

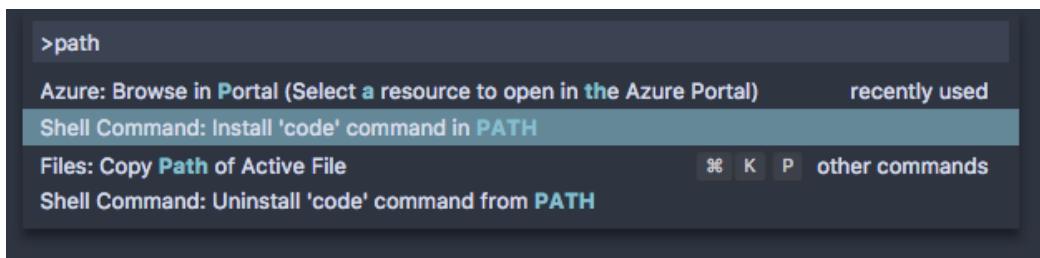
Configure CI/CD for Node application with Azure Pipelines

Overview

In this lab, we will present a scenario where Visual Studio Team Services (VSTS) can enable Node.js developers with continuous integration and deployment features. The scenario demonstrates how a Node.js developer using [Visual Studio Code](#) and various Azure-related VS Code extensions to create a new Azure App Service and use Git-based deployment. You'll learn how a Node.js developer can use VSTS to ensure that their code is deployed automatically to an Azure App Service when code is committed to a VSTS Git repository.

Prerequisites

1. If you don't already have a Visual Studio Dev Essentials and Visual Studio Team Services account, create one at [my.visualstudio.com](#).
2. If you don't already have an Azure subscription, create one [here](#).
3. Install Git if you don't already have it installed.
4. Install Node.js either by using the downloadable installers from [Node.js](#) or using various command-line tools. Mac users can install Node.js using [Homebrew](#) using the command `brew install node`.
5. Install [Visual Studio Code](#).
6. Install the [Azure App Service Tools](#) using the VS Code Extensions Palette (search for *azure*).
7. Once you have installed Visual Studio Code and the App Service Tools extension, open Visual Studio Code. In Code, use `Ctrl-Shift-P` (or `Cmd-Shift-P` on a Mac) to open up the command palette. Type the word `path` to filter the command menu, and then select the command **Install 'code' command in PATH** to make it easy to open Visual Studio Code from your command line.



Demo Steps

[Express](#) is a minimal, flexible web application framework for Node.js developers that provides a robust set of features for both web and mobile web developers. The [Express package](#) is available via [NPM](#), easily installable into any Node.js application using the command `npm install express`. In most cases Express needn't be installed as a global

package using the `-g` NPM switch, as the convention is that Express is installed independently into each Node.js app during development.

This demo will walk you through the process of creating a new Node.js web app that uses Express. Then you'll create a new Visual Studio Team Services project to use for storing the code and continuously deploying the app to Azure App Service.

Scaffolding a new Express app using Yeoman

To ease the process of bootstrapping a Node.js Express application, developers can use the [Yeoman](#) template engine. Yeoman - known as `yo` from its NPM CLI command name - has hundreds of scaffolders for all sorts of projects. Luckily, Yeoman is easily installable as an NPM package just like Express and many other Node.js components.

Install Yeoman and the Express generator

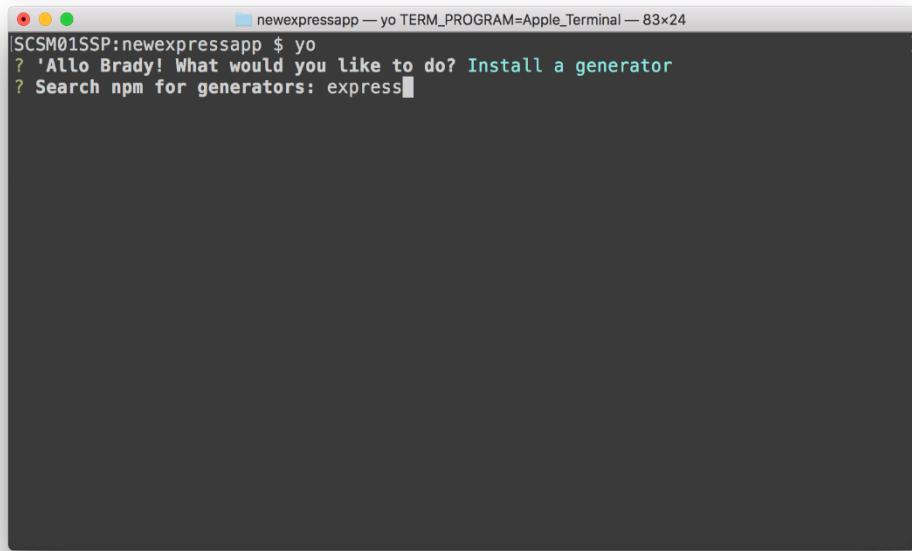
1. Run the command `npm install -g yo`.
2. Type `yo` at the command prompt.



A screenshot of a Mac OS X terminal window titled "newexpressapp — yo TERM_PROGRAM=Apple_Terminal — 83x24". The window shows the following text:

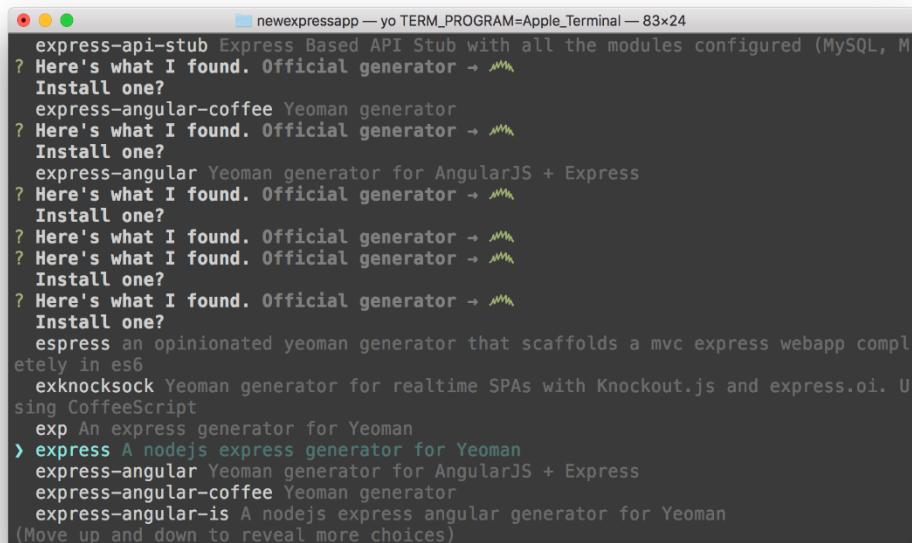
```
[SCSM01SSP:newexpressapp $ yo
? 'Allo Brady! What would you like to do?
Node
Update your generators
> Install a generator
Find some help
Clear global config
Get me out of here!
(Move up and down to reveal more choices)
```

3. Type `express` at the prompt and hit enter to search the Yeoman template database for Express-related scaffolders.



```
|SCSM01SSP:newexpressapp $ yo
? 'Allo Brady! What would you like to do? Install a generator
? Search npm for generators: express
```

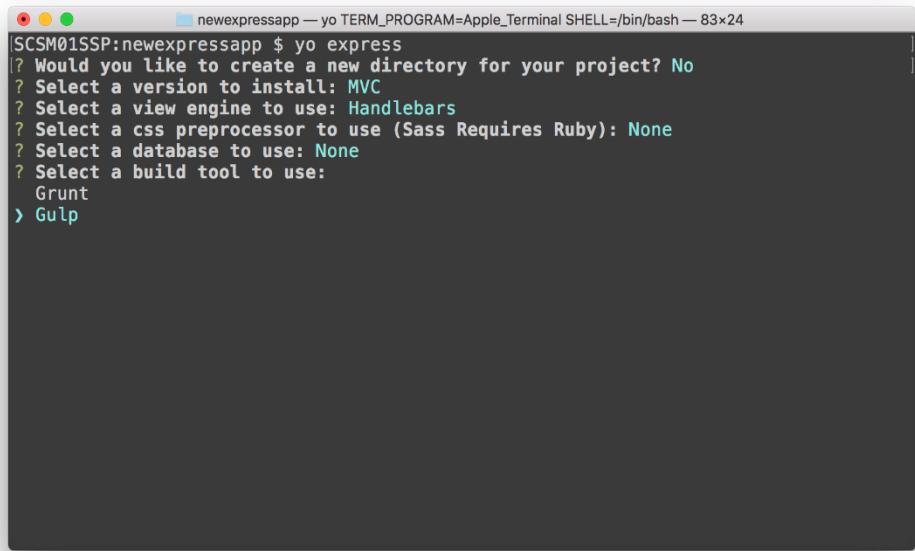
4. Scroll through the list of results until you see the generator named, simply, **express**, and then hit enter.



```
|SCSM01SSP:newexpressapp — yo TERM_PROGRAM=Apple_Terminal — 83x24
express-api-stub Express Based API Stub with all the modules configured (MySQL, M
? Here's what I found. Official generator → ↵
Install one?
express-angular-coffee Yeoman generator
? Here's what I found. Official generator → ↵
Install one?
express-angular Yeoman generator for AngularJS + Express
? Here's what I found. Official generator → ↵
Install one?
? Here's what I found. Official generator → ↵
? Here's what I found. Official generator → ↵
Install one?
? Here's what I found. Official generator → ↵
Install one?
express an opinionated yeoman generator that scaffolds a mvc express webapp compl
etely in es6
exknocksock Yeoman generator for realtime SPAs with Knockout.js and express.oi. U
sing CoffeeScript
exp An express generator for Yeoman
> express A nodejs express generator for Yeoman
express-angular Yeoman generator for AngularJS + Express
express-angular-coffee Yeoman generator
express-angular-is A nodejs express angular generator for Yeoman
(Move up and down to reveal more choices)
```

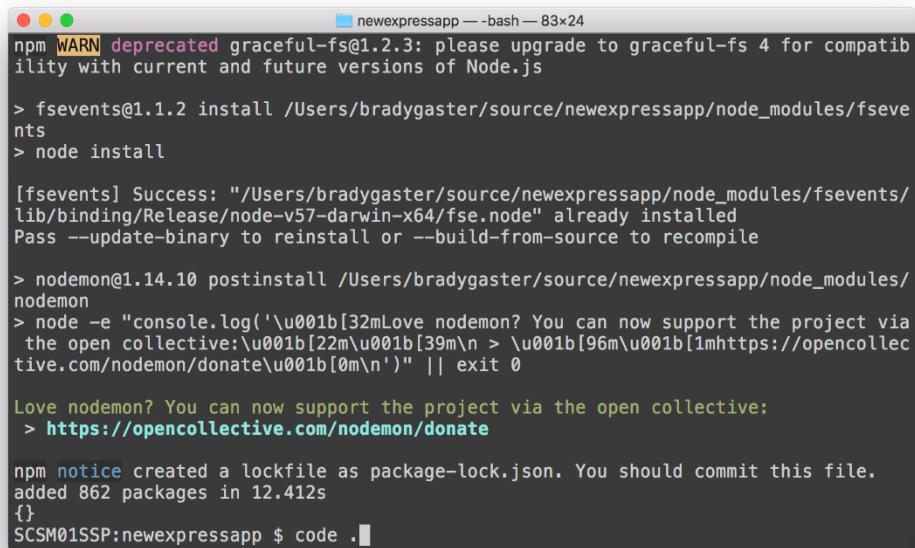
Generate a skeleton Express app

1. In the command prompt, enter the command **yo express** to create a new Express application.



```
[SCSM01SSP:newexpressapp $ yo express
? Would you like to create a new directory for your project? No
? Select a version to install: MVC
? Select a view engine to use: Handlebars
? Select a css preprocessor to use (Sass Requires Ruby): None
? Select a database to use: None
? Select a build tool to use:
  Grunt
> Gulp
```

2. Enter the command `code .` to start Visual Studio Code with the new Express app's folder open.



```
npm WARN deprecated graceful-fs@1.2.3: please upgrade to graceful-fs 4 for compatibility with current and future versions of Node.js

> fsevents@1.1.2 install /Users;bradygaster/source/newexpressapp/node_modules/fsevents
> node install

[fsevents] Success: "/Users;bradygaster/source/newexpressapp/node_modules/fsevents/lib/binding/Release/node-v57-darwin-x64/fse.node" already installed
Pass --update-binary to reinstall or --build-from-source to recompile

> nodemon@1.14.10 postinstall /Users;bradygaster/source/newexpressapp/node_modules/nodemon
> node -e "console.log('\u001b[32mLove nodemon? You can now support the project via the open collective:\u001b[22m\u001b[39m\n > \u001b[96m\u001b[1mhttps://opencollective.com/nodemon/donate\u001b[0m\n')" || exit 0

Love nodemon? You can now support the project via the open collective:
> https://opencollective.com/nodemon/donate

npm notice created a lockfile as package-lock.json. You should commit this file.
added 862 packages in 12.412s
{}
SCSM01SSP:newexpressapp $ code .
```

3. Edit `gulpfile.js` to turn off local hosting during the Gulp build. The code for the `default` task should be commented out:

```
gulp.task('default', [
  //['develop'
]);
```

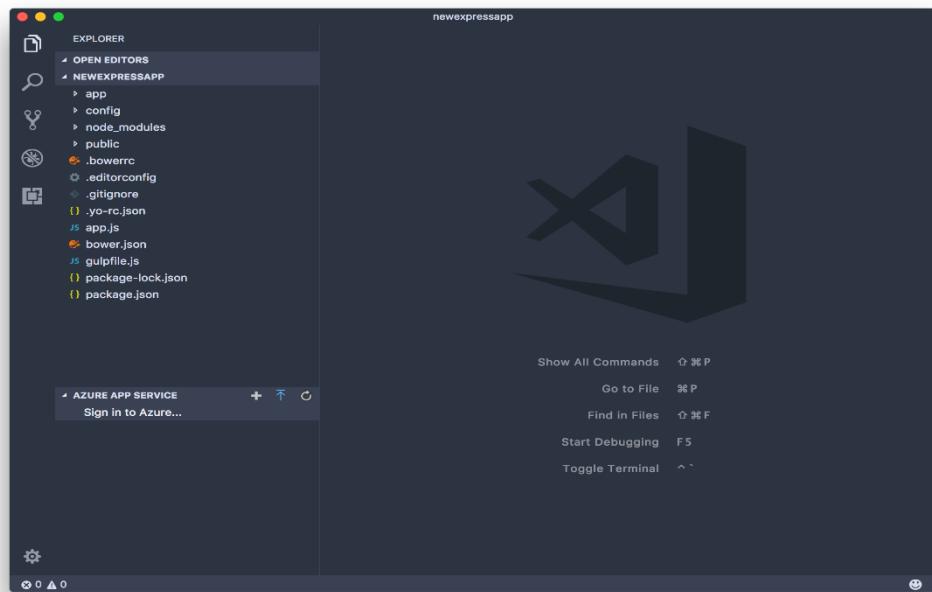
Create an Azure App Service using VS Code

There are a number of various extensions for VS Code that provide Azure functionality. Extensions like the [Azure CLI Tools](#) make it easy to write scripts that make use of the [Azure CLI](#) to perform administrative tasks in your Azure nn.

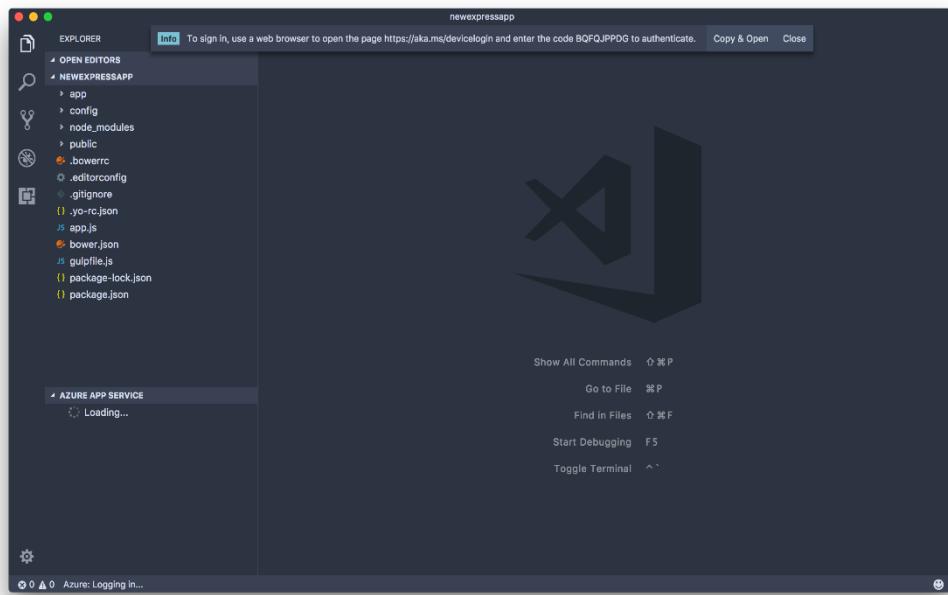
The [Azure App Service Tools for VS Code](#) provide a series of convenient features for creating new Azure App Services.

Login to your Azure subscription

1. Expand the Azure App Service Tools pane. Click the [Sign in to Azure](#) gesture in the explorer pane.



2. Click [Copy](#) and [Open](#) to open the browser and log in to your Azure subscription.



3. Hit **Ctrl-V** or **Cmd-V** on Mac, or use the context menu to paste the login code into the textbox.

Device Login

Enter the code that you received from the application on your device

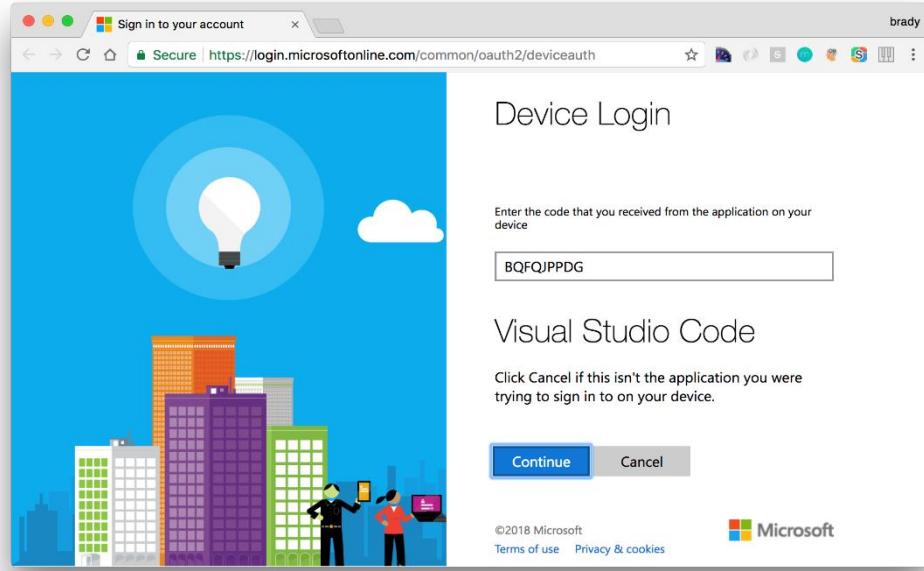
Code

Cut
Copy
Paste
Spelling and Grammar ►
Substitutions ►
Transformations ►
Font ►
Speech ►
Paragraph Direction ►

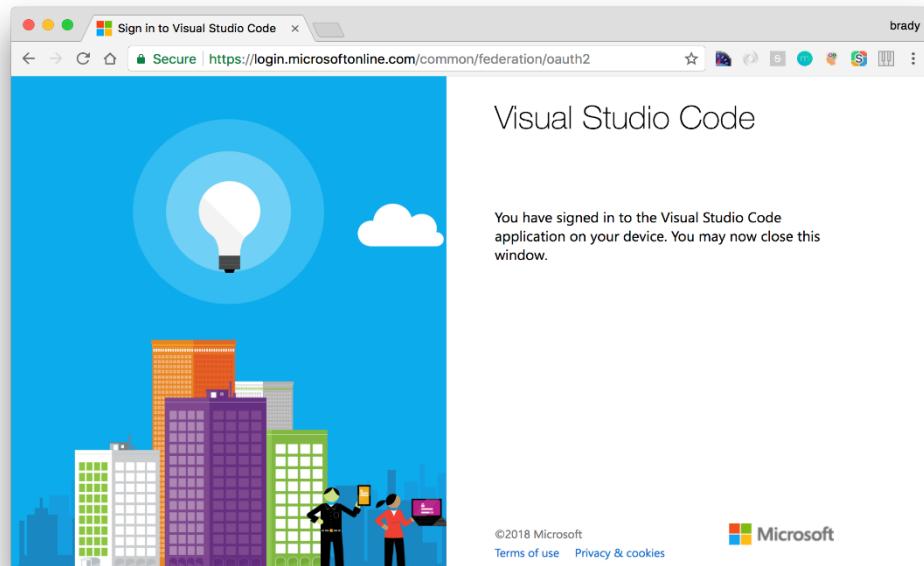
© 2017 Microsoft
[Terms of use](#) [Privacy & Cookies](#)

Microsoft

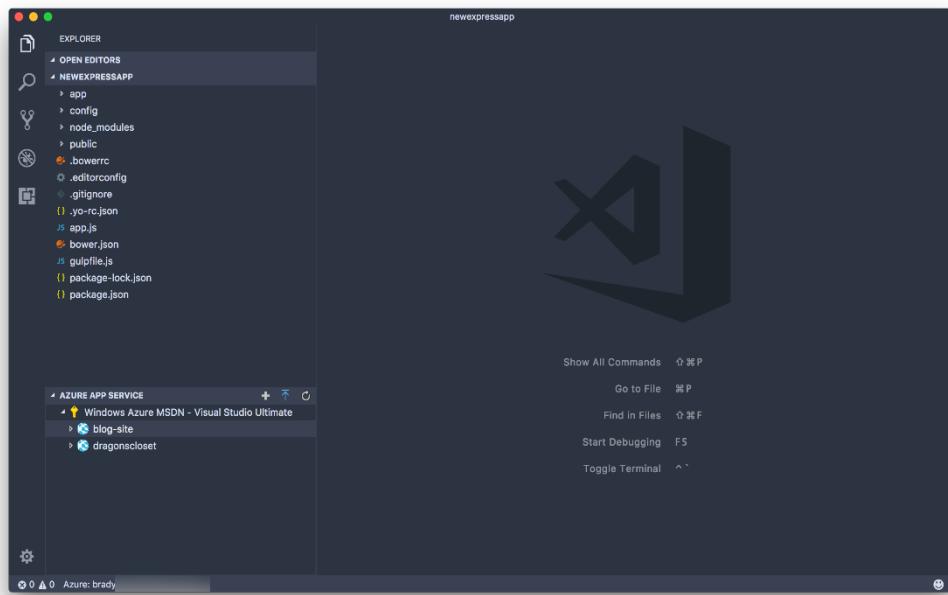
4. Click the **Continue** button to perform the login routine.



Once you click Continue, you'll be able to select which of the organizational or Microsoft accounts you want to use to login to your Azure subscription.



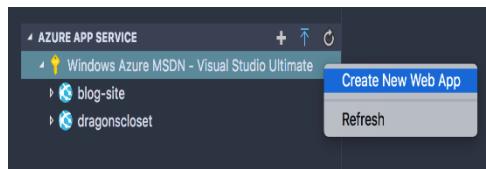
5. Once the login page loads, flip back to VS Code. You should see all of your Azure subscriptions in the App Service pane. When you expand the pane, all of your App Services are visible.



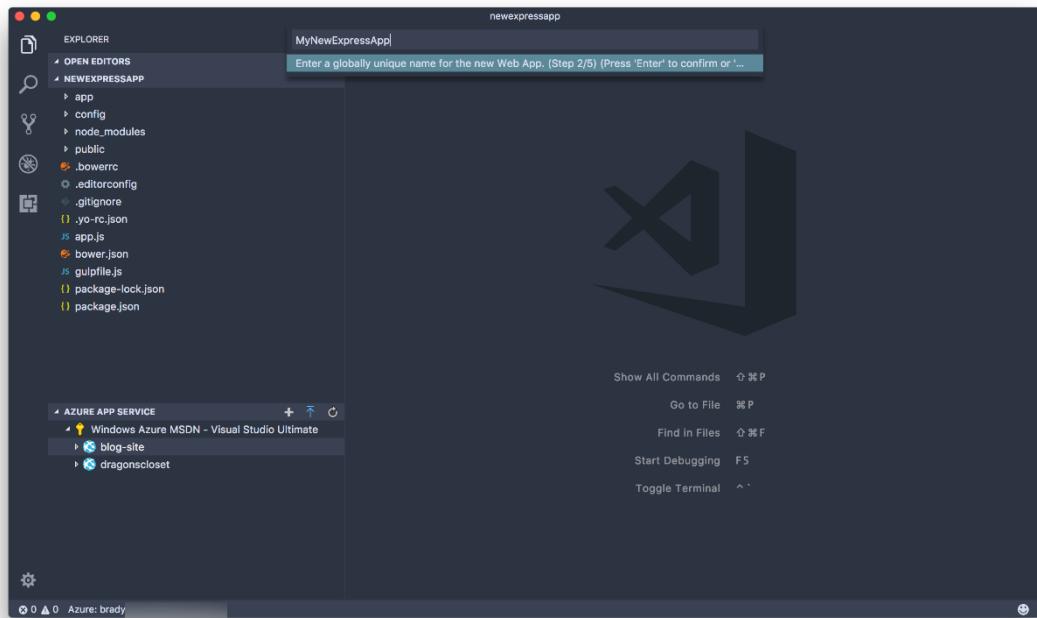
Create the App Service Web App

To setup continuous deployment to an Azure App Service, an Azure App Service resource must be created into which code can be deployed. Luckily, Visual Studio, Visual Studio for Mac, and Visual Studio Code all make this a rather easy task directly within the various IDEs.

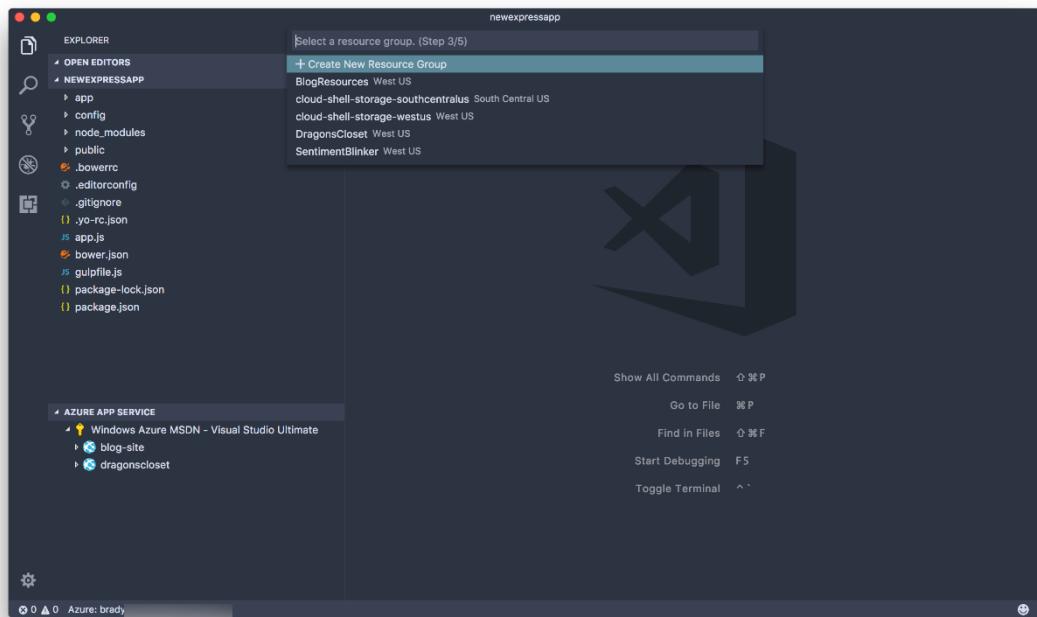
1. In VS Code, right-click the subscription you want to use and select the [Create New Web App](#) menu option.



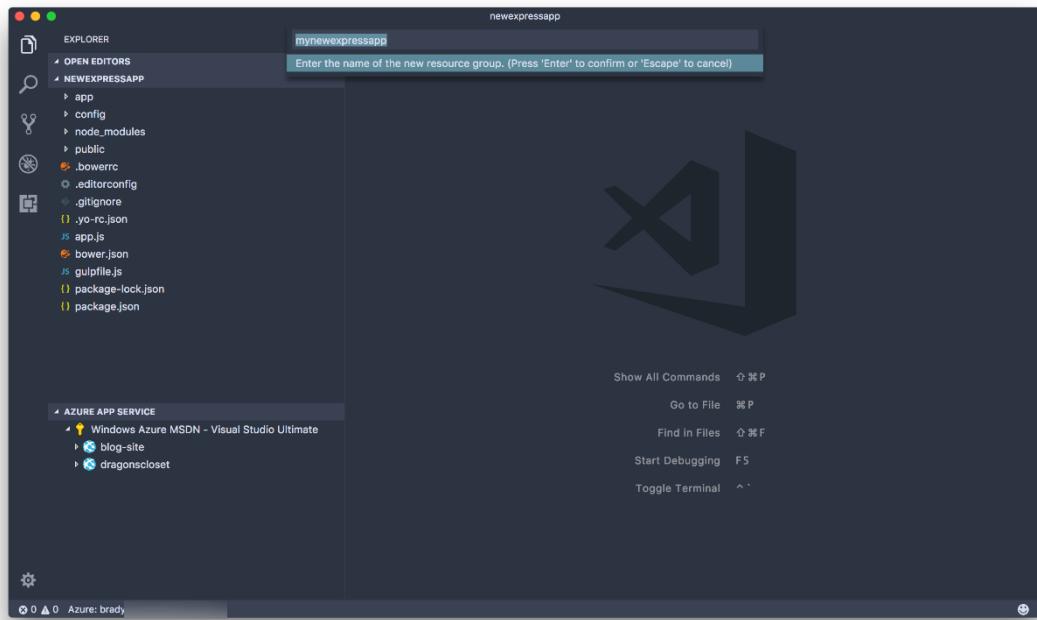
2. Type in a name for the App Service. Note, if an existing App Service with this name exists you'll be informed with an error message.



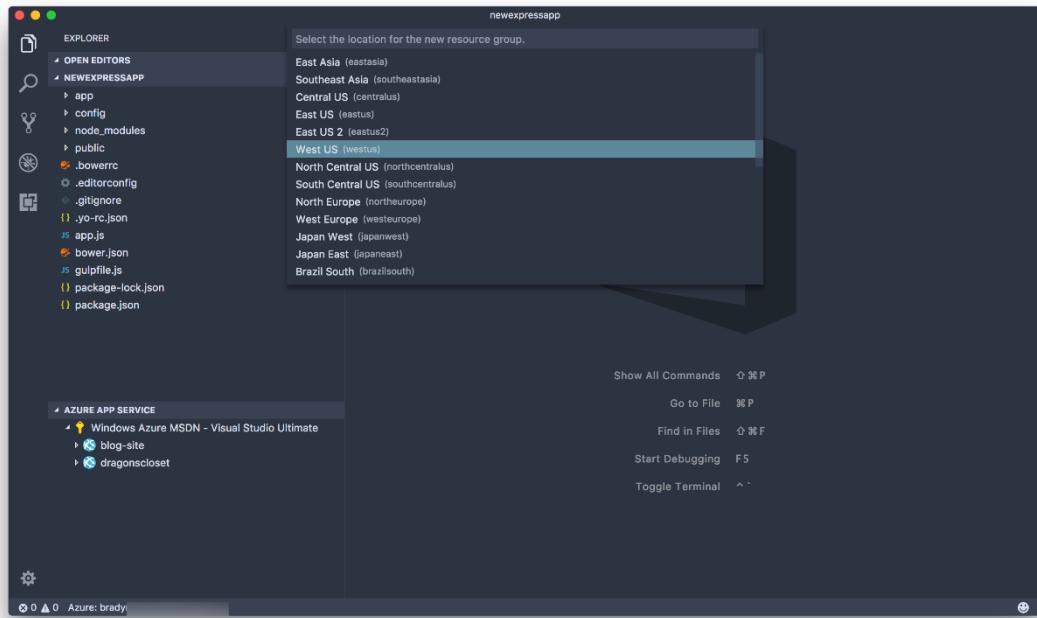
3. Create a new resource group for the App Service.



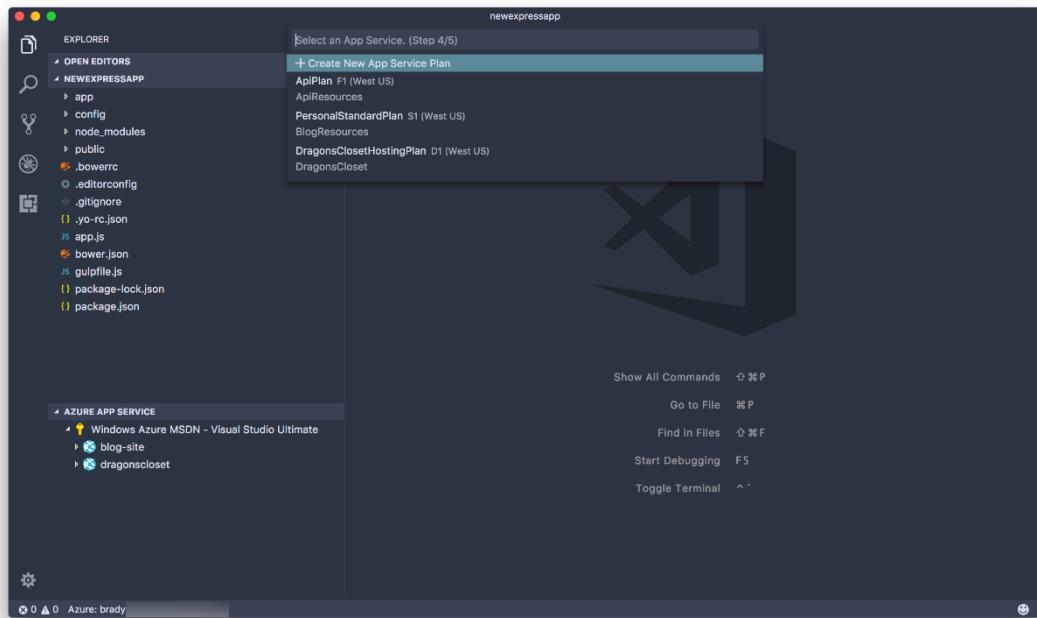
4. Give the new resource group a name.



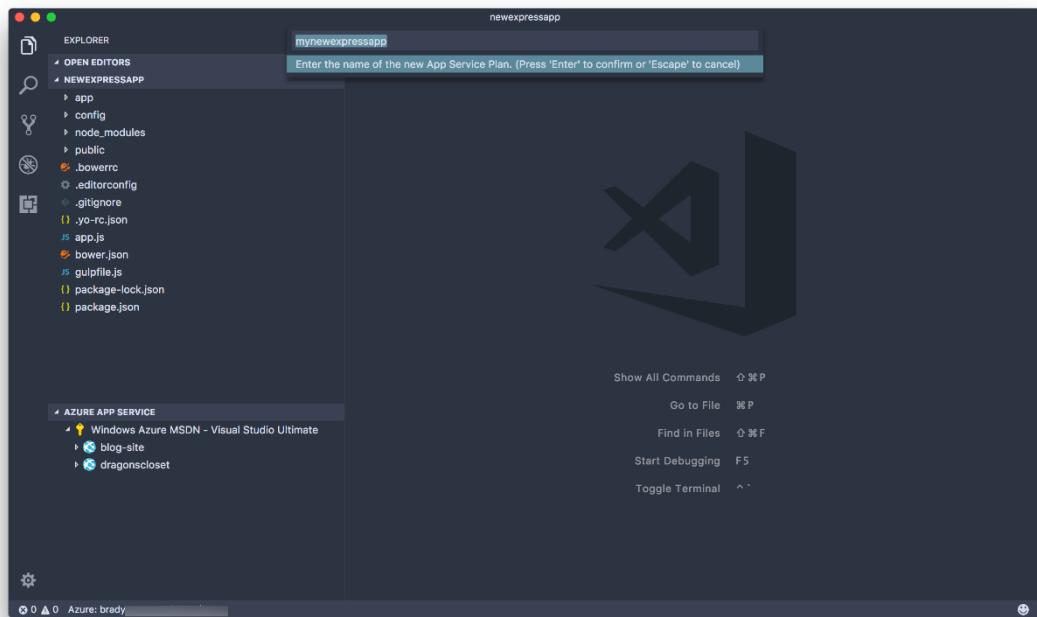
5. Select the geographic region from the menu.



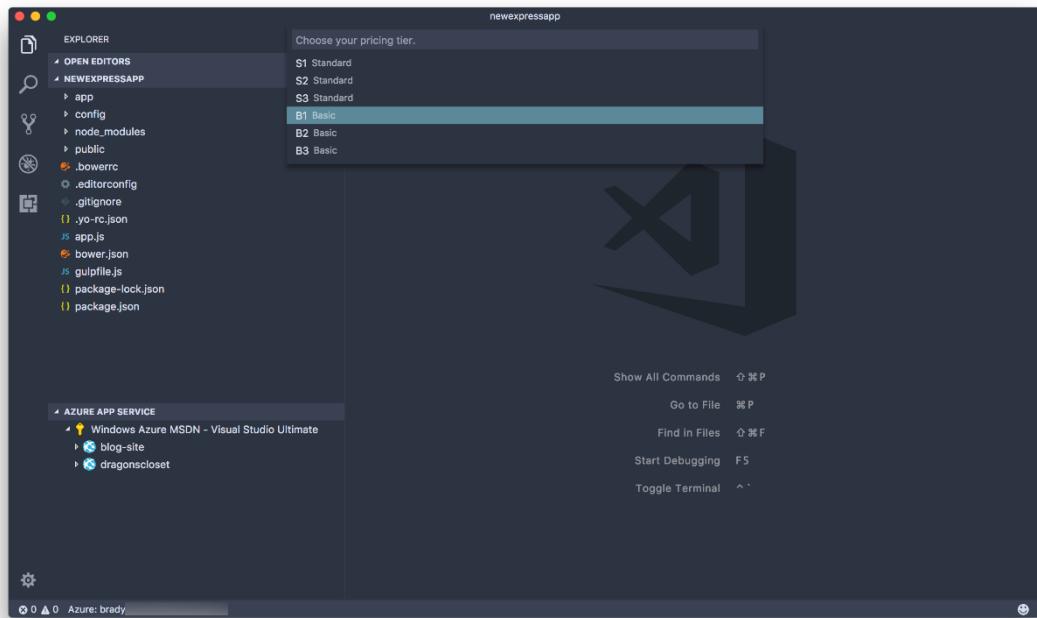
6. Select the option for creating a new App Service Plan.



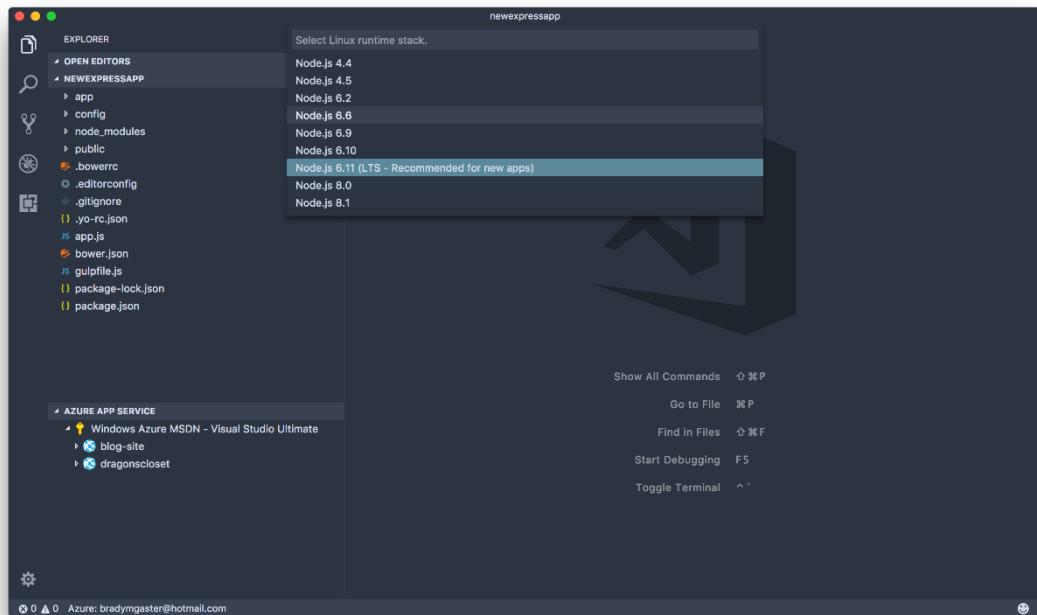
7. Give the new App Service Plan a name.



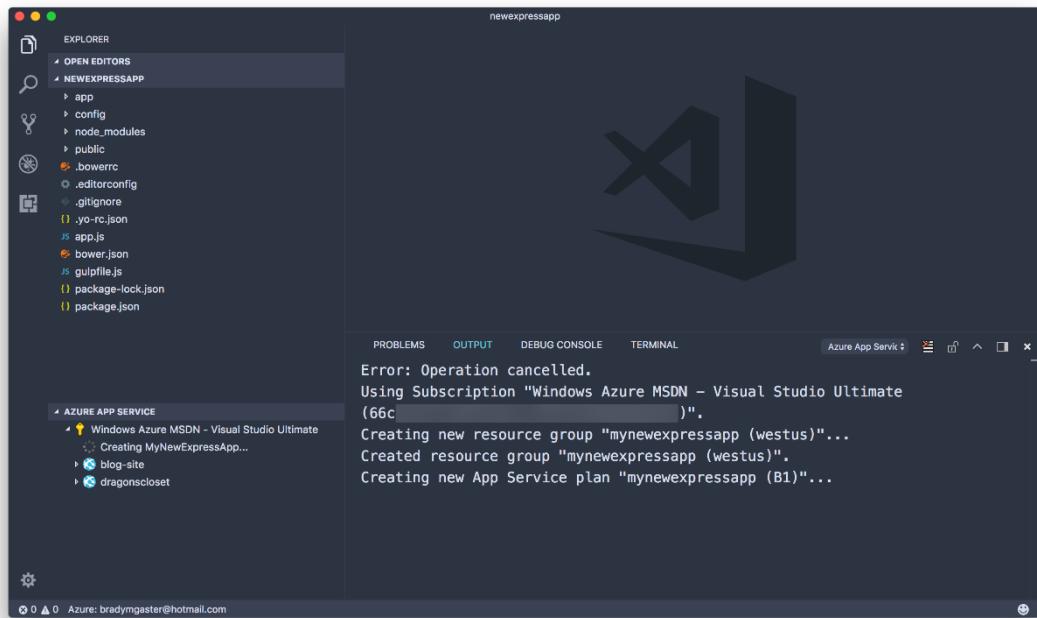
8. Select the tier of service.



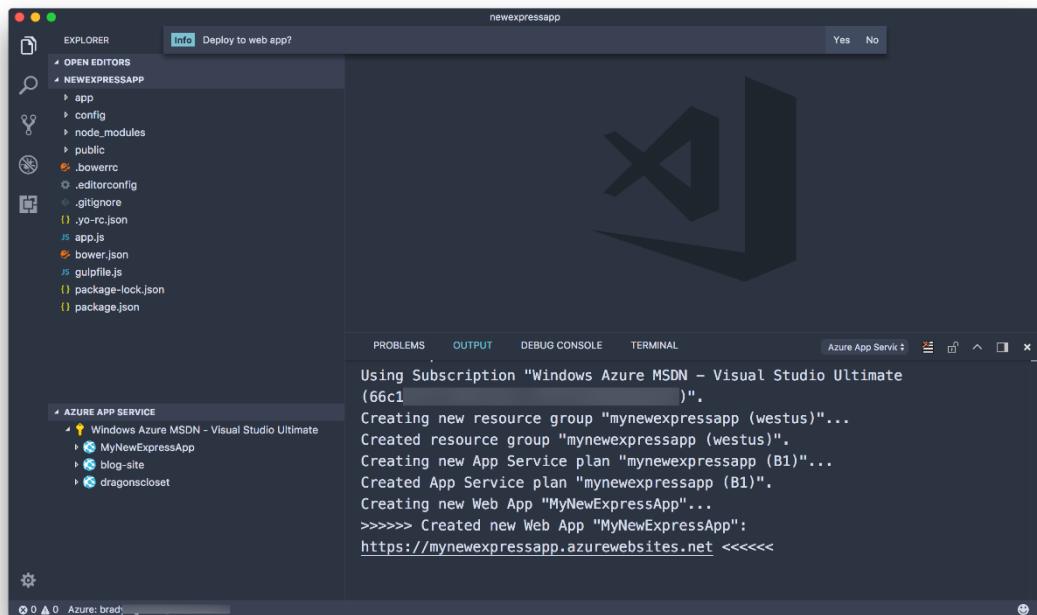
9. Select the Node.js version your app will need. VS Code will try to determine this for you but you have the option to choose your version.



10. Visual Studio Code will provide a stream of information as the resources are being created in your subscription.

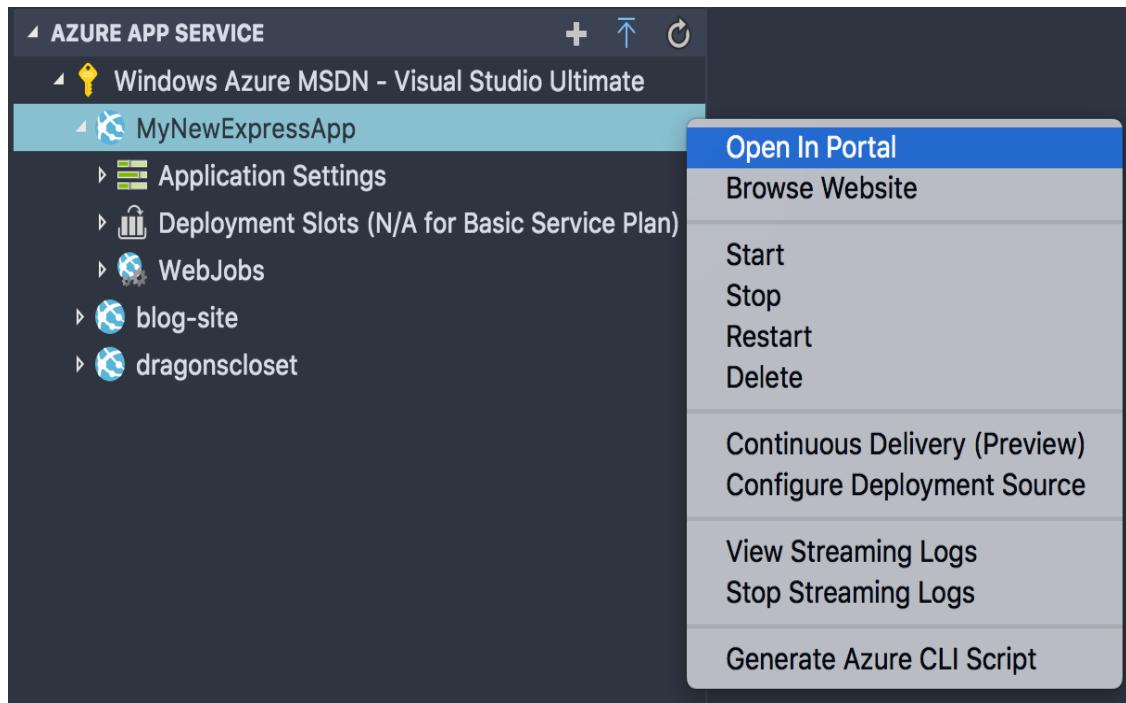


11. Once the App Service is created, it will appear in the App Service Explorer pane



Viewing the new App Service in the Azure Portal

1. In VS Code, right-click on the App Service to open the context menu. Select the **Open in Portal** option to open the new App Service in the Azure Portal.

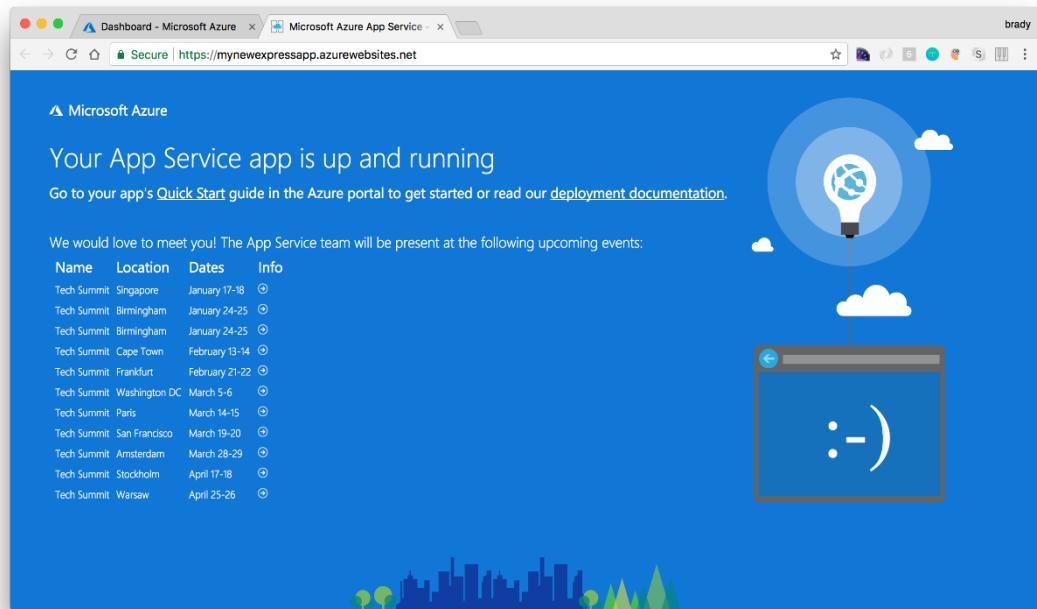


- Once you select an App Service to view within VS Code it'll open directly in the Azure portal in your default browser. Click the link in the portal's overview section to open the new App Service in a separate tab.

The screenshot shows the Azure portal's overview for the 'MyNewExpressApp' app service. Key details include:

- Resource group:** mynewexpressapp
- Status:** Running
- Location:** West US
- Subscription:** Windows Azure MSDN - Visual Studio Ultimate
- HTTP 5xx Errors:** A chart showing errors from 11 PM to 11:30 PM. Values: 100, 80, 60, 40, 20, 0.
- Data In:** A chart showing data transfer from 11 PM to 11:30 PM. Values: 100B, 80B, 60B, 40B, 20B, 0B.

- The App Service should open in your browser, displaying the default App Service start page.



Use Team Services to Enable Continuous Deployment

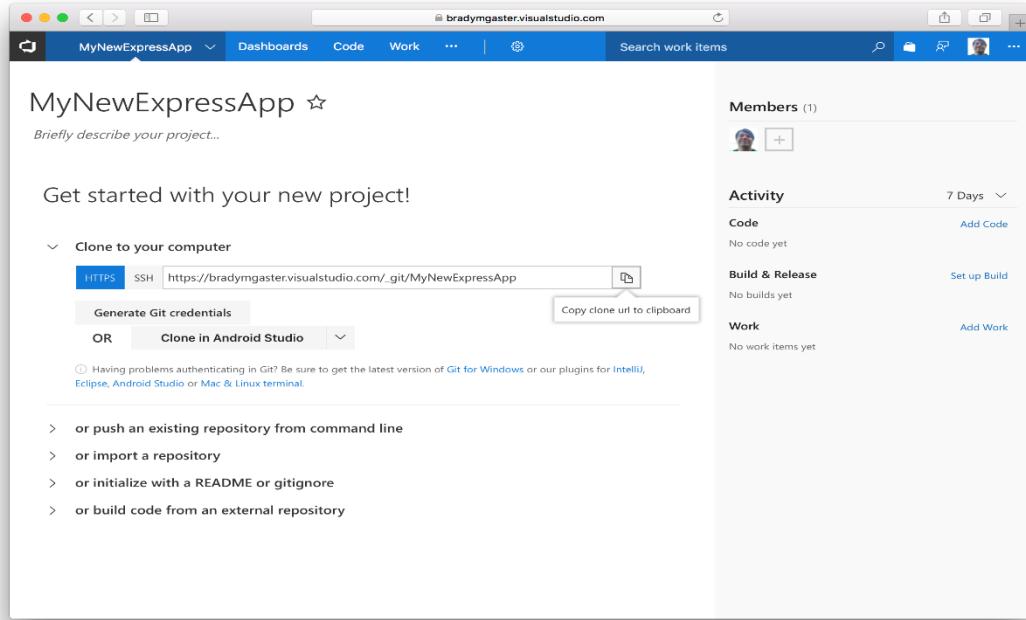
Now that the app is ready to be deployed we'll set up a new Team Services project and set it up with Git so the code can be easily committed and continuously deployed.

Create the new Team Services Project

1. Browse to your Team Services account at [https://\[youraccount\].visualstudio.com](https://[youraccount].visualstudio.com). Give the project a name and click **Create**.

A screenshot of the Microsoft Team Services (VSTS) interface. The user is on the 'Create new project' page. The top navigation bar shows 'bradymgaster.visualstudio.com'. The main area has a blue background with icons related to software development. The title 'Good morning, Brady Gaster' is displayed. Below the title, there are several project management icons. The 'Create new project' section contains fields for 'Project name' (set to 'MyNewExpressApp'), 'Description' (empty), 'Version control' (set to 'Git'), and 'Work item process' (set to 'Agile'). At the bottom right are 'Create' and 'Cancel' buttons.

2. If you haven't already created Git credentials click the **Generate Git credentials** button, then provide a username and password to be used when interacting with this project. Then, click the **Copy clone url to clipboard** button as shown below.



3. At a command prompt in your project's folder, type the command `git init` to initialize a new repository locally. Then, type `git remote add origin {copied URL}` to add the remote to your local repository. Then, `add` and `commit` the files to the repository. Finally, use `git push origin master` to push the Express app code up to the VSTS project's Git repository.

```
bradygaster@Bradys-MacBook-Pro ~/source/NewExpressApp: git init
Initialized empty Git repository in /Users;bradygaster/source/NewExpressApp/.git/
bradygaster@Bradys-MacBook-Pro (master #) ~/source/NewExpressApp: git remote add origin
https://bradygaster.visualstudio.com/_git/MyNewExpressApp
bradygaster@Bradys-MacBook-Pro (master #) ~/source/NewExpressApp: git add .
bradygaster@Bradys-MacBook-Pro (master +) ~/source/NewExpressApp: git commit -m 'initial creation'
[master (root-commit) 11890cc] initial creation
 19 files changed, 6017 insertions(+)
 create mode 100644 .DS_Store
 create mode 100644 .bowerrc
 create mode 100644 .editorconfig
 create mode 100644 .gitignore
 create mode 100644 .yo-rc.json
 create mode 100644 app.js
 create mode 100644 app/controllers/home.js
 create mode 100644 app/models/article.js
 create mode 100644 app/views/error.handlebars
 create mode 100644 app/views/index.handlebars
 create mode 100644 app/views/layouts/main.handlebars
 create mode 100644 app/views/partials/welcome.handlebars
 create mode 100644 bower.json
 create mode 100644 config/config.js
 create mode 100644 config/express.js
 create mode 100644 gulpfile.js
 create mode 100644 package-lock.json
 create mode 100644 package.json
 create mode 100644 public/css/style.css
bradygaster@Bradys-MacBook-Pro (master) ~/source/NewExpressApp: git push origin master
```

- Click on the **Code** tab in your Team Services browser.

Name	Last change	Commits
app	7 minutes ago	11890ccb initial creation bradygaster
config	7 minutes ago	11890ccb initial creation bradygaster
public/css	7 minutes ago	11890ccb initial creation bradygaster
.bowerrc	7 minutes ago	11890ccb initial creation bradygaster
.DS_Store	7 minutes ago	11890ccb initial creation bradygaster
editorconfig	7 minutes ago	11890ccb initial creation bradygaster
.gitignore	7 minutes ago	11890ccb initial creation bradygaster
.yo-rc.json	7 minutes ago	11890ccb initial creation bradygaster
app.js	7 minutes ago	11890ccb initial creation bradygaster
bower.json	7 minutes ago	11890ccb initial creation bradygaster
gulpfile.js	7 minutes ago	11890ccb initial creation bradygaster
package-lock.json	7 minutes ago	11890ccb initial creation bradygaster
package.json	7 minutes ago	11890ccb initial creation bradygaster

Create the Build Definition

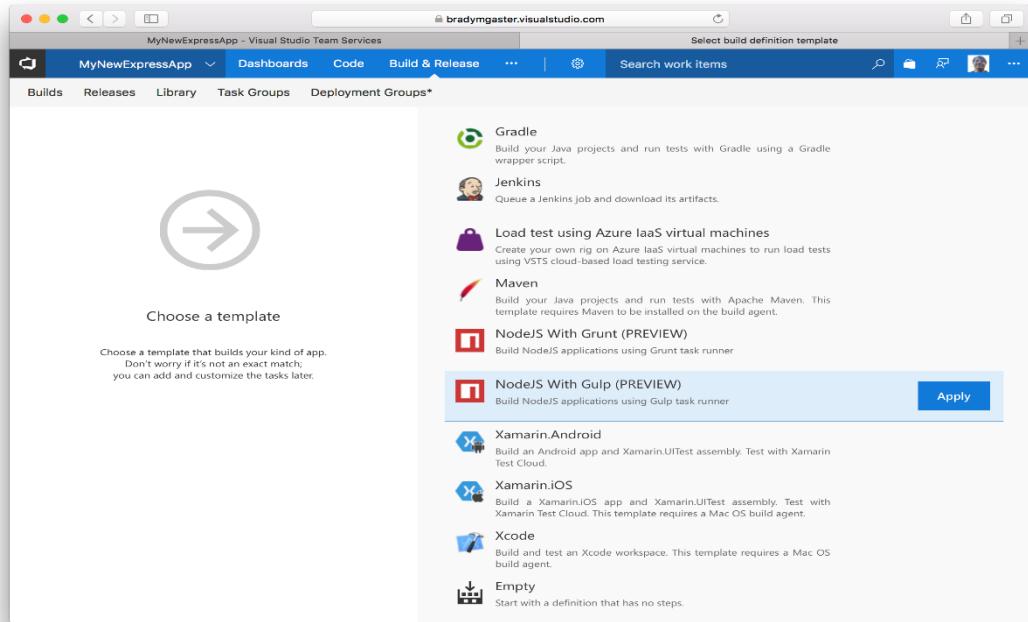
- Click the **Set up build** button in the **Files** page in the browser.

+ New file Upload file(s) ↴

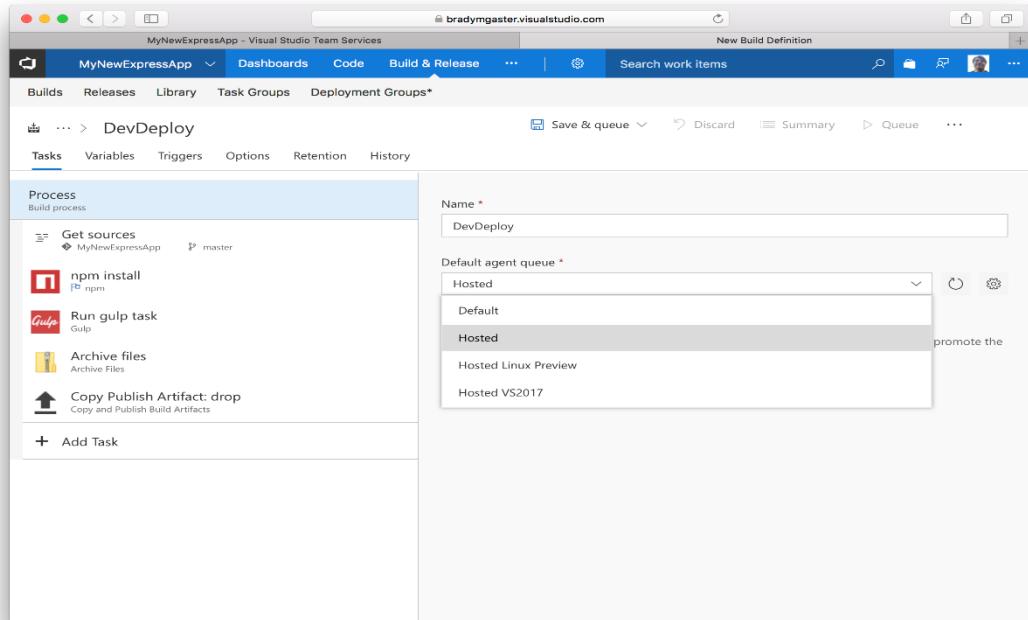
Clone ⌂ ⌂

Set up build

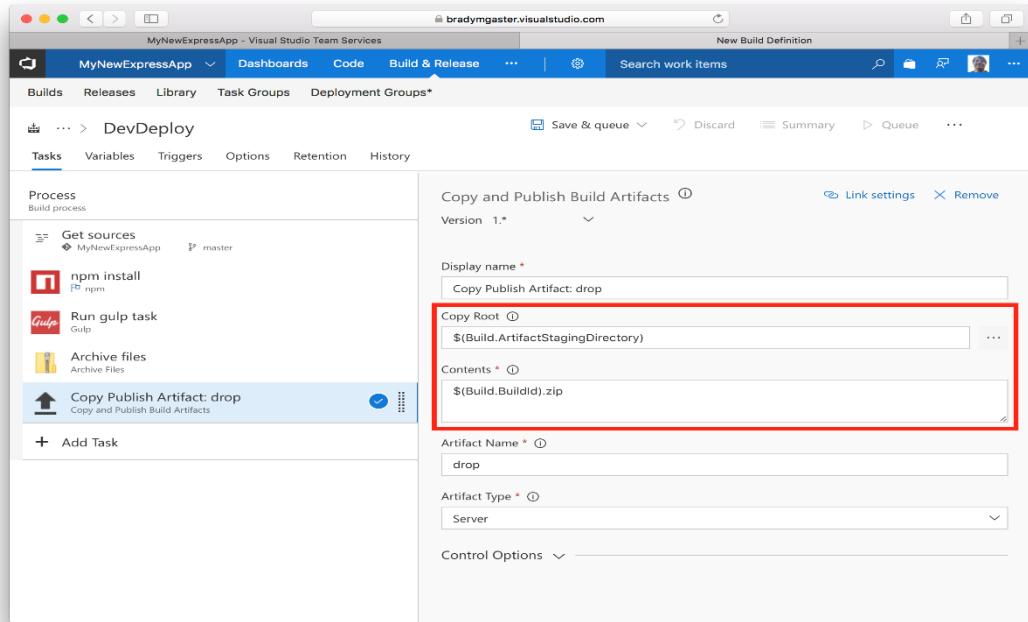
- Find the build template named **NodeJS with Gulp** and click the **Apply** button.



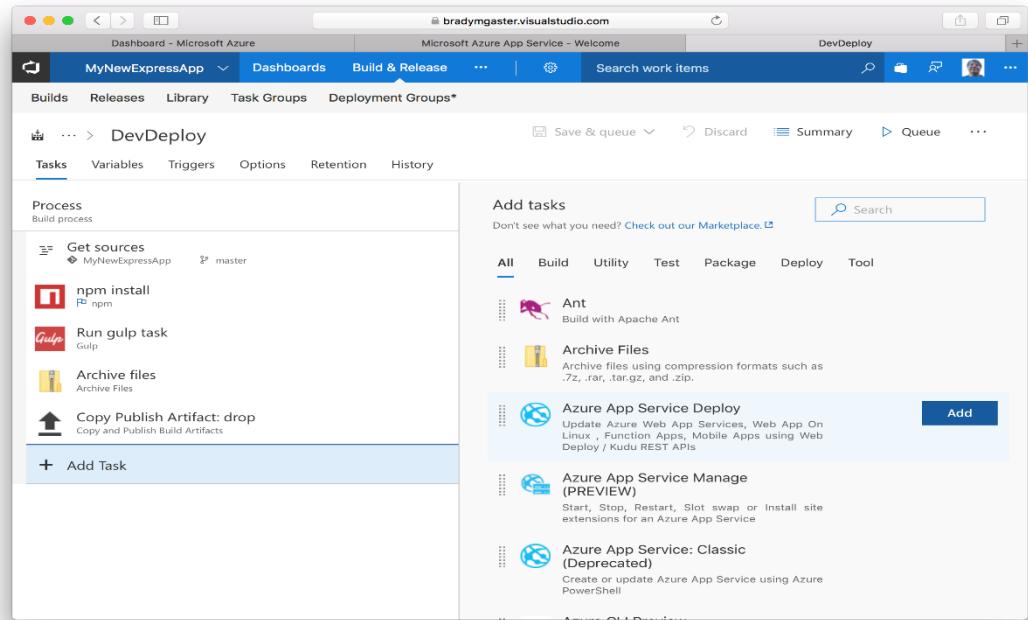
3. Give the build definition the name **DevDeploy** and select **Hosted** from the **Default agent queue** menu.



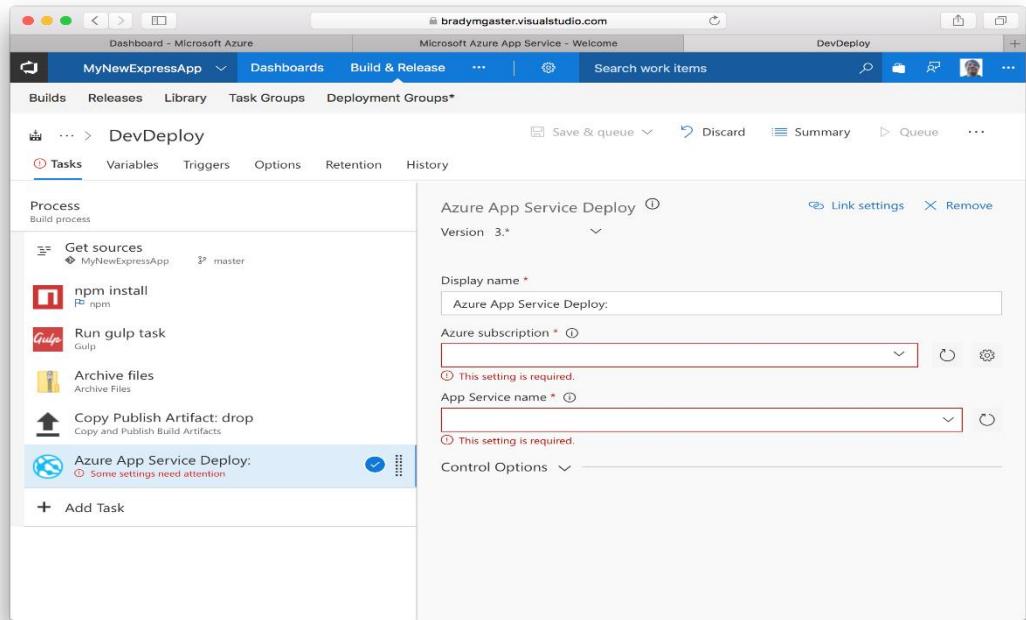
4. Click the **Copy Publish Artifact: drop** build task to see the details of this task.



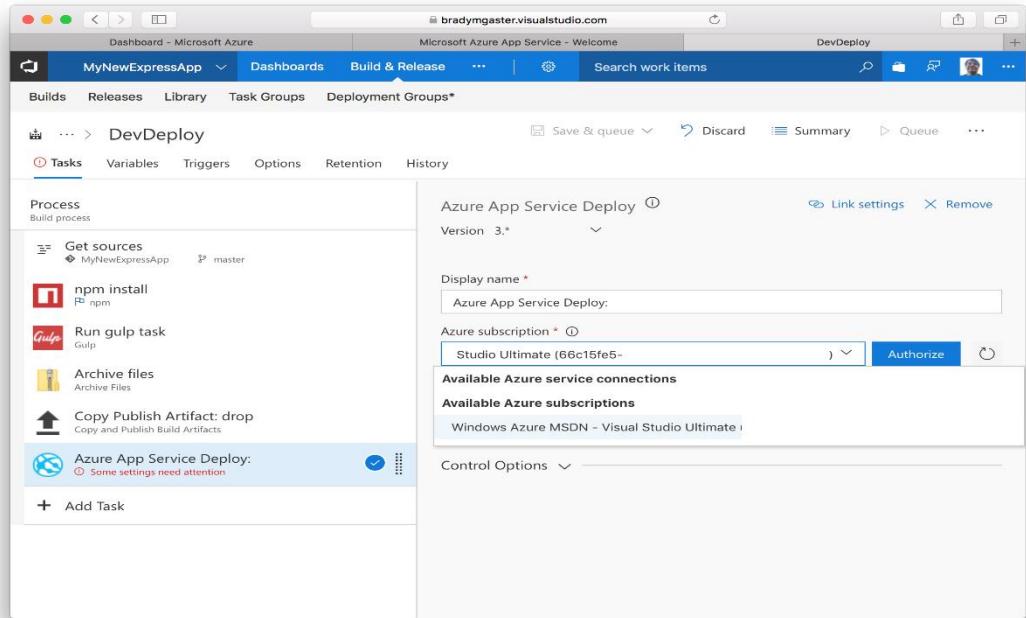
- Click the **Add Task** button under the list of build tasks. Select the **Azure App Service Deploy** task and click the **Add** button next to it.



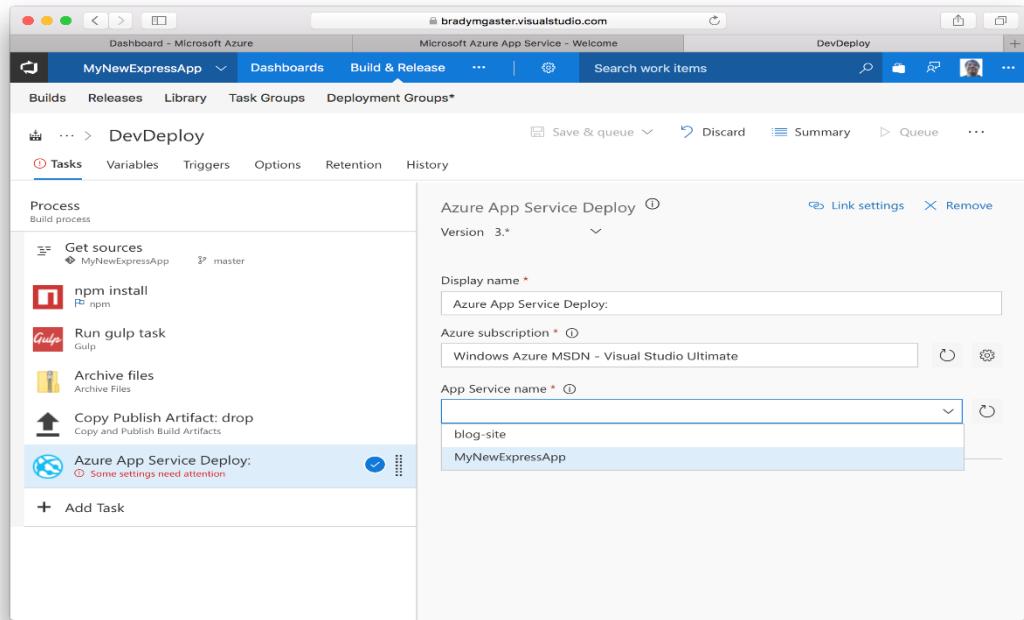
- Note that the **Azure subscription** and **App Service name** menus are both required.



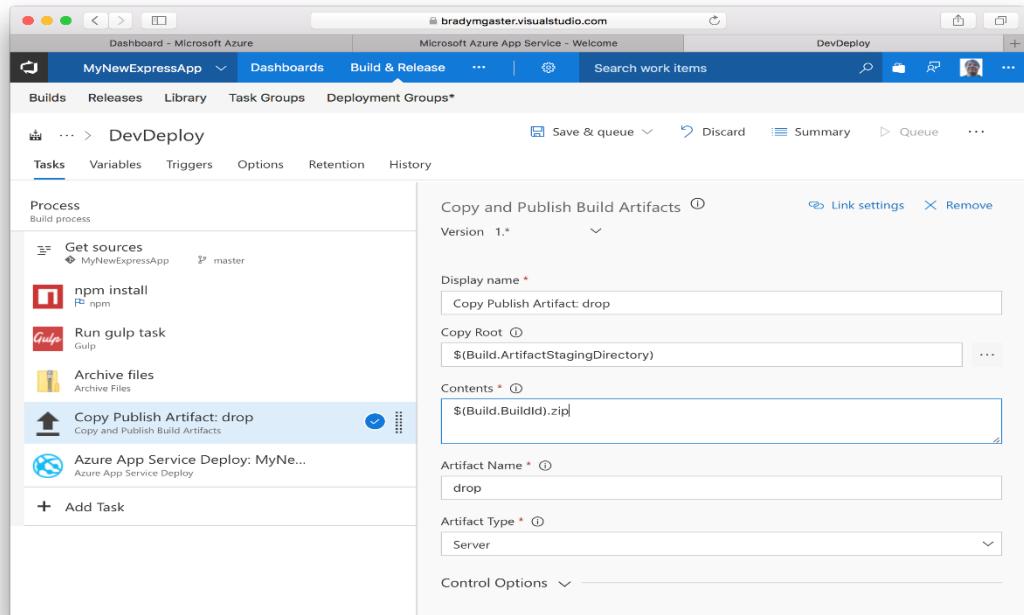
7. Select the Azure subscription in which your destination App Service is housed.



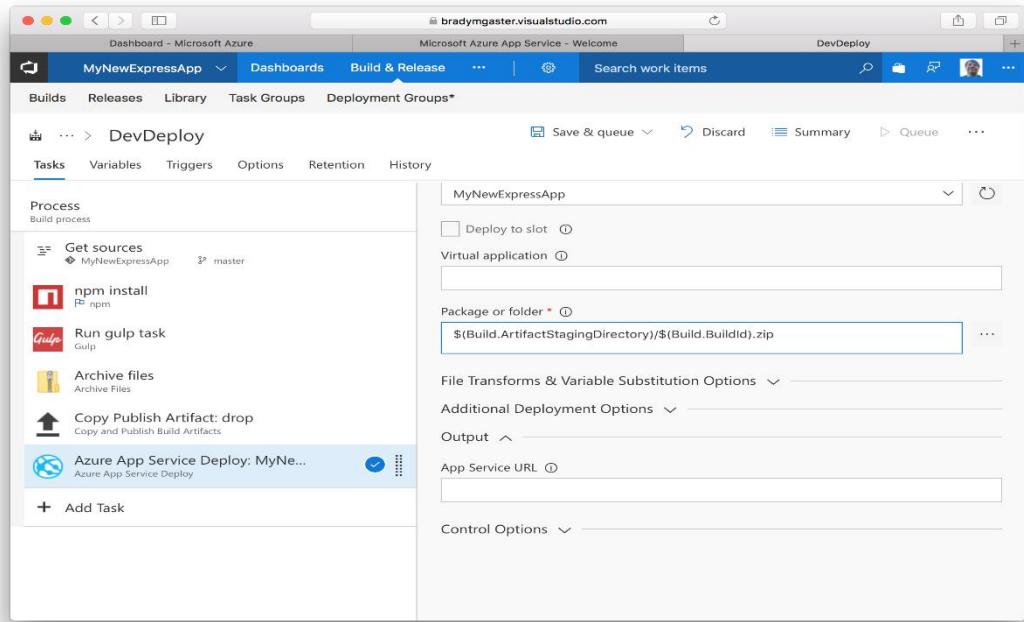
8. Select the App Service you created earlier within VS Code to use as the deployment target for your Node.js source code.



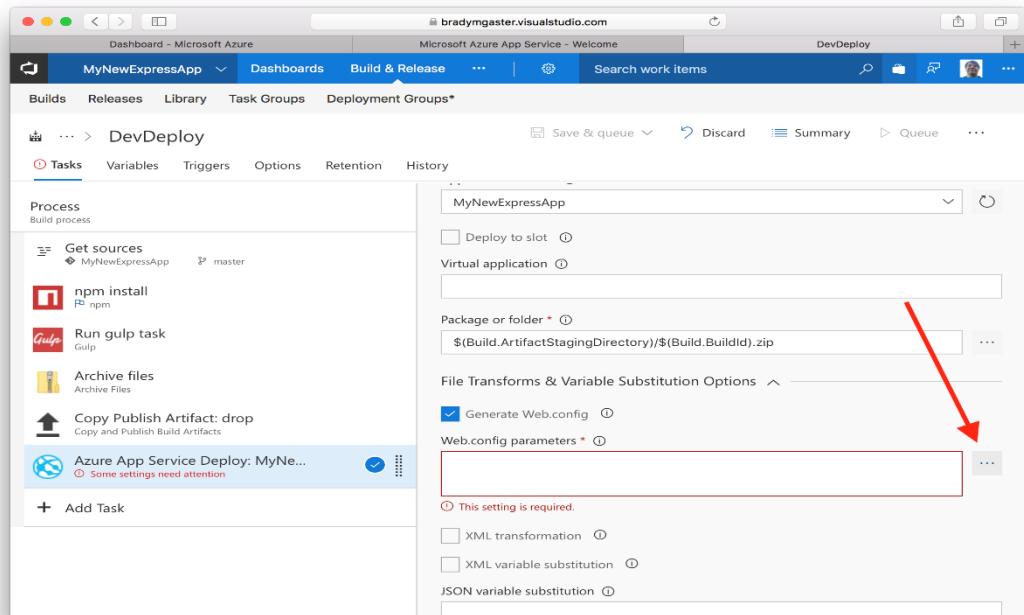
9. Click the previous build task (the "Copy Publish Artifact: drop" task) and take note of the **Copy root** and **Contents** text boxes.



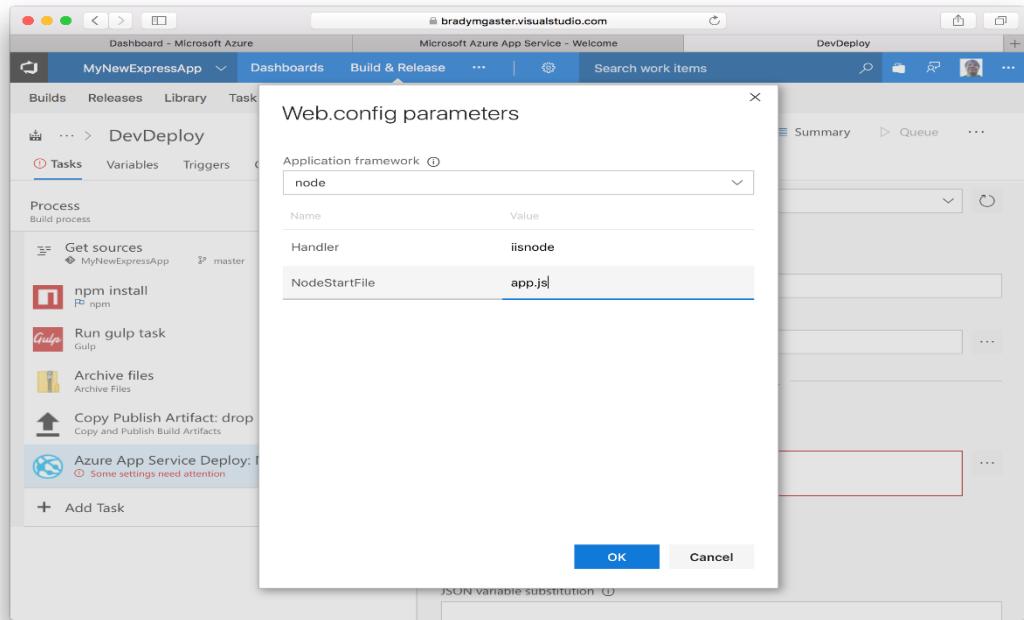
10. Once you've copied these two value and pasted them both into the Deploy build task, the value of the task's **Package or folder** textbox should represent the full path the ZIP file being built by your VSTS build.



11. Check the checkbox labelled **Generate Web.config** to enable that section of the task window. Then click the ellipse button next to the text box to open up the **Web.config parameters** dialog.



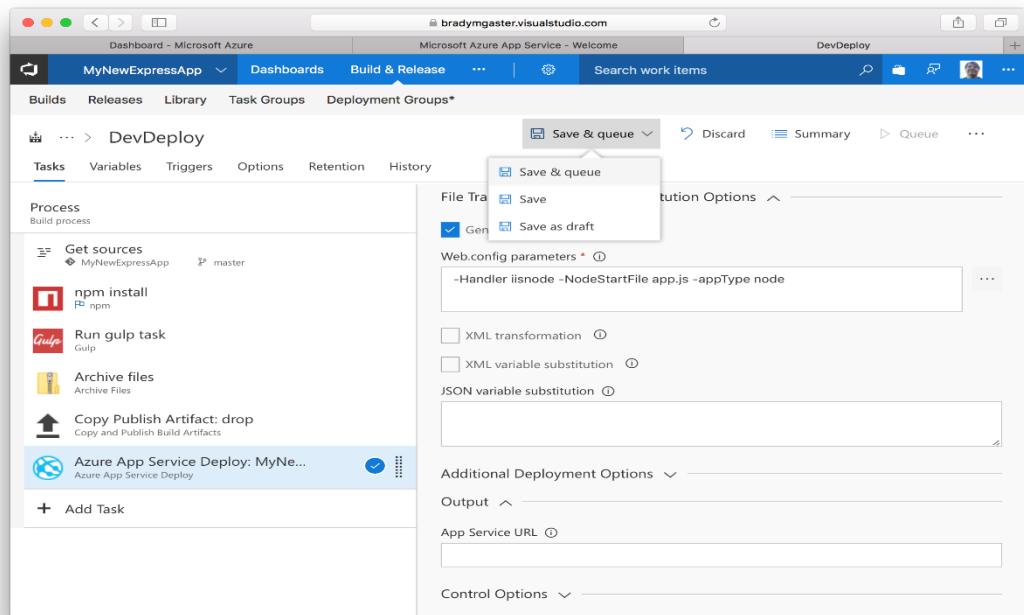
12. Select **node** from the Web.config parameters screen's **Application framework** menu. Then, change the **NodeStartFile** value to be **app.js** since the Express app's default filename is **app.js**.



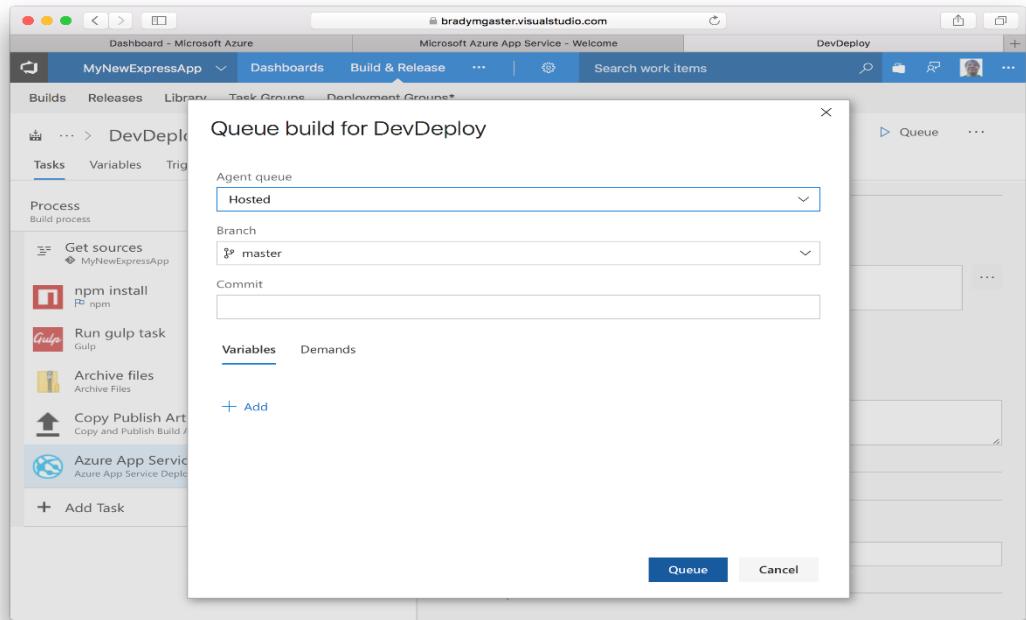
Run the Build

Now the build definition is complete running it will result in the code being deployed to the Azure App Service.

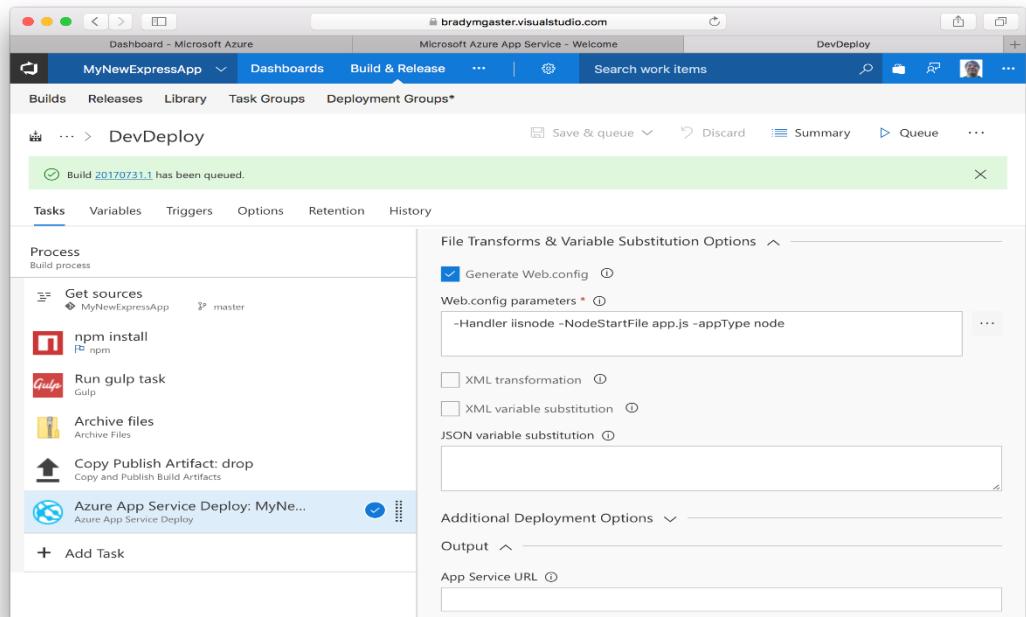
1. To kick off the manual build, click the **Save and Queue** spin button in the build definition page. Then, select the **Save and queue** option to save the definition and trigger the build.



2. Click the **Queue** button on the **Queue build for deploy** dialog.



- Click the queued build link to drill into the running build.



- The build log is visible in the browser.

The screenshot shows the Microsoft Azure DevDeploy interface for a build named "Build 20170731.1". The "Build" task is selected. The "Build Started" section indicates the build is running for 8 seconds on a Hosted Agent. The "Console" tab displays the command-line output of the build process. The logs show the execution of various tasks: Initialize Agent, Initialize Job, Get Sources, npm install, Run gulp task, Archive files, Copy Publish Artifact: drop, Azure App Service Deploy: MyNewExpressApp, Post Job Cleanup, and finally the "Starting: npm install" command. The output also includes a note about experimental changes and committing them.

```

remote: VSTSv          STSVSTSVTsvS
remote: VSTSvSTSVST
remote: VSTSvSTvS
remote: VSTSv (TM)
remote: Microsoft (R) Visual Studio (R) Team Services
From https://bradymgaster.visualstudio.com/_git/MyNewExpressApp
 * [new branch]      master    -> origin/master
git checkout --progress --force 11890ccb8f9835f6f1a3d8ebc2f2eab951820486c
Note: checking out 11890ccb8f9835f6f1a3d8ebc2f2eab951820486c.
You are in 'detached HEAD' mode.
You can experiment with features without impacting any branches you make in this
state without impacting any branches you make. If you want to create a new branch to retain commits you create, do
so now and later switch to that branch with the checkout command again. Example:
  git checkout -b <new-branch-name>
HEAD is now at 11890ccb... initial creation
* * * * *
Finishing: Get Sources
* * * * *
* * * * *
Starting: npm install
* * * * *

Task          : npm
Description   : Run an npm command
Version      : 0.2.20
Author       : Microsoft Corporation
Help         : [More information](https://go.microsoft.com/fwlink/?LinkId=613746)

C:\Program Files\nodejs\npm.cmd config list

```

5. Wait until the build has completed.

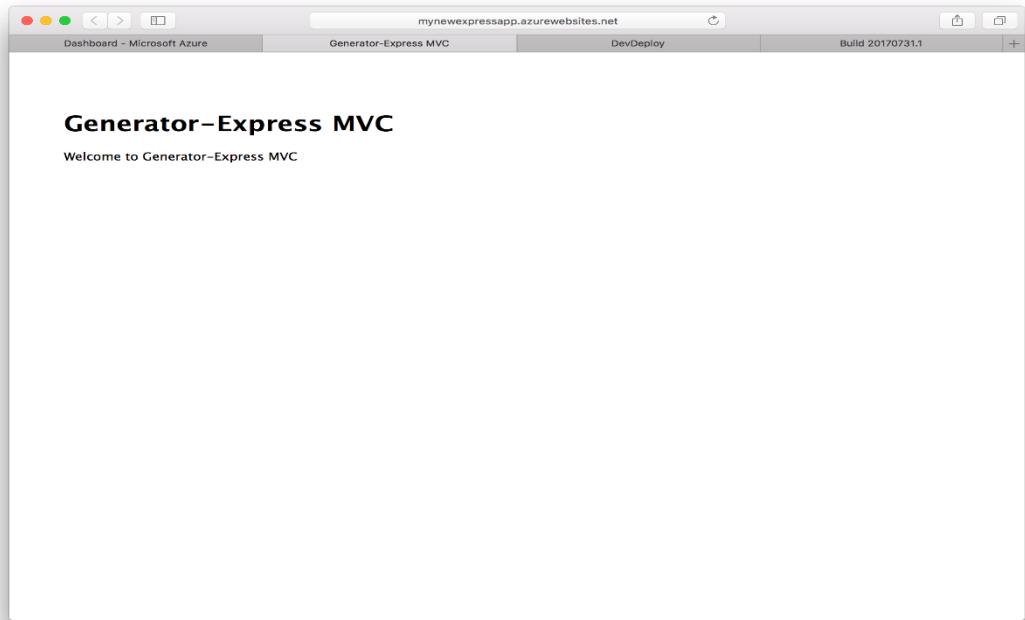
The screenshot shows the Microsoft Azure DevDeploy interface for the same build. The "Build" task is selected. The "Build succeeded" section indicates the build ran for 3 minutes on a Hosted Agent and completed 2 seconds ago. The "Logs" tab displays the detailed log output. The logs show the build process from start to finish, including the successful deployment of the web package to the Azure App Service. The output includes informational messages about file additions, deployment details, and the final success message.

```

Info: Adding file (MyNewExpressApp\node_modules\yargs\index.js)
Info: Adding file (MyNewExpressApp\node_modules\yargs\lib\array-push.js).
Info: Adding file (MyNewExpressApp\node_modules\yargs\lib\parser.js)
Info: Adding file (MyNewExpressApp\node_modules\yargs\lib\usage.js)
Info: Adding file (MyNewExpressApp\node_modules\yargs\lib\validation.js)
Info: Adding file (MyNewExpressApp\node_modules\yargs\lib\wrap.js)
Info: Adding file (MyNewExpressApp\node_modules\yargs\package.json)
Info: Adding file (MyNewExpressApp\package-lock.json)
Info: Adding file (MyNewExpressApp\package.json)
Info: Adding file (MyNewExpressApp\style.css)
Info: Adding file (MyNewExpressApp\web.config)
Total changes: 10067 (10067 added, 0 deleted, 0 updated, 0 parameters changed, 34685256 bytes copied)
Successfully deployed web package to App Service with configuration details
Successfully updated deployment History at https://mynewexpressapp.scm.azurewebsites.net/deployments/0752308338
* * * * *
Finishing: Azure App Service Deploy: MyNewExpressApp
* * * * *
* * * * *
Starting: Post Job Cleanup
* * * * *
Cleaning any cached credential from repository: MyNewExpressApp (Git)
git remote set-url origin https://bradymgaster.visualstudio.com/_git/MyNewExpressApp
git remote set-url --push origin https://bradymgaster.visualstudio.com/_git/MyNewExpressApp
* * * * *
Finishing: Post Job Cleanup
* * * * *
Finishing: Build
* * * * *

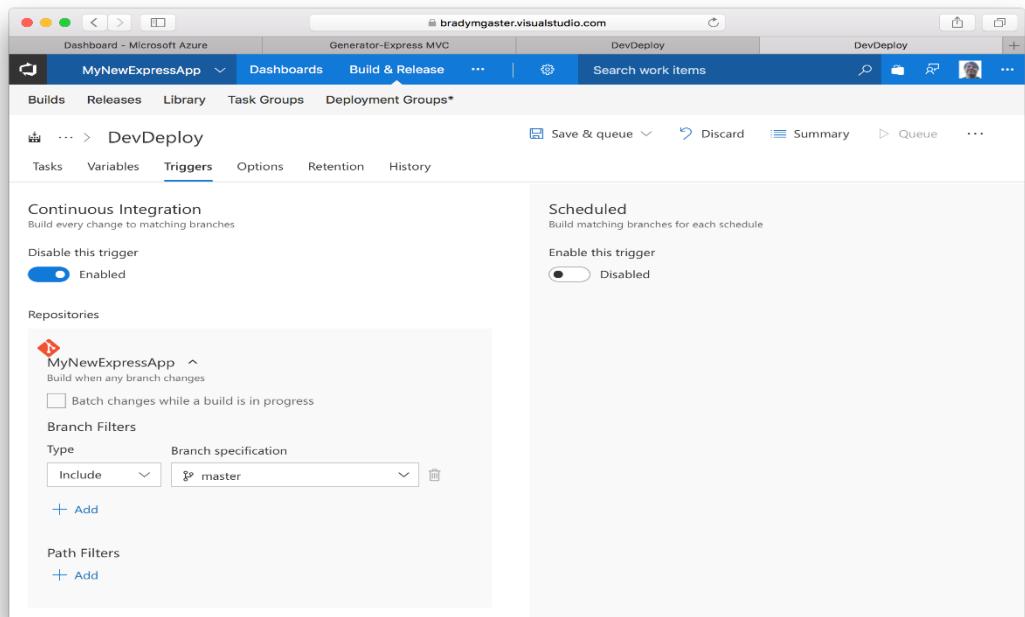
```

6. Open up a browser (or refresh the already-open browser tab) on the App Service URL.



Enabling triggered builds

1. Click the **Triggers** tab in Team Services.



2. Edit the code in `controllers/home.js` so that the `articles` variable in the `get` method is instantiated using the code below.

```
var articles = [
  new Article({
    'title': 'First Article',
    'text' : 'This is the text of the first article'
  }),
```

```

        new Article({
          'title': 'Second Article',
          'text' : 'This is the text of the second article'
        })
      ];

```

```

var express = require('express'),
  router = express.Router(),
  Article = require('../models/article');

module.exports = function (app) {
  app.use('/', router);
};

router.get('/', function (req, res, next) {
  //var articles = [new Article(), new Article()];
  var articles = [
    new Article({
      'title': 'First Article',
      'text' : 'This is the text of the first article'
    }),
    new Article({
      'title': 'Second Article',
      'text' : 'This is the text of the second article'
    })
  ];
  res.render('index', {
    title: 'Generator-Express MVC'
  });
}

```

3. Edit the code in `view\index.handlebars` to add a `dl` list containing a presentation of the articles in the view model.

```

<dl>
  {{#each articles}}
    <dt>{{title}}</dt>
    <dd>{{text}}</dd>
  {{/each}}
</dl>

```

The screenshot shows the Visual Studio Code interface with the title bar "index.handlebars — NewExpressApp". The left sidebar is titled "EXPLORER" and lists files like "home.js", "index.handlebars", "error.handlebars", and "index.handlebars" again under "NEWEXPRESSAPP". The main editor area contains the following Handlebars template code:

```
<h1>{{title}}</h1>
<p>Welcome to {{title}}</p>

<dl>
  {{#each articles}}
    <dt>{{title}}</dt>
    <dd>{{text}}</dd>
  {{/each}}
</dl>
```

The status bar at the bottom indicates "Ln 11, Col 8" and "Handlebars".

4. Commit the code to the repository.

The screenshot shows the Visual Studio Code interface with the title bar "index.handlebars — NewExpressApp". The left sidebar is titled "SOURCE CONT..." and shows a commit message "updated ux to include articles" in the "CHANGES" section. The main editor area contains the same Handlebars template code as the previous screenshot. The status bar at the bottom indicates "Ln 11, Col 8" and "Handlebars".

5. Go back to the build definition in Team Services and take note that a new build has been queued.

The screenshot shows the Microsoft Azure Dev & Release dashboard for the project "MyNewExpressApp". The "Build Definitions" tab is active, displaying a list of builds. One build, "#20170731.2", is currently in progress on the "master" branch. The success rate is shown as 100.00%. Other sections visible include "Details", "Branches", "Analytics", and "History".

6. Refresh the site in the browser.

The screenshot shows the Microsoft Azure website for the application "mynewexpressapp.azurewebsites.net". The page title is "Generator-Express MVC". The content area displays two articles: "First Article" (with the text "This is the text of the first article") and "Second Article" (with the text "This is the text of the second article"). The URL in the address bar is "mynewexpressapp.azurewebsites.net".