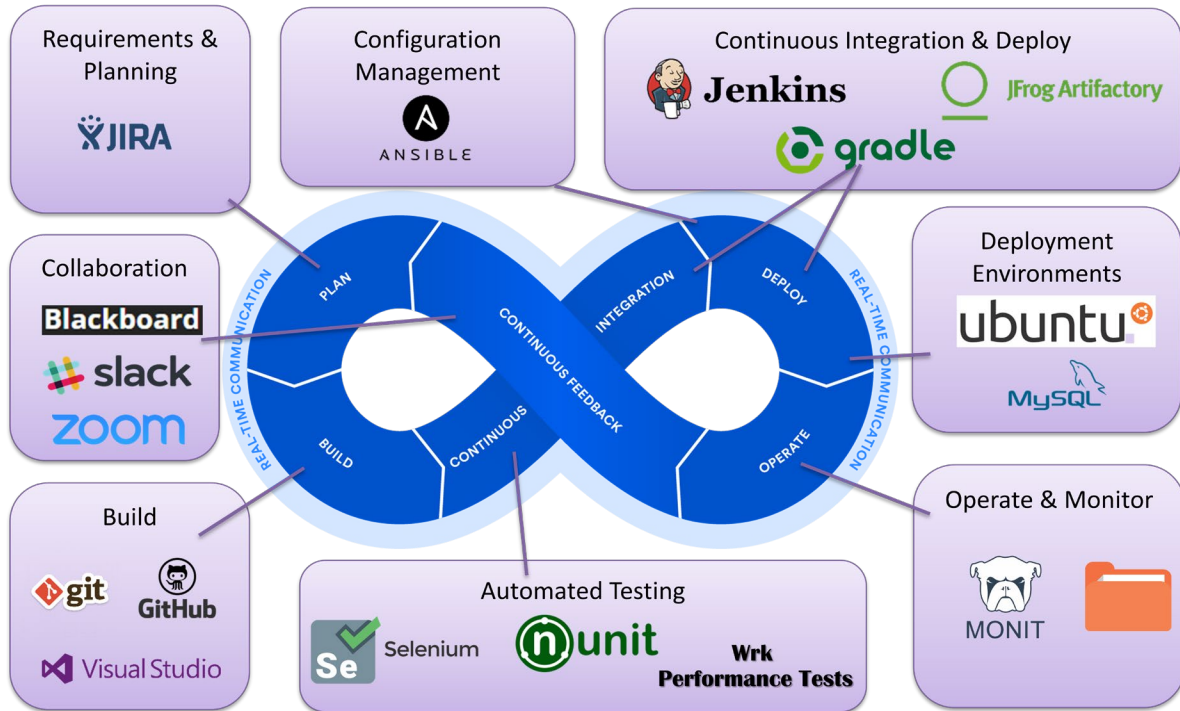
- Effort spend over time
- Stories / features completed

DevOps Process and Environments

DevOps Process

DevOps is a software engineering culture and practice that aims at unifying software development (Dev) and software operation (Ops).

Figure 2 - DevOps Process



Source: 1: <https://en.wikipedia.org/wiki/DevOps>

DevOps Environments

We need to select tools and environments to be configured for the following areas:

- Plan
 - Requirements & Planning (Jira, Slack, Blackboard, Zoom)
 - Architecture & Design (Draw.io)
- Build
 - Language tools (Visual Studio Code or Visual Studio Community 2017)
 - Unit Testing (Nunit)
 - Source Control (Git, GitHub)
 - Automated Testing (Selenium)
- Continuous Integration (Jenkins)
- Deploy
 - Configuration Management (Nuget, Ansible)
 - Artifact Management (jFrog Artifactory)
 - Deployment Environments (Linux Ubuntu 16.04)
- Operate
 - Operate and Monitor (Monit)

Updated: 02/04/2019

Created by: Charles Aylward

- Continuous Feedback (Slack, Jira, Zoom, Blackboard)

Ansible

Introduction

Ansible is a universal language, unraveling the mystery of how work gets done. Turn tough tasks into repeatable playbooks. Roll out enterprise-wide protocols with the push of a button. Give your team the tools to automate, solve, and share.

WHAT IS ANSIBLE?

Ansible is powerful IT automation that you can learn quickly. It's simple enough for everyone in your IT team to use, yet powerful enough to automate even the most complex deployments. Ansible handles repetitive tasks, giving your team more time to focus on innovation.

Simple. Powerful. Agentless.

With Ansible you can start to do real work in just minutes due to its simple, human-readable language. Altogether its powerful capabilities allow orchestration of your entire application lifecycle regardless of where it's deployed. And Ansible's agentless architecture means it is one less thing to keep secure.

Control. Knowledge. Delegation.

Red Hat® Ansible® Tower expands IT automation to your enterprise adding control, knowledge, and delegation on top of Ansible's automation engine.

Ready to Accelerate?

The answer is clear. We need to stretch beyond the boundaries of manual effort. When you automate, you accelerate. And when you build a culture of automation across your company, anything is possible. Ansible helps smart people do smarter work by having automation do the rest.

Download

The following is a link to an open-source version of Ansible: <https://www.ansible.com/>

Benefits

Automate: Deploy apps. Manage systems. Crush complexity.

Accelerate: Solve problems once and share the results with everyone.

Collaborate: Break down silos, create a culture of automation.

Integrate: Automate the technologies you already use.

Jenkins

With the Ansible Plugin in Jenkins, this brings the automation integration support into Jenkins. This integration allows the build jobs to deploy to different servers and automate different jobs within the server.

Video: <https://www.ansible.com/resources/videos/quick-start-video>

Artifactory

Introduction

Artifactory is a product by JFrog that serves as a binary repository manager. It is a natural extension to the source code repository, as it stores the outcome of the build process called artifacts. This will store individual application components that can later be assembled into a full product, allowing a build to be broken into smaller chunks making more efficient use of resources.

Download

The following is a link to an open-source version of Artifactory from JFrog

<https://jfrog.com/open-source/>

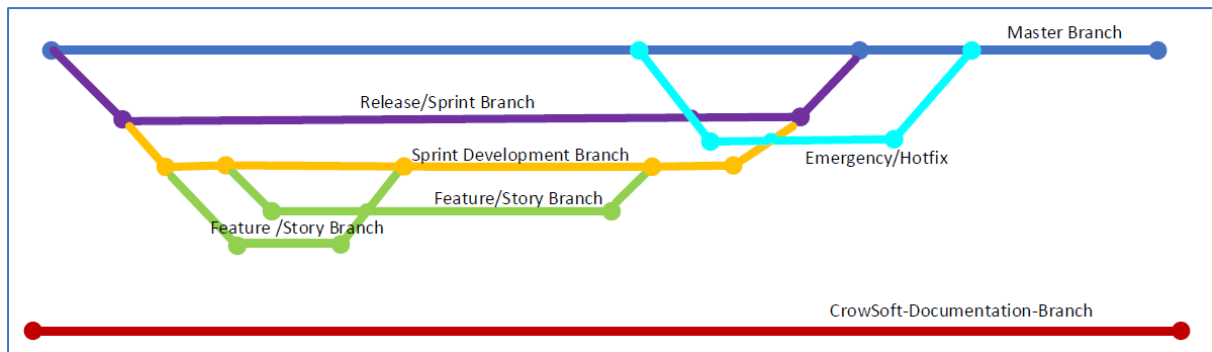
Benefits

The binary repository allows to host all the artifacts in one central location making the management much simpler for teams. This fits into the DevOps mentality of Build once , Deploy always. After the binary is sent to artifactory it can be deployed into different platforms. Confirming that the code that works in Dev, and then pushed to Prod, will work there also. Artifactory provides an end-to-end automated solution for tracking artifacts from development to production. One of the main benefits of running builds through Artifactory is, fully reproducible builds

Jenkins

With the Artifactory Plugin in Jenkins, this brings the artifactory's build integration support into Jenkins. This integration allows the build jobs to deploy artifacts and resolve dependencies to and from Artifactory and then have them linked to the build job that created them.

CrowSoft Branching Strategy



- Master Branch complete code running on Production (no changes pushed to master)
- Release /Sprint branch will be taken from the Master prefixed by release and sprint identifier
- Each Sprint/Release will have a Development branch branched from the Release.
- Each Feature/Story for a Release will be branched from the Development Branch.
- When a feature is complete, tested and peer reviewed a pull request is made to the Development Branch.
- When all development and testing is complete a pull request will be made to the Release Branch from Development.
- When Release is fully verified as complete a pull request will be made to the Master Branch From the Release Branch.
- **Emergency/hotfix branch** to be taken from Master with pull request made to Master for hotfix
- **CrowSoft-Documentation-Branch** is for documentation only not to be merged with the Master Branch

Build Server configuration

Op System: Ubuntu 16.04

IP address : 172.28.25.123

Install Java:

Download Site:

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

Version to download: `jdk-11.02_linux-x64_bin.tar.gz`

`x64` for 64 bit systems and `tar.gz` for Ubuntu Op Systems

Steps:

1. create a directory in the `/usr/lib` directory called java

Cd Downloads directory

```
/Downloads$ sudo tar -xzf jdk-11.02_linux-x64_bin.tar.gz -C /usr/lib/java
```

`X` = extract, `v` = verbose, `z` = gzip/unzipping `f` = extract in filesystem `tar` = to create and extract tape archives

`-C` specifies a different directory other than the current working directory

2. Add the file path for java to the `/etc/environment` file as `JAVA_HOME`

```
cd /etc
```

```
sudo nano environment
```

#Edit the file with this line.

```
JAVA_HOME="/usr/lib/java/jre1.8.0_201"
```

3. Next show Ubuntu a link to java

```
sudo update-alternatives --install "/usr/bin/java" "java" "/usr/lib/java/jre1.8.0_201/bin/java" 1
```

Set this version of java as the default

```
sudo update-alternatives --set java /usr/lib/java/jre1.8.0_201/bin/java
```

4. To check that java is install

```
java -version This should return:
```

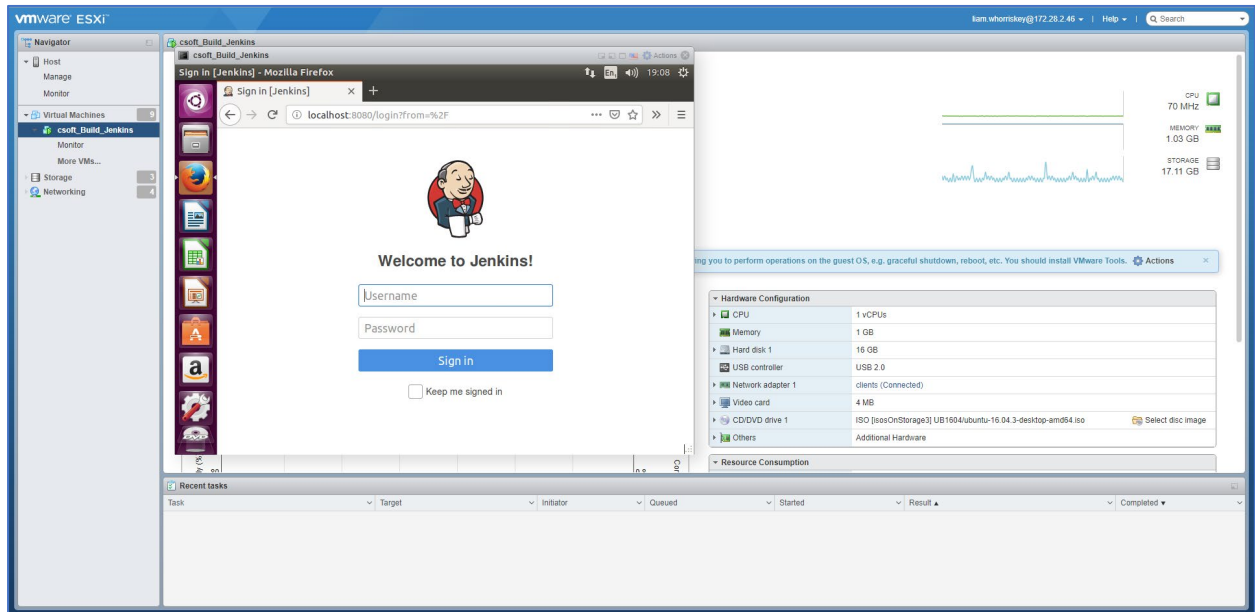
```
java version "1.8.0_201"
```

```
Java(TM) SE Runtime Environment (build 1.8.0_201-b09)
```

```
Java HotSpot(TM) 64-Bit Server VM (build 25.201-b09, mixed mode)
```

@Authors
Mary Walsh McGinty
Liam Whorriskey

Jenkins



Jenkins Setup Complete with suggested plugins.

- Name: Dev Ops
- User: admin
- Pass: *****19
- email: 100113360@student.lyit.ie
- Jenkins Url: <http://localhost:8080/>

Plugins Installed

- Ant Plugin
- Build Timeout
- Email Extension Plugin
- GitHub Branch Source Plugin
- Gradle Plugin
- LDAP Plugin
- Matrix Authorisation Strategy Plugin
- SWASP Markup Formatter Plugin
- PAM Authentication Plugin
- Pipeline
- Pipeline: GitHub Groovy Libraries
- SSH Slaves Plugin
- Subversion plugin
- Timestamper
- Workspace Cleanup Plugin
- Artifactory plugin
- Audit log
- Backup plugin

@Authors
Mary Walsh McGinty
Liam Whoriskey

Jenkins Security

- Added Users with permissions
- Disabled “Remember Me” option
- Set up Matrix-based security

Artifactory

Install: JFrog Artifactory-OSS Version 5.8.3. An open source version of JFrog Artifactory

Created: Maven, Gradle and Generic Repositories

Set up: Users with permissions

Install Gradle:

After Java JDK has been installed install Gradle.

Steps

1. Download the zip Gradle file

```
cd /tmp
```

```
wget https://services.gradle.org/distributions/gradle-4.10.2-bin.zip
```

This will download the file into the tmp directory

```
sudo unzip -d /opt/gradle /tmp/gradle-*.zip
```

This extracts the file into the /opt/gradle/gradle-4.10.2 directory

2. Configure Ubuntu Environment Variables

```
sudo nano /etc/profile.d/gradle.sh
```

This creates a new file called gradle.sh in the /etc/profile.d directory. Inside this file the following is added.

```
export GRADLE_HOME=/opt/gradle/gradle-4.10.2
```

```
export PATH=${GRADLE_HOME}/bin:${PATH}
```

Exit and save the file ‘ctrl + x’ then y.

When done run the following commands to make the file executable.

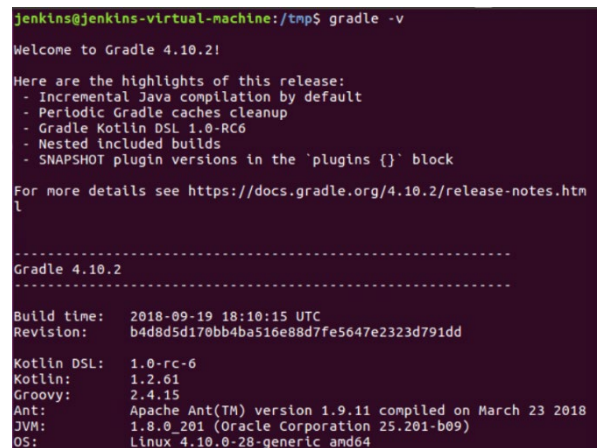
```
sudo chmod +x /etc/profile.d/gradle.sh
```

```
source /etc/profile.d/gradle.sh
```

3. To check if Gradle installed correctly

```
gradle -v
```

It should return this →



```
jenkins@jenkins-virtual-machine:/tmp$ gradle -v
Welcome to Gradle 4.10.2!

Here are the highlights of this release:
- Incremental Java compilation by default
- Periodic Gradle caches cleanup
- Gradle Kotlin DSL 1.0-RC6
- Nested included builds
- SNAPSHOT plugin versions in the 'plugins {}' block

For more details see https://docs.gradle.org/4.10.2/release-notes.html

-----
Gradle 4.10.2
-----

Build time: 2018-09-19 18:10:15 UTC
Revision: b4d8d5d170bb4ba510e88d7fe5647e2323d791dd

Kotlin DSL: 1.0-rc-6
Kotlin: 1.2.61
Groovy: 2.4.15
Ant: Apache Ant(TM) version 1.9.11 compiled on March 23 2018
JVM: 1.8.0_201 (Oracle Corporation 25.201-b09)
OS: Linux 4.10.0-28-generic amd64
```

@Authors
Mary Walsh McGinty
Liam Whoriskey

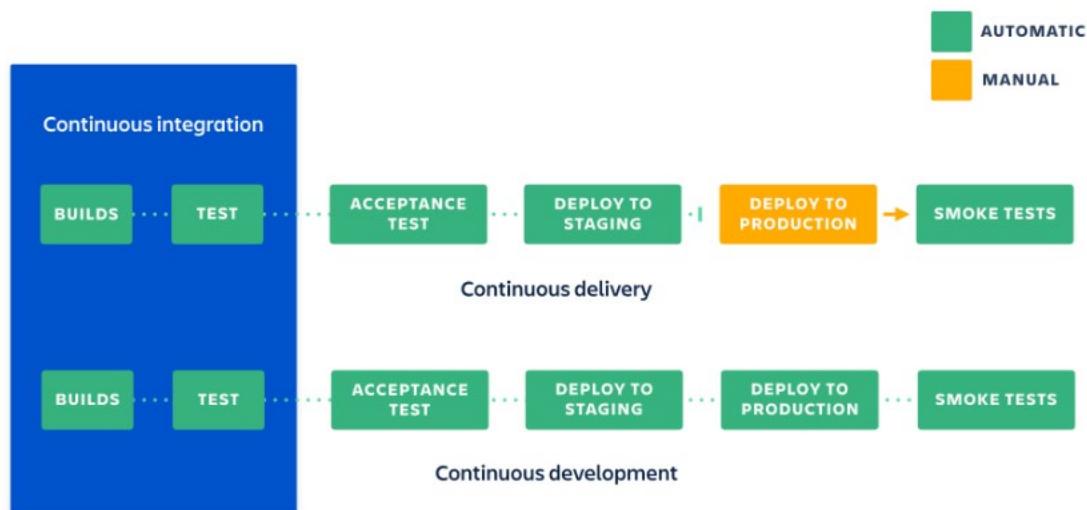
CI/CD

CI and CD are acronyms to represent modern development practices.

CI stands for continuous integration, which is a practice that focuses on developers integrating code regularly into a shared repository. Check-ins are verified by automated builds and testing to ensure bugs are caught often and early.

CD, on the other hand, can either mean continuous delivery or continuous deployment. Both are very similar but have a distinct difference. Continuous delivery is automating the release process by deploying changes on a schedule or at the click of a button. Continuous deployment on the other hand, deploys changes automatically after passing all stages of a production pipeline.

Continuous deployment and continuous delivery are very similar in practice and both are an extension of continuous integration in an automated DevOps environment.



Some tools for CD:

Many CI tools can be doubled up as deployment tools:

Jenkins:

Although Jenkins is primarily a CI server, it can be configured to be used as a deployment tool with the many plugins available to it.

Tools that are designed specifically for automated deployment:

Octopus:

Octopus is an automated deployment and release management server. It is designed to simplify deployment of ASP.NET applications, windows services and databases.

GoCD:

Go is an open-source tool to help in the continuous deployment of software. It supports several version control tools, including Git. Similar to Jenkins, plugins can be installed to extend its features.

These plugins support authentication and authorization software, version control, build tools, notification and chat tools and cloud computing providers.

Recommendation:

Although Octopus and Go are viable options: Octopus as it is the de facto for .NET applications and it integrates with Jenkins and GoCD as it is an open-source tool which can be extended with its plugins. Due to the familiarity with Jenkins, it is already being used as our CI server and with plugins we can extend its features for deployments I recommend we use Jenkins as our CD server.

CI/CD

CI and CD are acronyms to represent modern development practices.

CI stands for continuous integration, which is a practice that focuses on developers integrating code regularly into a shared repository. Check-ins are verified by automated builds and testing to ensure bugs are caught often and early.

CD, on the other hand, can either mean continuous delivery or continuous deployment. Both are very similar but have a distinct difference. Continuous delivery is automating the release process by deploying changes on a schedule or at the click of a button. Continuous deployment on the other hand, deploys changes automatically after passing all stages of a production pipeline.

Continuous deployment and continuous delivery are very similar in practice and both are an extension of continuous integration in an automated DevOps environment.

Some tools for CI:

Jenkins:

Jenkins is an open-source CI tool written in Java. Some of its main features are:

- Compatible with most OS and versions of Linux, Mac OS and Windows.
- Easy installation
- Easily set up and configured with its web interface
- Plugins – there are over 1500 plugins to

TeamCity:

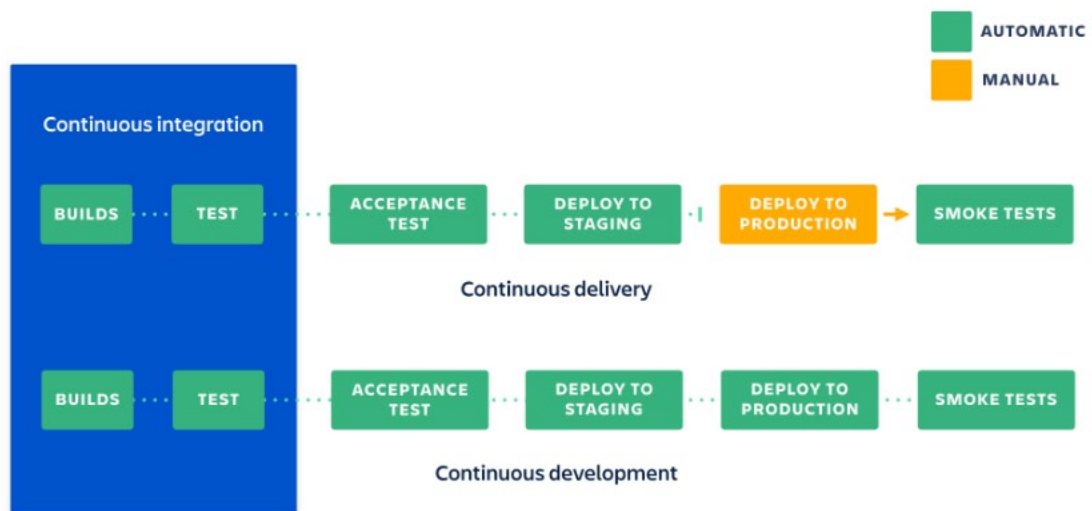
TeamCity, also written in Java, main features include:

- Configure builds in DSL
- Project level cloud profiles
- Remote run and pre-tested commit

Travis CI:

TravisCI is free for open source projects. Its main features include:

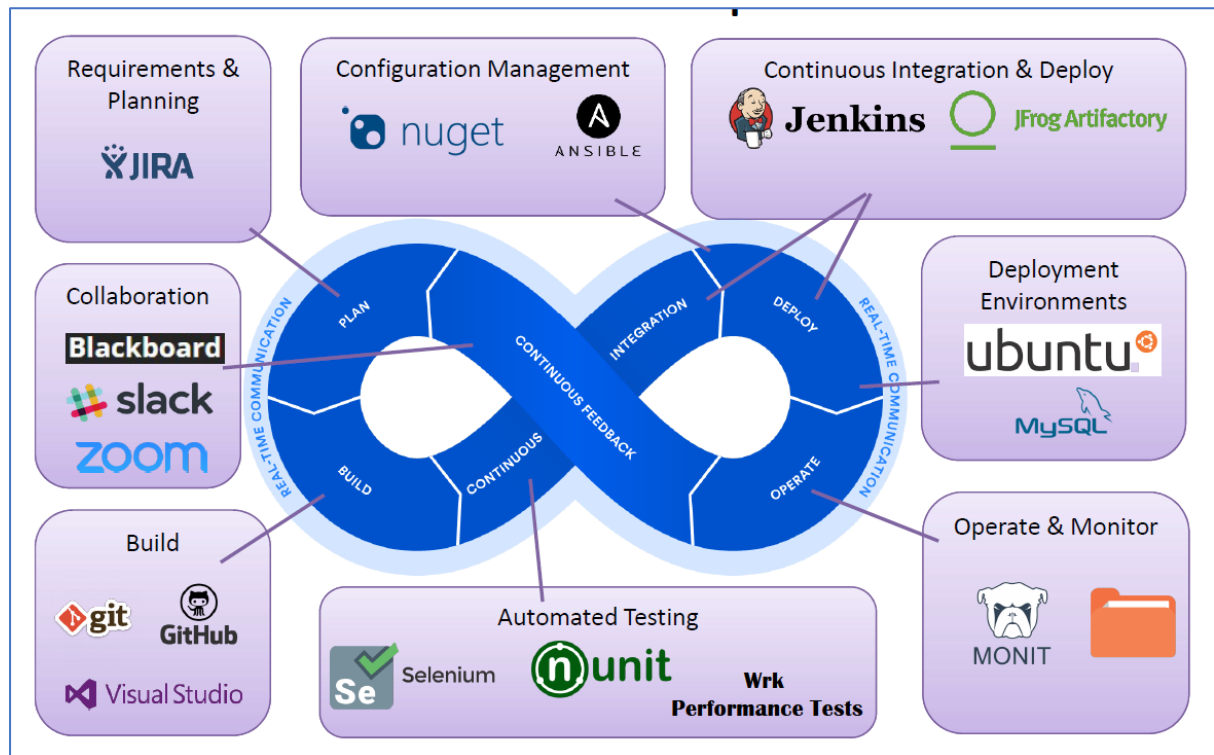
- Hosted – not dependant on any platform
- Parallel testing
- Easy setup, no installation



Recommendation:

Due to its flexibility, compatibility, ease of use and as an open-source tool, I recommend we use Jenkins as our continuous integration server.

CrowSoft DevOps Tools



CrowSoft Database Design

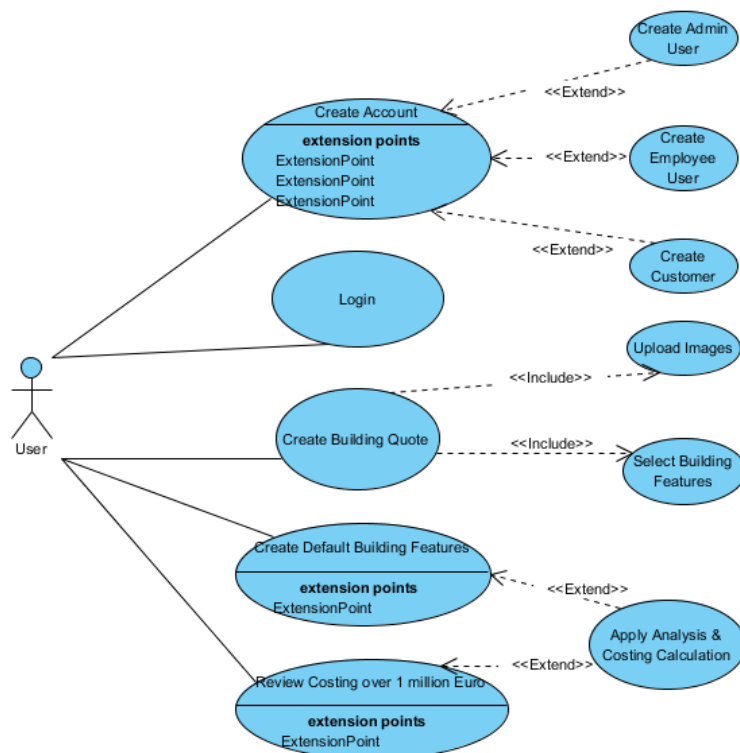
Date Created: 20/03/2019

Date Updated: 21/03/2019

CrowSoft Use Case

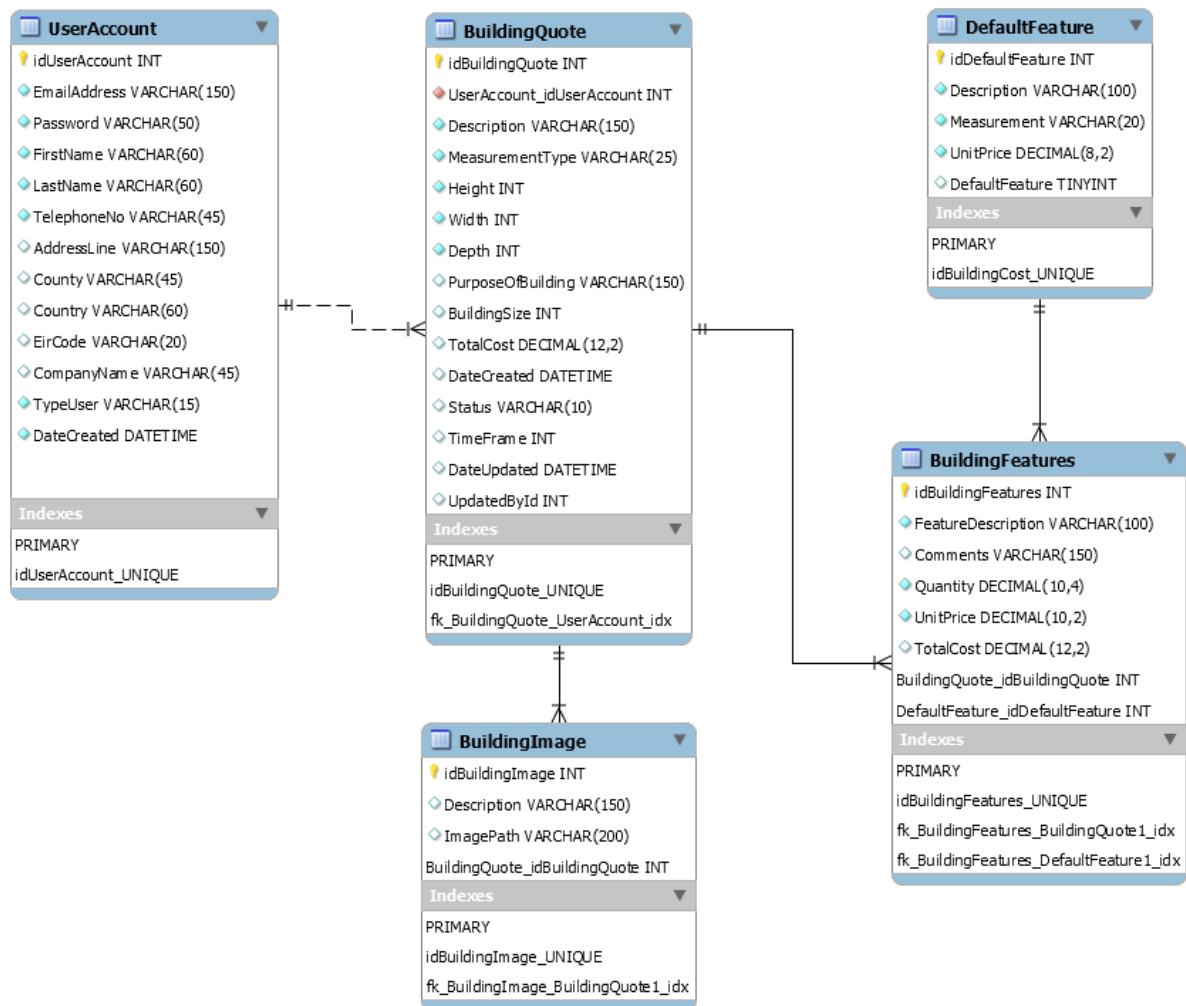
The use case below is base on CrowSoft functional requirements.

- Users require creating an account, and can either be an Admin, Employee or Customer user.
- User requires to login into the system and the system needs to validate what role the user belongs to
- A customer/client user can create a building quote for analysis and costing. The customer can upload multiple images and also select custom features to be added to the costs.
- The Admin user can add default building features for analysis and costing
- An admin user can review the building analysis and costing, update changes to the costs/features and apply the costing again, or approve the costs.



CrowSoft Database Table Structures

CrowSoft ERD



User Account Table

This table will be used for multipurpose users; for employees, admin users and customers. This will use a role-base security at three levels, admin (all access), employee user (access to view costing and building quotes), customer (allow to create building quote including features). Features on the UI will be visible according to the role of the user.

Column Name	Description	Mandator y	Sample Data	Validation	type
idUserAccount	Primary Key - Auto Increment	Y	1		int
EmailAddress	Email Address and Username	Y	joe@email.com	Valid Email Address	string

Password	Hash Password using Cryptography in ASP.Net Core before saving to DB	Y	\$1\$O3JMY.Tw\$AdLn	Validate Login	string
FirstName	First Name	Y	Joe	Not null	string
LastName	Last Name	Y	Blogg	Not null	string
TelephoneNo	Telephone No	Y	555-555-5555	Not null	string
AddressLine1	Address Line 1	N	123 Street		string
AddressLine2	Address Line 2	N	My Area		string
County	County	N	Donegal		string
Country	Country	N	Ireland		string
EirCode	Postal Code	N	F90 A1A1		string
CompanyName	CompanyName	Y	CrowSoft	Not null	string
TypeUser	Drop Down Box (Admin, Employee, Customer)	Y	Admin	Not null	string
DateCreated	Auto Generated by MySQL (CURRENT_TIMESTAMP')	Y	2019-02-31 20:00:00'	Not null	DateTi me

Default Feature Table

The default feature table is a configuration table for both default features to be added to each and every building quote, but also where a customer can select multiple features to be added to the quote and to be calculated.

Column Name	Description	Mandatory	Sample Data	Validation	type
idDefaultFeature	Primary Key - Auto Increment	Y	1		int
Description	Description of the Feature	Y	E.g. Concrete Walls, Kitchen, Tiled Roof	Not null	string
Measurement	Type of measurement used to calculate unit price	Y	E.g. Qty, Time, Sqr Meters, Sqr Foot	Not null	string
UnitPrice	Unit Price of feature	Y	€100.00	Not null	int
DefaultFeature	This is a Boolean field. If this field = 1, then this feature will be automatically be added to the total cost, if 0, then it will display as an option for the customer to select.	Y	1 or 0	Not null	tinyint

Building Quote Table

This is the header table when a customer creates a new quote for costing. A user/customer has to register for an account before they can create a quote for costing. A separate process will automatically run when a customer click confirms. This calculation will add up all default and customer features required and add it to the total cost.

Column Name	Description	Mandato ry	Sample Data	Validation	type
idBuildingQuote	Primary Key - Auto Increment	Y	1		int
UserAccount_idUserAccount	Many to One Foreign Key to UserAccess Table	Y	1	User has to be logged in	int
Description	Description of the building the customer requires	Y	Offices for Joe Blogg	Not null	string
MeasurementType	Drop Down for Metrics or Imperial	Y	Metric	Not null	string
Height	Height	Y	50	Not null	int
Width	Width	Y	200	Not null	int
Depth	Depth	Y	200	Not null	int
PurposeOfBuilding	Customer explains purpose of the building	N	Office Block		string
BuildingSize	Calculated from Height, Width and Depth	N		Auto calculation	int
TotalCost	System calculates total cost of all features	N	€130,000.00	If cost is over €1 mill, change status to Withhold	decimal
DateCreated	Auto Generated by MySQL (CURRENT_TIMESTAMP)	Y	2019-02-31 20:00:00'	Not null	datetime
Status	Status of the quote: Confirmed, Approved, Withhold, Cancelled	Y	Approved	Not null	string
TimeFrame	This is the time in months. Calculated from labour cost.	N	6	Auto calculation	decimal
DateUpdated	Updated when record is updated by Admin	N	2019-02-31 20:00:00'		datetime
UpdatedBy	Admin Id who updated the quote	N	1		int

Building Features Table

This table is to save the customer selected features to be added to the final costing.

Column Name	Description	Mandatory	Sample Data	Validation	type
idBuildingFeatures	Primary Key - Auto Increment	Y	1		int
Comments	Comments of whats required	N	Colour of the walls white		string
FeatureDescription	Copied from default feature select		Concrete	Auto generate from default features selected	string
Quantity	Qty required	Y	150	Not null	int
UnitPrice		Y	€50	Not null	decimal
TotalCost	Total cost calculated	N	€7,500	Auto calculate	decimal
BuildingQuote_idBuildingQuote	Foreign key for building quote	Y	1	Not null	int
DefaultFeature_idDefaultFeature	Foreign key for Default Feature	Y	1	Not null	int

Building Image Table

The customer will be able to upload multiple images and this is where the image path will be stored. This has a many to one relationship with the Building Quote header table.

Column Name	Description	Mandatory	Sample Data	Validation	type
idBuildingImage	Primary Key - Auto Increment	Y	1		int
Description	Image Description	Y	Building Plans		string
ImagePath	Path of the folder where images is stored	Y	d:/crowsoft/images	Auto generate filename	string
BuildingQuote_idBuildingQuote	Foreign key for building quote	Y	1	Not null	int

Security Tools for the DevOps Pipeline

Contents

Introduction	1
Security Development Tools	1
Pre-commit Version Control Security Tools.....	1
Continuous Integration Security Tools.....	2
Continuous Delivery Security Tools	2
Operation Security tools	3

Introduction

These are some of the available Open Source security tools for the DevOps pipeline. I have only included the tools that are going to be relevant to the project. I have included both Python and Java for an example. https://mattboegner.com/secure_cicd_pipeline_2/ This is an interesting article on securing the DevOps pipeline if anybody is interested in reading it.

Security Development Tools

Mittn: For Python development, allows the developer to create checks based on use cases. Developers can also code check to catch mistakes that they previously made. A security testing tool for CI. <https://github.com/F-Secure/mittn>

BDD-Security: For Java development, works the same way as Mittn. A testing framework used for functional security, infrastructure security and application security testing.

Jasmine: JavaScript

QUnit: JavaScript

Static and Dynamic code Analysis

SonarQube

OWASP Zap & OWASP Zapper (Jenkins plugin): Allow automating attack proxy to test for possible attacks

Pre-commit Version control Security

Git-hound: A free security tool to provide automated checks to ensure that no sensitive data is committed into code.

OWASP Threat Dragon: Building security into the design of the application. A threat modeling web application, including system diagramming.

SonarLint: An IDE extension (IntelliJ, Eclipse, Visual Studio). Helps to detect and fix quality issues as the code is written. Bug detection issues are detected and reported, pinpointing the problem, and gives corrective recommendations.

Puma Scan: An IDE extension (Visual Studio) that provides real time continuous source code analysis for C#, .NET. Vulnerabilities are displayed immediately inside the development environment.

Continuous Integration Security

ESLint: A static code analysis tool. Debugging is done by examining the port without executing the port. A linter tool for identifying and reporting on patterns in JavaScript. Helps to maintain code quality with ease.

Mocha: A JavaScript test framework running on NodeJS and in the browser making asynchronous testing simple and easy. Runs serially allowing for flexible and accurate reporting while mapping uncaught exceptions to the correct test cases.

OWASP Dependency Check: A utility that identifies project dependencies and checks if there are any known publicly disclosed vulnerabilities . A security audit tool with plugins for build-tools, Maven, Jenkins, gradle and Ant-task. It automatically updates itself.

Docker Bench: A script that checks for dozens of common best practices around deploying docker containers in production. Currently supports multiple versions of docker and docker-bench will determine the test set to run based on the docker version on the host machine. It scans the docker environments, start the host level and inspect all the aspects of the host, the docker daemon and its configuration.

Continuous Delivery Security. (Before, during and after Deployment)

SSL Labs-scan: A command-line scanning tool that doubles as the reference API client of SSL Labs. Security smoke tests. Scans web PKIs. Designed for automated or bulk testing. SSL Labs API exposes the complete SSL/ TLS server testing functionality in a programmatic fashion allowing for scheduled and bulk assessments.

OSQuery: Uses basic SQL commands to leverage a relational data-model to describe a device. Safety checks. When attackers leave a malicious process running but delete the original binary on the disk. This query returns any process whose original binary has been deleted, which could be an indicator of a suspicious process. Under Apache license.

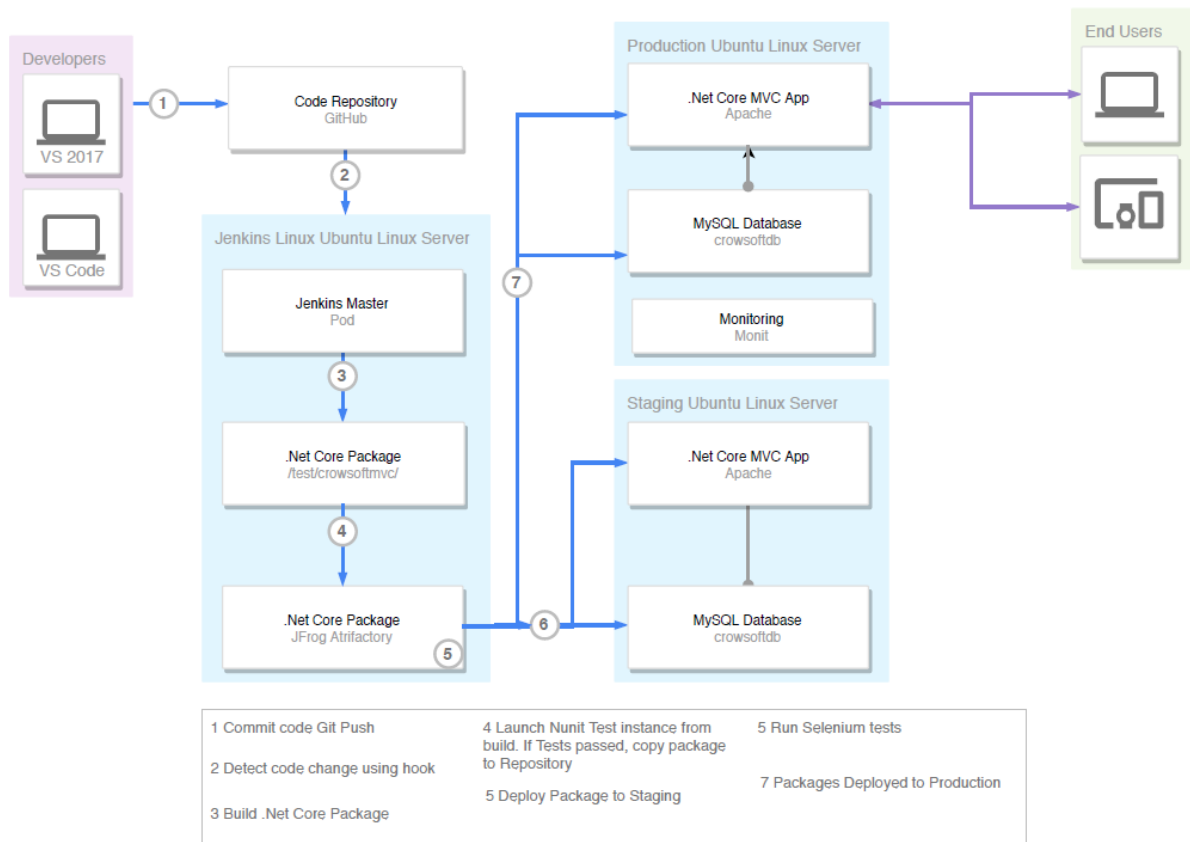
Ansible vault: A feature of Ansible that allows keeping sensitive data in encrypted files. These can then be placed in source code.

Samhain: A host-based intrusion detection system. It provides file integrity checking and log file monitoring/analysis as well as rootkit detection, port monitoring, detection of rogue SUID executables and hidden process. Designed to monitor multiple hosts with potentially different operating systems. It provides centralized logging and maintenance and can be used as a standalone application on a single host.

Operations Security (Continuous security, monitoring, testing, audit and compliance checks)

- Chaos Monkey:** It injects faults to randomly terminate virtual machine and container instances that run inside a production environment. Exposing engineers to failures more frequently in order to help them build more resilient services.
- Open SCAP:** An ecosystem which provides multiple tools to assist administrators and auditors with assessment, measurement and enforcement of security baselines. Provides a wide variety of hardening guides and configuration baseline. Enabling you to choose a security policy that best suits the needs of your organisation.
- Grafana:** A metric analytics and visualization suite for continuous monitoring. Used for time series data for infrastructure and application analytics. It allows to query, visualise, alert and understand the matrices no matter where they are stored. It creates an explore and share dashboard with your schemas.
- ElastAlert:** A tool for continuous monitoring. A simple framework for alerting on anomalies , inconsistencies, spikes or other patterns of interest from data in Elasticsearch. It works by combining Elasticsearch with two types of components, rule types and alerts. Elasticsearch is periodically queried and the data is passed to the rule type, which determines when a match is found. When a match occurs, it is given to one or more alerts, which take action based on the match. This is configured by a set of rules, each of which defines a query, a rule type and a set of alerts.

DevOps Pipeline



CrowSoft

The customer would like an on-line system to take in building details for analysis. CrowSoft is an engineering and tech company. The product that they wish to market is BusIntelligence. The system must take details for buildings and to provide easy to use analysis features. For example, they may wish to have a review all buildings which cost more than €1 million to build and have special features as this may indicate future maintenance projects that may be exploited. Keep the analysis simple. The analysis system should be clean and simple. The system needs to take into account the usual details and present simplified graphics. It must be possible to upload files or images. The administrator should be able to access detailed information and edit as appropriate. Once the client enters details it should not be able to be changed by the client.

1. Functional Requirements

Client/User Requirements

1. Sign-In/Sign-out

1.1 The Client/User shall enter sign-in credentials (username and password).

1.1.1 On successful sign-in the user shall be brought to the signed-in home page.

1.1.2 On unsuccessful sign-in the user shall be displayed an error and be brought back to sign-in page.

1.2 The Client/User shall sign-out and confirmation page/home page is displayed.

2. Create Account (Sign-up)

2.1 Client/User shall create an account via the sign-in page.

2.2 Client/User shall be able to enter in following sign-up credentials:

- First name
- Last name
- Telephone/Mobile
- Address line 1
- Address line 2
- County
- Country
- Eircode
- Email
- Company name

2.2.1 On successful sign-up the Client/User shall be redirected to the sign-in page.

2.2.2 On unsuccessful sign-up the Client/User shall be displayed an appropriate error message.

3. Enter Details of Buildings

3.1 Client/User can create a new entry

3.2 Client/User must be able to upload images from file

- 3.2.1 Client/User must be able to upload images from mobile device
- 3.3 Client/User selects measurement types from drop down boxes
 - 3.3.1.1 Select imperial or metric standard
 - 3.3.1.2 Select height, width, depth
 - 3.3.1.3 Select material type of building
 - 3.3.1.4 Select purpose of building
- 3.4 Client/User will view selected items and confirm/cancel
- 3.5 Client/User can select various analysis features (cost per sqm, buildings for purpose, age of building etc.)
- 3.6 Client/User will receive confirmation email when required analysis feature is activated, viewed and confirmed.
- 3.7 Client/User cannot edit details once confirmed

Admin Requirements

1. Sign-In/Sign-out

- 1.1 The admin shall enter sign-in credentials (username and password).
 - 1.1.1 On successful sign-in the admin shall be brought to the signed-in home page.
 - 1.1.2 On unsuccessful sign-in the admin shall be displayed an error and be brought back to sign-in page.
- 1.2 The admin shall sign-out and confirmation page/home page is displayed.

2. Add new client/user

- 2.1 The admin can create an account for new client/user
- 2.2 The admin shall be able to enters the following client/user details:
 - client/user ID
 - First name
 - Last name
 - Contact number
 - Address
 - Email
- 2.2.1 If client/user account is created successfully the admin shall be redirected to the admin homepage.
- 2.2.2 If employee account is unsuccessful an error message shall appear.

3. Edit client/user

- 3.1 The admin shall enter unique Employee ID.
- 3.2 Employee details page will appear, edit details and save any changes.
 - 3.2.1 A successful save will redirect the admin to the admin homepage.
 - 3.2.2 An unsuccessful save will prompt an error.

4. Edit client/user building details

- 4.1 The admin shall enter unique Client/User ID.
- 4.2 Client/user details page will appear, edit details and save any changes.
 - 4.2.1 A successful save will redirect the admin to the admin homepage.
 - 4.2.2 An unsuccessful save will prompt an error.

5. Remove client/user

- 5.1 The admin shall enter a client/user ID which they wish to remove.
- 5.2 The admin shall select a Remove client/user button.
- 5.3 A message prompt will ask to confirm, select yes.

- 5.3.1 A successful remove will redirect the admin to the admin homepage.
- 5.3.2 An unsuccessful remove will prompt an error.

2. Non-Functional Requirements

Availability

- 1. Application must be available 24 hours a day.
- 2. Application must be available 365 days a year.

Reliability

- 1. Application must always present the correct information for each user.
- 2. Application must analyse the correct information and display to screen.
- 3. Invalid inputs must be properly reported as such.
- 4. Application must always maintain data integrity
 - 4.1 Table relationships must always be consistent.
 - 4.2 Foreign key fields must agree with the primary key that is referenced by the foreign key.
 - 4.3 Ensuring that changes made to the database by authorized users do not result in the loss or conflict of data consistency.
- 5. The application failure rate shall be less than 1 failure per 1000 hours of operation.

Performance

- 1. Application must not exceed given amount of memory.
- 2. Web application must be performed within milliseconds.
- 3. Application must not lag or freeze when in use.

Usability

- 1. Layout should be simple and clean.
- 2. Buttons must be of equal size and shape.
- 3. Application interface must be visually appealing and easy to use.
- 4. Use at least font size 17 with contrasting backgrounds.

Scalability

- 1. Application must be able to work with different languages.

Compatibility

- 1. The application must be compatible on every browser i.e Chrome, Firefox, IE, etc.

Security

- 1. User application must lock login after 5 incorrect password attempts.
 - user must contact administrator to reactivate log in.
- 2. User data must be encrypted during application synchronization with back end database to provide secure transfer of sensitive information such as personal and financial details.

Marking Scheme:

An individual report is required from each learner to describe the project in terms of each of the areas below. Each team member must be familiar with each area regardless of who creates the original implementation. The project is small so that you can focus on Why more than How. Refer to the following draft marking scheme:

Desc	Pts	Desc	Pts
Performance Management Tool(s)	5	Agile dev/team collaboration	5
Unit Testing – automated Sonarqube or similar	5	Scrum	10
Jira	5	GIT & GitHub (pull – push)	5
Security Tools (Greenlight)	5	Staging/Production Environments	10
Artifactory	5	Conclusions	20
Automated documentation	5		
Code Created (include consideration for security, performance, etc.)	10		
Jenkins Pipeline	10		
Total	50		50

CrowSoft MySQL Script & Admin User

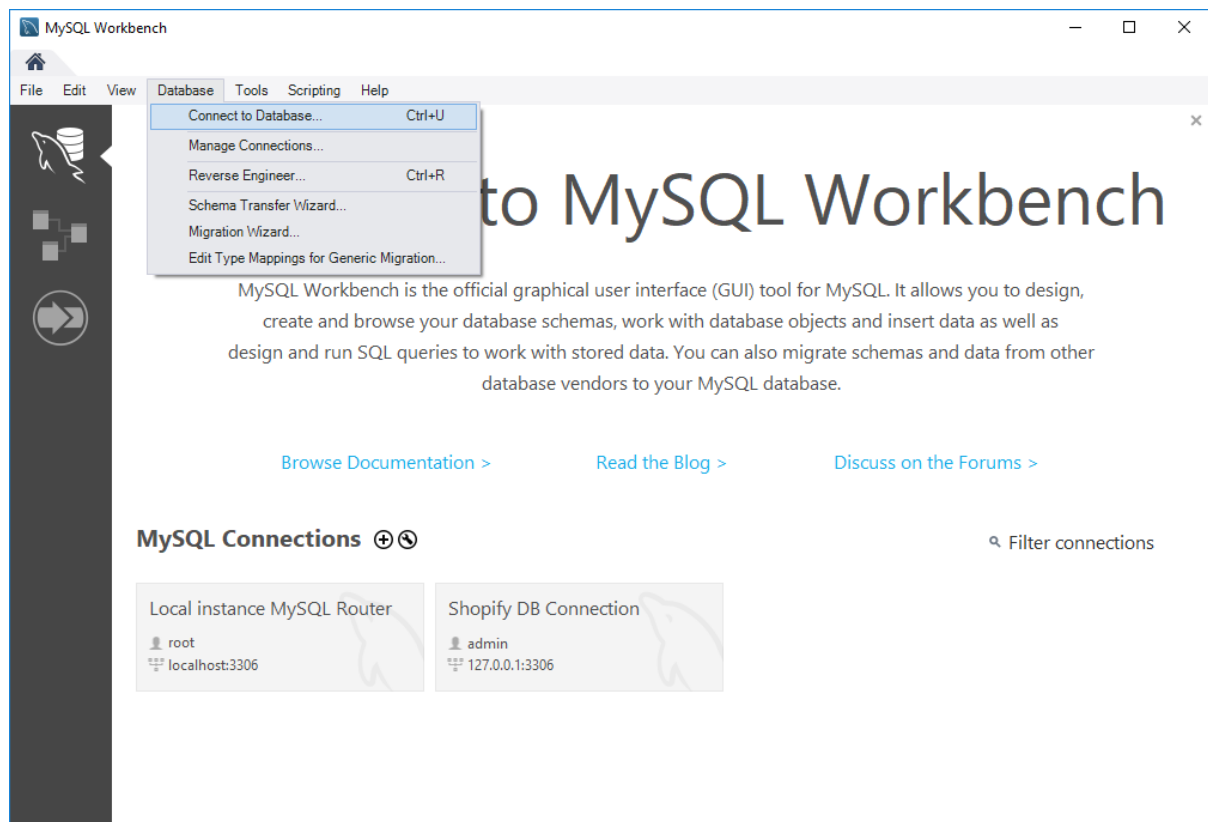
Contents

Log into MySQL Database Remotely.....	1
Create Database Schema	3
Adding a admin user	4
Run SQL script	5

Log into MySQL Database Remotely

You will need MySQL Workbench if you would like to access the MySQL database from you desktop / home pc. Here is the link (It's open source): <https://dev.mysql.com/downloads/workbench/>

Open MySQL Workbench, and go to Database, Connect to Database



Date: 26/03/2019

Created by: Charles Aylward

Click on Connection Method and select: Standard TCP/IP over SSH

Change the following settings

- SSH Hostname: 172.28.25.133:22
- SSH Username: devadmin
- SSH Password: refer to dev server password posted on Slack (You can store the password in Vault)
- MySQL Hostname: 172.28.25.133
- MySQL Server port: 3306
- Username: dbadmin
- Password: refer to dev server password posted on Slack (You can store the password in Vault)

Then click on OK

The screenshot shows a 'Connect to Database' dialog box with the following fields and values:

- Stored Connection:** (empty dropdown)
- Connection Method:** Standard TCP/IP over SSH
- Parameters tab:**
 - SSH Hostname:** 172.28.25.133:22
 - SSH Username:** devadmin
 - SSH Password:** Store in Vault ...
 - SSH Key File:** (empty text box)
 - MySQL Hostname:** 172.28.25.133
 - MySQL Server Port:** 3306
 - Username:** dbadmin
 - Password:** Store in Vault ...
 - Default Schema:** (empty text box)

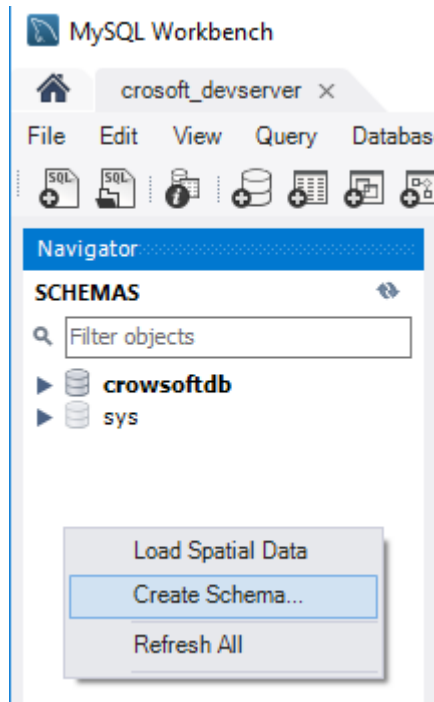
Buttons at the bottom: OK, Cancel

Date: 26/03/2019

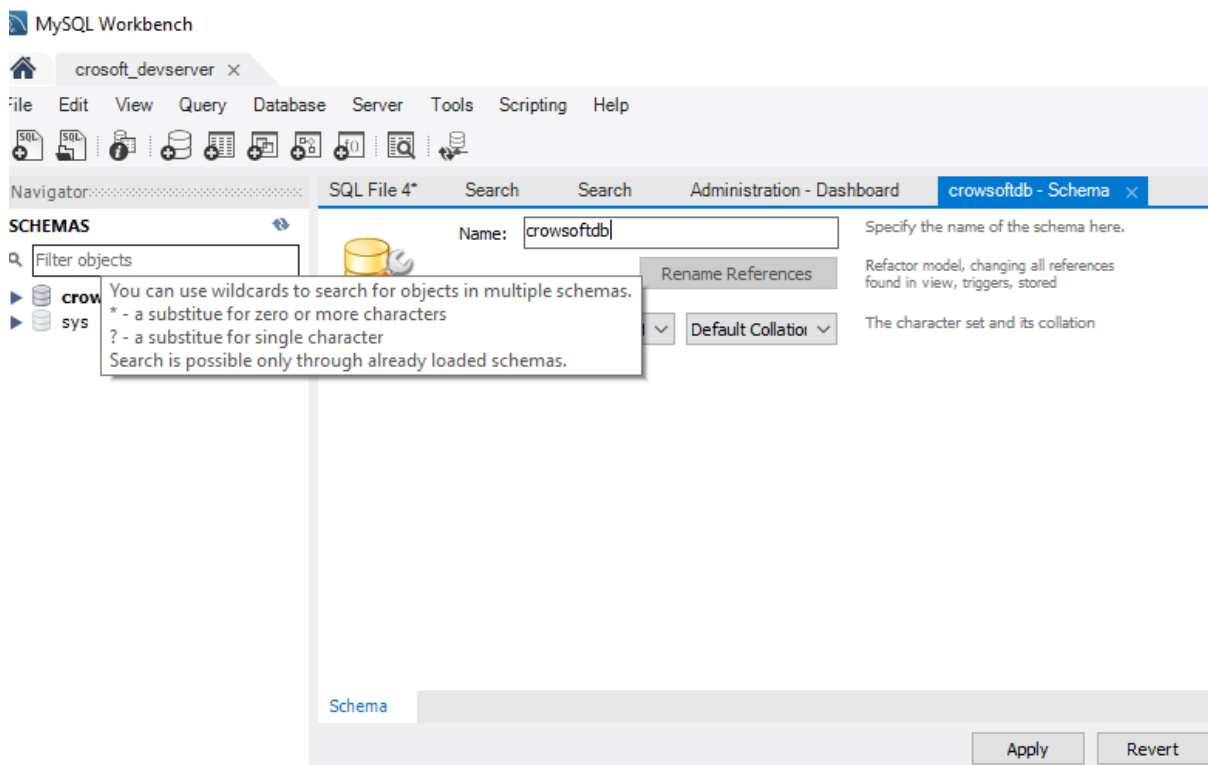
Created by: Charles Aylward

Create Database Schema

Go to the Schemas tab, and right click, then click on Create Schema.



Type in database name in Name column and apply.

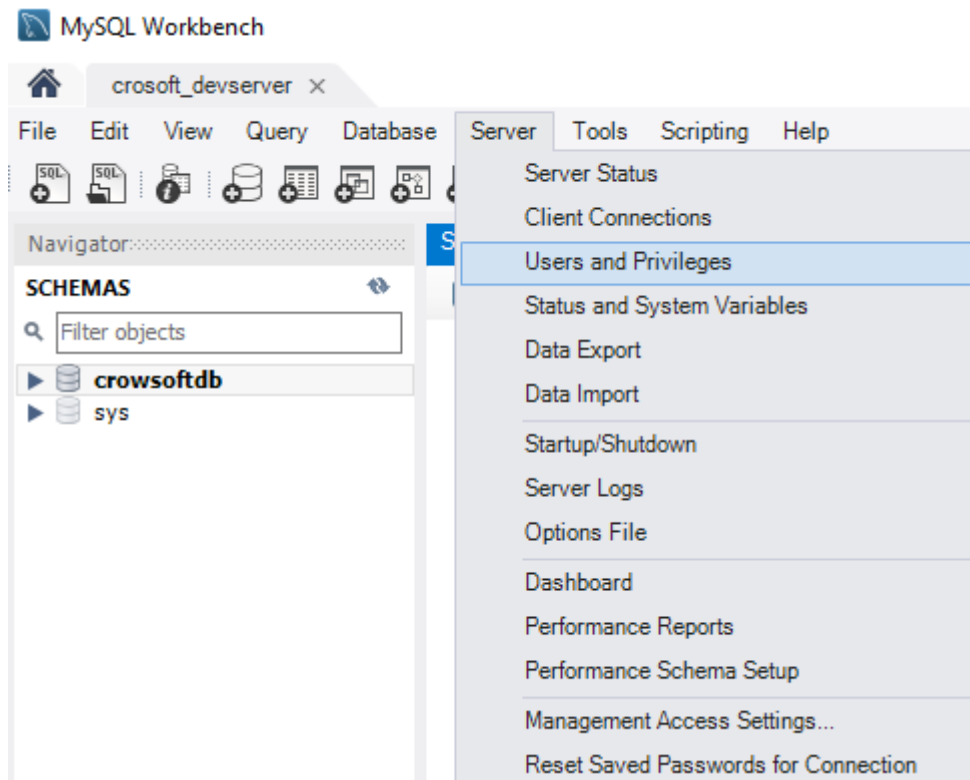


Date: 26/03/2019

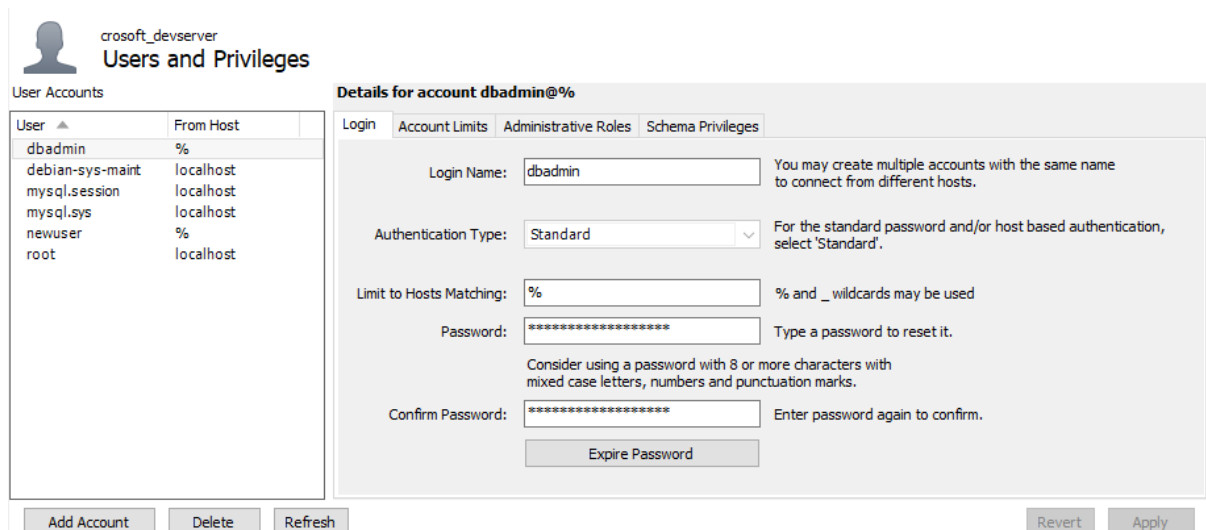
Created by: Charles Aylward

Adding a admin user

Go to Server and click on Users and Privileges



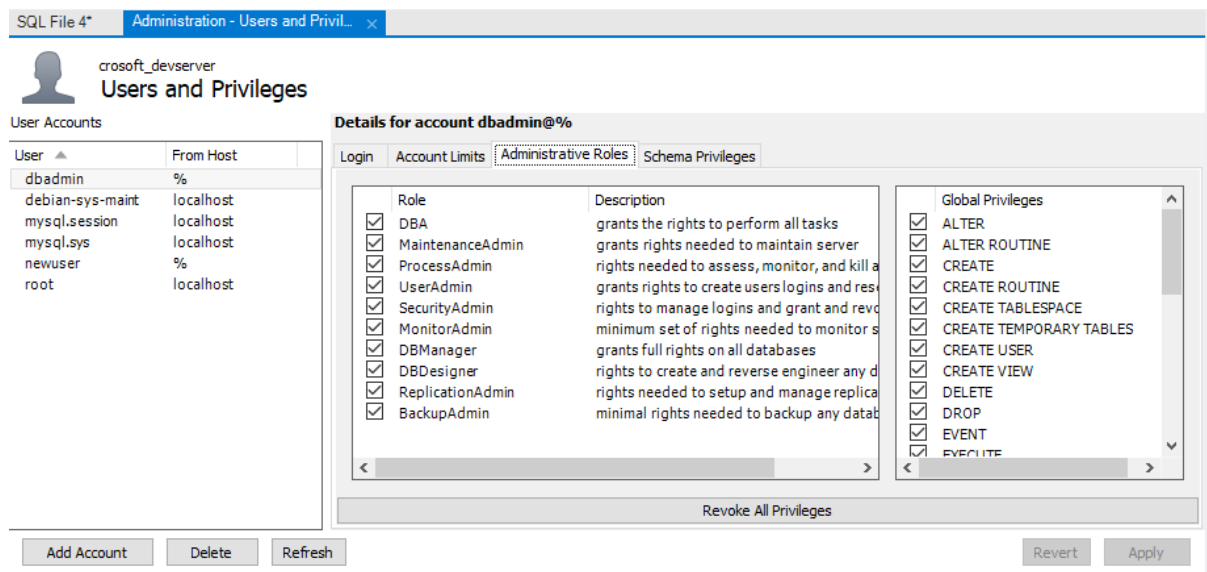
Click on Add Account, enter username and password.



Date: 26/03/2019

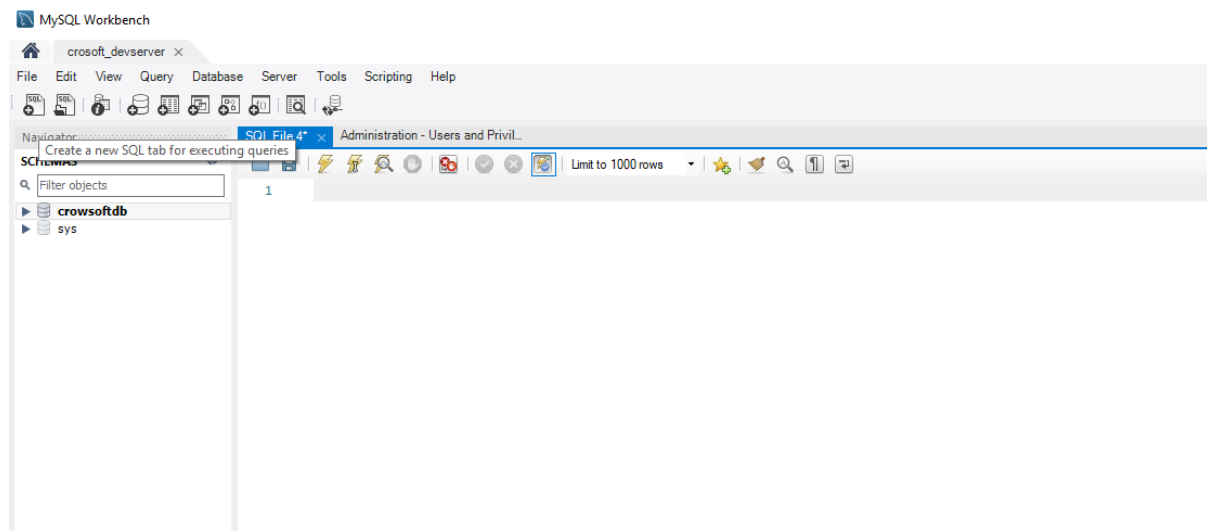
Created by: Charles Aylward

Click on Administrative Roles and all roles and all Global Privileges



Run SQL script

Click on Create a new SQL tab for executing queries Button.



Date: 26/03/2019
Created by: Charles Aylward

Copy and Paste SQL script (Find script in appendix) and click Execute Icon

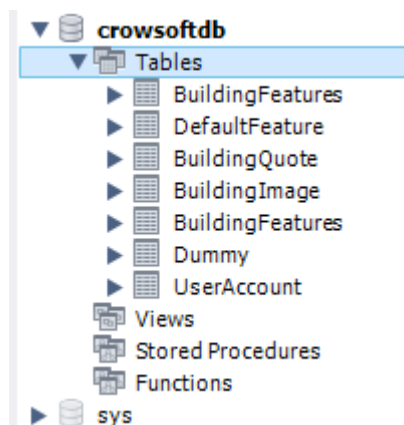


The screenshot shows the MySQL Workbench interface. The top pane is the SQL Editor, titled 'SQL File 4*' and 'Administration - Users and Privil...'. It contains a SQL script generated by MySQL Workbench, including comments and several SET statements to configure the environment. The bottom pane is the Output window, showing the 'Action Output' for the executed script. The output table lists the execution of each statement, including the time, the action performed, and the message returned by the database.

#	Time	Action	Message
4	09:17:14	SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;	0 row(s) affected
5	09:17:14	SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CH...	0 row(s) affected
6	09:17:14	SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE=TRADITIONAL,ALLOW_INVALI...	0 row(s) affected
7	09:17:14	CREATE SCHEMA IF NOT EXISTS 'crowsoftdb' DEFAULT CHARACTER SET utf8	1 row(s) affected, 1 warning(s): 1007 Can't create database 'crowsoftdb'; da
8	09:17:14	USE 'crowsoftdb'	0 row(s) affected
9	09:17:14	CREATE TABLE IF NOT EXISTS 'crowsoftdb'. 'UserAccount' ('idUserAccount' INT NOT N...	0 row(s) affected
10	09:17:15	CREATE TABLE IF NOT EXISTS 'crowsoftdb'. 'DefaultFeature' ('idDefaultFeature' INT NO...	0 row(s) affected
11	09:17:15	CREATE TABLE IF NOT EXISTS 'crowsoftdb'. 'BuildingQuote' ('idBuildingQuote' INT NOT ...	0 row(s) affected
12	09:17:15	CREATE TABLE IF NOT EXISTS 'crowsoftdb'. 'BuildingFeatures' ('idBuildingFeatures' INT ...	0 row(s) affected
13	09:17:16	CREATE TABLE IF NOT EXISTS 'crowsoftdb'. 'BuildingImage' ('idBuildingImage' INT NOT ...	0 row(s) affected

Right click on crowsoftdb and click Refresh All.

Open Tables and view tables.



Date: 26/03/2019

Created by: Charles Aylward

Appendix

```
-- MySQL Script generated by MySQL Workbench
-- Thu Mar 21 09:02:11 2019
-- Model: New Model   Version: 1.0
-- MySQL Workbench Forward Engineering
```

```
SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='TRADITIONAL,ALLOW_INVALID_DATES';
```

```
-----
-- Schema crowsoftdb
-----
```

```
-----
-- Schema crowsoftdb
-----
```

```
CREATE SCHEMA IF NOT EXISTS `crowsoftdb` DEFAULT CHARACTER SET utf8 ;
USE `crowsoftdb` ;
```

```
-----
-- Table `crowsoftdb`.`UserAccount`
-----
```

```
CREATE TABLE IF NOT EXISTS `crowsoftdb`.`UserAccount` (
  `idUserAccount` INT NOT NULL AUTO_INCREMENT,
  `EmailAddress` VARCHAR(150) NOT NULL COMMENT 'User email address used as username as well. ',
  `Password` VARCHAR(50) NOT NULL,
  `FirstName` VARCHAR(60) NOT NULL,
  `LastName` VARCHAR(60) NOT NULL,
  `TelephoneNo` VARCHAR(45) NOT NULL,
  `AddressLine` VARCHAR(150) NULL,
  `County` VARCHAR(45) NULL,
  `Country` VARCHAR(60) NULL,
  `EirCode` VARCHAR(20) NULL,
  `CompanyName` VARCHAR(45) NULL,
  `TypeUser` VARCHAR(15) NOT NULL,
  `DateCreated` DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`idUserAccount`),
  UNIQUE INDEX `idUserAccount_UNIQUE` (`idUserAccount` ASC))
ENGINE = InnoDB;
```

```
-----
-- Table `crowsoftdb`.`DefaultFeature`
-----
```

```
CREATE TABLE IF NOT EXISTS `crowsoftdb`.`DefaultFeature` (
  `idDefaultFeature` INT NOT NULL AUTO_INCREMENT,
  `Description` VARCHAR(100) NOT NULL,
  `Measurement` VARCHAR(20) NOT NULL,
  `UnitPrice` DECIMAL(8,2) NOT NULL,
  `DefaultFeature` TINYINT NULL,
  PRIMARY KEY (`idDefaultFeature`),
  UNIQUE INDEX `idBuildingCost_UNIQUE` (`idDefaultFeature` ASC))
ENGINE = InnoDB;
```

```
-----
-- Table `crowsoftdb`.`BuildingQuote`
-----
```

```
CREATE TABLE IF NOT EXISTS `crowsoftdb`.`BuildingQuote` (
  `idBuildingQuote` INT NOT NULL AUTO_INCREMENT,
  `UserAccount_idUserAccount` INT NOT NULL,
  `Description` VARCHAR(150) NOT NULL,
  `MeasurementType` VARCHAR(25) NOT NULL,
  `Height` INT NOT NULL,
  `Width` INT NOT NULL,
  `Depth` INT NOT NULL,
  `PurposeOfBuilding` VARCHAR(150) NULL,
  `BuildingSize` INT NULL,
```

Date: 26/03/2019

Created by: Charles Aylward

```
`TotalCost` DECIMAL(12,2) NULL,
`DateCreated` DATETIME NULL DEFAULT CURRENT_TIMESTAMP,
`Status` VARCHAR(10) NULL,
`TimeFrame` INT NULL,
`DateUpdated` DATETIME NULL,
`UpdatedById` INT NULL,
PRIMARY KEY (`idBuildingQuote`),
UNIQUE INDEX `idBuildingQuote_UNIQUE` (`idBuildingQuote` ASC),
INDEX `fk_BuildingQuote_UserAccount_idx` (`UserAccount_idUserAccount` ASC),
CONSTRAINT `fk_BuildingQuote_UserAccount`
  FOREIGN KEY (`UserAccount_idUserAccount`)
    REFERENCES `crowsoftdb`.`UserAccount` (`idUserAccount`)
  ON DELETE NO ACTION
  ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

-- Table `crowsoftdb`.`BuildingFeatures`

```
CREATE TABLE IF NOT EXISTS `crowsoftdb`.`BuildingFeatures` (
  `idBuildingFeatures` INT NOT NULL AUTO_INCREMENT,
  `FeatureDescription` VARCHAR(100) NOT NULL,
  `Comments` VARCHAR(150) NULL,
  `Quantity` DECIMAL(10,4) NOT NULL,
  `UnitPrice` DECIMAL(10,2) NOT NULL,
  `TotalCost` DECIMAL(12,2) NULL,
  `BuildingQuote_idBuildingQuote` INT NOT NULL,
  `DefaultFeature_idDefaultFeature` INT NOT NULL,
  PRIMARY KEY (`idBuildingFeatures`, `BuildingQuote_idBuildingQuote`, `DefaultFeature_idDefaultFeature`),
  UNIQUE INDEX `idBuildingFeatures_UNIQUE` (`idBuildingFeatures` ASC),
  INDEX `fk_BuildingFeatures_BuildingQuote1_idx` (`BuildingQuote_idBuildingQuote` ASC),
  INDEX `fk_BuildingFeatures_DefaultFeature1_idx` (`DefaultFeature_idDefaultFeature` ASC),
  CONSTRAINT `fk_BuildingFeatures_BuildingQuote1`
    FOREIGN KEY (`BuildingQuote_idBuildingQuote`)
      REFERENCES `crowsoftdb`.`BuildingQuote` (`idBuildingQuote`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_BuildingFeatures_DefaultFeature1`
    FOREIGN KEY (`DefaultFeature_idDefaultFeature`)
      REFERENCES `crowsoftdb`.`DefaultFeature` (`idDefaultFeature`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

-- Table `crowsoftdb`.`BuildingImage`

```
CREATE TABLE IF NOT EXISTS `crowsoftdb`.`BuildingImage` (
  `idBuildingImage` INT NOT NULL AUTO_INCREMENT,
  `Description` VARCHAR(150) NULL,
  `ImagePath` VARCHAR(200) NULL,
  `BuildingQuote_idBuildingQuote` INT NOT NULL,
  PRIMARY KEY (`idBuildingImage`, `BuildingQuote_idBuildingQuote`),
  UNIQUE INDEX `idBuildingImage_UNIQUE` (`idBuildingImage` ASC),
  INDEX `fk_BuildingImage_BuildingQuote1_idx` (`BuildingQuote_idBuildingQuote` ASC),
  CONSTRAINT `fk_BuildingImage_BuildingQuote1`
    FOREIGN KEY (`BuildingQuote_idBuildingQuote`)
      REFERENCES `crowsoftdb`.`BuildingQuote` (`idBuildingQuote`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;
```

```
SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```

Date Created: 03/04/2019 Updated: 07/04/2019

Open Visual Studio Code, and go to your `crowsoftmvcworkspace`.

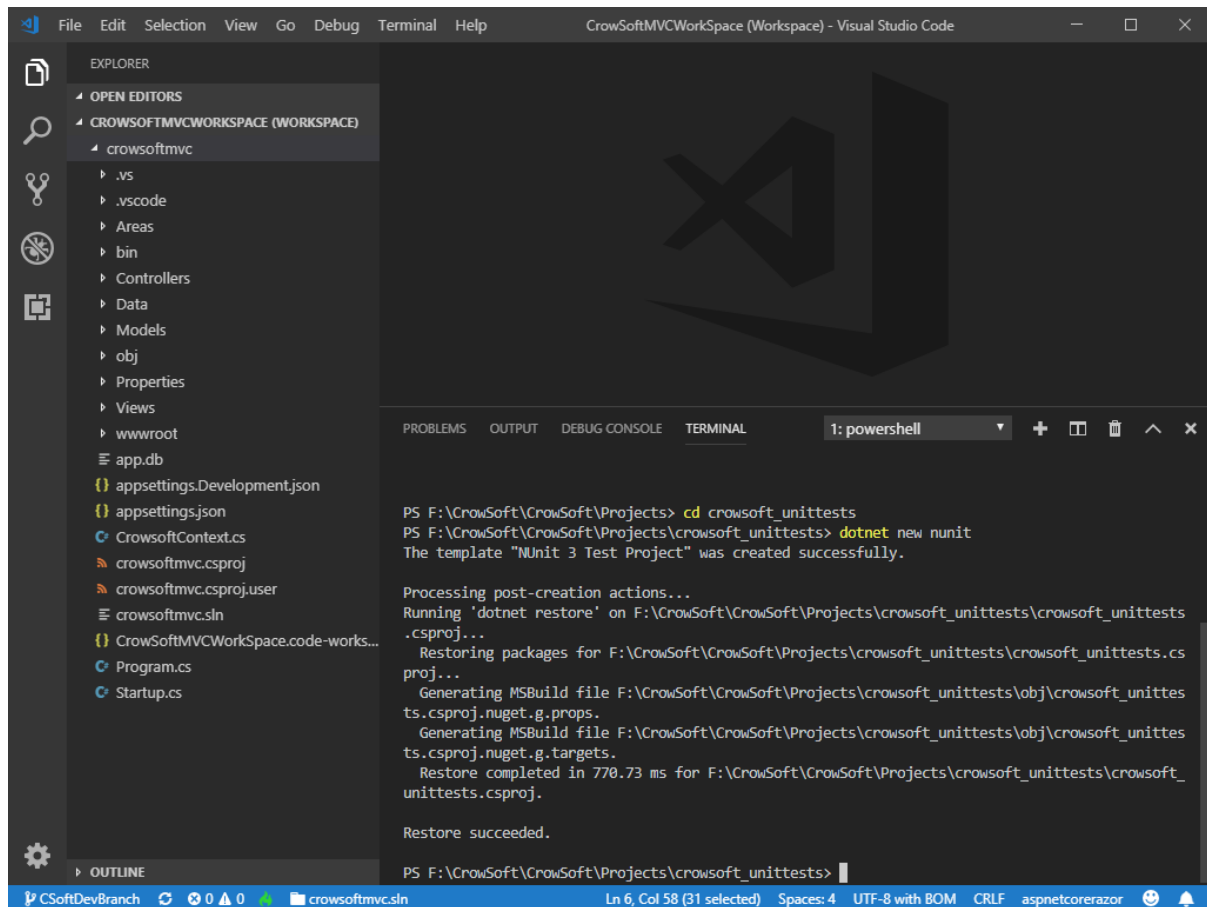
The screenshot shows the Visual Studio Code interface with a dark theme. The Explorer view on the left displays the file structure of a project named 'CROWSOFTMVCWORKSPACE (WORKSPACE)', including folders like '.vs', '.vscode', 'Areas', 'bin', 'Controllers', 'Data', 'Models', 'obj', 'Properties', 'Views', 'wwwroot', and 'app.db', as well as files like 'appsettings.Development.json', 'appsettings.json', 'CrowsoftContext.cs', 'crowsoftmvc.csproj', 'crowsoftmvc.csproj.user', 'crowsoftmvc.sln', and 'CrowSoftMVCWorkSpace.code-works...'. The Terminal view at the bottom shows a Windows PowerShell session with the following commands and output: 'cd ..', 'mkdir crowsoft_unittests', and 'Directory: F:\CrowSoft\CrowSoft\Projects'. The status bar at the bottom indicates the current file is 'crowsoftmvc.sln' at line 6, column 58, with 31 selected characters. The bottom right corner shows the UTF-8 encoding and BOM (Byte Order Mark) settings.

Created by: Charles Aylward

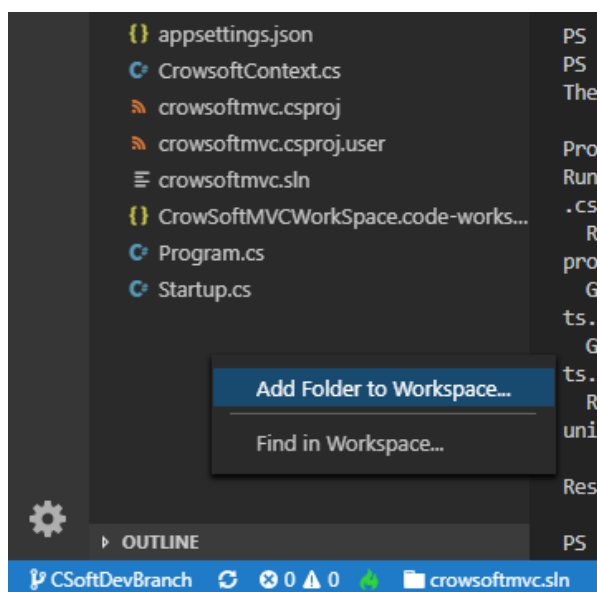
Date Created: 03/04/2019 Updated: 07/04/2019

Go into the new directory by using `cd`

Type **`dotnet new nunit`**, and press enter. Note: This will create the project structure for your nunit tests.



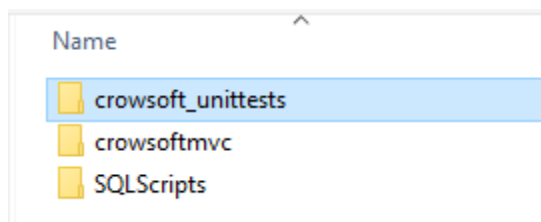
Right click on the open space below the code, then click Add Folder to Workspace.



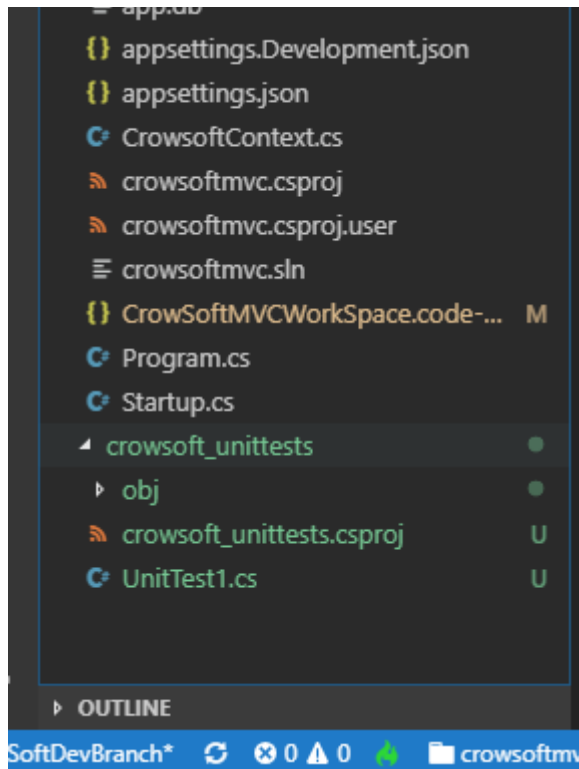
Created by: Charles Aylward

Date Created: 03/04/2019 Updated: 07/04/2019

Select your folder, crowsoft_unittests, and click Add



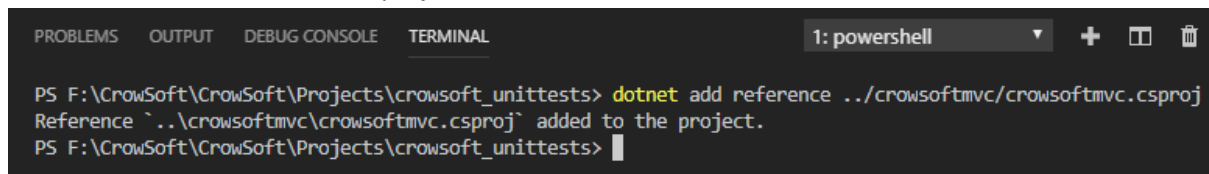
You can see below, crowsoft_unittests folder added to your workspace.



Now we need to reference of our main project to the unit test project.

Open Terminal, and type in the following code and press enter: dotnet add reference

../crowsoftmvc/crowsoftmvc.csproj



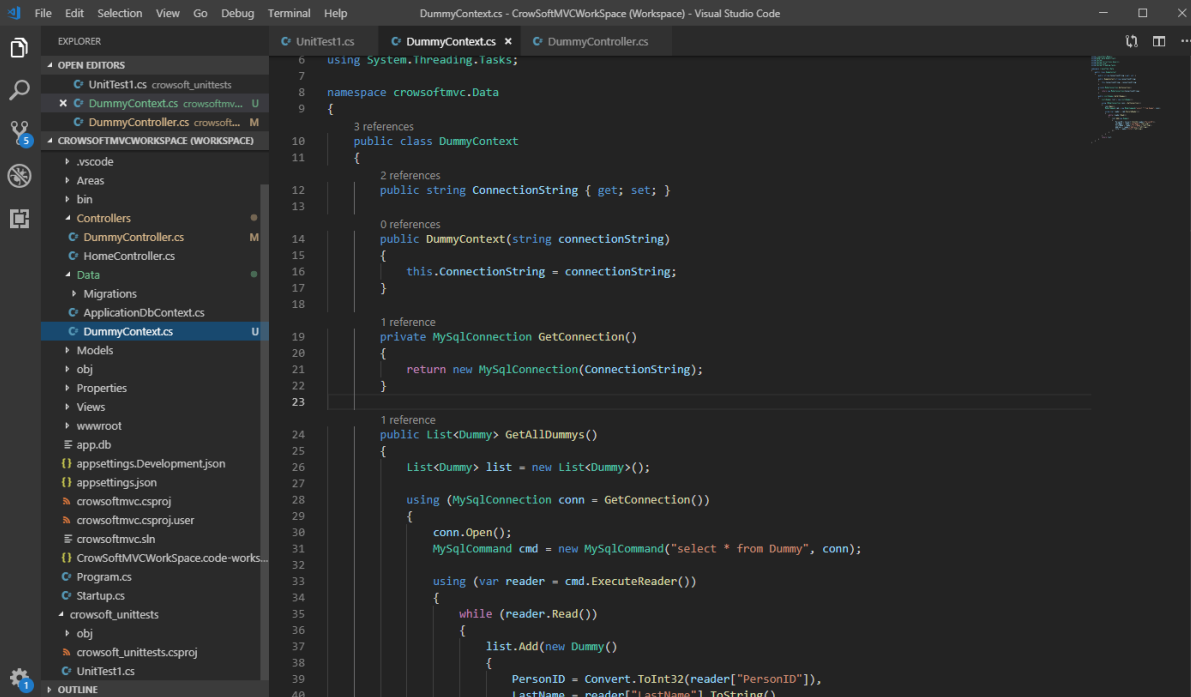
Now the crowsoftmvc.proj is added to the unittests project.

Created by: Charles Aylward

Date Created: 03/04/2019 Updated: 07/04/2019

For the unit test to work, I moved and renamed CrowsoftContext to the Data folder and called it DummyContext. The reason for that is because the context is specific to the Dummy table.

Here is the change below:



The screenshot shows the Visual Studio Code interface with the 'DummyContext.cs' file open in the editor. The file is located in the 'Data' folder of the 'CrowsoftMVCWorkspace' project. The code defines a 'DummyContext' class within the 'crowsoftmvc.Data' namespace. It includes a 'ConnectionString' property, a constructor, a 'GetConnection()' method, and a 'GetAllDummys()' method. The 'GetAllDummys()' method uses a 'MySQLConnection' to execute a query and return a list of 'Dummy' objects. The Explorer pane on the left shows the project structure, and the Outline pane on the right shows the class structure.

```
using System.Threading.Tasks;

namespace crowsoftmvc.Data
{
    3 references
    public class DummyContext
    {
        2 references
        public string ConnectionString { get; set; }

        0 references
        public DummyContext(string connectionString)
        {
            this.ConnectionString = connectionString;
        }

        1 reference
        private MySqlConnection GetConnection()
        {
            return new MySqlConnection(ConnectionString);
        }

        1 reference
        public List<Dummy> GetAllDummys()
        {
            List<Dummy> list = new List<Dummy>();

            using (MySqlConnection conn = GetConnection())
            {
                conn.Open();
                MySqlCommand cmd = new MySqlCommand("select * from Dummy", conn);

                using (var reader = cmd.ExecuteReader())
                {
                    while (reader.Read())
                    {
                        list.Add(new Dummy()
                        {
                            PersonID = Convert.ToInt32(reader["PersonID"]),
                            LastName = reader["LastName"].ToString();
                        });
                    }
                }
            }
        }
    }
}
```

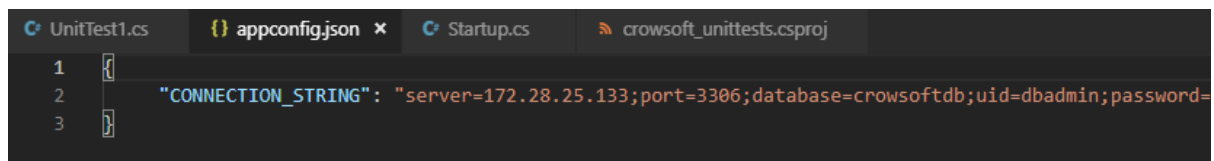
Include the following Package References to your crowsoft_unittests.csproj file.

```
<PackageReference Include="Microsoft.AspNetCore.Mvc.Core" Version="2.2.2" />
<PackageReference Include="Microsoft.AspNetCore.Razor.Design" Version="2.2.0" />
<PackageReference Include="Microsoft.AspNetCore.TestHost" Version="2.2.0" />
<PackageReference Include="Microsoft.EntityFrameworkCore.Sqlite" Version="2.2.3" />
<PackageReference Include="Microsoft.Extensions.Configuration" Version="2.2.0" />
<PackageReference Include="Microsoft.Extensions.Configuration.Json" Version="2.2.0" />
<PackageReference Include="Microsoft.VisualStudio.Web.CodeGeneration.Design" Version="2.2.3" />
<PackageReference Include="Moq" Version="4.10.1" />
<PackageReference Include="MySQL.Data" Version="8.0.15" />
```

Created by: Charles Aylward

Date Created: 03/04/2019 Updated: 07/04/2019

Add a appconfig.json file. This is to reference the connection string for MySQL.



```
{
  "CONNECTION_STRING": "server=172.28.25.133;port=3306;database=crowsoftdb;uid=dbadmin;password="
}
```

Open UnitTest1.cs, and added connection string code to setup a connection string for all tests.

First, read the appconfig.json file into a variable called config.

```
var config = new ConfigurationBuilder()
    .AddJsonFile("appconfig.json")
    .Build();
```

Then, read the connectionstring into a local variable, called connection_string.

```
3  using System.Collections.Generic;
4  using crowsoftmvc.Data;
5  using Microsoft.Extensions.Configuration.Json;
6  using Microsoft.Extensions.Configuration;
7
8  namespace Tests
9  {
10     0 references
11     public class Tests
12     {
13         2 references
14         private string connection_string;
15
16         // This sets up the test and reads the connectionstring from the appconfig.json file
17         [SetUp]
18         0 references
19         public void Setup()
20         {
21             var config = new ConfigurationBuilder()
22                 .AddJsonFile("appconfig.json")
23                 .Build();
24             connection_string = config["CONNECTION_STRING"];
```

Make sure the following using statements are added:

```
using System.Collections.Generic;
using crowsoftmvc.Data;
using Microsoft.Extensions.Configuration.Json;
using Microsoft.Extensions.Configuration;
```

Created by: Charles Aylward

Date Created: 03/04/2019 Updated: 07/04/2019

Add a new method to test if Dummy records are returned. It should return 2 record, which are Greater than 0.

```
// This is a example test, that test if Dummy records are available in the MySQL Database
[Test]
public void Test_GetDummyList()
{
    DummyContext context = new DummyContext(connection_string);
    List<crowsoftmvc.Models.Dummy> myDummyList = context.GetAllDummies();

    Assert.Greater(myDummyList.Count, 0, "Error No Dummy Records Returned");
}
```

Go to crowsoft_unittests.csproj and add the following into the ItemGroup:

```
<None Update="appconfig.json">
  <CopyToOutputDirectory>Always</CopyToOutputDirectory>
</None>
```

Example:

```
<ItemGroup>
  <PackageReference Include="Microsoft.AspNetCore.Mvc.Core" Version="2.2.2" />
  <PackageReference Include="Microsoft.AspNetCore.Razor.Design" Version="2.2.0" />
  <PackageReference Include="Microsoft.AspNetCore.TestHost" Version="2.2.0" />
  <PackageReference Include="Microsoft.EntityFrameworkCore.Sqlite" Version="2.2.3" />
  <PackageReference Include="Microsoft.Extensions.Configuration" Version="2.2.0" />
  <PackageReference Include="Microsoft.Extensions.Configuration.Json" Version="2.2.0" />
  <PackageReference Include="Microsoft.VisualStudio.Web.CodeGeneration.Design" Version="2.2.3" />
  <PackageReference Include="Moq" Version="4.10.1" />
  <PackageReference Include="MySQL.Data" Version="8.0.15" />
  <PackageReference Include="nunit" Version="3.11.0" />
  <PackageReference Include="NUnit3TestAdapter" Version="3.11.0" />
  <PackageReference Include="Microsoft.NET.Test.Sdk" Version="15.9.0" />
  <None Update="appconfig.json">
    <CopyToOutputDirectory>Always</CopyToOutputDirectory>
  </None>
</ItemGroup>
```

Open Terminal for crowsoft_unittests, and type in dotnet test

You should get the following results:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL
1: powershell

Test execution time: 5.2851 Seconds
PS F:\CrowSoft\CrowSoft\Projects\crowsoft_unittests> dotnet test
Build started, please wait...
Build completed.

Test run for F:\CrowSoft\CrowSoft\Projects\crowsoft_unittests\bin\Debug\netcoreapp2.2\crowsoft_unittests.dll(.NETCoreApp,Version=
Microsoft (R) Test Execution Command Line Tool Version 15.9.0
Copyright (c) Microsoft Corporation. All rights reserved.

Starting test execution, please wait...

Total tests: 1. Passed: 1. Failed: 0. Skipped: 0.
Test Run Successful.
Test execution time: 1.6014 Seconds
PS F:\CrowSoft\CrowSoft\Projects\crowsoft_unittests>
```

Finish..

Pokerbot

#example how to configure for poker session

/deal config options 1,2,3,5,8

#example how to play a session Adding user @<persons_Slack_name>

/deal @Mary Walsh McGinty @MatthewMcC @Colin @Charles @Michael @JOJI please enter 1, 3, 5,
or 8

@Bharathi

@Liam

Scrum Master roles & responsibilities.

Author	Description	Version	Date
Michael McFadden	Scrum Master Roles	1	03/25/2019

Definition:

Scrum is about continuous learning and continuous improvement

The scrum master is the team role responsible for ensuring the team lives agile values and principles and follows the processes and practices that the team agreed they would use.

The responsibilities of this role include:

- Clearing obstacles
- Establishing an environment where the team can be effective
- Addressing team dynamics
- Ensuring a good relationship between the team and [product owner](#) as well as others outside the team
- Protecting the team from outside interruptions and distractions.

Roles & responsibilities for Team Crowsoft

- Never Commit the Team to Anything Without Consulting Them First
- Help the team get passed any impediments
- Ensure the PO & all team members are available for Sprint Planning. (no team member will be assigned a story if absent, Scrum Master needs to approve late stories into Sprint)
- Ensure Team update their status daily on csoft_daily_scrum
- Ensure all team members contribute to retrospective, regardless of work done/not done in that Sprint
- Facilitate the meetings and ensure that all relevant parties are in attendance
- Encourage the Team to be self-organising
- Listen to team members and help your team to learn how to solve problems on their own.
- Analyse the Jira dashboard and reporting tools for KPI's for continuous improvement
Process of how the team works can and should be adapted according to previous Sprint
- Keep morale high amongst the team
- Work with Team to assign story points to backlog items **prior** to Sprint planning

Selenium Testing Setup

Version	Date	Affected Section	Author
1.0	04/04/2019	Initial Draft	Matthew Mc Colgan

Contents

Introduction	2
Prerequisites	2
Steps.....	2
References	5

Introduction


Selenium WebDriver is an opensource web automation framework to execute tests against different browsers. You can write your tests in multiple languages including C#, Java, PHP etc. It can operate across multiple OS's and can be integrated with many frameworks including NUnit and other applications like Jenkins.

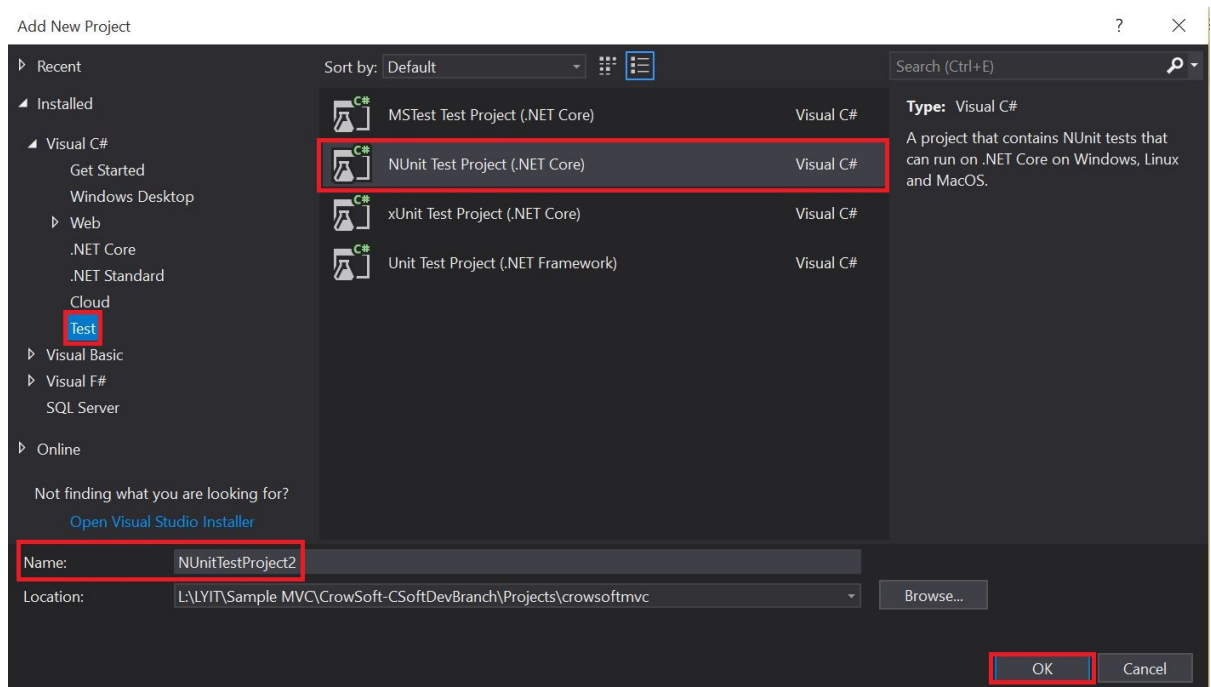
Prerequisites

Visual Studio Community Edition

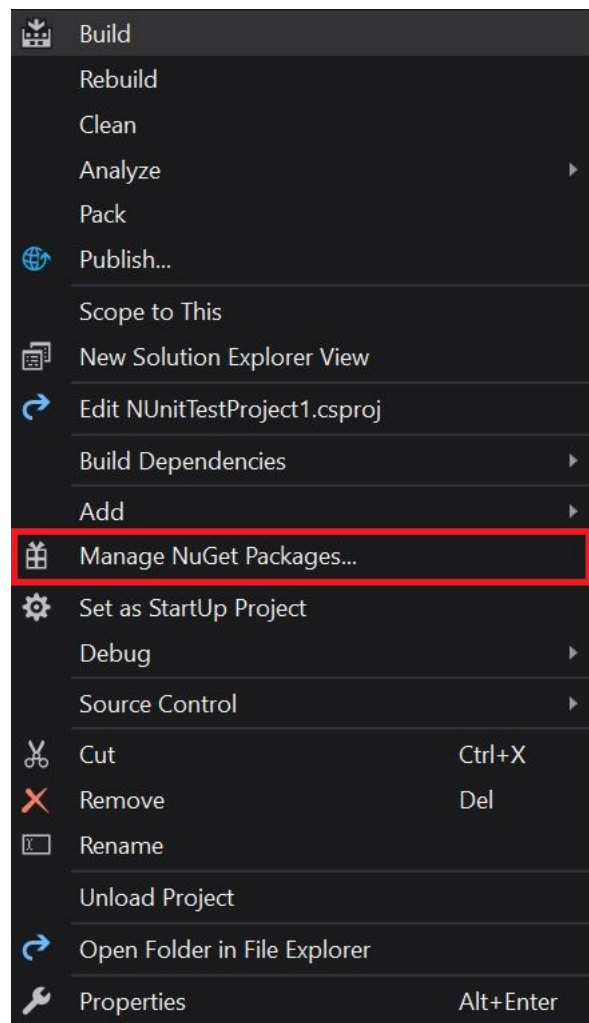
Sample MVC app created by Charles. Can be downloaded/cloned from GitHub @ <https://github.com/rleannon/CrowSoft/tree/CSoftDevBranch/Projects/crowsoftmvc>

Steps

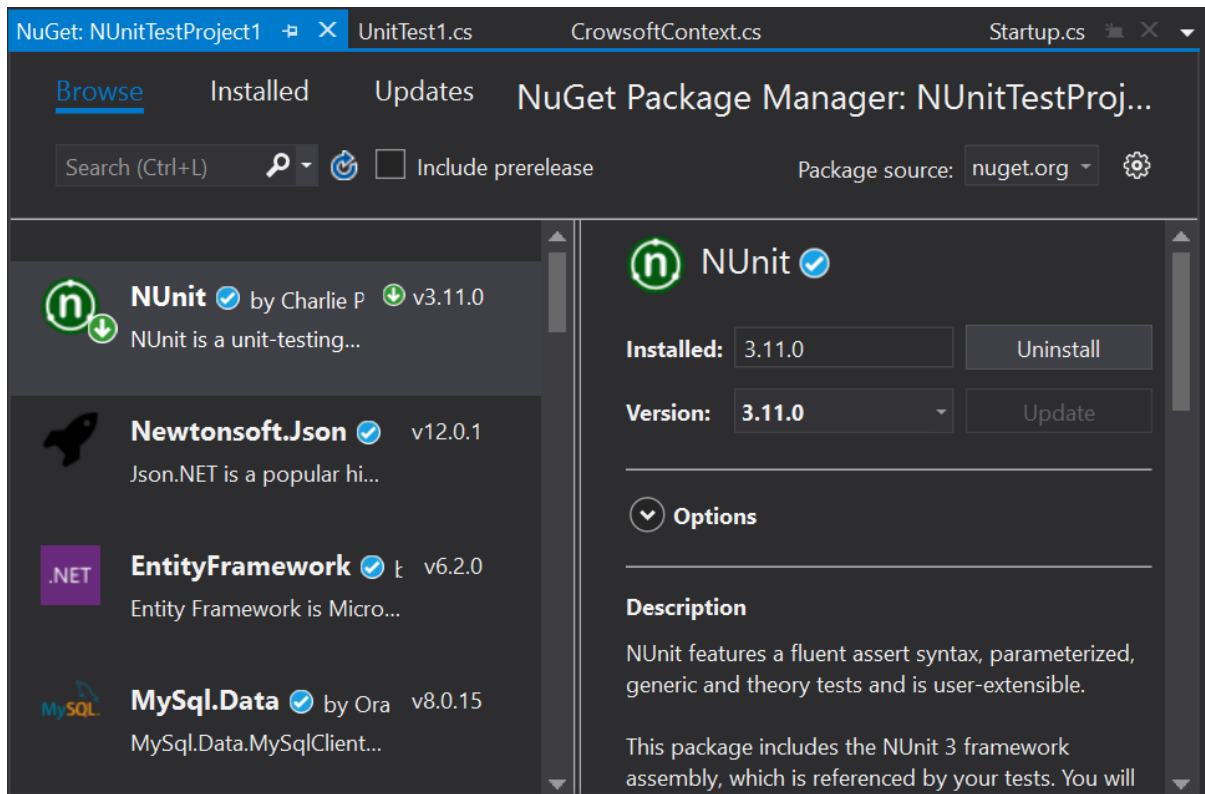
1. Open the sample MVC app in VS Community
2. Right click on the Solution

3. Select "Add" and choose "New Project".
4. Select "Test" -> "NUnit Test Project (.NET Core)" and give it a name. Click "OK" when done.



5. Right click on the newly created project and select "Manage NuGet Packages"



6. You can browse for the required NuGet packages using the Browse tab. Select the required package and click install on the pane in the right.



7. Install all of the following NuGet packages:
 - a. Microsoft.NET.Test.SDK
 - b. Microsoft.NETCore.TestHost
 - c. Selenium.Chrome.WebDriver
 - d. Selenium.WebDriver
8. Download chromedriver.exe from the following link <http://chromedriver.chromium.org/downloads>. Make sure to download the driver that corresponds to the version of chrome you are using.
9. Place the chromedriver.exe in your project folder (see code below)**
10. Open the UnitTest1.cs file that was automatically generated when creating the project and paste the following code:

```
using NUnit.Framework;
using OpenQA.Selenium;
using OpenQA.Selenium.Chrome;

namespace Tests
{
    public class Tests
    {
        public class SeleniumTests
        {
            IWebDriver driver;

            [SetUp]
            public void startBrowser()
            {
                driver = new ChromeDriver("L:/LYIT/Sample MVC/CrowSoft-
CSoftDevBranch/Projects/NUnitTestProject1/bin/Debug/netcoreapp2.2");
            } **Change path to suit where your chromedriver is installed**
        }
    }
}
```

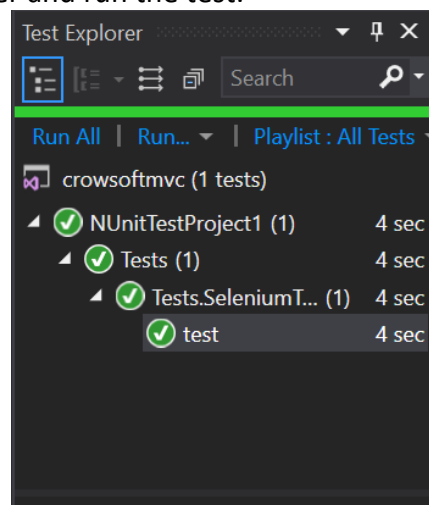
```

[Test]
public void test()
{
    driver.Url = "https://localhost:44380";
}

[TearDown]
public void closeBrowser()
{
    driver.Close();
}
}
}
}

```

11. Open the Test Explorer and run the test.



References

<http://chromedriver.chromium.org/downloads>

<https://www.lambdatest.com/blog/13-reasons-why-selenium-webdriver-should-be-your-first-choice-for-automation-testing/>

<https://www.guru99.com/selenium-csharp-tutorial.html>

Service Level Agreement (SLA)
for *Ruth Lennon*
by
CrowSoft Technologies

Effective Date: 8-04-2019

Document Owner:	CrowSoft Technologies
------------------------	-----------------------

Version

Version	Date	Description	Author
1.0	7-04-2019	Service Level Agreement	Charles Aylward

Approval

(By signing below, all Approvers agree to all terms and conditions outlined in this Agreement.)

Approvers	Role	Signed	Approval Date
CrowSoft Director	Service Provider		
Customer	Customer		

*source of this template: (Author, NA) <http://www.slatemplate.com/>

Table of Contents

1.	Agreement Overview	3
2.	Goals & Objectives	3
3.	Stakeholders	3
4.	Periodic Review	4
5.	Service Agreement.....	4
5.1.	Service Scope.....	4
5.2.	Customer Requirements.....	5
5.3.	Service Provider Requirements.....	5
5.4.	Service Assumptions.....	5
6.	Service Management.....	6
6.1.	Service Availability	6
6.2.	Service Requests	6

1. Agreement Overview

This Agreement represents a Service Level Agreement (“SLA” or “Agreement”) between **CrowSoft Technologies**, called the provider and **Ruth Lennon**, called the customer for the provisioning of IT services required to support and sustain the CrowSoft Building Analysis & Business Intelligence Solution.

This Agreement remains valid until superseded by a revised agreement mutually endorsed by the stakeholders.

This Agreement outlines the parameters of all IT services covered as they are mutually understood by the primary stakeholders. This Agreement does not supersede current processes and procedures unless explicitly stated herein.

2. Goals & Objectives

The **purpose** of this Agreement is to ensure that the proper elements and commitments are in place to provide consistent IT service support and delivery to the Customer(s) by the Service Provider(s).

The **goal** of this Agreement is to obtain mutual agreement for IT service provision between the Service Provider(s) and Customer(s).

The **objectives** of this Agreement are to:

- Provide clear reference to service ownership, accountability, roles and/or responsibilities.
- Present a clear, concise and measurable description of service provision to the customer.
- Match perceptions of expected service provision with actual service support & delivery.

3. Stakeholders

The following Service Provider(s) and Customer(s) will be used as the basis of the Agreement and represent the **primary stakeholders** associated with this SLA:

IT Service Provider(s): CrowSoft Technologies. (“Provider”)

IT Customer(s): Ruth Lennon (“Customer”)

4. Periodic Review

This Agreement is valid from the **Effective Date** outlined herein and is valid until further notice. This Agreement should be reviewed at a minimum once per fiscal year; however, in lieu of a review during any period specified, the current Agreement will remain in effect.

The **Business Relationship Manager** (“Document Owner”) is responsible for facilitating regular reviews of this document. Contents of this document may be amended as required, provided mutual agreement is obtained from the primary stakeholders and communicated to all affected parties. The Document Owner will incorporate all subsequent revisions and obtain mutual agreements / approvals as required.

Business Relationship Manager: CrowSoft Technologies

Review Period: Bi-Yearly (6 months)

Previous Review Date: 08-04-2019 (TBD)

Next Review Date: 08-10-2019

5. Service Agreement

The following detailed service parameters are the responsibility of the Service Provider in the ongoing support of this Agreement.

5.1. Service Scope

The following Services are covered by this Agreement;

- Manned telephone support
- Monitored email support
- Remote assistance using Remote Desktop and a Virtual Private Network where available
- Planned or Emergency Onsite assistance (extra costs apply)
- Monthly system health check

5.2. Customer Requirements

Customer responsibilities and/or requirements in support of this Agreement include:

- Payment for all support costs at the agreed interval.
- Reasonable availability of customer representative(s) when resolving a service related incident or request.

5.3. Service Provider Requirements

Service Provider responsibilities and/or requirements in support of this Agreement include:

- Meeting response times associated with service related incidents.
- Appropriate notification to Customer for all scheduled maintenance.

5.4. Service Assumptions

Assumptions related to in-scope services and/or components include:

- Changes to services will be communicated and documented to all stakeholders.

6. Service Management

Effective support of in-scope services is a result of maintaining consistent service levels. The following sections provide relevant details on service availability, monitoring of in-scope services and related components.

6.1. Service Availability

Coverage parameters specific to the service(s) covered in this Agreement are as follows:

- Telephone support : 9:00 A.M. to 5:00 P.M. Monday – Friday
 - Calls received out of office hours will be forwarded to a mobile phone and best efforts will be made to answer / action the call, however there will be a backup answer phone service
- Email support: Monitored 9:00 A.M. to 5:00 P.M. Monday – Friday
 - Emails received outside of office hours will be collected, however no action can be guaranteed until the next working day
- Onsite assistance guaranteed within 72 hours during the business week

6.2. Service Requests

In support of services outlined in this Agreement, the Service Provider will respond to service related incidents and/or requests submitted by the Customer within the following time frames:

- 0-8 hours (during business hours) for issues classified as **High** priority.
- Within 48 hours for issues classified as **Medium** priority.
- Within 5 working days for issues classified as **Low** priority.

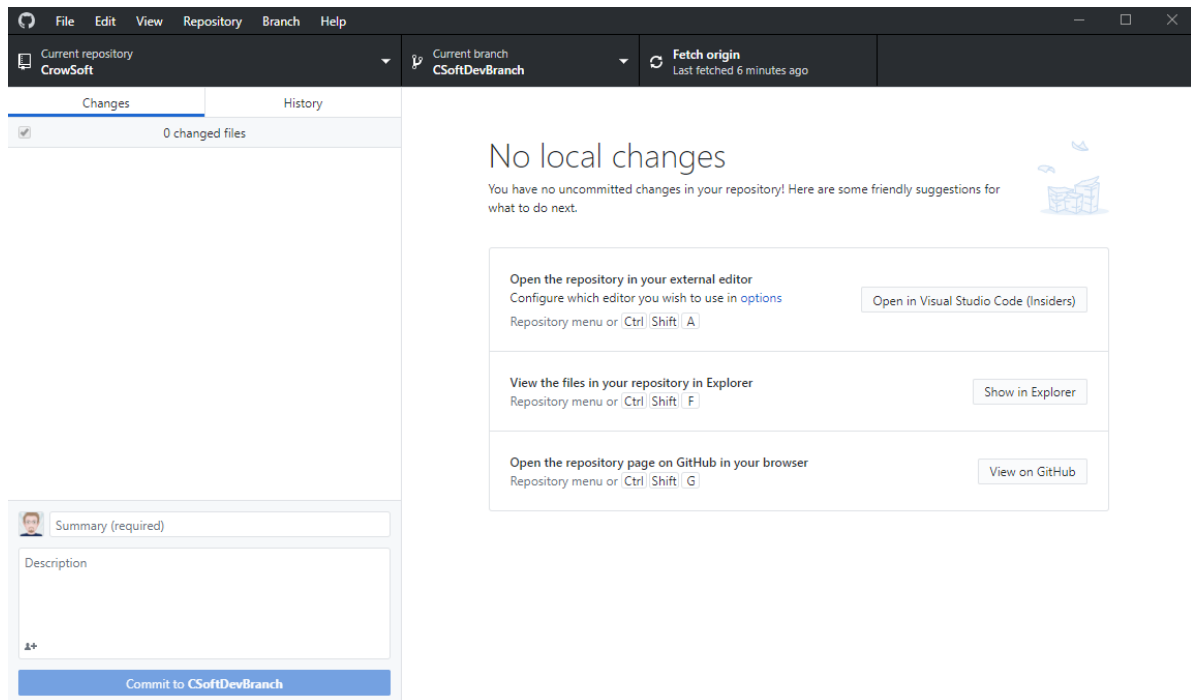
Remote assistance will be provided in-line with the above timescales dependent on the priority of the support request.

Source Control on Visual Studio Code and Github

I use Github Desktop for windows.

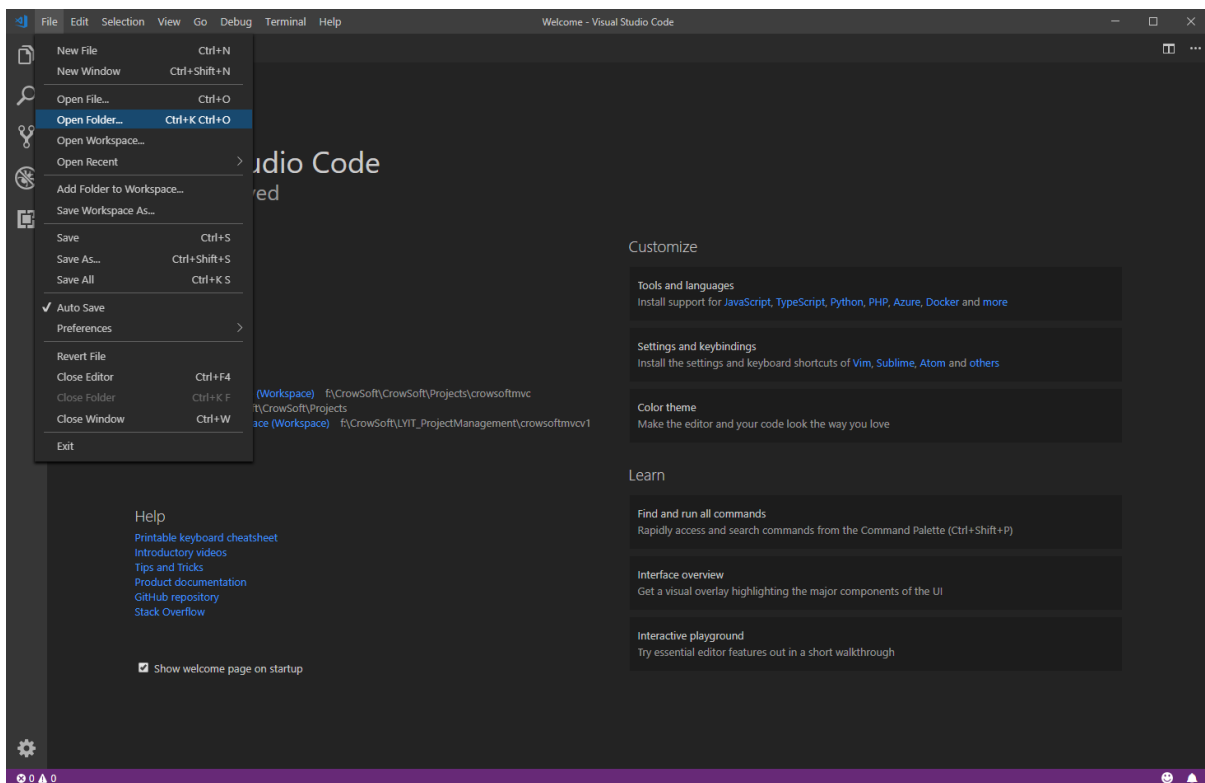
Select your CSoftDevBranch, and then fetch origin.

```
On linux, use: git-pull - Fetch from and integrate with local branch
```

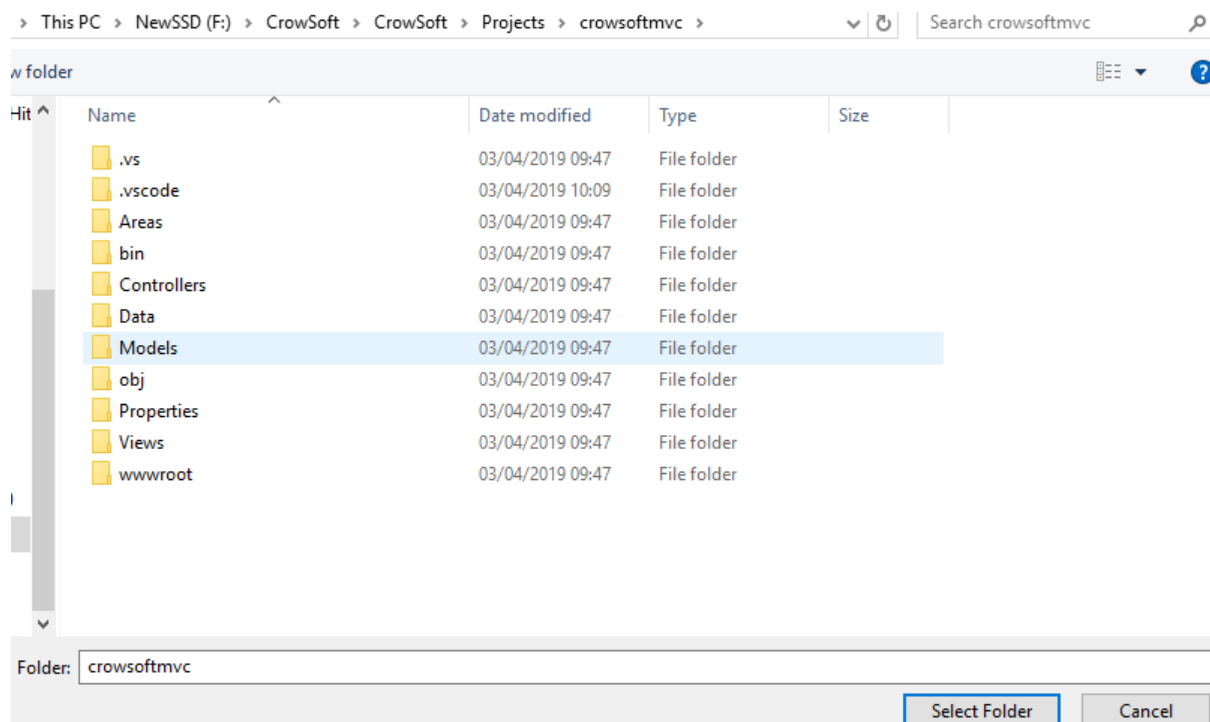


Created by: Charles Aylward
Updated: 03/04/2019


Open Visual Studio Code. Go to File, Open folder.

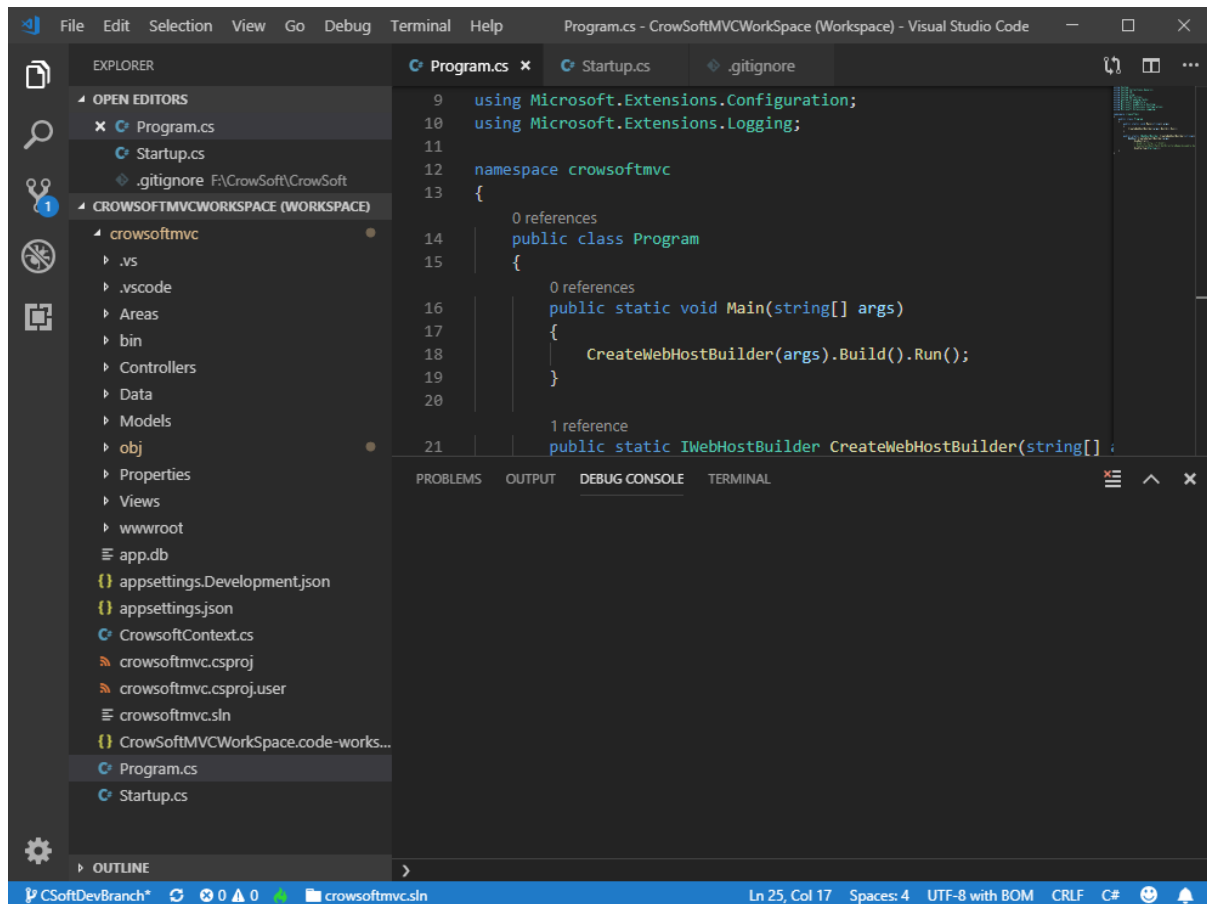


Select the folder on your local drive. **NOTE:** If you see the **CrowSoftMVCWorkspace.code-workspace** file, then go back, and open the workspace file.

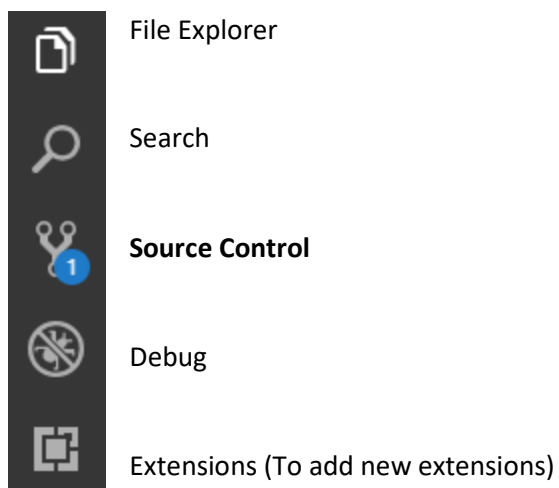


Created by: Charles Aylward
Updated: 03/04/2019


Note  on bottom left of the IDE. This will automatically link to your Github branch if you select the local git folder.

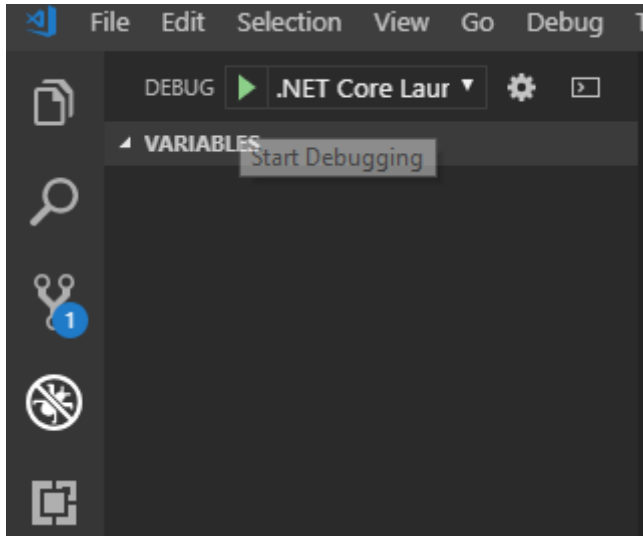


Below is where you can find Source Control.

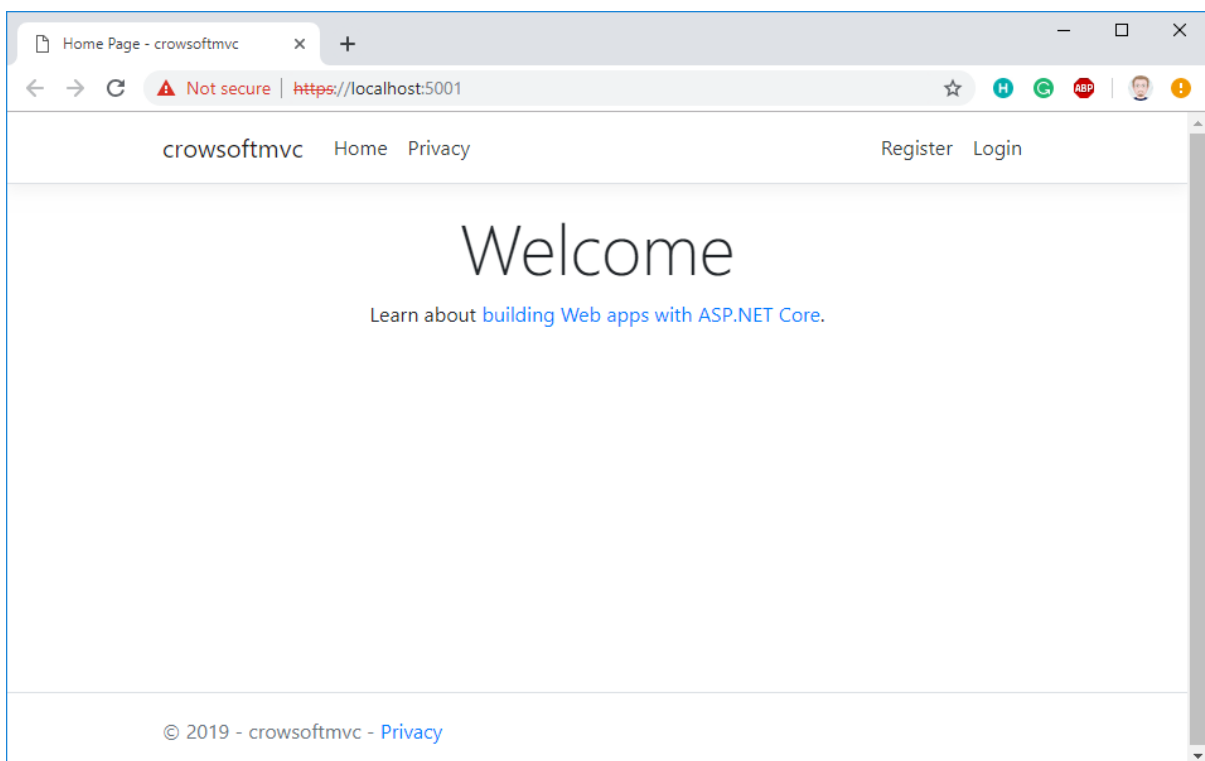


Created by: Charles Aylward
Updated: 03/04/2019

Let's debug and run the app. Click on the Debug Icon, then click 

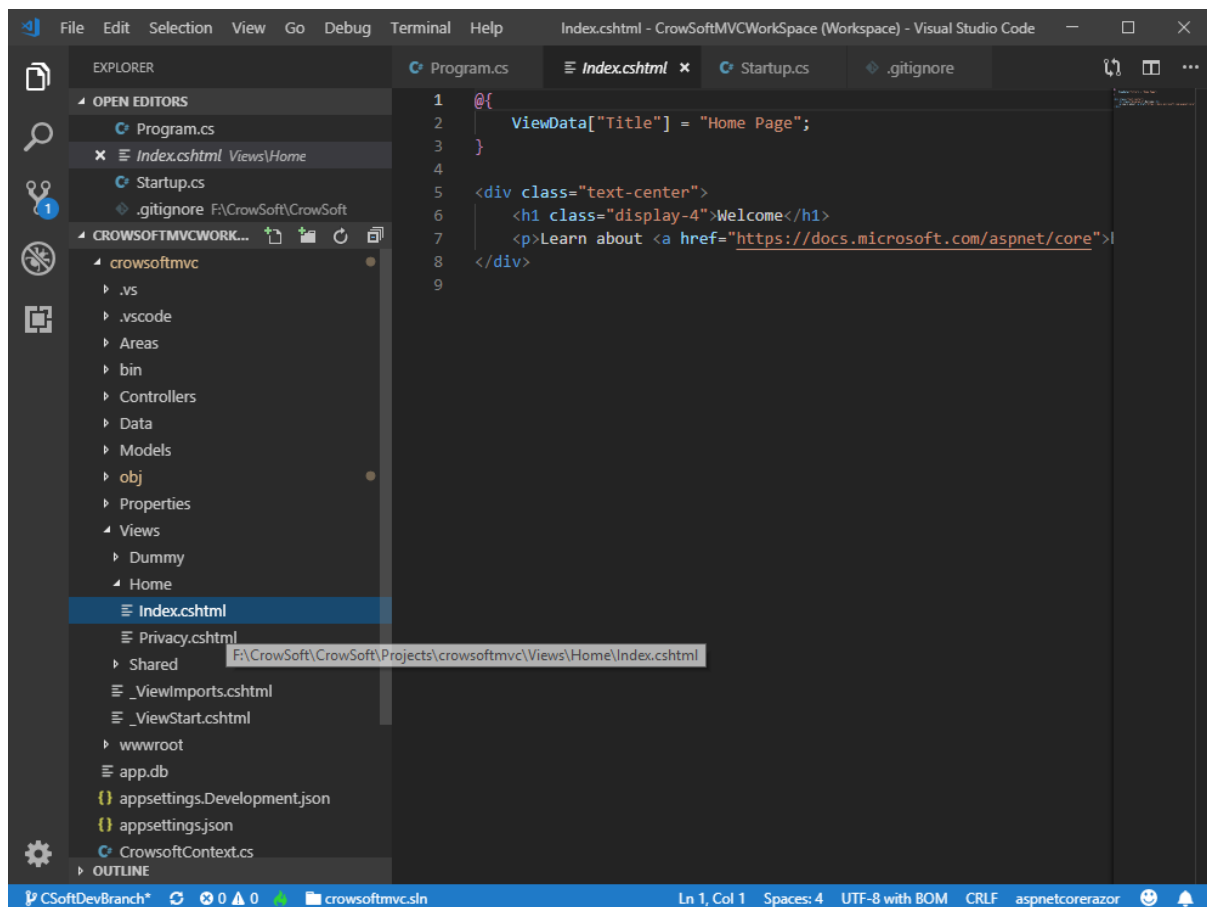


This should open your default browser and run the web application as shown below.

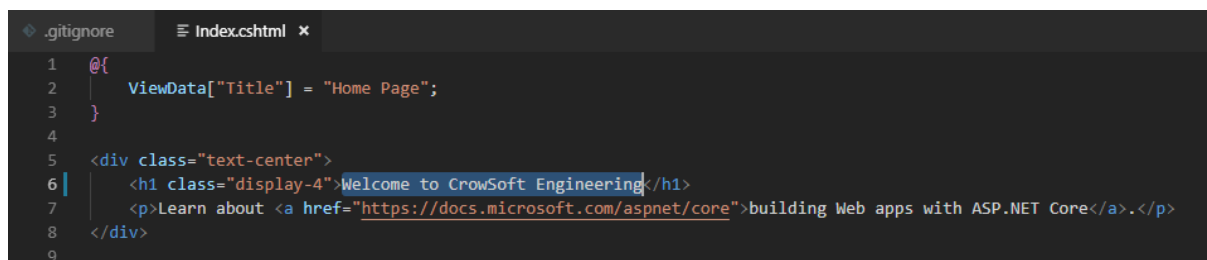


Created by: Charles Aylward
Updated: 03/04/2019

Let's make a change to the code and check it in. Go to Views/Home/Index.cshtml




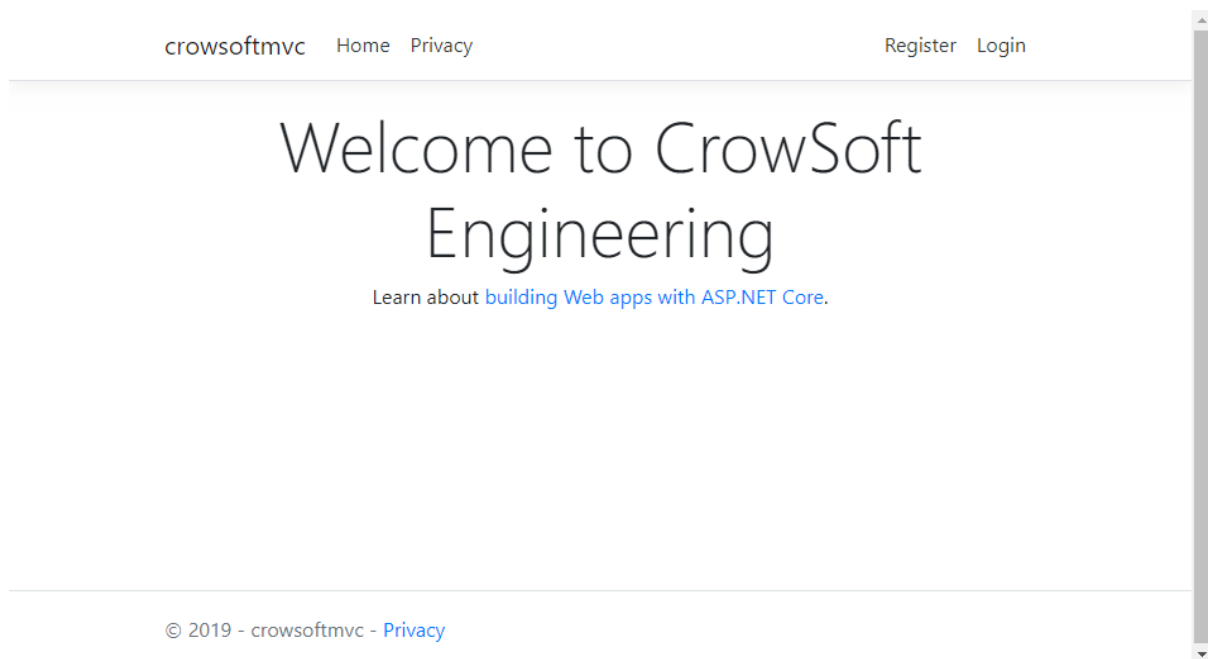
Change the display-4 message to: **Welcome to CrowSoft Engineering**



Click File / Save

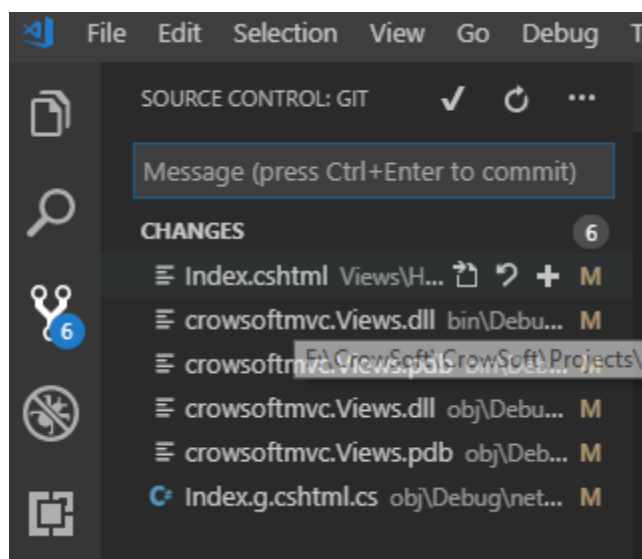
Created by: Charles Aylward
Updated: 03/04/2019

Then Debug and click  again. You can see then change on the web app below.



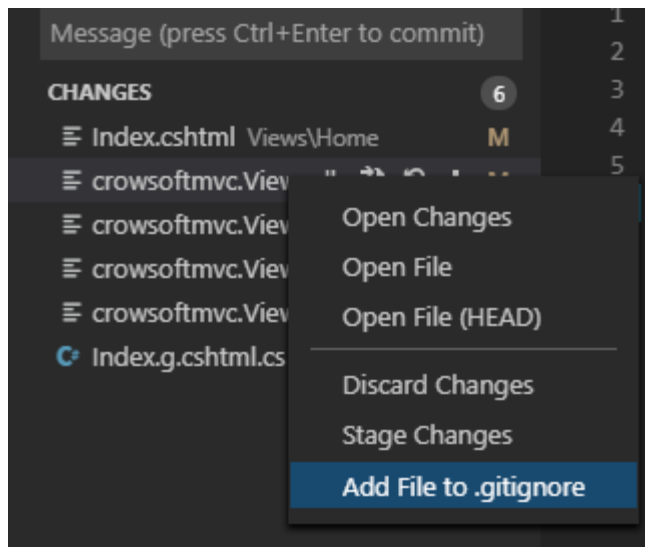
Now let's check in our changes.

Click on Source Control button. You will see a few CHANGES on the list.

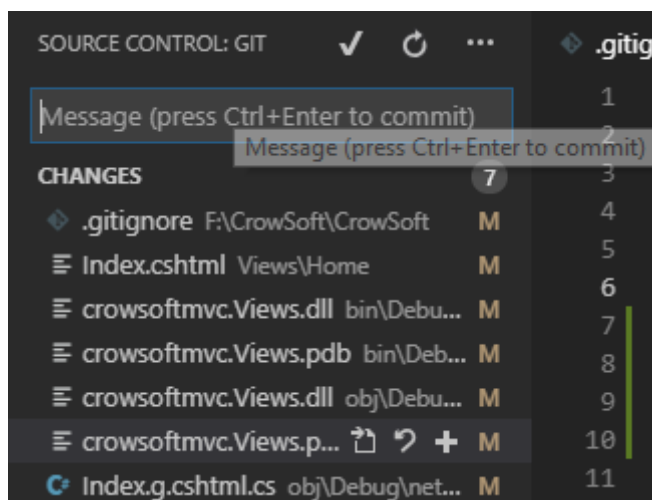


Created by: Charles Aylward
Updated: 03/04/2019

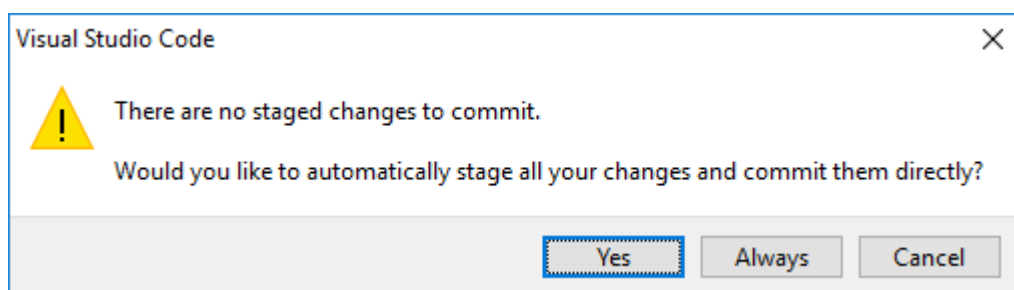
Any dll's and pdb files, must be ignored and not checked in. You can right click on those, and click Add File to .gitignore.



Enter your message below field, and click the commit button or press Ctrl-Enter.

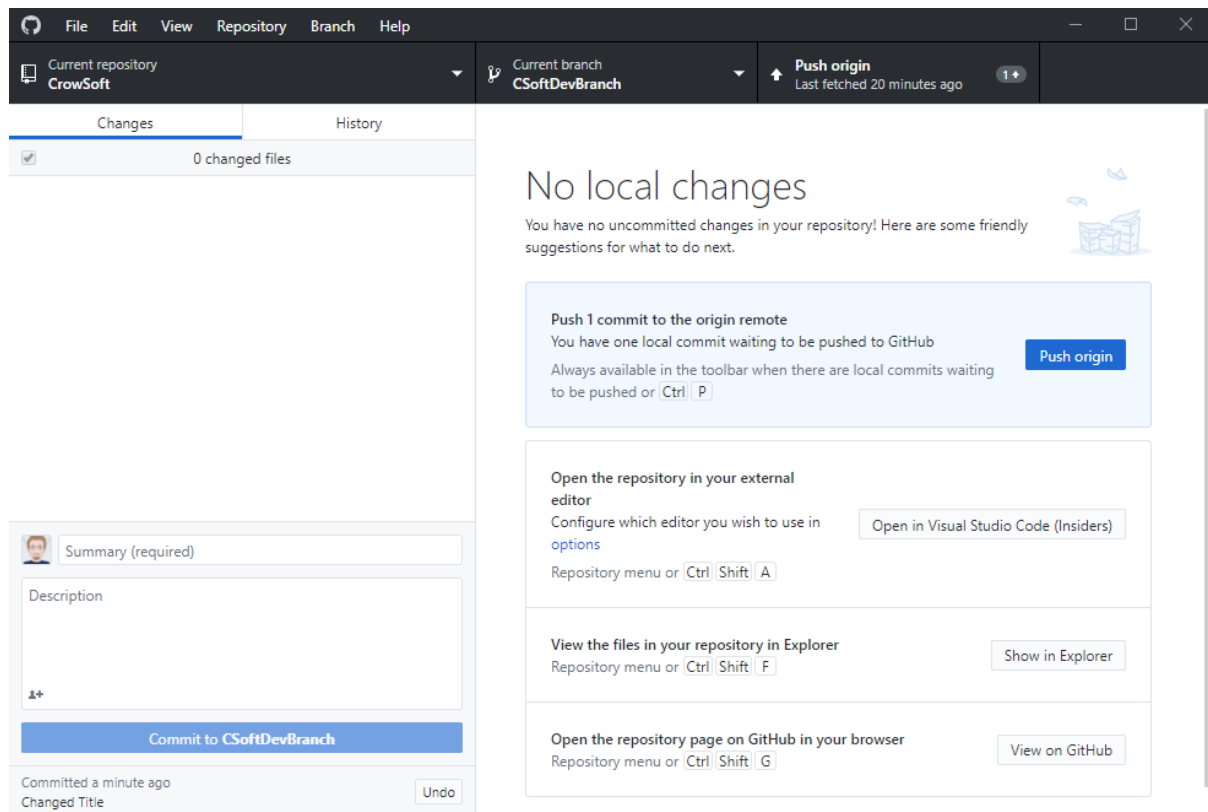


The following screen will appear. Click Yes. **NOTE:** This will only commit code locally. You still need to Push it to Github.

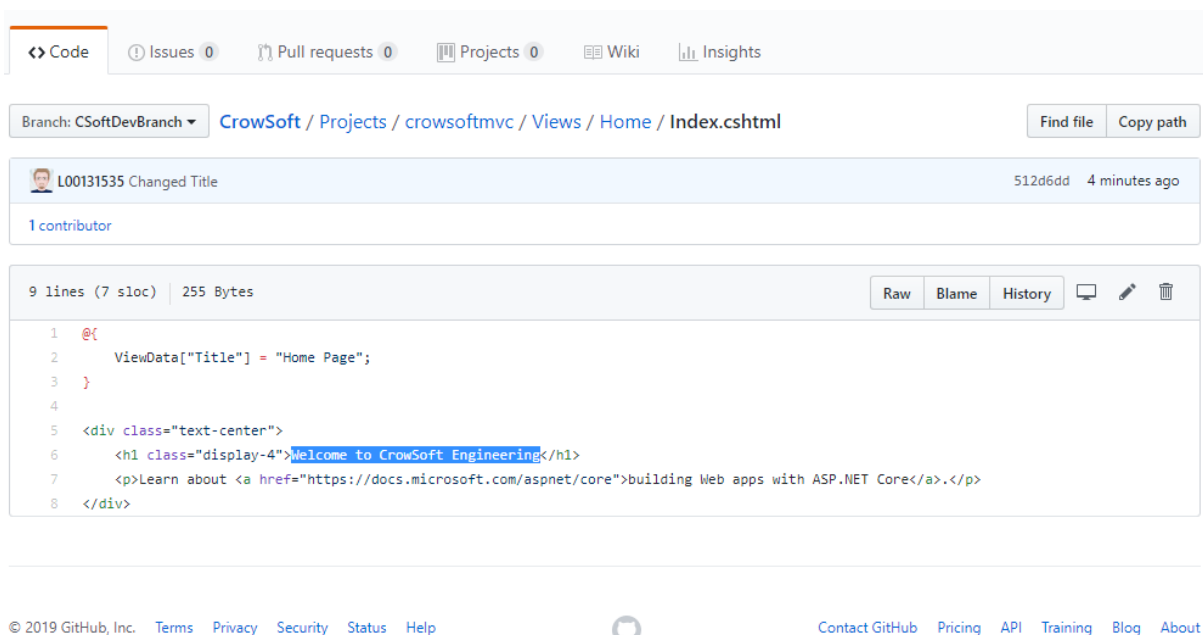


Created by: Charles Aylward
Updated: 03/04/2019

To push changes to Github, open Github Desktop, then Click Push origin or
for Linux or command line users: ***git push origin CSoftDevBranch***



Go to Github.com, then open file in this location:
CrowSoft/Projects/crowsoftmvc/Views/Home/Index.cshtml



Make sure the change is showing as seen above.

Bobs your uncle! 😊 Finish!

CrowSoft Technologies

Project Management

Jira:

CrowSoft project has been created

Project tasks to be discussed, estimated and tracked.

Environment and automation tools

These technologies are being proposed based on current course work.

To be discussed and agreed for inclusion on the project

Importance and sequence to be decided.

- VMs – Ubuntu 16.04
- Application Language – C#
- MySQL DB
- Ansible – run ssh connections to run configuration tasks – Python based
- Jenkins – create VM install OS with Jenkins instance
- Github account has been created –
- Artifactory jfrog – repository to store application artifacts executables
Installing jfrog Jenkins plugin – setting up trial jfrog account
- Testing – Nunit / Selenium
- Application logging -- log4net
- Security implementation for user/information stored

Functional Requirements:

Confirm requirements with product owner

- User logs in
- User adds files
- User adds images
- User queries the system (analysis features)
- Validation on information provided
- Security on information provided

Links:

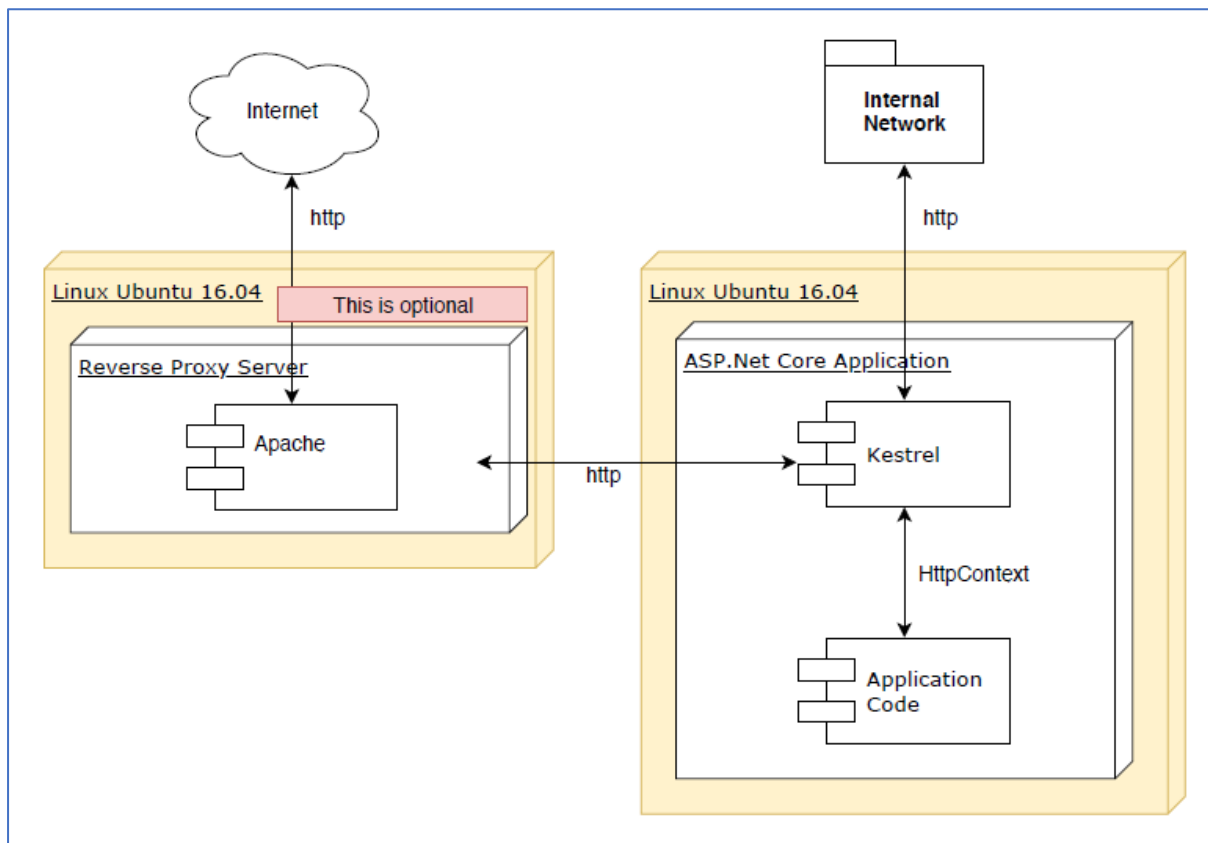
<https://studentjira.lyit.ie/projects/CSOFT/issues/CSOFT-2?filter=allopenissues>

<https://github.com/rlennon/CrowSoft>

<https://git-scm.com/docs> -- useful reference link

https://www.owasp.org/index.php/OWASP_Proactive_Controls - security info

Technology Stack



CrowSoft WebApp Copy and Publish to Dev Server

Contents

The web application service	2
Service Configuration	2
How to Check the Service Status	2
How to stop the service	2
How to start the service	2
Check if the web application is working	3
Copy Web App Project Files to Dev Server	3
To publish the Web Application	4

Date Created: 20/03/2019
Created By: Charles Aylward

The web application service

A service is required to be able to run the application without having to manually start the application in the console.

Service Configuration

The service name is: **crowsoft-webapp.service**

The config file of the service: located at /etc/systemd/system/crowsoft-webapp.service

```
devadmin@csoft-dev-ubuntu-golden: /home
GNU nano 2.5.3 File: /etc/systemd/system/crowsoft-webapp.service

[Unit]
Description=Crowsoft .NET MVC Web API App running on Ubuntu

[Service]
WorkingDirectory=/var/www/crowsoftwww
ExecStart=/usr/bin/dotnet /var/www/crowsoftwww/crowsoftmvc.dll
Restart=always
# Restart service after 10 seconds if the dotnet service crashes:
RestartSec=10
KillSignal=SIGINT
SyslogIdentifier=dotnet-example
User=www-data
Environment=ASPNETCORE_ENVIRONMENT=Development

[Install]
WantedBy=multi-user.target
```

How to Check the Service Status

To check the status of the service: *sudo systemctl status crowsoft-webapp.service*

NB: Make sure the status are: **active (running)**

```
devadmin@csoft-dev-ubuntu-golden:/home$ sudo systemctl status crowsoft-webapp.service
● crowsoft-webapp.service - Crowsoft .NET MVC Web API App running on Ubuntu
   Loaded: loaded (/etc/systemd/system/crowsoft-webapp.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2019-03-26 17:37:04 GMT; 1 weeks 3 days ago
     Main PID: 864 (dotnet)
    CGroup: /system.slice/crowsoft-webapp.service
            └─864 /usr/bin/dotnet /var/www/crowsoftwww/crowsoftmvc.dll

Mar 26 17:37:12 csoft-dev-ubuntu-golden dotnet-example[864]: warn: Microsoft.AspNetCore.DataProtection.KeyManagemen
Mar 26 17:37:12 csoft-dev-ubuntu-golden dotnet-example[864]: Neither user profile nor HKLM registry available
Mar 26 17:37:12 csoft-dev-ubuntu-golden dotnet-example[864]: info: Microsoft.AspNetCore.DataProtection.KeyManagemen
Mar 26 17:37:12 csoft-dev-ubuntu-golden dotnet-example[864]: Creating key {9eb7f333-1a9d-425f-bf3b-a5c204c921
Mar 26 17:37:12 csoft-dev-ubuntu-golden dotnet-example[864]: warn: Microsoft.AspNetCore.DataProtection.KeyManagemen
Mar 26 17:37:12 csoft-dev-ubuntu-golden dotnet-example[864]: No XML encryptor configured. Key {9eb7f333-1a9d-
Mar 26 17:37:14 csoft-dev-ubuntu-golden dotnet-example[864]: Hosting environment: Development
Mar 26 17:37:14 csoft-dev-ubuntu-golden dotnet-example[864]: Content root path: /var/www/crowsoftwww
Mar 26 17:37:14 csoft-dev-ubuntu-golden dotnet-example[864]: Now listening on: http://[::]:5123
Mar 26 17:37:14 csoft-dev-ubuntu-golden dotnet-example[864]: Application started. Press Ctrl+C to shut down.
lines 1-17/17 (END)
```

How to stop the service

To stop the service: *sudo systemctl stop crowsoft-webapp.service*

```
devadmin@csoft-dev-ubuntu-golden:/home$ sudo systemctl stop crowsoft-webapp.service
```

How to start the service

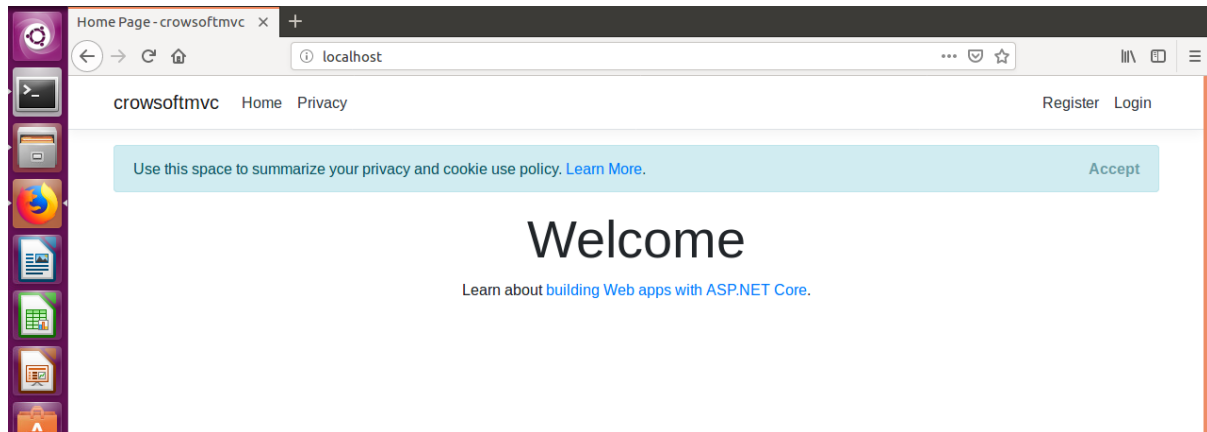
To start the service: *sudo systemctl start crowsoft-webapp.service*

```
devadmin@csoft-dev-ubuntu-golden:/home$ sudo systemctl start crowsoft-webapp.service
```

Date Created: 20/03/2019
Created By: Charles Aylward

Check if the web application is working

After the service is running, test the web app, open FireFox, and go enter localhost



Copy Web App Project Files to Dev Server

On the Dev Server, the folder for the project files should copy into the following folder:

```
devadmin@csoft-dev-ubuntu-golden:~/dev/crowsoftmvcv1$ ls
app.db          Controllers     Data           Properties
appsettings.Development.json  CrowsoftContext.cs  Models        Startup.cs
appsettings.json  crowsoftmvc.csproj  obj           Views
Areas            crowsoftmvc.csproj.user  Program.cs    wwwroot
bin              crowsoftmvc.sln        Program.cs.save
```

NOTE: The Staging and Production server will not have this folder. We should publish from either Jenkins or Artifactory for the final build.

Date Created: 20/03/2019
Created By: Charles Aylward

To publish the Web Application

Steps to publish new code:

1. Make sure the service is stopped. Command to stop: ***sudo systemctl stop crowsoft-webapp.service***
2. Run the following command Command: ***sudo dotnet publish ~/dev/crowsoftmvc1/crowsoftmvc.csproj -c release -o /var/www/crowsoftwww***
3. Now start the service. Command to start: ***sudo systemctl start crowsoft-webapp.service***
4. Optional: Check status of service: ***sudo systemctl status crowsoft-webapp.service***
5. This is how the folder content should look like.

```
devadmin@csoft-dev-ubuntu-golden:~$ sudo dotnet publish ~/dev/crowsoftmvc1/crowsoftmvc.csproj -c release -o /var/www/crowsoftwww
Microsoft (R) Build Engine version 15.9.20+g88f5fadf8e for .NET Core
Copyright (c) Microsoft Corporation. All rights reserved.

Restore completed in 79.62 ms for /home/devadmin/dev/crowsoftmvc1/crowsoftmvc.csproj.
Models/Dummy.cs(10,33): warning CS0169: The field 'Dummy.context' is never used [/home/devadmin/dev/crowsoftmvc1/crowsoftmvc.csproj]
crowsoftmvc -> /home/devadmin/dev/crowsoftmvc1/bin/release/netcoreapp2.2/crowsoftmvc.dll
crowsoftmvc -> /home/devadmin/dev/crowsoftmvc1/bin/release/netcoreapp2.2/crowsoftmvc.Views.dll
crowsoftmvc -> /var/www/crowsoftwww/
devadmin@csoft-dev-ubuntu-golden:~$
```

```
root@csoft-dev-ubuntu-golden:/var/www/crowsoftwww# ls
app.db
appsettings.Development.json
appsettings.json
crowsoftmvc.deps.json
crowsoftmvc.dll
crowsoftmvc.pdb
crowsoftmvc.runtimeconfig.json
crowsoftmvc.Views.dll
crowsoftmvc.Views.pdb
dotnet-aspnet-codegenerator-design.dll
Google.Protobuf.dll
Microsoft.CodeAnalysis.CSharp.Workspaces.dll
Microsoft.CodeAnalysis.Workspaces.dll
Microsoft.Data.Sqlite.dll
Microsoft.EntityFrameworkCore.Abstractions.dll
Microsoft.EntityFrameworkCore.dll
Microsoft.EntityFrameworkCore.Relational.dll
Microsoft.EntityFrameworkCore.Sqlite.dll
Microsoft.VisualStudio.Web.CodeGeneration.Contracts.dll
Microsoft.VisualStudio.Web.CodeGeneration.Core.dll
Microsoft.VisualStudio.Web.CodeGeneration.dll
Microsoft.VisualStudio.Web.CodeGeneration.EntityFrameworkCore.dll
Microsoft.VisualStudio.Web.CodeGeneration.Templating.dll
Microsoft.VisualStudio.Web.CodeGeneration.Utils.dll
Microsoft.VisualStudio.Web.CodeGenerators.Mvc.dll
MySQL.Data.dll
NuGet.Frameworks.dll
Remotion.Linq.dll
runtimes
SQLitePCLRaw.batteries_green.dll
SQLitePCLRaw.batteries_v2.dll
SQLitePCLRaw.core.dll
SQLitePCLRaw.provider.e_sqlite3.dll
System.Composition.AttributedModel.dll
System.Composition.Convention.dll
System.Composition.Hosting.dll
System.Composition.Runtime.dll
System.Composition.TypedParts.dll
```