

DevSecOps
User Guide

Michelle Monfort

Jeroen Soeurt

Robert Wilson

University of Maryland Global Campus
SWEN 670 Software Engineering Project

Professor Dr. Mir Assadullah

Summer 2021

Table of Contents

Table of Figures:	3
Introduction	5
Git	6
Introduction	6
Installing Git	6
Windows	6
Linux	8
MacOS	8
Basic Git usage	8
Cloning	10
Branches	11
Staging	11
Commit	12
Push	12
Pull Request	12
Reviewing a pull request	13
Merging	16
Resolving merge conflict	17
Pipelines – GitHub Actions	21
Retrieving the APK	22
Advanced Development Factory	25
Introduction	25
Installing Docker	25
Windows	25
Linux	26
MacOS	26
Clone the ADF repository	26
From a terminal, clone the ADF repository using	26
• Visual Studio Code:	27
Build the Docker image	28
Running the ADF locally	28

Connecting to the running container.....	28
Flutter development in the container	34
What else is there?.....	35
How to install extra software.....	36
VIM	36
Emacs	37
NetBeans.....	38
Project idea website	39
Introduction	39
Accessing the website.....	39
Submitting an idea	39

Table of Figures:

Figure 1: GIT Setup Screen 1 - License.....	6
Figure 2: GIT Setup Screen 2 - Common Options.....	7
Figure 3: : GIT Setup Screen 3 - Use Unix style line endings	8
Figure 4: GitHub Clone Repository Url.....	10
Figure 5: Command Line - Clone Command.....	11
Figure 6: GitHub Pull Request Creation.....	13
Figure 7: GitHub Base Branch Selection	13
Figure 8: GitHub Check Status of Pipeline.....	14
Figure 9: GitHub Review Files	15
Figure 10: GitHub Review Files - Insert Comment.....	16
Figure 11: GitHub Review Files - Request Changes	16
Figure 12: GitHub Pull Request - Merge Conflict	17
Figure 13: GitHub Pull Request - Resolve Conflicts Button	18
Figure 14: GitHub Resolve Conflict - Identify Number of Conflicts	18
Figure 15: GitHub Resolve Conflict - Identify Merge Markers.....	19
Figure 16: GitHub Resolve Conflict - Edit document.....	20
Figure 17: GitHub Resolve Conflict - Commit Merge	20
Figure 18: GitHub Resolve Conflict - Merge Resolved Pull Request.....	20
Figure 19: GitHub Actions - Actions YAML Configuration File.....	21
Figure 20: GitHub Actions - Workflows	22
Figure 21: GitHub Actions - Artifacts	23
Figure 22: GitHub Actions - Downloaded Artifact.....	23
Figure 23: GitHub Actions - Running APK Artifact on Local Device	24
Figure 24: Docker - Images	25
Figure 25: Docker - GitHub Repository Sync.....	26
Figure 26: Docker - Edit startup.sh	27

Figure 27: Docker - Notepad++ EOL Conversion.....	27
Figure 28: Docker - Microsoft Remote Desktop Connection Screen.....	29
Figure 29: Docker - Remmina Remote Desktop Preference Screen Part 1	30
Figure 30: Docker - Remmina Remote Desktop Preference Screen Part 2	31
Figure 31: ADF - Login Screen	32
Figure 32: ADF - Welcome Screen via Remmina on Linux.....	33
Figure 33: ADF - Welcome Screen via Remote Desktop Connection on Windows	33
Figure 34: ADF - VS Code with Attached Emulator	35
Figure 35: ADF - System Info	36
Figure 36: ADF - Installed VIM	37
Figure 37: ADF - Installed Emacs	37
Figure 38: ADF - Installed NetBeans IDE.....	38
Figure 39: CaPPMS - Submit Project Idea Page.....	39
Figure 40: CaPPMS - Extend Page for Other Than Self Sponsor Information.....	40
Figure 41: CaPPMS - Project List Page.....	40
Figure 42: CaPPMS - Project List Idea View	41
Figure 43: CaPPMS Project Idea PDF Export	41
Figure 44: CaPPMS - Project Idea Comments and Menu Section.....	42

Introduction

The DevSecOps team has completed several projects during the summer semester. The deliverables of these projects can help current and future developers in the UMGC software engineering capstone course in their work. They include and containerized development environment (Advanced Development Factory), and a website to manage project ideas and suggestions.

The other part of the DevSecOps team's responsibilities included setting up and maintaining the git repositories for the development teams on GitHub. This user guide will cover all three areas. It starts with the use of Git, then covers the ADF, and finally describes the use of the website.

Git

Introduction

Git is a version control system originally developed by Linus Torvalds in the mid-2000s.

It is free and open source, and is used by online repository providers like GitHub, Azure

DevOps, and BitBucket. For this course we will be using git in combination with GitHub.

GitHub provides free and paid repositories. As a UMGC student you can get the GitHub Student Developer Pack, which gives you a pro account and several software products and services for free. More information about this can be found at <https://education.github.com/pack>.

Installing Git

Windows

Navigate to <https://git-scm.com/> and download the latest installer.

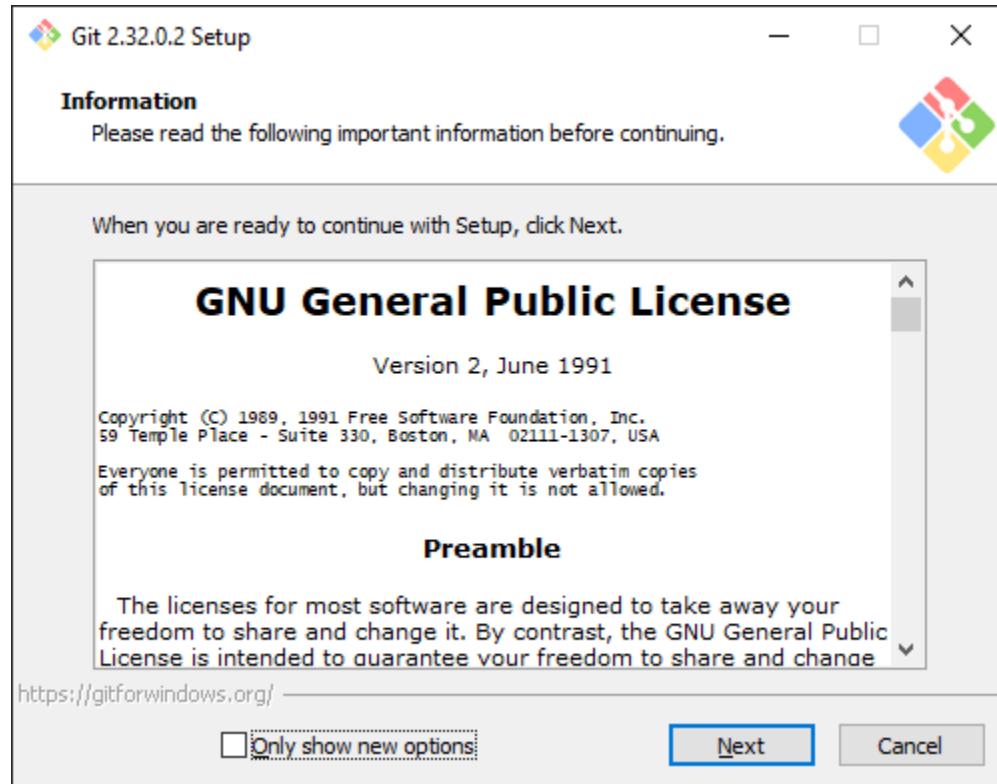


Figure 1: GIT Setup Screen 1 - License

Accept the GPL by clicking next. The default options on the next screen can be left as-is.

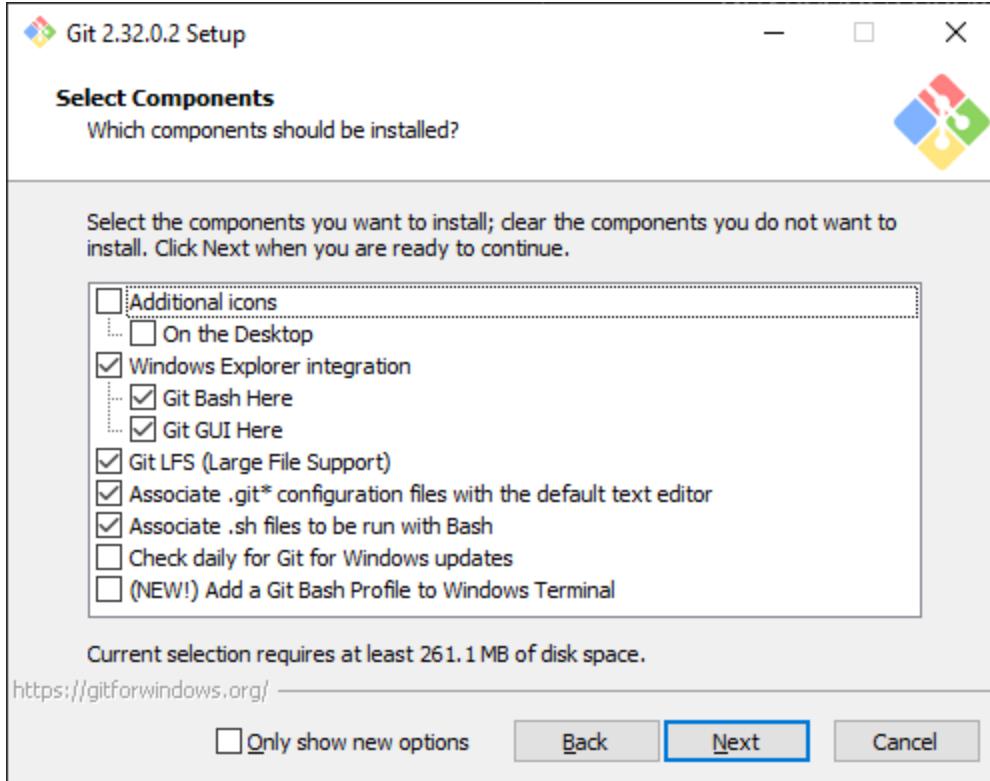


Figure 2: GIT Setup Screen 2 - Common Options

Click next several times until you get to this screen:

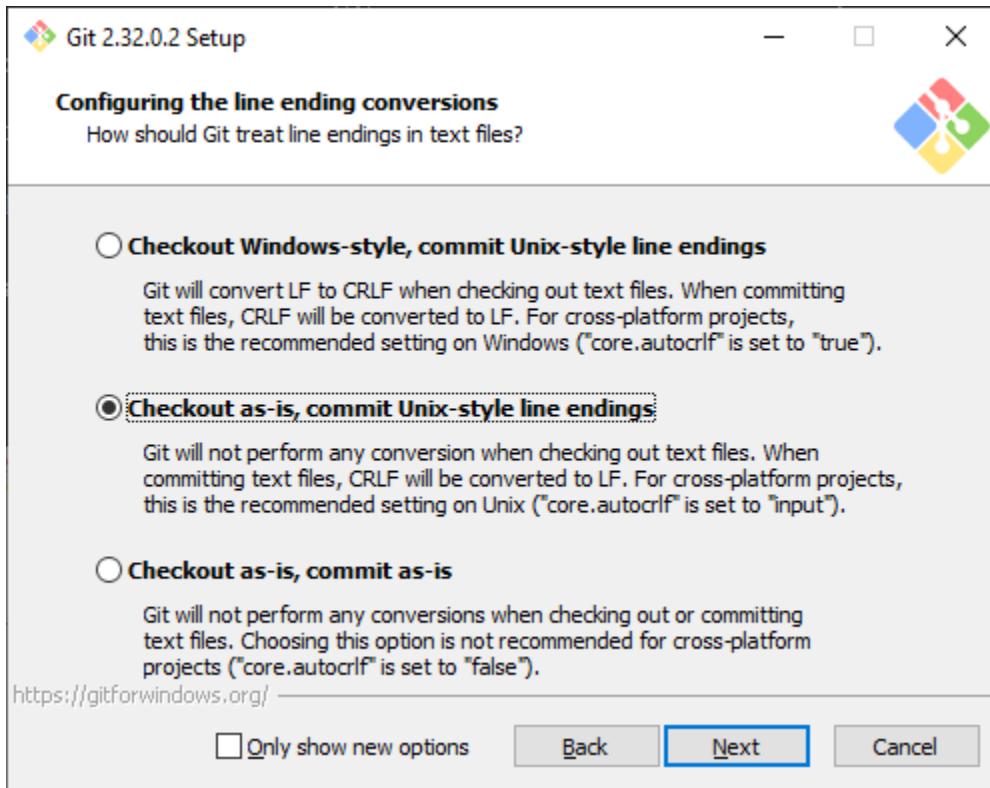


Figure 3: : GIT Setup Screen 3 - Use Unix style line endings

Here it is important to select Checkout as-is. Otherwise, you run into problems when cloning the ADF in the next chapter. All other default options on the next screens are ok as-is. Click install to complete the installation.

Linux

Git is included in most distributions. For Debian based (Ubuntu, Mint) use apt-get:

- sudo apt-get install git

For RPM based distros (RedHat, Centos, Fedora), use yum:

- sudo yum install git

MacOS

Git is included with Xcode, so you could install XCode from the app store. If you don't want to install Xcode, you can also install Git using Homebrew:

- brew install git

Basic Git usage

The DevSecOps team will provide you with access to the UMGC capstone organization in GitHub (<https://github.com/umgc/>). Here you will find repositories from previous terms.

DevSecOps has also created a repository this term's teams. Navigate to your team's repository to get started.

Cloning

On your repository, click on the green Code button, and copy the URL listed.

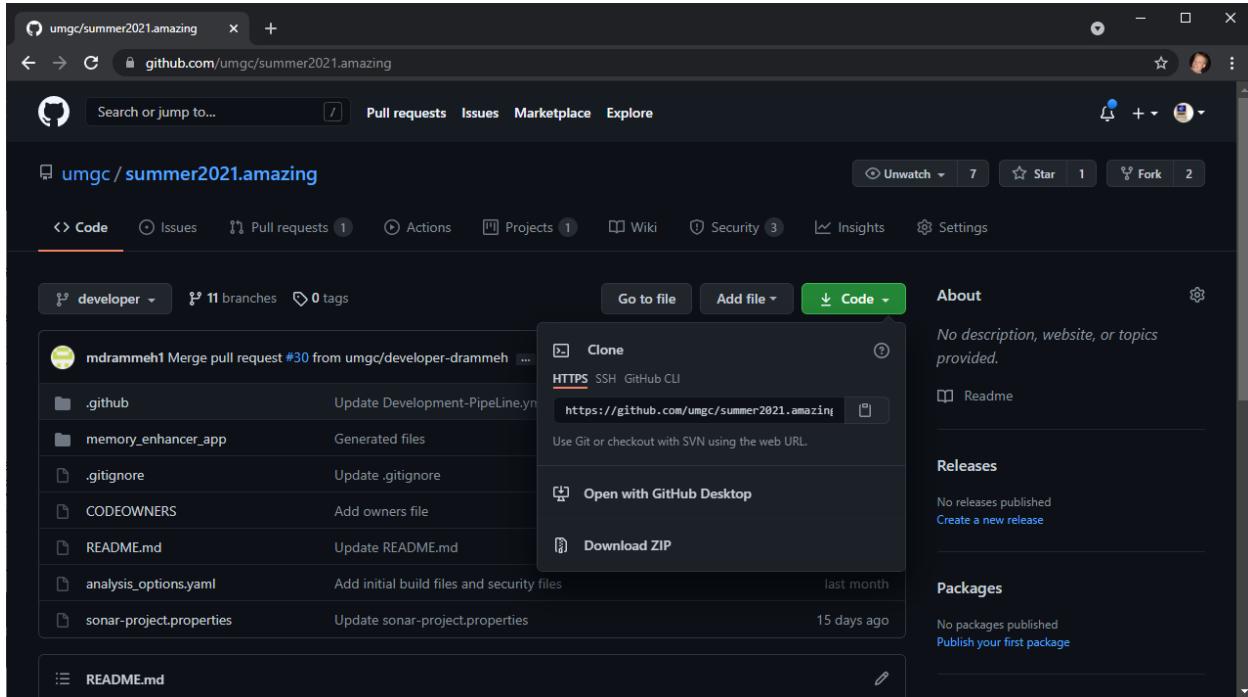


Figure 4: GitHub Clone Repository Url

Now open a new terminal window, and clone the repository using the following command:

- `git clone <url>`

Example:

- `git clone https://github.com/umgc/summer2021.amazing.git`

This will create a new folder with the name of the repository, and download the repository.

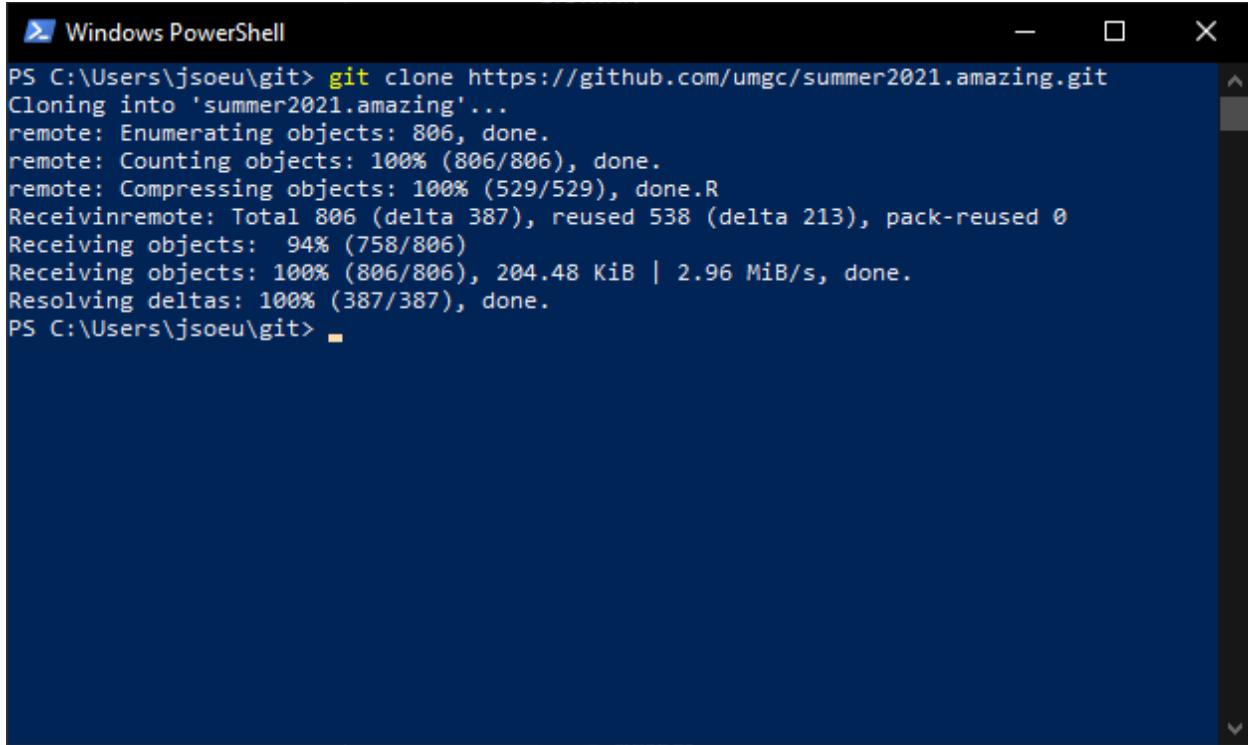
A screenshot of a Windows PowerShell window titled "Windows PowerShell". The window shows the command "git clone https://github.com/umgc/summer2021.amazing.git" being run. The output of the command is displayed, showing the progress of cloning a repository from GitHub. The progress includes: Cloning into 'summer2021.amazing'...; remote: Enumerating objects: 806, done.; remote: Counting objects: 100% (806/806), done.; remote: Compressing objects: 100% (529/529), done.R; Receiving objects: Total 806 (delta 387), reused 538 (delta 213), pack-reused 0; Receiving objects: 94% (758/806); Receiving objects: 100% (806/806), 204.48 KiB | 2.96 MiB/s, done.; Resolving deltas: 100% (387/387), done. The command prompt PS C:\Users\jsoeu\git> is visible at the bottom.

Figure 5: Command Line - Clone Command

Branches

Git uses the concept of branches. You can create a new branch using:

- git checkout -b <branchname>

Example:

- git checkout -b jersoe/encryption

This creates a new local branch. You can make changes to the code. You can also switch back to other branches using:

- git checkout <branchname>

Example:

- git checkout developer

You generally want to create a new branch for your work. When you are satisfied with your changes, you have to perform a few steps:

Staging

First, you have to stage the file that you have changed. This can be a specific file, or all files that have changed.

All files:

- `git add .`

A specific file:

- `git add crypto.java`

Commit

Next, you have to commit your changes. To do so, use the following command:

- `git commit -m "<message>"`

Example:

- `git commit -m "Added encryption."`

Push

The next step in your workflow is to push your changes to the remote repository on GitHub. Now there are two options. If you created a new local branch, then this branch will not exist on the remote repository. In that case, use:

- `git push --set-upstream origin <branch>`

Example:

- `git push --set-upstream origin jersoe/crypto`

If you've previously pushed this branch to remote, then the following will suffice:

- `git push`

Pull Request

The final step is to request that your code be merged into the development branch. You can request this by creating a pull request. To do so, navigate to the GitHub repository, click on Pull Requests, and use the green “New pull request” button.

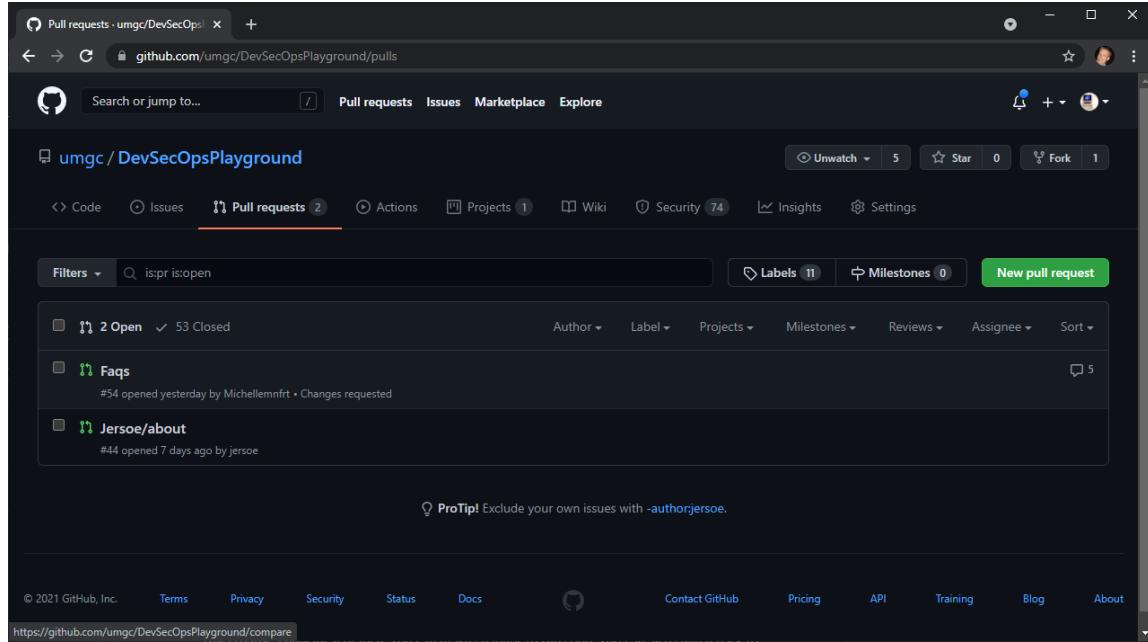


Figure 6: GitHub Pull Request Creation

At the top, choose the target branch (developer), and the source branch (your work):

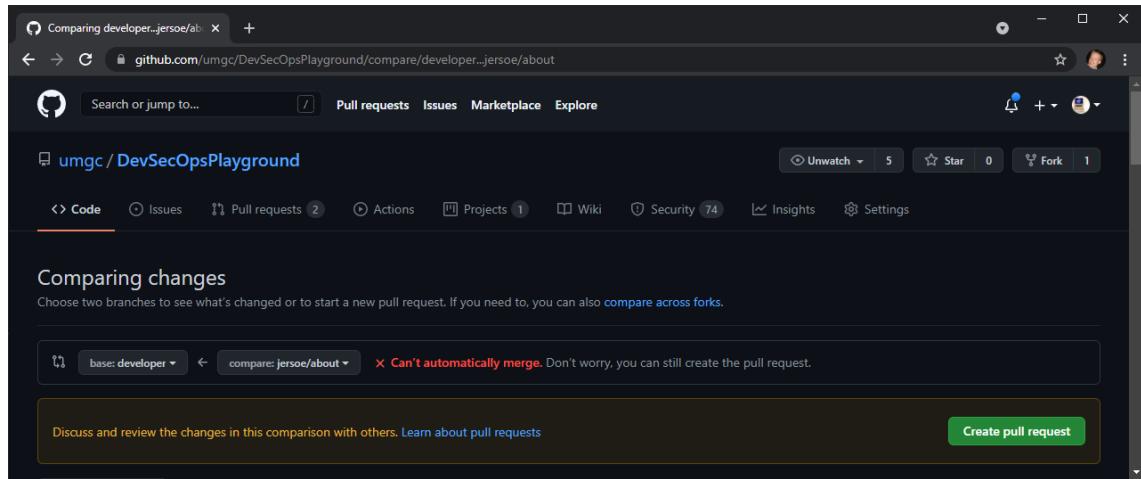


Figure 7: GitHub Base Branch Selection

Next click the green Create pull request button.

The pull request will now be reviewed by the project manager of your team, and by DevSecOps.

Reviewing a pull request

The repositories have been set up to require reviews. As a project manager or DevSecOps team member you can review open pull requests. To do so, navigate to the repository, click on Pull request, and choose the PR you would like to review.

There are two areas to pay particular attention to: Check, and Files changed.

- Checks show the result of the pipeline that ran. This lets you know if there are any formatting issues, or vulnerabilities for example.

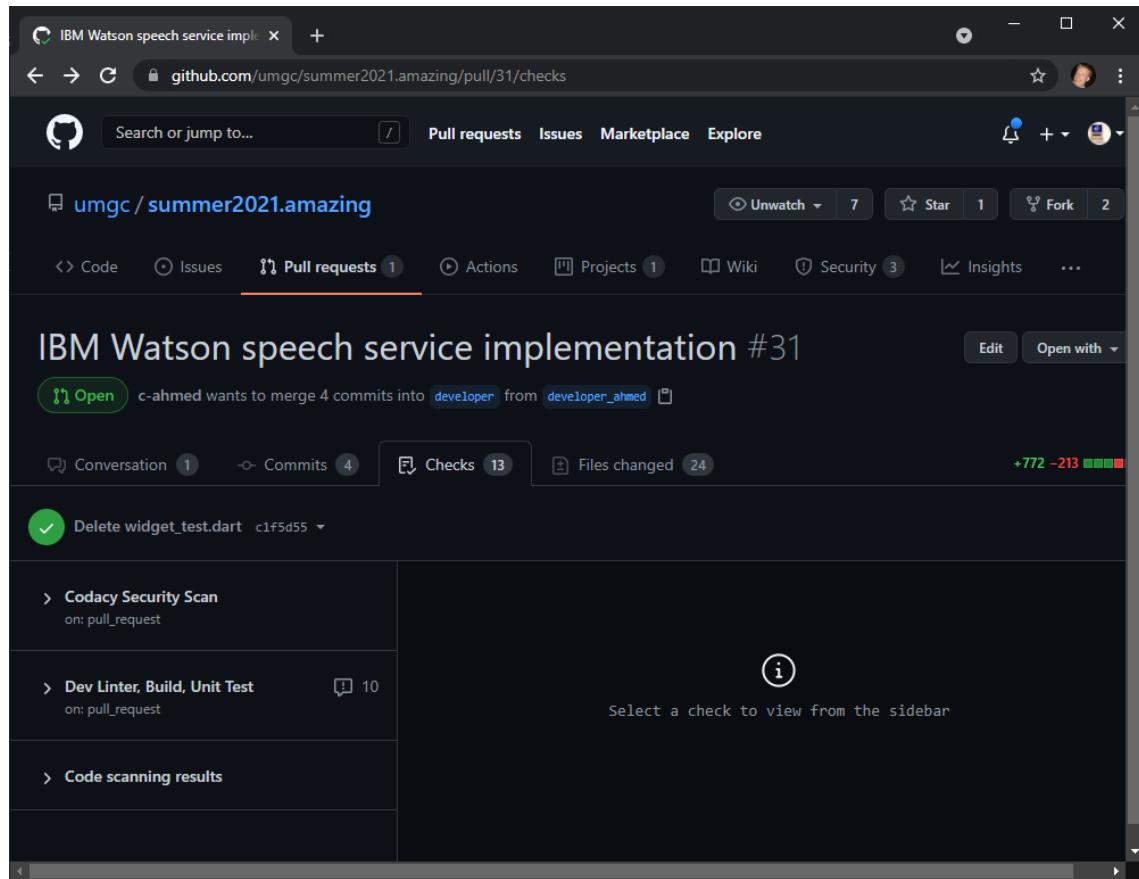


Figure 8: GitHub Check Status of Pipeline

- The Files changed tab show all old/new/replaced code. Here, pay attention that there are no issues such as:
 - Commented out code.
 - Left-over boilerplate code.
 - Naming-convention issues.
 - Bad practices.

- Secrets being part of a commit.

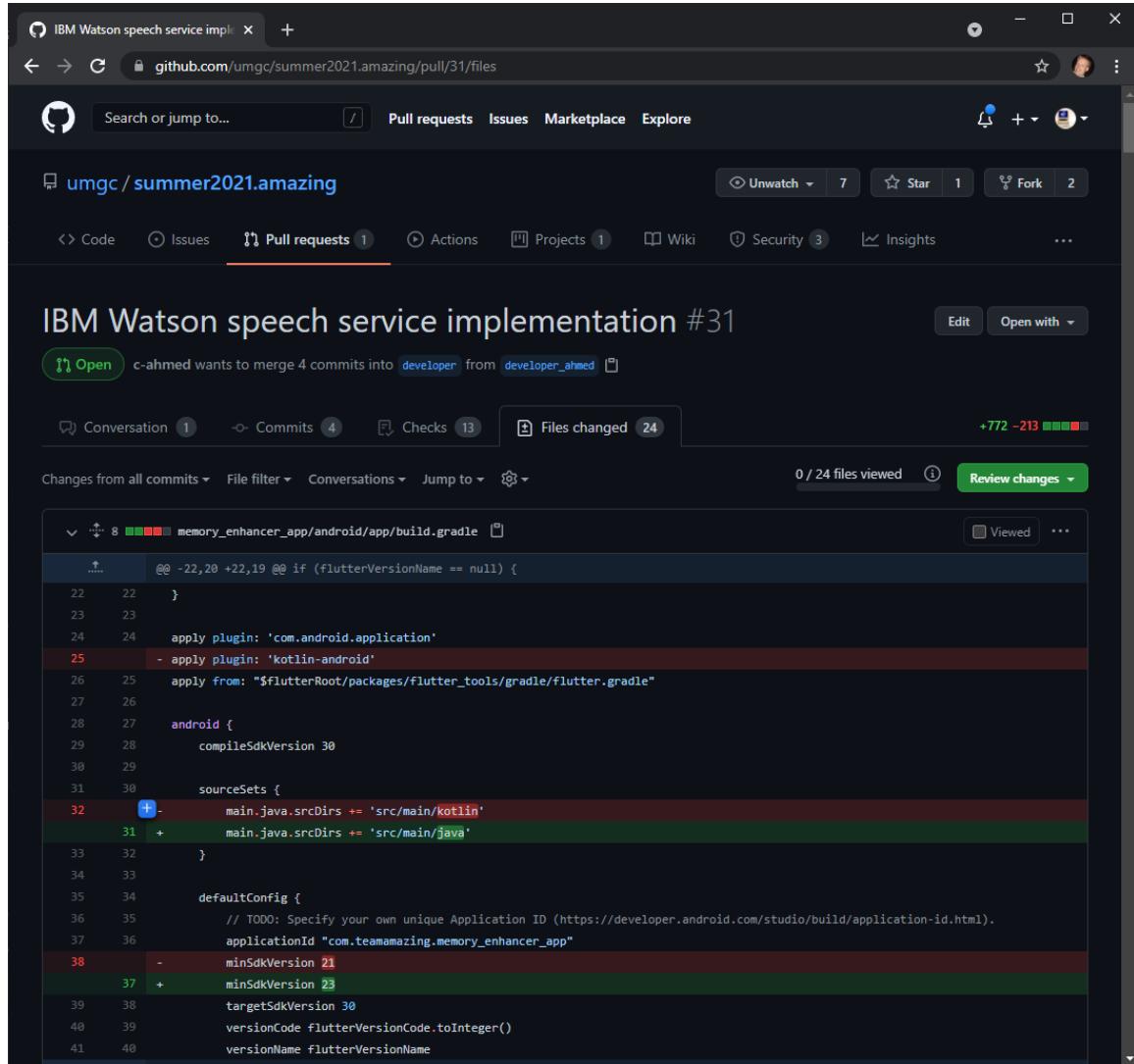


Figure 9: GitHub Review Files

As a reviewer, you can approve a PR, comment on a PR, and request changes from the submitter.

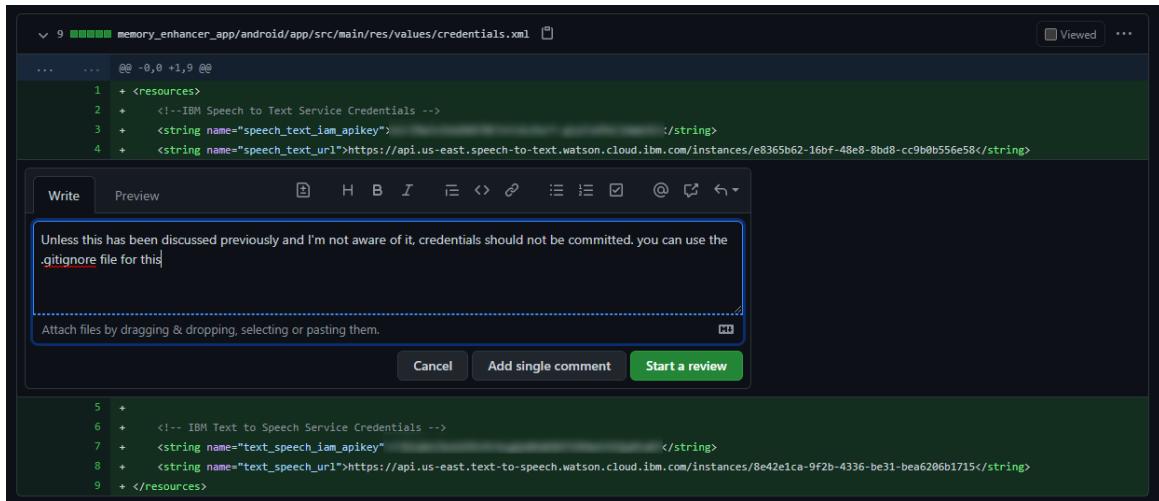


Figure 10: GitHub Review Files - Insert Comment

You can add comments to code by clicking the + button in front of a line of code.

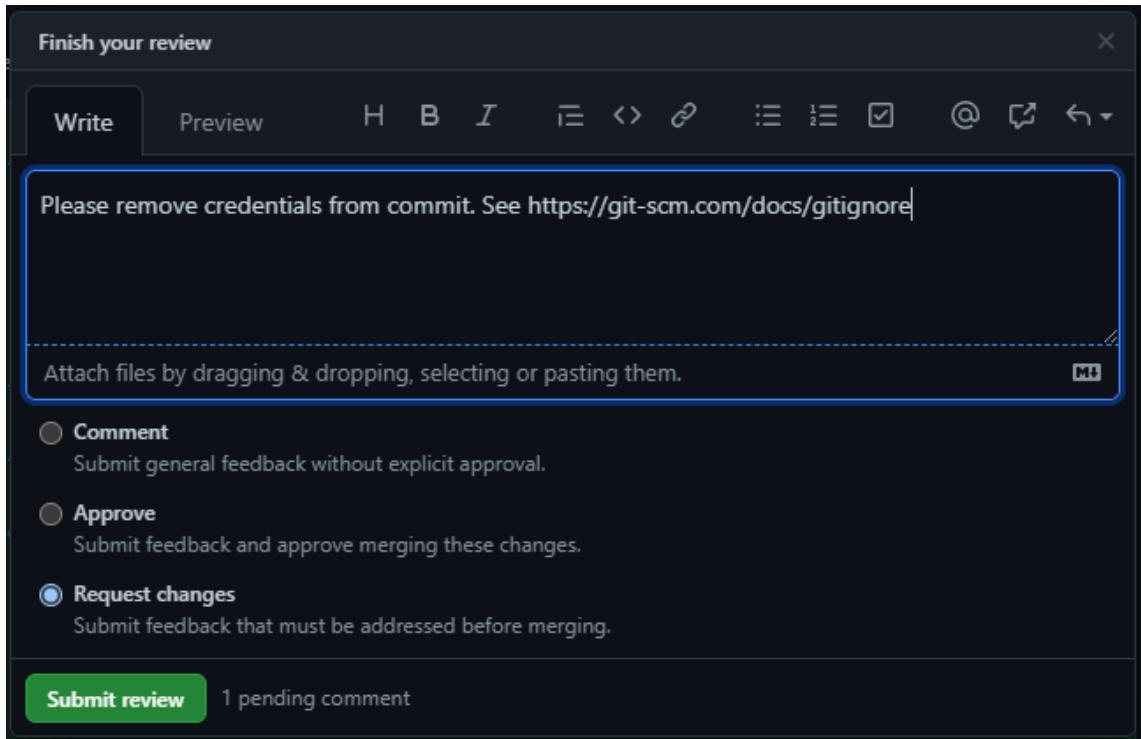


Figure 11: GitHub Review Files - Request Changes

Finally, approve the request if you do not see any issues, or request changes if needed.

Merging

If a pull request has been approved, you will be able to merge it. To do so, click the green Merge pull request button on the pull request page.

Resolving merge conflict

Working on a multiple person team in or around the same area of a project can lead to merge conflicts. This usually happens because two different people editing the same code at nearly the same time and one PR is merged before the other. It may also happen that you have multiple PRs open and they might touch the same code as well, even if in a minor way. Luckily GitHub makes it super easy to resolve these conflicts right in the browser.

Let's take the example where one person added a section to the CaPPMS ReadMe.md file while another person was working on their section of the ReadMe file.

During creation of a PR in GitHub, it is obvious that there is a merge conflict. After selecting the base branch, a strong red not appears letting you know that this cannot be automatically merged.

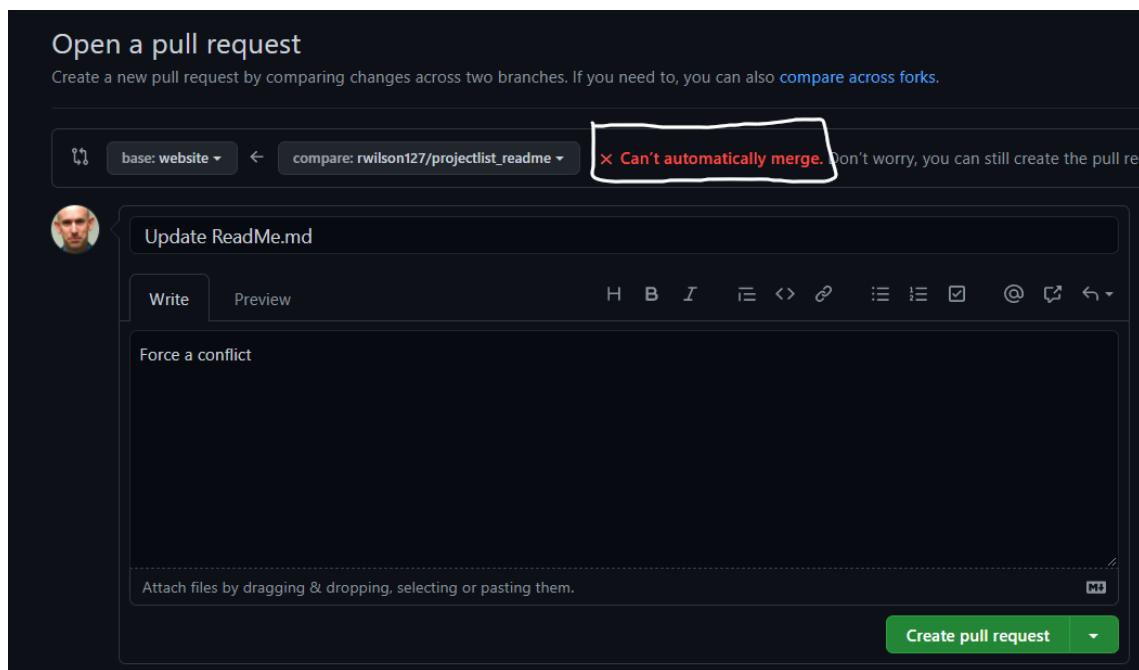


Figure 12: GitHub Pull Request - Merge Conflict

- Create the pull request
- Click the Resolve Conflicts button

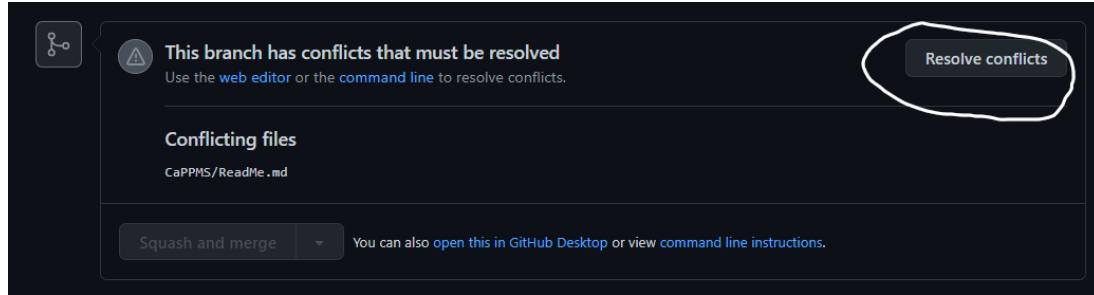


Figure 13: GitHub Pull Request - Resolve Conflicts Button

At the top of the page, there are some hints to help you resolve the problems:

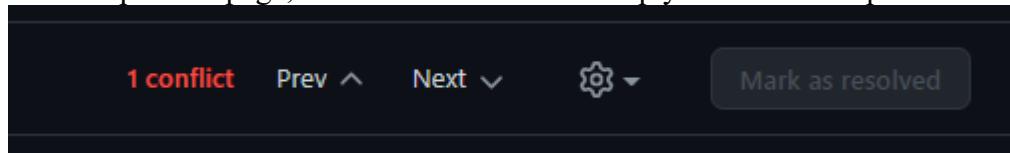


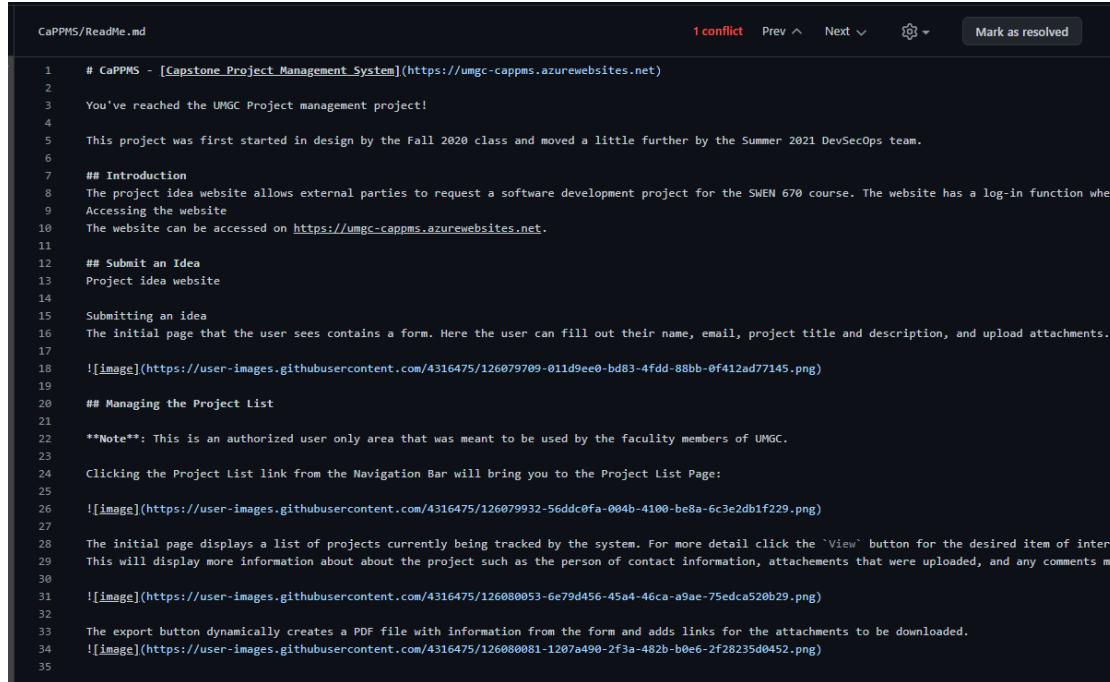
Figure 14: GitHub Resolve Conflict - Identify Number of Conflicts

- Some other hints include a line that identifies the area of the conflicted code. In the picture below, the red arrow helps locate this. The yellow underlined section help locate merge markers. These markers help a user understand what code was added on what branch.

```
1 # CaPPMS - [Capstone Project Management S
2
3 You've reached the UMGC Project managemen
4
5 This project was first started in design
6
7 <<<<< rwilson127/projectlist_readme
8 ## Managing the Project List
9
10 **Note**: This is an authorized user only
11
12 Clicking the Project List link from the N
13
14 ![[image]](https://user-images.githubusercontent.com/123456789/123456789.png)
15
16 The initial page displays a list of projec
17 This will display more information about
18
19 ![[image]](https://user-images.githubusercontent.com/123456789/123456789.png)
20
21 The export button dynamically creates a P
22 ![[image]](https://user-images.githubusercontent.com/123456789/123456789.png)
23 =====
24 ## Introduction
25 The project idea website allows external
26 Accessing the website
27 The website can be accessed on https://umgc-project-management-system.rwilson127.repl.co
28
29 ## Submit an Idea
30 Project idea website
31
32 Submitting an idea
33 The initial page that the user sees conta
34
35 ![[image]](https://user-images.githubusercontent.com/123456789/123456789.png)
36
37
38 >>>> website
39
```

Figure 15: GitHub Resolve Conflict - Identify Merge Markers

- Depending on the desired end result, the user might take only from their own branch, the remote branch, or a combination of the two branches. Whatever the case, the merge markers need to be removed before the conflict can be considered to be resolved. In this case we will take from both branches but move the website code above the user branch code section.



The screenshot shows a GitHub conflict resolution interface for a file named 'CapPPMS/ReadMe.md'. The code content is as follows:

```

1 # CapPPMS - [Capstone Project Management System](https://umgc-cappms.azurewebsites.net)
2
3 You've reached the UMGC Project management project!
4
5 This project was first started in design by the Fall 2020 class and moved a little further by the Summer 2021 DevSecOps team.
6
7 ## Introduction
8 The project idea website allows external parties to request a software development project for the SHEN 670 course. The website has a log-in function wher
9 Accessing the website
10 The website can be accessed on https://umgc-cappms.azurewebsites.net.
11
12 ## Submit an Idea
13 Project idea website
14
15 Submitting an idea
16 The initial page that the user sees contains a form. Here the user can fill out their name, email, project title and description, and upload attachments.
17
18 
19
20 ## Managing the Project List
21
22 **Note**: This is an authorized user only area that was meant to be used by the faculty members of UMGC.
23
24 Clicking the Project List link from the Navigation Bar will bring you to the Project List Page:
25
26 
27
28 The initial page displays a list of projects currently being tracked by the system. For more detail click the 'View' button for the desired item of intere
29 This will display more information about about the project such as the person of contact information, attachments that were uploaded, and any comments ma
30
31 
32
33 The export button dynamically creates a PDF file with information from the form and adds links for the attachments to be downloaded.
34 
35

```

At the top right of the interface, there are buttons for '1 conflict', 'Prev ⌂', 'Next ⌂', a conflict icon, and 'Mark as resolved'.

Figure 16: GitHub Resolve Conflict - Edit document

- With the code re-arranged and the merge markers removed, the next step is to click “Mark as resolved” in the upper right of the pane. This will result in a green check mark and let us Commit merge.

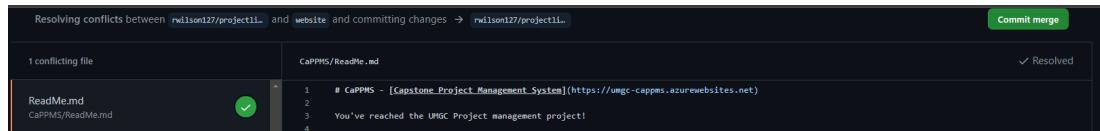


Figure 17: GitHub Resolve Conflict - Commit Merge

- The end result is the PR can now be merged onto the desired branch without further conflict.

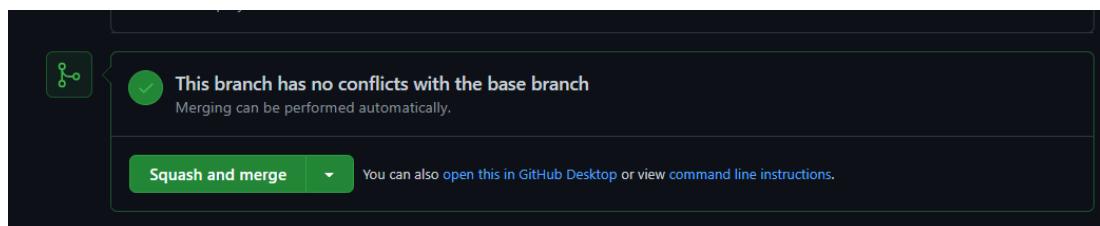
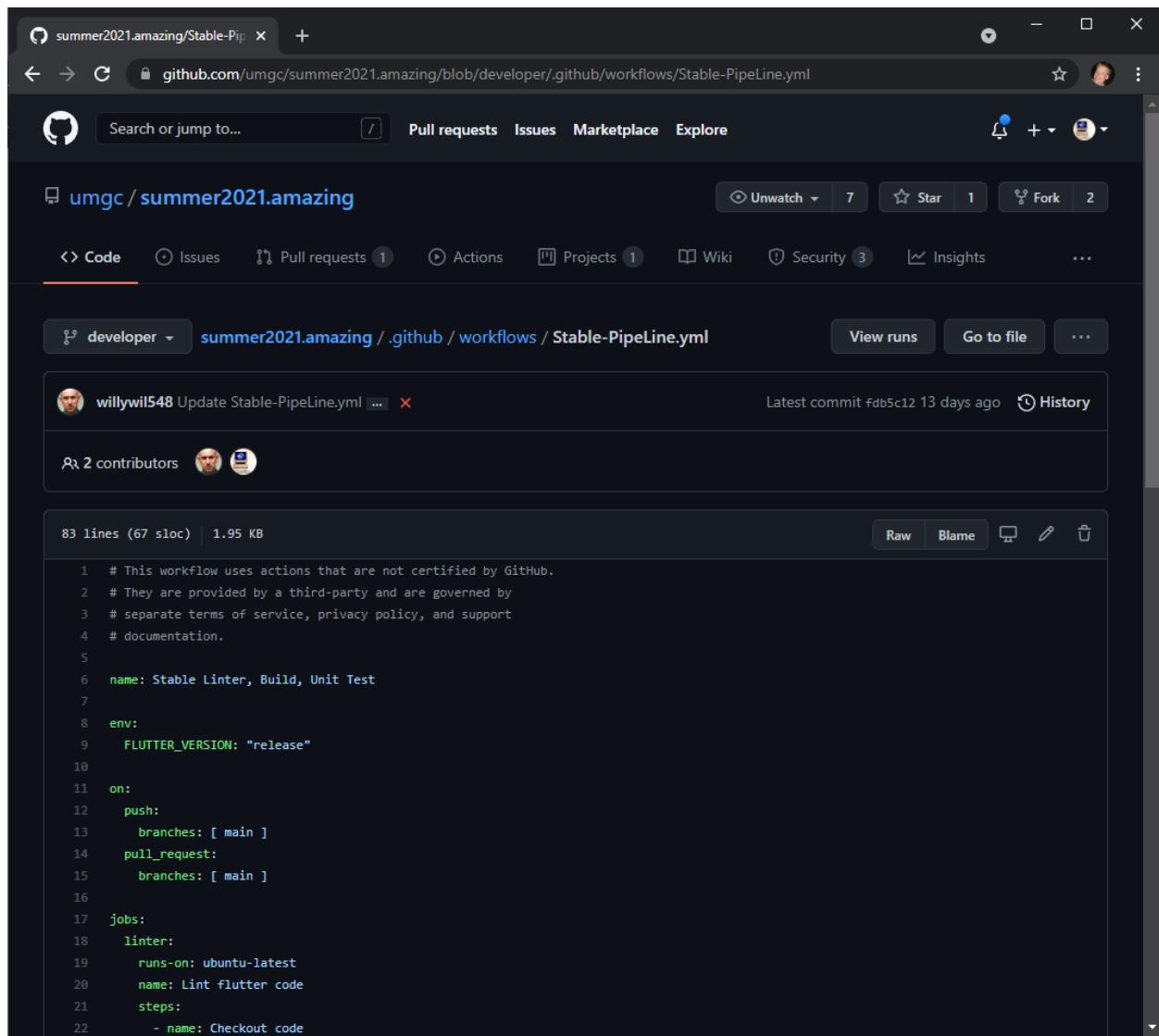


Figure 18: GitHub Resolve Conflict - Merge Resolved Pull Request

Pipelines – GitHub Actions

The DevSecOps team has set up several pipelines in your repository. These consist of scripts that run when a certain action occurs, for example when a pull request is submitted. They contain steps to check the code, check formatting, and depending on the branch, deploy your application.

These scripts can be seen in the .github directory in the root of your repository.



The screenshot shows a GitHub repository page for 'umgc / summer2021.amazing'. The 'Code' tab is selected, displaying the contents of the '.github / workflows / Stable-PipeLine.yml' file. The file contains YAML configuration for a GitHub Action pipeline. The code is as follows:

```
1 # This workflow uses actions that are not certified by GitHub.
2 # They are provided by a third-party and are governed by
3 # separate terms of service, privacy policy, and support
4 # documentation.
5
6 name: Stable Linter, Build, Unit Test
7
8 env:
9   FLUTTER_VERSION: "release"
10
11 on:
12   push:
13     branches: [ main ]
14   pull_request:
15     branches: [ main ]
16
17 jobs:
18   linter:
19     runs-on: ubuntu-latest
20     name: Lint flutter code
21     steps:
22       - name: Checkout code
```

Figure 19: GitHub Actions - Actions YAML Configuration File

The result of this pipeline can be seen in the pull request screen under the Checks tab, but also on the main repository screen under Actions.

The pipelines set up currently build your project and store the resulting APK file.

Retrieving the APK

To retrieve the APK, navigate to your repository, then Actions, and choose a pipeline.

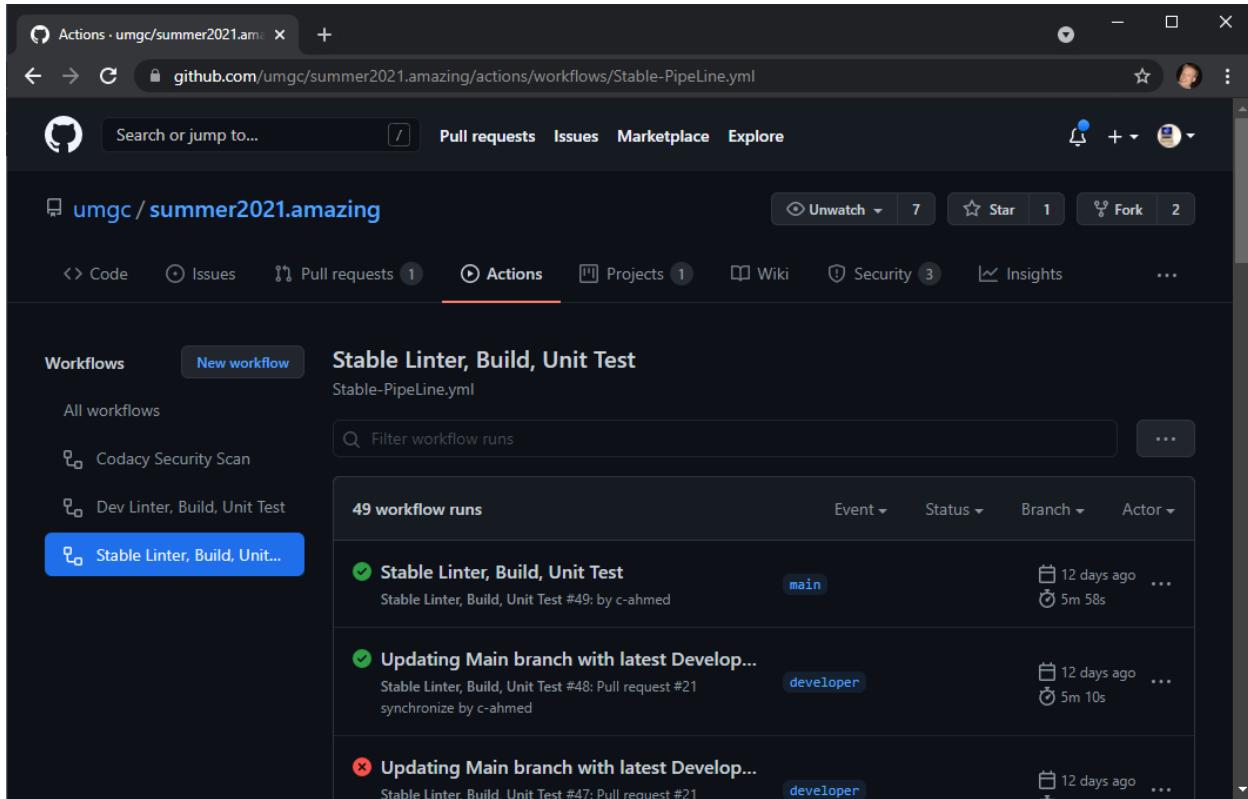


Figure 20: GitHub Actions - Workflows

Next, scroll all the way to the bottom of the page. Here, under artifact, you can download the built APK.

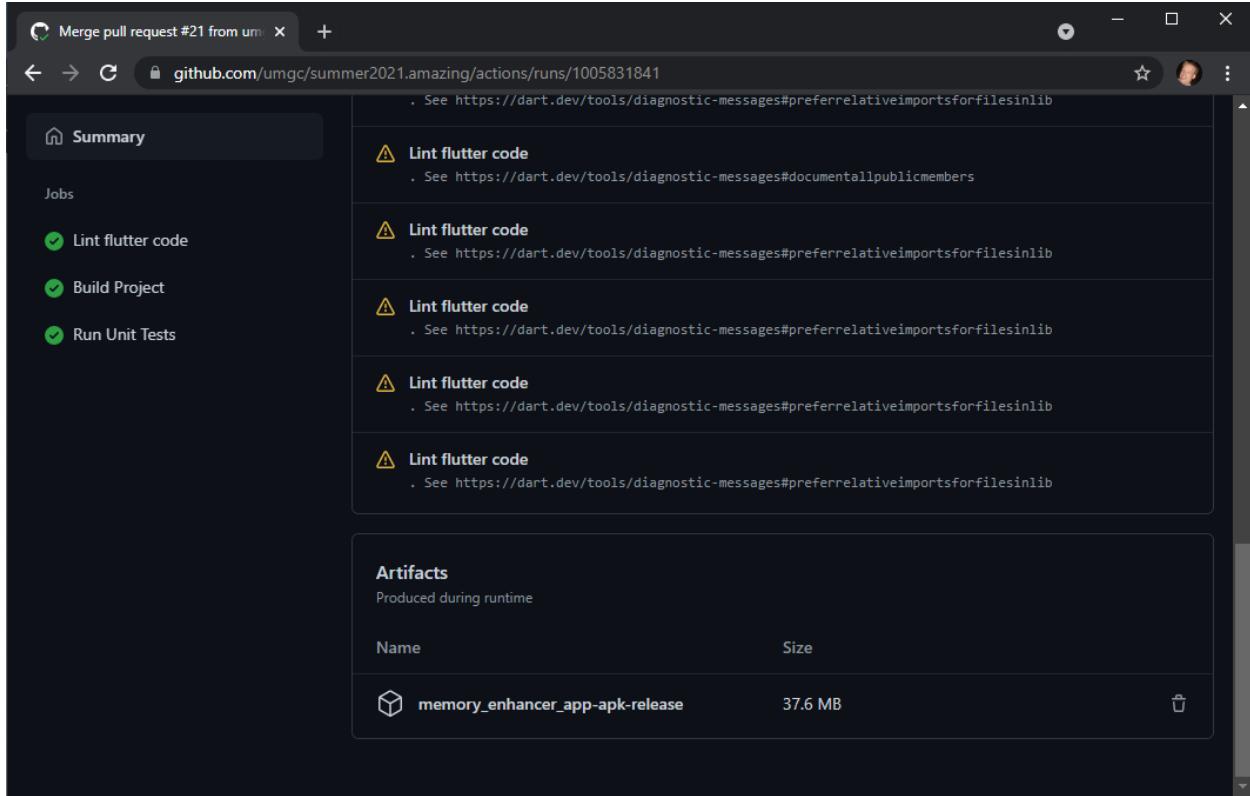


Figure 21: GitHub Actions - Artifacts

From here, you can copy the extracted APK to an Android device over USB.

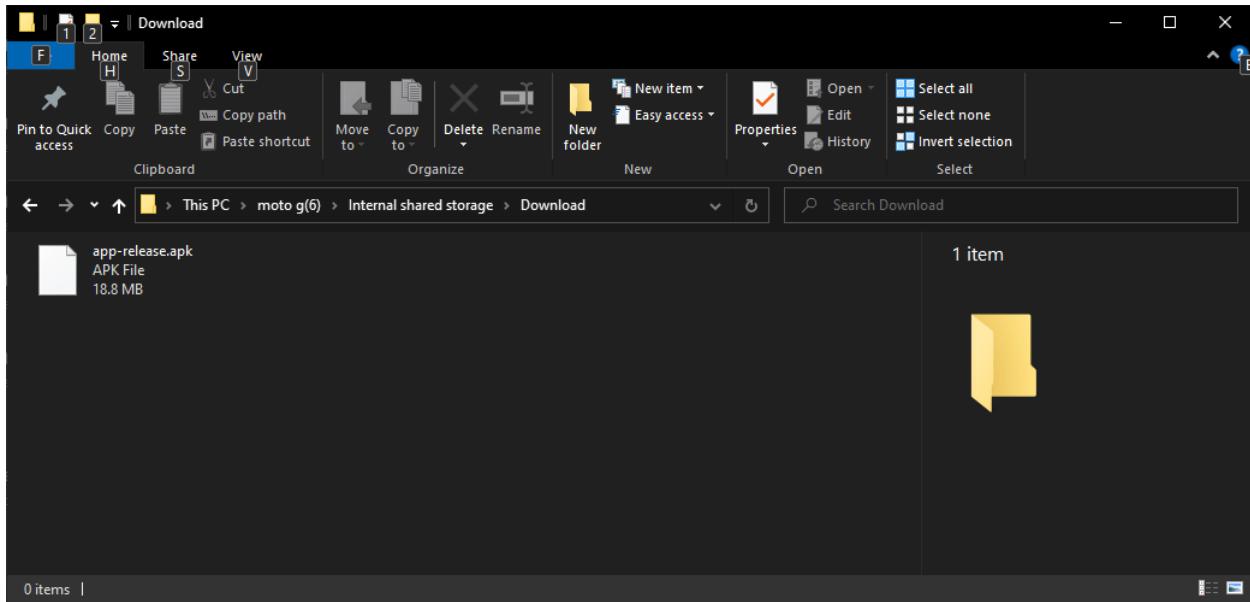


Figure 22: GitHub Actions - Downloaded Artifact

Now, on your device, there will be several warnings. This is because the app does not originate in the Google Play store.

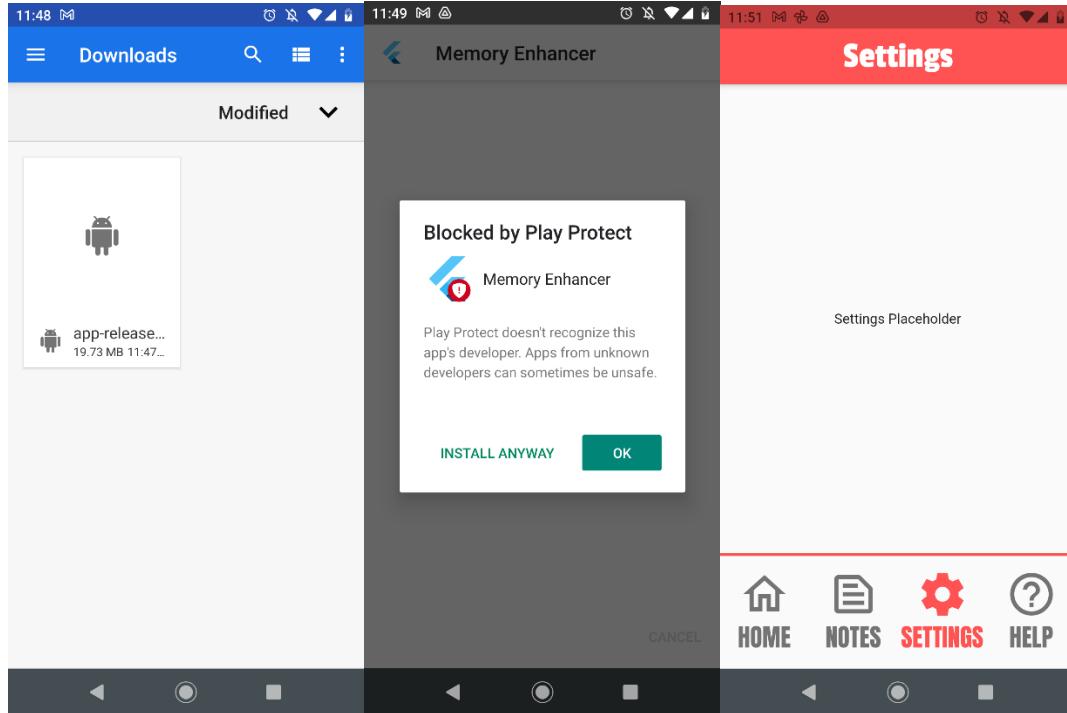


Figure 23: GitHub Actions - Running APK Artifact on Local Device

Advanced Development Factory

Introduction

The Advanced Development Factory, or ADF, is a fully containerized development environment. It contains a full Linux desktop system with several browsers, Visual Studio Code, Android development tools, Flutter, Dart, and the Android emulator. DevSecOps has also installed several other tools that might help future classes, such as Dotnet.

Installing Docker

The first step to get the ADF running on your local machine is to install Docker. Docker is a containerization platform that runs on Windows, Mac, and Linux.

Windows

Install Docker Desktop from <https://hub.docker.com/editions/community/docker-ce-desktop-windows>. Most Docker containers require a Linux kernel. Windows facilitates this through Windows Subsystem for Linux.

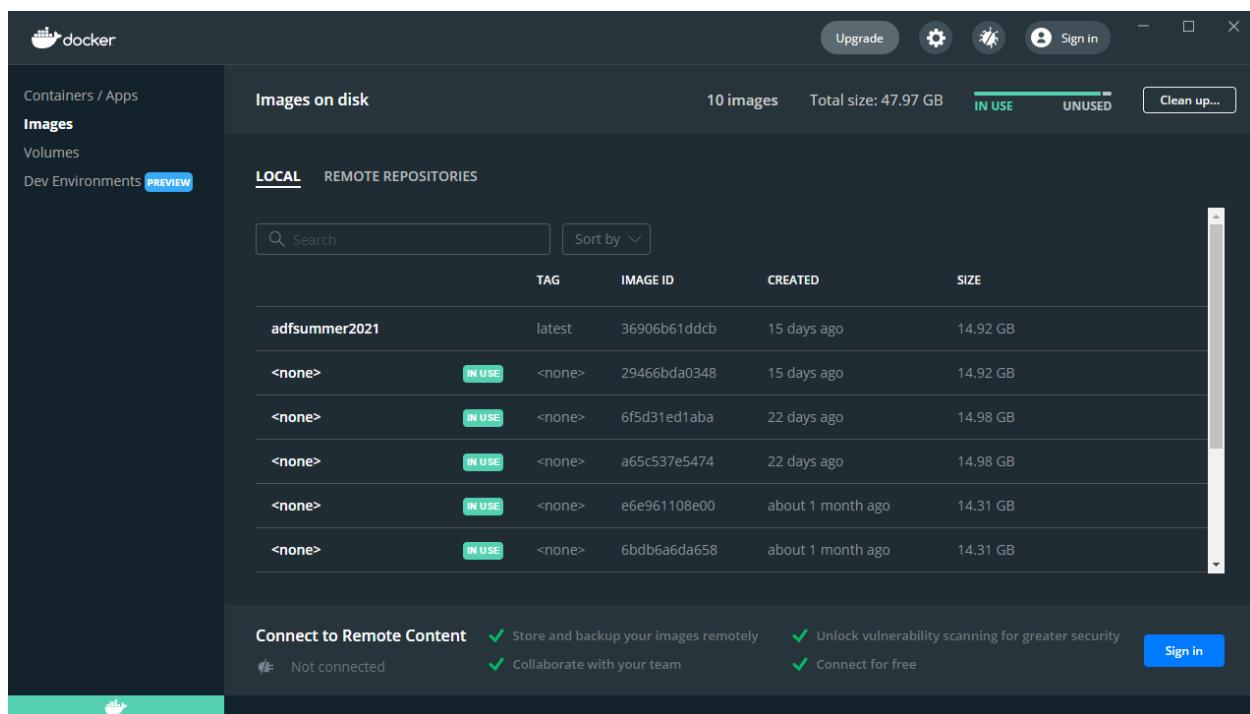


Figure 24: Docker - Images

Linux

Follow the steps for your distribution at <https://docs.docker.com/engine/install/>.

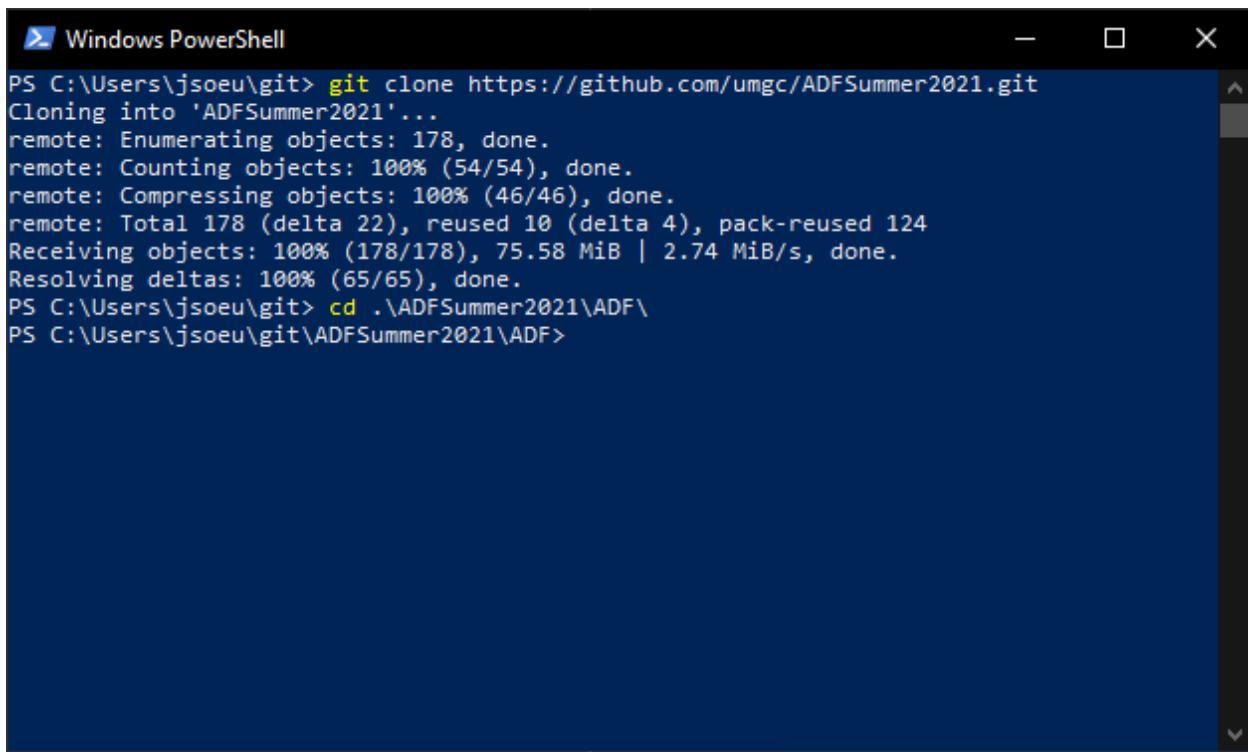
MacOS

Install Docker Desktop from <https://hub.docker.com/editions/community/docker-ce-desktop-mac>.

Clone the ADF repository

From a terminal, clone the ADF repository using

- git clone <https://github.com/umgc/ADFSummer2021.git>
- cd .\ADFSummer2021\ADF\



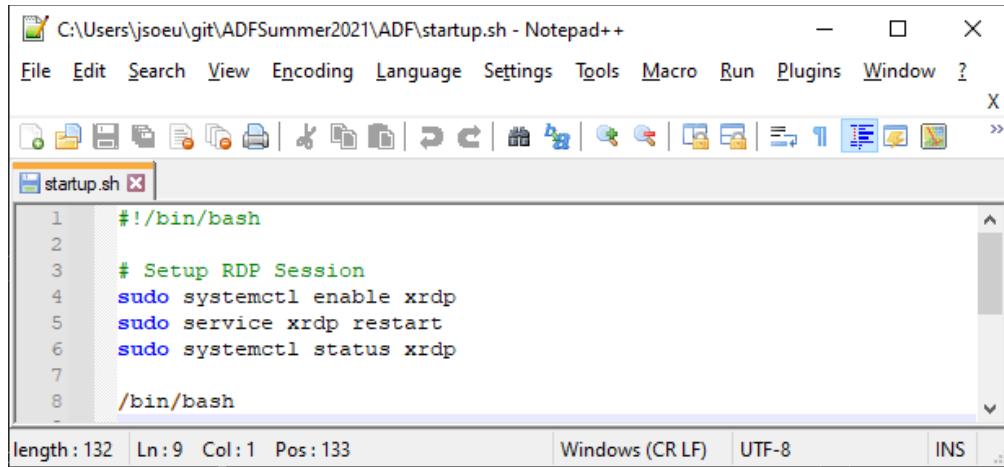
A screenshot of a Windows PowerShell window titled "Windows PowerShell". The command "git clone https://github.com/umgc/ADFSummer2021.git" is run, followed by "cd .\ADFSummer2021\ADF\". The output shows the cloning process, including object enumeration, counting, compressing, receiving objects, and resolving deltas.

```
PS C:\Users\jsoeu\git> git clone https://github.com/umgc/ADFSummer2021.git
Cloning into 'ADFSummer2021'...
remote: Enumerating objects: 178, done.
remote: Counting objects: 100% (54/54), done.
remote: Compressing objects: 100% (46/46), done.
remote: Total 178 (delta 22), reused 10 (delta 4), pack-reused 124
Receiving objects: 100% (178/178), 75.58 MiB | 2.74 MiB/s, done.
Resolving deltas: 100% (65/65), done.
PS C:\Users\jsoeu\git> cd .\ADFSummer2021\ADF\
PS C:\Users\jsoeu\git\ADFSummer2021\ADF>
```

Figure 25: Docker - GitHub Repository Sync

Important: If you are on Windows, and installed Git with Checkout Windows Style, the line-endings of startup.sh will be set to CRLF. This mean that when the file is copied into the Linux container at the end of the build, it will cause an error. To fix this, download Notepad++ from <https://notepad-plus-plus.org/downloads/>. Open up the file. At the bottom is will show you Windows (CR LF) in the status bar:

- Notepad++



The screenshot shows the Notepad++ application window with the file 'startup.sh' open. The code in the editor is:

```
1 #!/bin/bash
2
3 # Setup RDP Session
4 sudo systemctl enable xrdp
5 sudo service xrdp restart
6 sudo systemctl status xrdp
7
8 /bin/bash
```

The status bar at the bottom indicates: length: 132 Ln: 9 Col: 1 Pos: 133. The encoding is set to Windows (CR LF) and the character set is UTF-8.

Figure 26: Docker - Edit startup.sh

Choose *Edit > EOL Conversion > Unix (LF)*, and hit save.

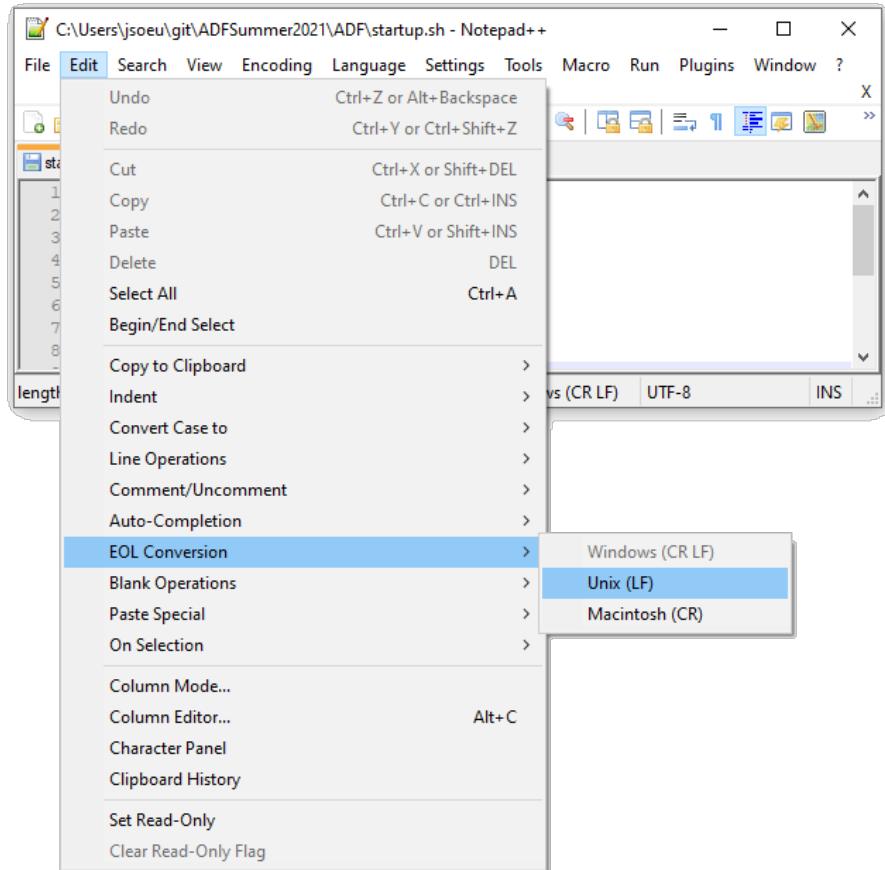


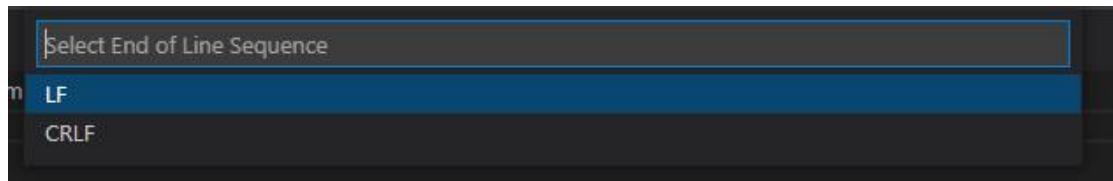
Figure 27: Docker - Notepad++ EOL Conversion

- **Visual Studio Code:**

Click the CRLF/LF from the status bar located in the bottom right of the screen:



Select "LF"



Build the Docker image

From the command line, run the following command:

- docker build --pull --rm -f "ADF/dockerfile" -t adfsummer2021:latest "ADF"

This process will take a while depending on your internet connection.

Running the ADF locally

You can now run the image using:

- docker run -dit -p 3389:3389 --rm --privileged adfsummer2021:latest

Note: on Windows port 3389 might be occupied by the Windows Remote Desktop server. In that case, run it with 63389:3389, and in the next step connect to localhost:63389.

Connecting to the running container

Use your favorite RDP client to connect to localhost:3389 (or 63389 on Windows).

On Windows, use Remote Desktop:

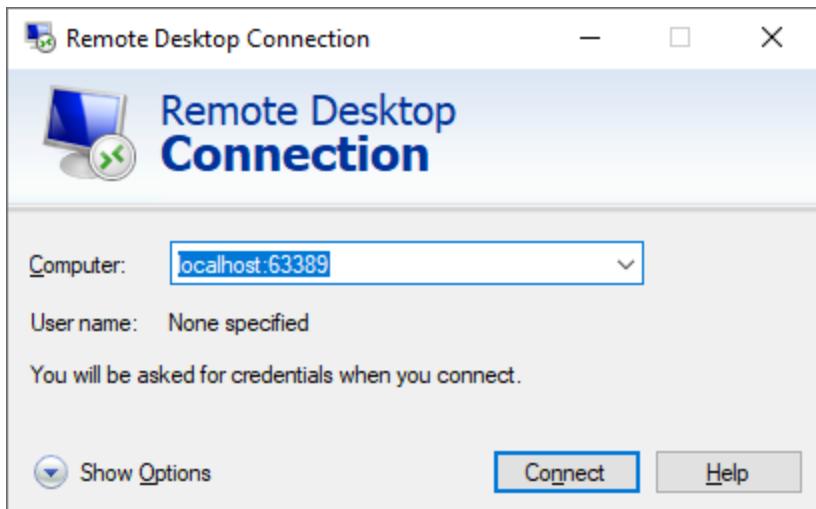


Figure 28: Docker - Microsoft Remote Desktop Connection Screen

On Linux, you can

install Remmina.

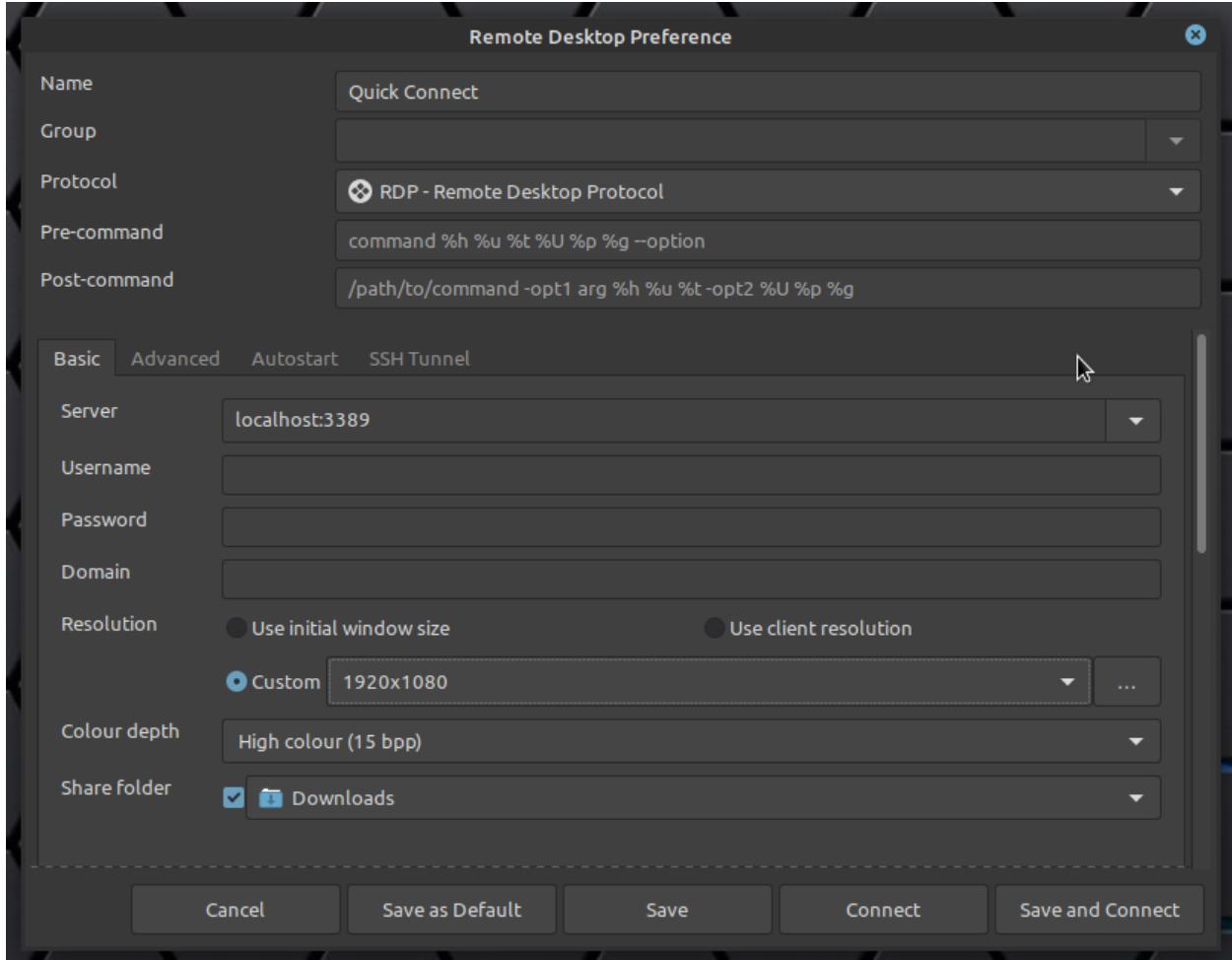


Figure 29: Docker - Remmina Remote Desktop Preference Screen Part I

Make sure to enable sound:

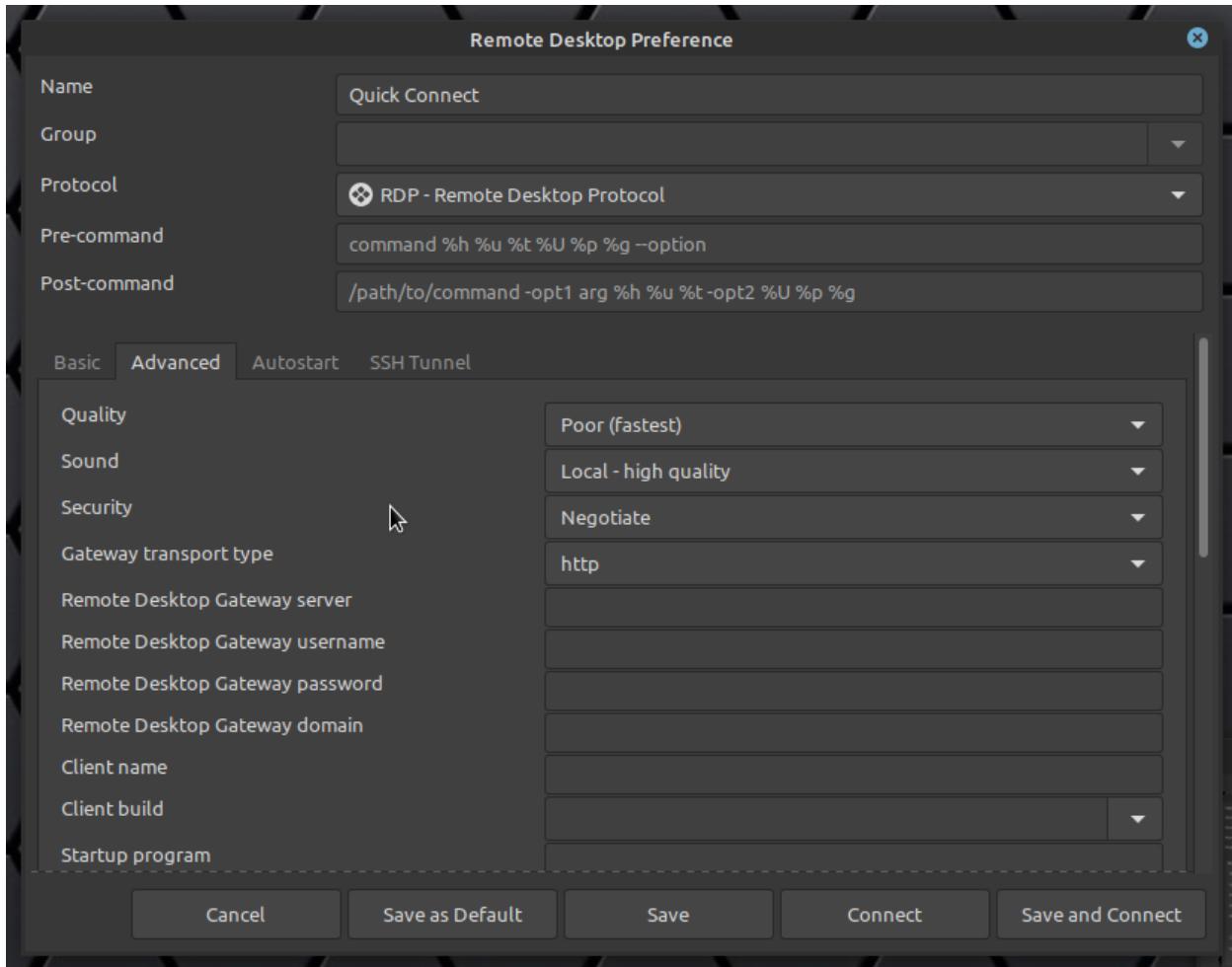


Figure 30: Docker - Remmina Remote Desktop Preference Screen Part 2

Next connect and you'll see a login screen.

- Username: developer
- Password: password

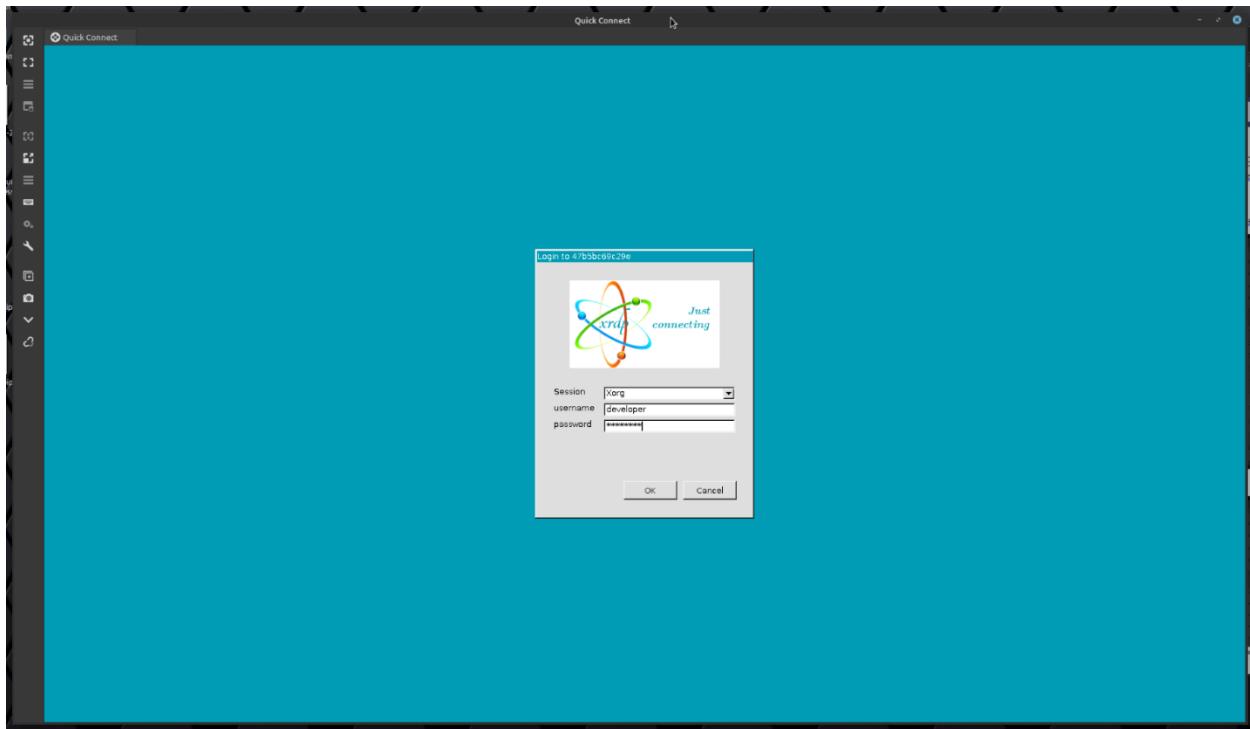


Figure 31: ADF - Login Screen

After logging in you'll be greeted by a welcome screen:
Linux:

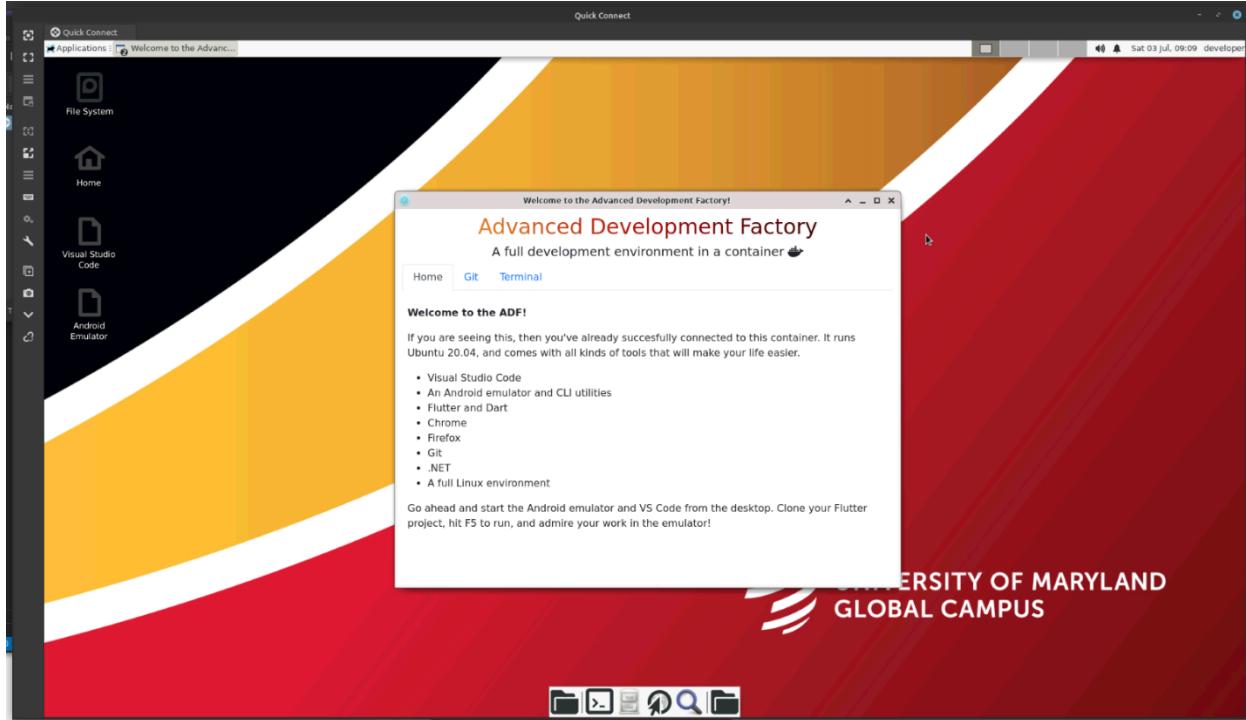


Figure 32: ADF - Welcome Screen via Remmina on Linux

Windows:

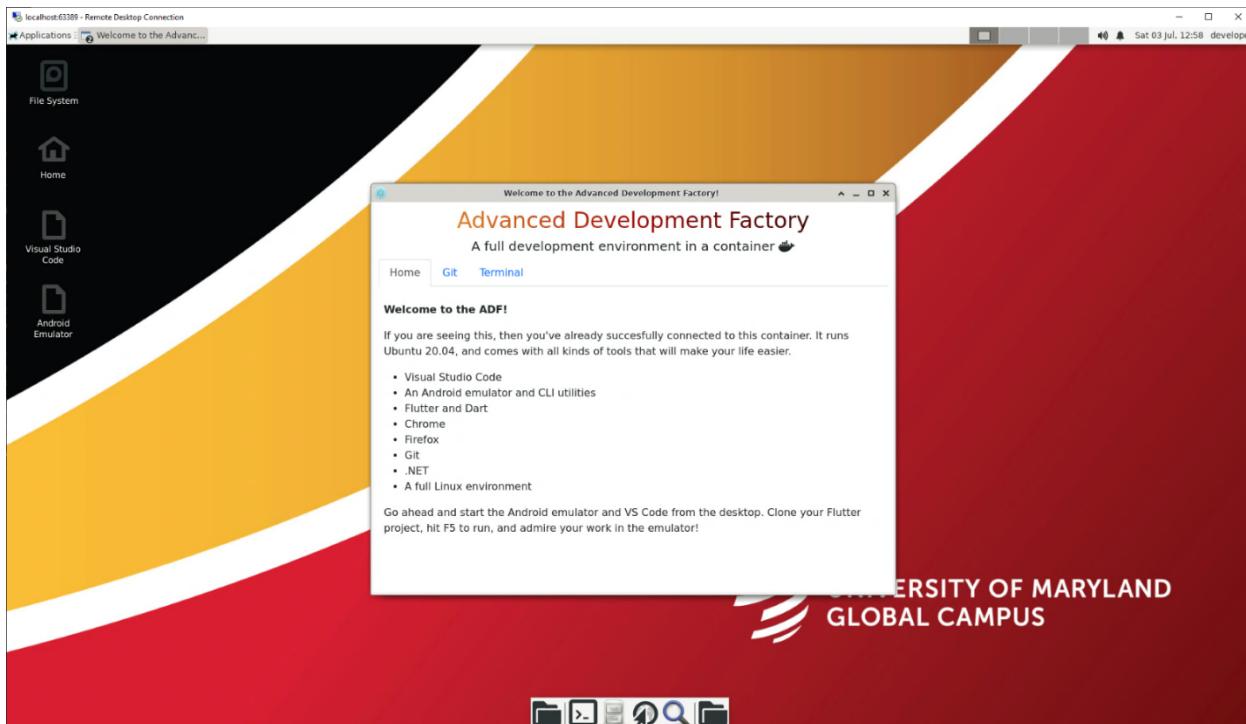


Figure 33: ADF - Welcome Screen via Remote Desktop Connection on Windows

Flutter development in the container

Start Visual Studio Code from the desktop.

Open a new terminal, and clone the repository:

- `git clone https://github.com/umgc/summer2021.charlie.git`

Move into the root directory of the app

- `cd summer2021.charlie`

Start the emulator using (**Note:** The emulator does not work in a container host by Microsoft Windows) via either of the two following ways:

1. `flutter emulators --launch flutter_emulator`
2. from the desktop using the icon.

Run the app using

- flutter run

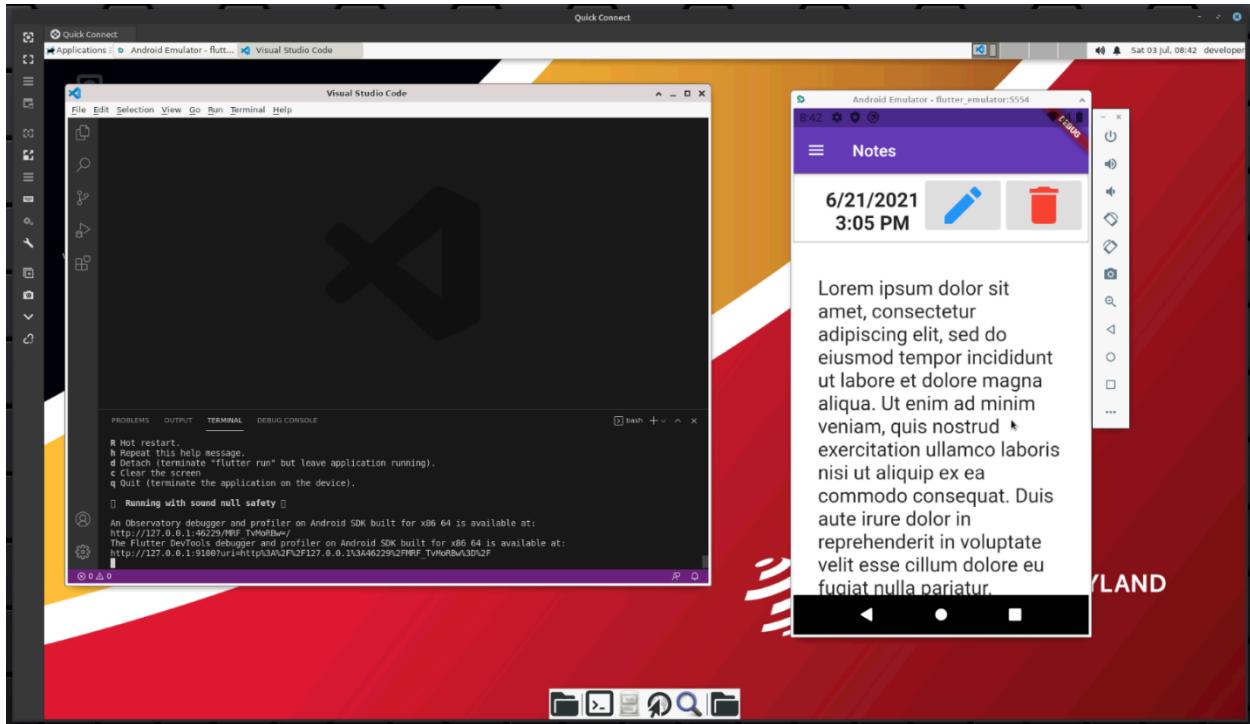


Figure 34: ADF - VS Code with Attached Emulator

If you want to make changes, then you first need to tell Git who you are:

- git config --global user.email you@example.com
- git config --global user.name "Your Name"

Then first create a new branch:

- git checkout -b jersoe/buttonfix

Make your changes, and then stash and commit them:

- git add .
- git commit -m "Finally fixed that button"

Then push your changes to GitHub:

- git push -u origin jersoe/buttonfix

What else is there?

This Docker image provides you with a full Linux environment. The developer account

has sudo access, so you can install additional software if you want to.

Note: sudo access means that you can run any command with root privileges by preceding it with the command sudo (super-user do). Ie: sudo apt-get install vim.

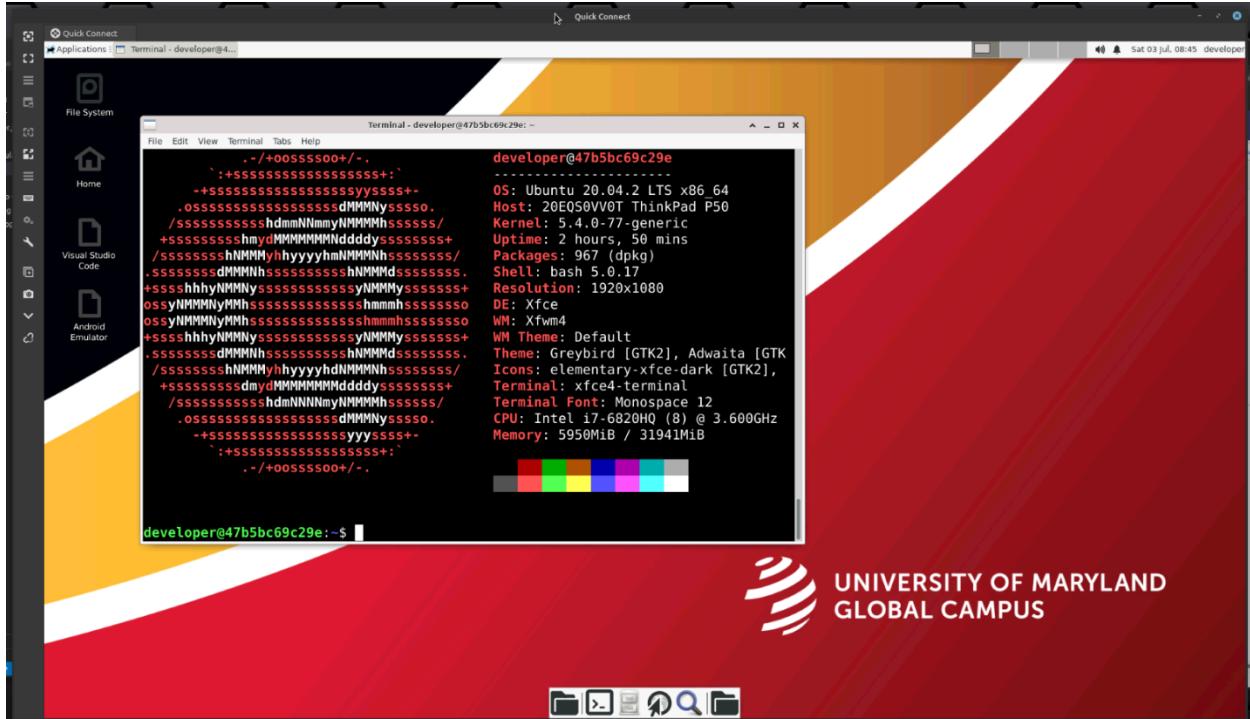


Figure 35: ADF - System Info

How to install extra software

First update the deb repositories using

- sudo apt-get update

Next, install the piece of software you need using

- Sudo apt-get install <packagename>

Examples:

VIM

sudo apt-get install vim

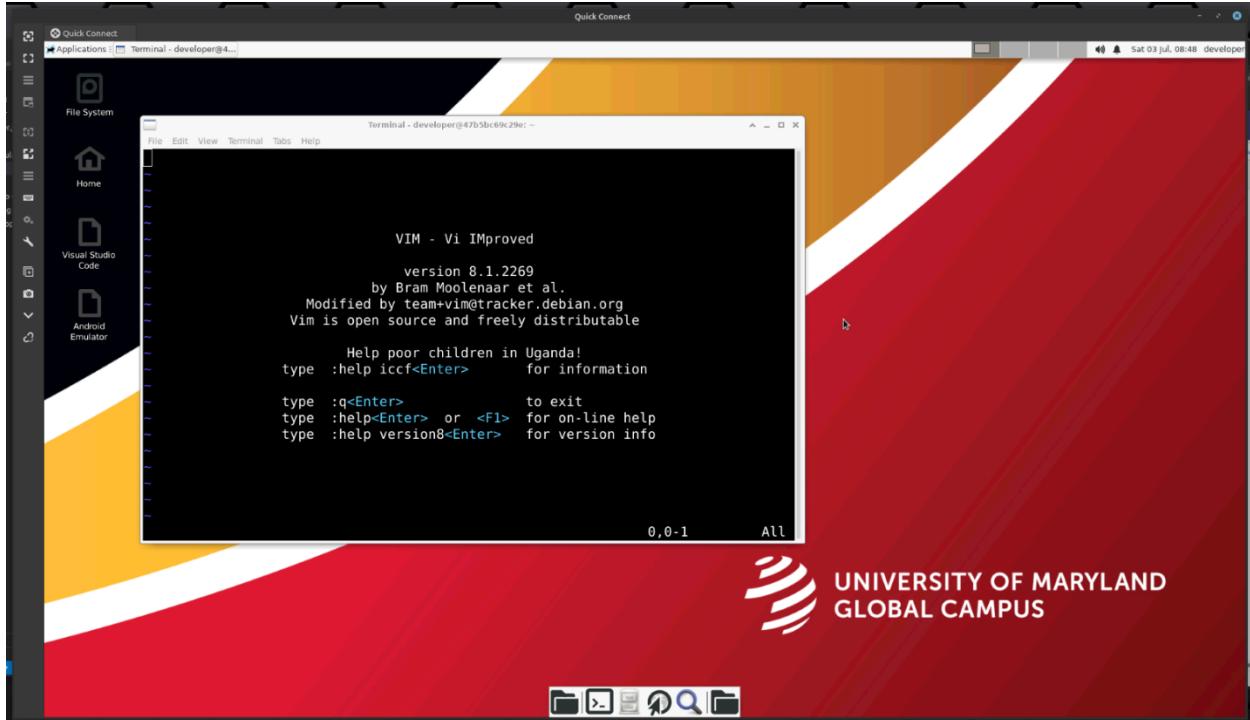


Figure 36: ADF - Installed VIM

Emacs

- sudo apt-get install emacs

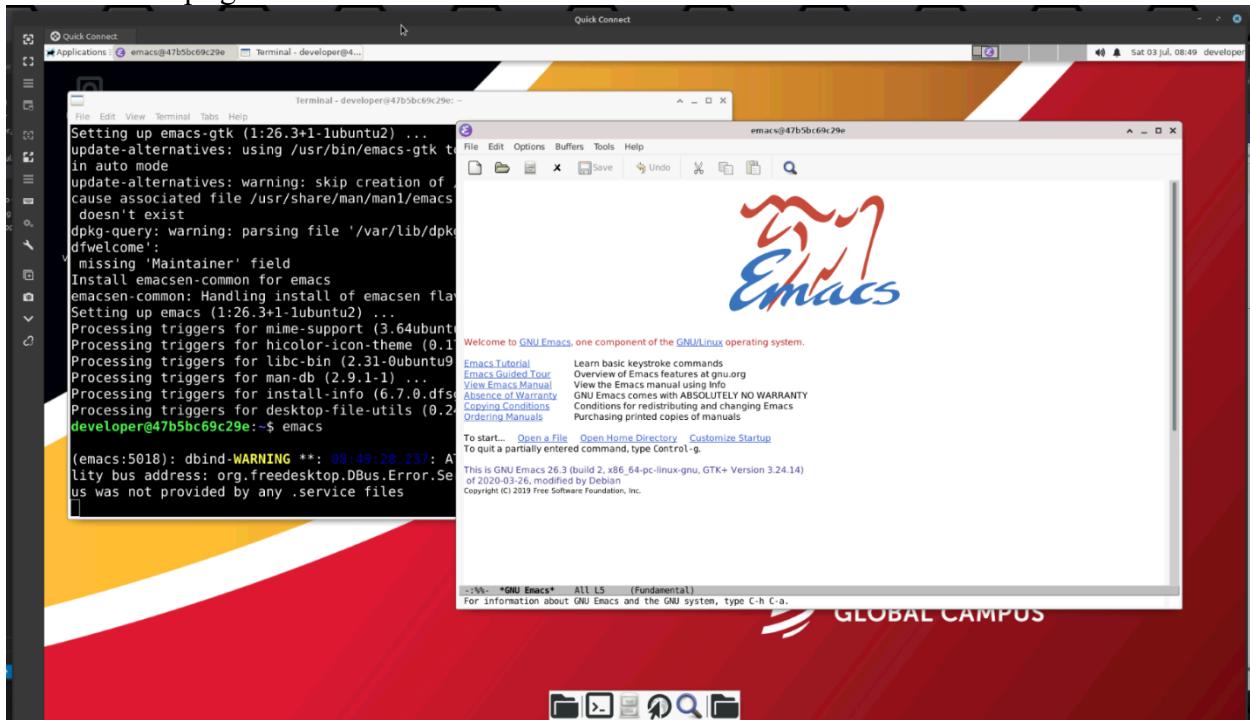


Figure 37: ADF - Installed Emacs

NetBeans

- sudo apt-get install netbeans

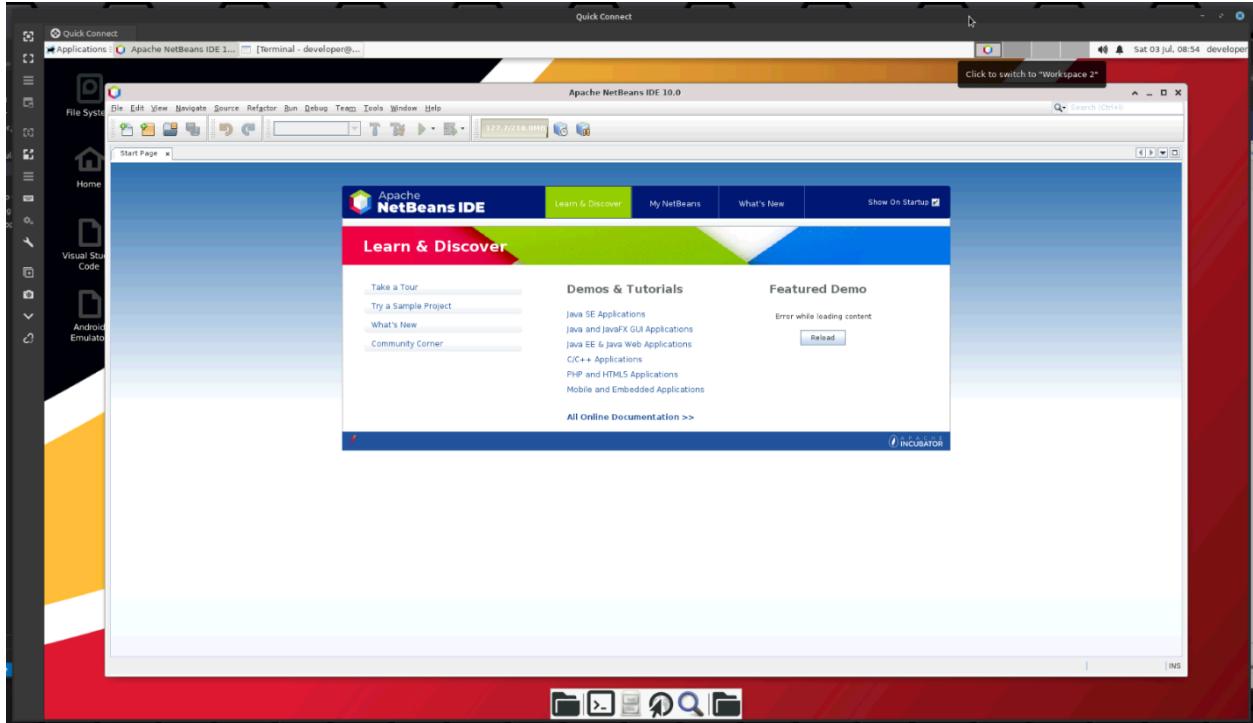


Figure 38: ADF - Installed NetBeans IDE

Project idea website

Introduction

The project idea website allows external parties to request a software development project for the SWEN 670 course. The website has a log-in function where faculty will be able to review, approve, or deny requests. It was developed using the Microsoft Blazor framework, and runs on Microsoft Azure.

Accessing the website

The website can be accessed on <https://umgc-cappms.azurewebsites.net>.

Submitting an idea

The initial page that the user sees contains a form. Here the user can fill out their name, email, project title and description, and upload attachments.

The screenshot shows a web browser window with the URL umgc-cappms.azurewebsites.net. The page title is "Software Development Project Proposal". The form fields are as follows:

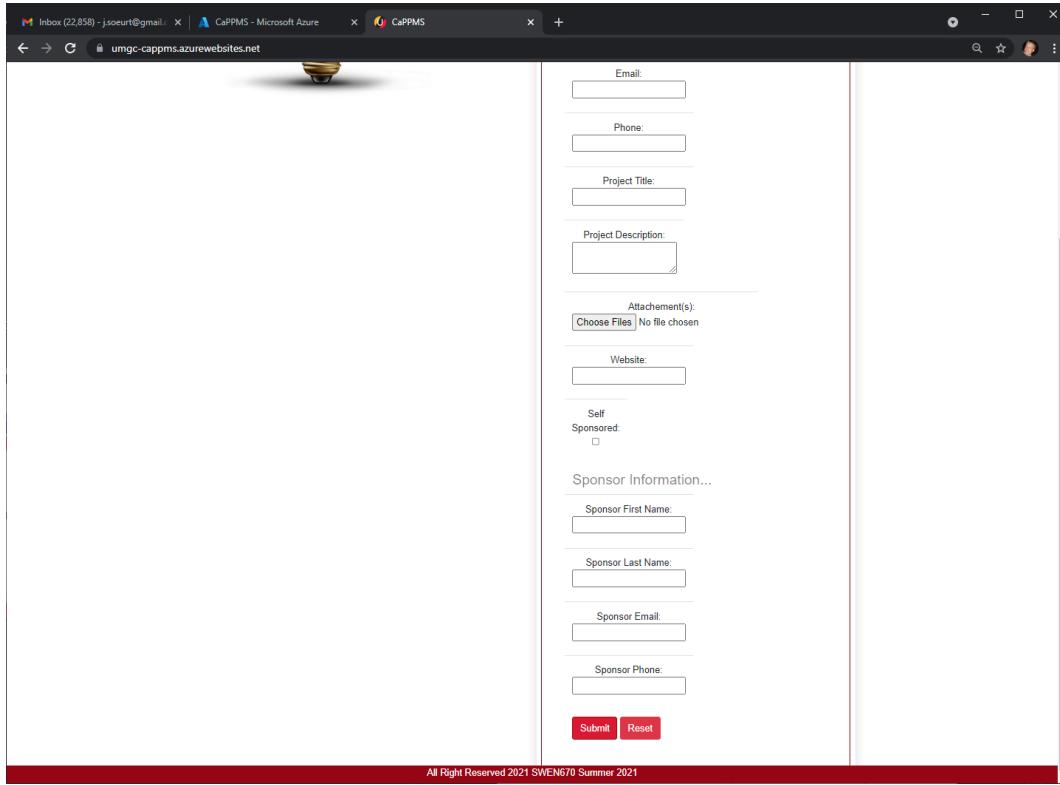
- First Name: Peter
- Last Name: Project
- Email: peter@umgc.edu
- Phone: 123-456-7890
- Project Title: UberEats for cat food
- Project Description:

I'm proposing an UberEats service for cat food. Please see the attached picture of a hungry Furry McPurfecte
- Attachment(s): Choose Files purrymcpurface.jpg
- Website: (empty input field)
- Self Sponsored:

At the bottom are "Submit" and "Reset" buttons.

Figure 39: CaPPMS - Submit Project Idea Page

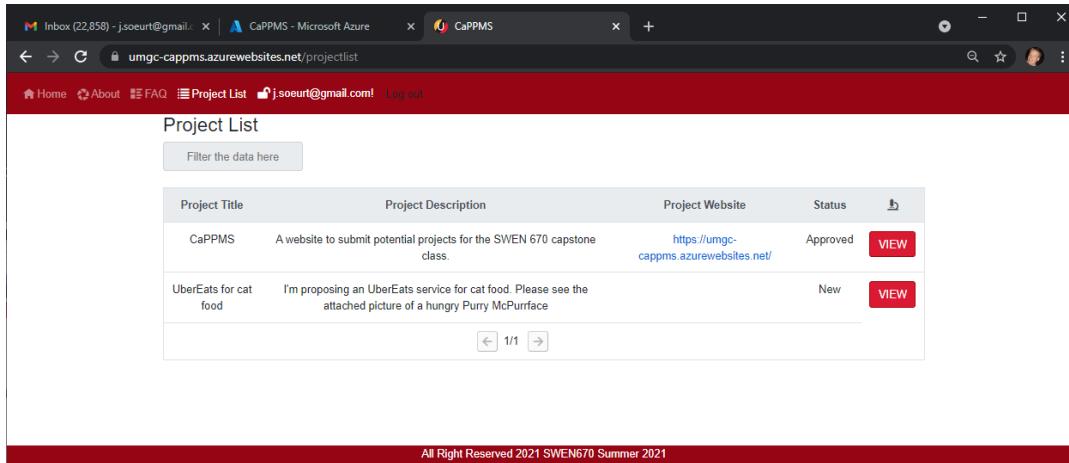
If the user unchecks the Self Sponsored checkbox, the form expands to show sponsor information.



The screenshot shows a web browser window with three tabs: 'Inbox (22,858) - j.soeurt@gmail.com', 'CaPPMS - Microsoft Azure', and 'CaPPMS'. The main content area displays a form for project submission. The 'Self Sponsored' checkbox is unchecked, which triggers an expansion of the form to include fields for 'Sponsor First Name', 'Sponsor Last Name', 'Sponsor Email', and 'Sponsor Phone'. The expanded form also includes fields for 'Email', 'Phone', 'Project Title', 'Project Description', 'Attachment(s)', 'Website', and 'Submit' and 'Reset' buttons. The footer of the page reads 'All Right Reserved 2021 SWEN670 Summer 2021'.

Figure 40: CaPPMS - Extend Page for Other Than Self Sponsor Information

If the user is logged in, they can navigate to the Project List:



The screenshot shows a web browser window with three tabs: 'Inbox (22,858) - j.soeurt@gmail.com', 'CaPPMS - Microsoft Azure', and 'CaPPMS'. The main content area displays a 'Project List' table. The table has columns for 'Project Title', 'Project Description', 'Project Website', 'Status', and 'View'. There are two rows of data: one for 'CaPPMS' (Approved, View button) and one for 'UberEats for cat food' (New, View button). A 'Filter the data here' input field is located above the table. The footer of the page reads 'All Right Reserved 2021 SWEN670 Summer 2021'.

Project Title	Project Description	Project Website	Status	
CaPPMS	A website to submit potential projects for the SWEN 670 capstone class.	https://umgc-cappms.azurewebsites.net/	Approved	VIEW
UberEats for cat food	I'm proposing an UberEats service for cat food. Please see the attached picture of a hungry Purry McPurrface		New	VIEW

Figure 41: CaPPMS - Project List Page

Clicking on View shows all the details.

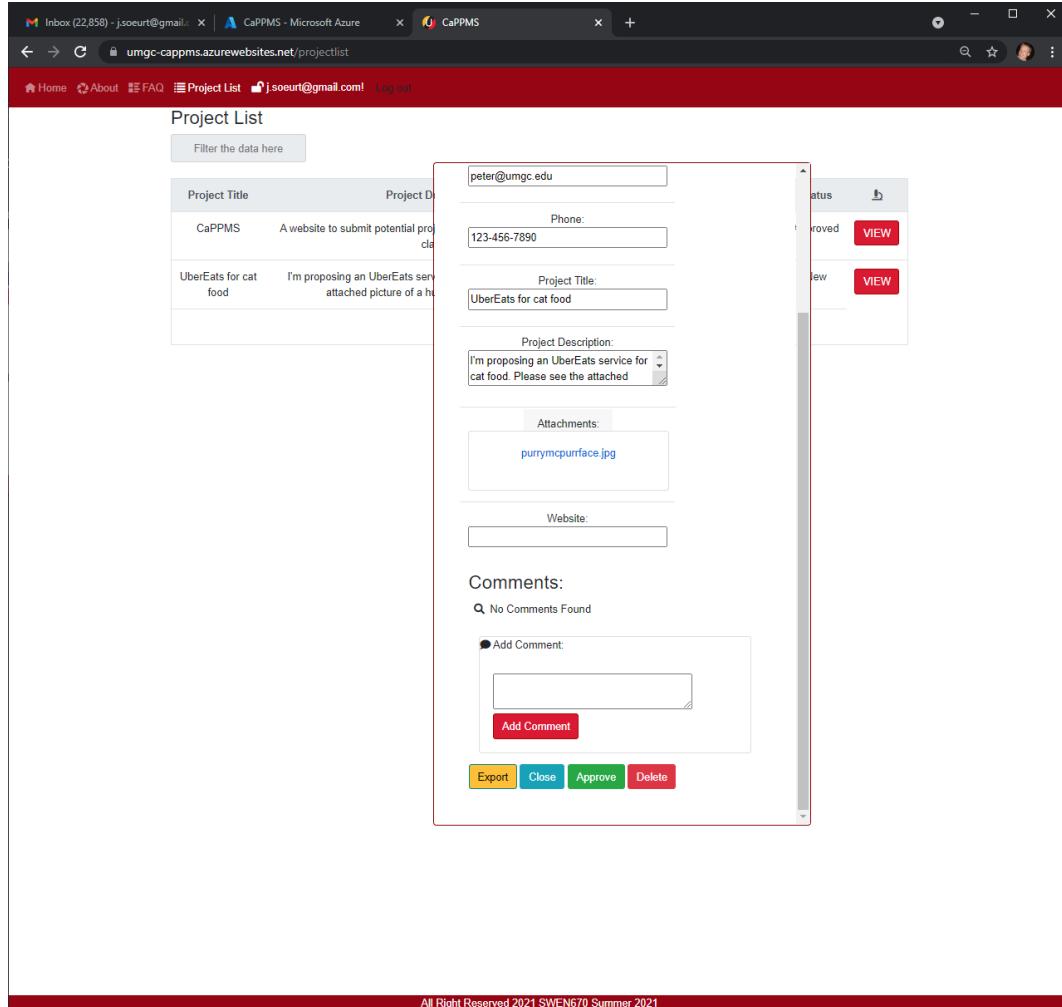


Figure 42: CaPPMS - Project List Idea View

Here the user can also choose to export. In that case, a PDF is generated containing the project proposal details.

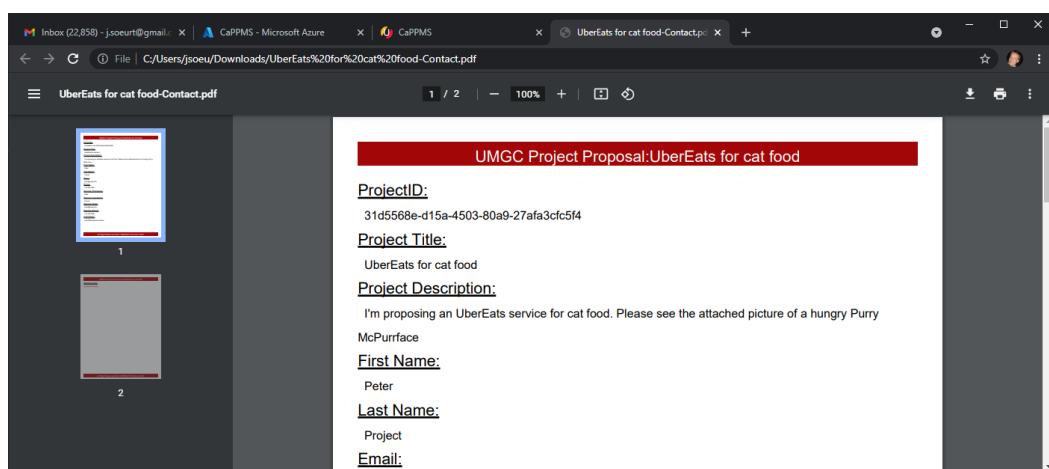


Figure 43: CaPPMS Project Idea PDF Export

The logged in user can also leave comments, approve, or delete the proposal.

Comments:

 No Comments Found

 Add Comment:

Add Comment

Export**Close****Approve****Delete**

Figure 44: CaPPMS - Project Idea Comments and Menu Section