COSC 4P02

Software Engineering II

Final Report

April 30th, 2023

Group Members

Parth Bhaveshbhai Patel	**Product Owner/Developer**	**pp19oo**
Akshar Patel	**Product Owner/Developer**	**ap18zr**
Naitik Hareshbhai Chovatiya	**Scrum Master/Developer**	**nc18us**
Aum Pandya	**Developer**	**ap19xt**
Anishka Shetty	**Developer**	**as18cq**
Darshakkumar Bambharoliya	**Developer**	**db18hn**
Sneh Patel	**Developer**	**sp18oo**

Table of Contents

Introduction **3**
User Stories **3**
Installation Manual **3**
Clone the repository 3
Open the project in Unity 4
Using XCode to Build the IOS App 6
Development Tools and Platform **8**
Development Process **8**
Sprint Planning 8
Daily Stand-up Meeting 13
Citations **14**
Test Documents/Result **16**
GitHub Logs **21**
CLIO - Our Main Unity Repo 21
CLIO-React 22
Documents 23

Introduction

The purpose of this final report is to provide an overview of our AR (Augmented Reality) application with all working elements, including the user interface design, 3D models, image tracking components, audio scripts, animations, and application architecture which is used to create a functional AR application. Additionally, this document will discuss the development tools and methods. We have also mentioned the challenges that we faced while developing the applications and how we have overcome them.

User Stories

- As a user, I want to be able to look at museum artifacts on my mobile phone so that I can have more information about the artifacts.
- As a User, I want to have a guided tour of the app, so I know how to utilize every feature.
- As a user, I want to have audio descriptions of every object so it is easier for me to understand the history of that artifact better.
- As a user, I want to have a visually aesthetic user interface and a simple app so that I can use it easily.
- As the museum manager, I want the app to have a link to our museum website so that users can have more information about the museum and also donate to the museum's cause.
- As a manager, I want the app to be kid-friendly and also usable by older visitors so everyone can enjoy it.
- As a manager, I want the app to be accessible so everyone can feel included.
- As a museum manager, I want the app to respect the user's privacy and follow the museum's ethical guidelines.
- As a manager, I want the app to be downloadable to all our visitors so everyone can enjoy the app.
- As a manager, I want the app to represent all events factually so that wrong information is not displayed to the users.

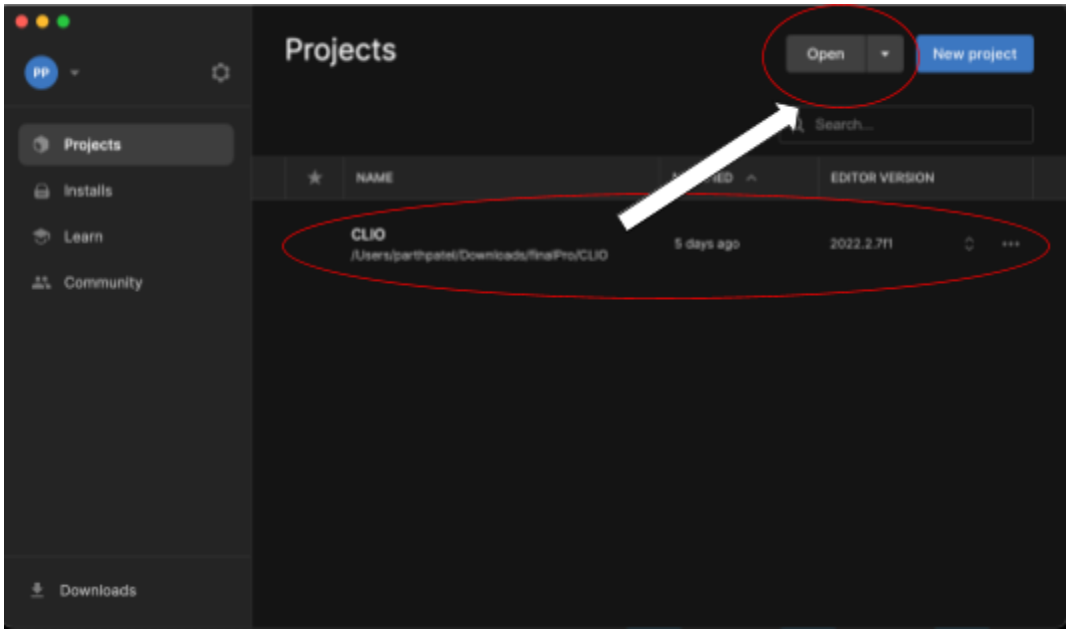
Installation Manual

Clone the repository:

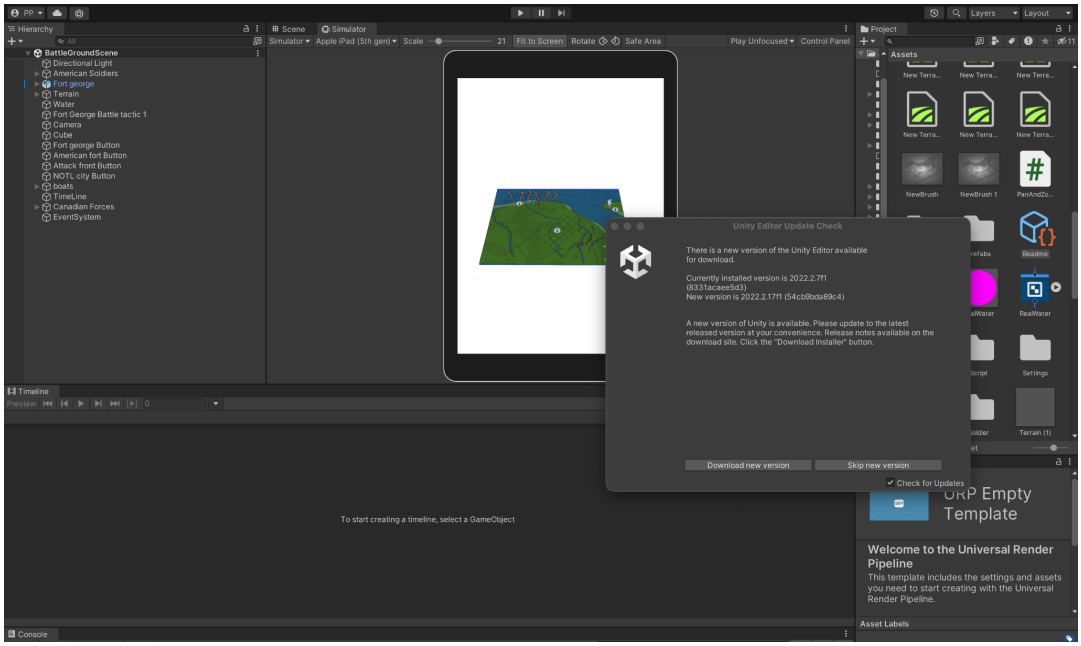
```
git clone git@github.com:COSC-4P02-Team-REX/CLIO.git
```

Open the project in Unity:

1. Follow instructions on https://unity.com/download to download and install unity for your system. Install the Latest LTS editor available or you can use ***2022.2.7f1.***
2. Go to the cloned GitHub repo and open the Unity project.

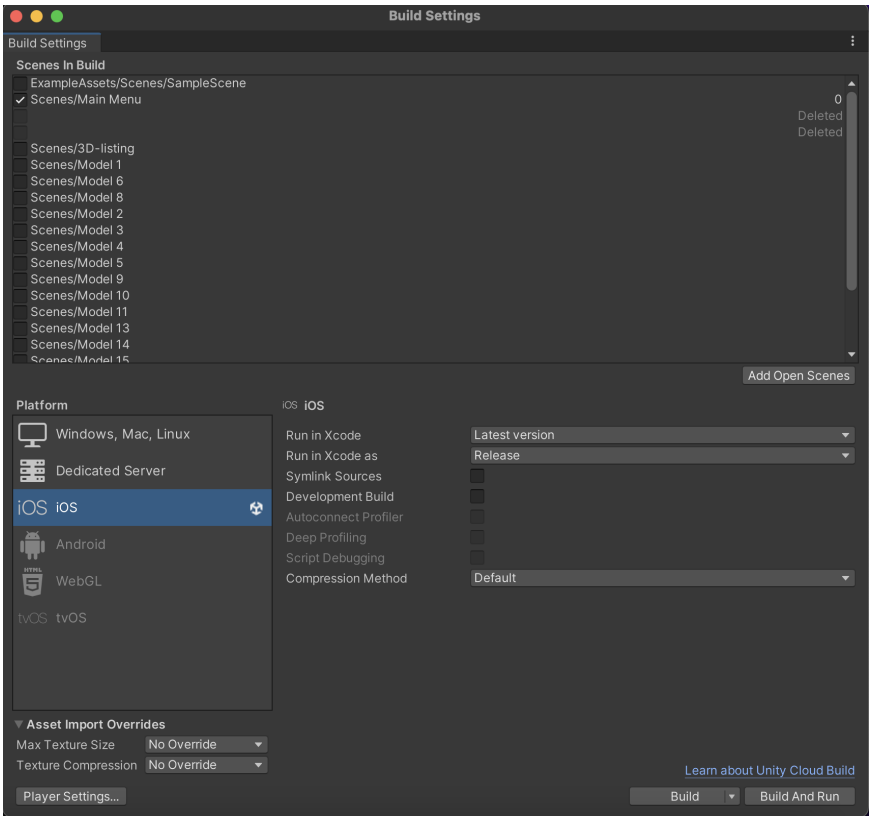


Note: The whole process can take up to 15 minutes, so you should grab a drink if you want to right now.



In the end you should see a window open up as the one above.

3. Change the platform to IOS in File>Build Settings by clicking “Switch Platform” in IOS, if required. (If not on IOS platform, the Build button shown in the figure below would say Switch Platform).



4. Click “Build” and choose your build directory.
    - Note: if there is no build directory click New Folder and title it Build. Press OK and Build here.
        - If there is an existing Build directory, choose it and click replace, then OK.

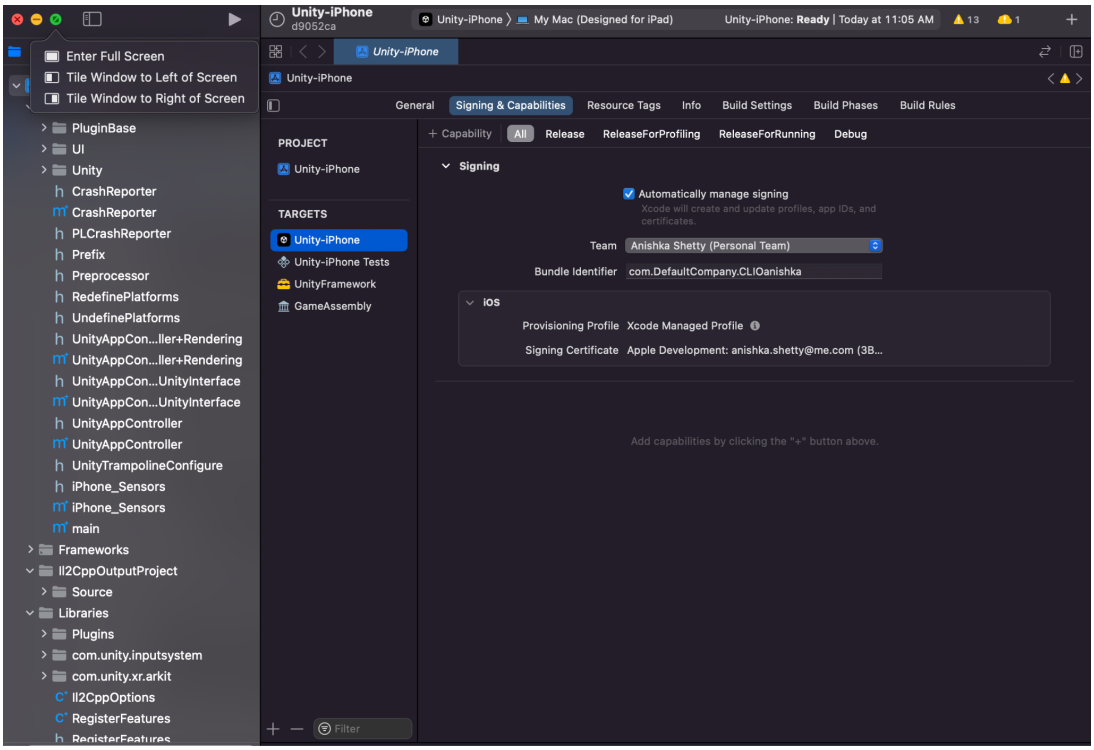
5. Build and Run the project!! It should take around 5-10 minutes.

Using XCode to Build the IOS App

1. (This not apply to all) Before beginning with Xcode, you will need to configure developer settings on your iPhone or iPad.

2. Once developer settings have been configured on your device, you can start building the app. Go to your Build folder, and look for the Unity-[iPhone/iPad].xcodeproj file. Double-click it!!

3. Try running the Xcode project. You may need to configure a signing signature that links the app to you.



The above figure shows an example of what your Signing and Capabilities should look like before running the project. Select your name under Team and add a unique bundle identifier beside .CLIO, such as your name.

4. After the app is built you should approve yourself as a Trusted Developer in your iPhone. Follow these steps:

- Go to Settings on your iPhone/iPad
- Go to General > VPN & Device Management
- Trust developer

5. Your app should be running now. Cheers. 🙂

Note: This is just a developer setup, Moving forward we want to host this on Apple's App Store and later have users download it directly from the App Store.
This will also allow us/Dev Team to push updates more quickly and allow us to have several tests placed in the deployment pipeline.

Development Tools and Platform

- Unity
- Metashape (Preferably using a mac studio with M1 Ultra or similar)
- Blender
- XCode
- Mac (Needed for iOS development)
- iPhones (for testing)

Development Process

For this project, we have used the agile scrum methodology. Scrum is an agile development process that can be applied to the developed AR applications. In this situation, Scrum contains segmenting the development process into a number of time-boxed units called Sprints, each of which contains a defined set of tasks that must be finished. A prioritized list of features and functionality is created by the development team in collaboration with stakeholders and is referred to as the Product Backlog. The team works on the Sprint Backlog, a part of the Product Backlog, during each Sprint.

Daily scrum meetings are held by the team to assess performance, spot blockers, and maintain attention on the Sprint objective. The team does a Sprint Review after each Sprint to present the finished product to stakeholders and request feedback. The Product Backlog for the following Sprint is informed by this feedback. The team also does a Sprint Retrospective to analyze the sprint's successes and failures and pinpoint areas for improvement.

The collaborative, iterative, and configurable nature of the development process for an AR software can be improved by using Scrum. It enables the team to concentrate on providing top-notch features and functionality in a timely and effective manner, as well as incorporating feedback from stakeholders and users to continuously enhance the app.

Sprint Planning:

During the sprint planning, the product owner and development team met together and reviewed the product backlog and previous work of the team. The product owner also works on the following tasks:

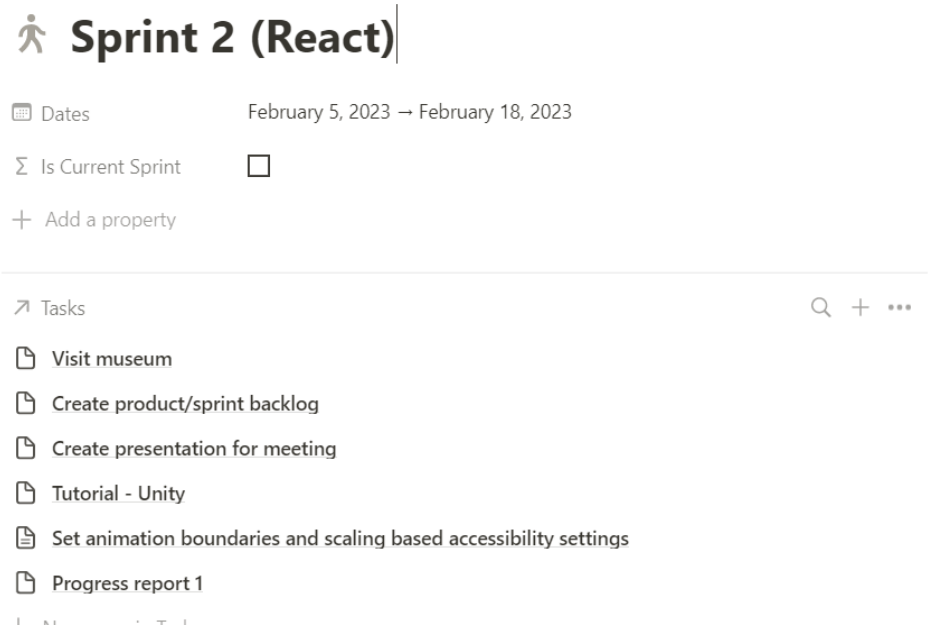
- Assign tasks to each team member
- Create the backlog

- Set the objective
- Define acceptance criteria
- Plan the sprint

We had a total of 6 sprints for the development. Each sprint was 2 weeks long. We have described the brief description of the all sprint below:

Sprint	Main focus
1	Set up the environment, created the system diagram, worked on UI mockups, and watched tutorials for learning tools.
2	Visited the museum, scanned the museum and trigger surfaces, started work on the Unity app.
3	Started to create 3D models of Museum artifacts, designed the main Menu and UI interface, and outlined the battleground scene.
4	Designed pop-up cards, added water on terrain for scene, created 3D model of soldier.
5	Implementation of the popup cards and text-to-speech accessibility features, button tracking with images.
6	Integration of all members' work , tested app on site, added tutorial for guiding the users, resolved conflicts and errors, removed bugs.

Screenshots of the Sprint Planning:



Sprint 3

Dates February 19, 2023 → March 4, 2023

Σ Is Current Sprint ☐

+ Add a property

↗ Tasks

- Gun Scene
- Create 3D Models for the Following Artifacts: Powder Flask and Horn, Canteen and Axe
- Main menu/ UI for App
- Battleground Scene - Create terrain for battle scene in unity
- Battleground Scene -Create 3d Models for soldiers for battle scene
- Set up development environment for Unity
- Take photos of the museum
- Visit museum to get user requirements
- ImageTracking Manager/Script
- Configure triggers for animation

Sprint 4

Dates March 4, 2023 → March 17, 2023

Σ Is Current Sprint ☐

+ Add a property

↗ Tasks

- Pop-up Cards for Artifacts
- Gun Scene
- Create 3D Models for the Following Artifacts: Powder Flask and Horn, Canteen and Axe
- Battleground Scene - Create terrain for battle scene in unity
- Battleground Scene - Add Water/ rain and mud effects to the unity scene
- Battleground Scene -Create 3d Models for soldiers for battle scene

Sprint 5

Dates March 15, 2023 → March 22, 2023

Is Current Sprint ☐

Add a property

Tasks

- Battleground Scene - Add Water/ rain and mud effects to the unity scene
- Battleground Scene -Create 3d Models for soldiers for battle scene
- Battleground Scene - Create terrain for battle scene in unity
- Poster/3d overlay or cards
- Untitled
- Battleground Scene - transition scene

Sprint 6

Dates April 2, 2023 → April 15, 2023

Is Current Sprint ☐

Add a property

Tasks

- Button tracking with images
- On screen components
- 3D models (Mirroring and merging)
- Battle animations
- Soldier model
- On screen photo (screenshots)
- NOTL Map anim
- Rain effects
- Add 3D models to posters
- Voice model for text-to-speech

Daily Stand-up Meeting:

We had a daily stand-up of 30 minutes on Google Meet. In these meetings we discussed the progress of our tasks and any blockers of development.
Main goals of the daily stand up:

- Check the progress since the last stand-up
- Discuss potential blockers or any errors in development
- Collaboration of team members
- Plan for the next day

Citations

Information on Pop-up Cards

"1812 Niagara." *The War of 1812: Niagara Frontier and York*, http://www.archives.gov.on.ca/en/explore/online/1812/niagara.aspx.

"Battle of Fort George." *The Canadian Encyclopedia*, https://www.thecanadianencyclopedia.ca/en/article/battle-of-fort-george.

"Battle of the Queenston Heights." *American Battlefield Trust*, https://www.battlefields.org/learn/war-1812/battles/battle-queenston-heights.

"Biography – Dickson, William – Volume VII (1836-1850) – Dictionary of Canadian Biography." *Home – Dictionary of Canadian Biography*, http://www.biographi.ca/en/bio/dickson_william_7E.html.

"Brief History of the War of 1812." *Niagara 1812*, http://discover1812.com/about/history/.

Canada, Textile Museum of. "Daniel Servos Coat." *Narrative Threads*, 12 Aug. 2015, http://surl.li/fqjkb.

Defence, National. "Laura Secord." *Canada.ca*, / Gouvernement Du Canada, 21 June 2018, https://www.canada.ca/en/services/defence/caf/militaryhistory/wars-operations/1812/laura-secord.html.

Government of Canada; Crown-Indigenous Relations and Northern Affairs Canada. "Indigenous Contributions to the War of 1812." *Government of Canada; Crown-Indigenous Relations and Northern Affairs Canada*, 13 June 2016, https://www.rcaanc-cirnac.gc.ca/eng/1338906261900/1607905474266.

"Isaac Brock - behind the Picture: Famous Faces of 1812: Multicultural Ontario Collections." *Vitacollections.ca*, https://vitacollections.ca/multiculturalontario/180/exhibit/8.

"John Norton and the War of 1812." *The Canadian Encyclopedia*, https://www.thecanadianencyclopedia.ca/en/article/john-norton-and-the-war-of-1812.

"Maj.-Gen. Sir Isaac Brock." *Brock University*, https://brocku.ca/about/maj-gen-sir-isaac-brock/.

"Military Gorget." *1812 History*, http://surl.li/fqjhz.

"Sword Shoulder-Belt-Plate Found in Canada." – *Works – New-York Historical Society*, http://surl.li/fqjgw.

"Treaty of Ghent." *The Canadian Encyclopedia*, https://www.thecanadianencyclopedia.ca/en/article/treaty-of-ghent.

"Uneasy Neighbors: Rival Political Systems in North America (U.S. National Park Service)." *National Parks Service*, U.S. Department of the Interior, https://www.nps.gov/articles/uneasy-neighbors-1.htm.

"Uniforms of the War of 1812." *Naval History and Heritage Command*, https://www.history.navy.mil/content/history/nhhc/our-collections/art/exhibits/conflicts-and-operations/the-war-of-1812/uniforms-of-the-war-of-1812.html.

"War of 1812 Facts." *American Battlefield Trust*, 28 Dec. 2021, https://www.battlefields.org/learn/articles/war-of-1812-faqs.

"War of 1812 Overview." *USS Constitution Museum*, 19 Mar. 2019, https://ussconstitutionmuseum.org/major-events/war-of-1812-overview/.

The War of 1812: Militia and Civilian Life, http://www.archives.gov.on.ca/en/explore/online/1812/militia.aspx.

Website, War of 1812. "The Provincial Marine at Amherstburg in the War of 1812." *Re*, https://www.warof1812.ca/provmarine.htm.

Website, War of 1812. "War of 1812 Glengarry Light Infantry Fencibles - History and Uniform." *Re*, https://www.warof1812.ca/gli.htm.

Libraries/Packages

GoodEnough. "IOS TTS (Native Speech Synthesizer): Integration ." *Unity Asset Store*,https://assetstore.unity.com/packages/tools/integration/ios-tts-native-speech-synthesizer-127543.

Technologies, Unity. "TextMeshPro." *Unity*, https://docs.unity3d.com/Manual/com.unity.textmeshpro.html.

"Ui Button Pack 2 - 1200 Button: 2d Icons." *Unity Asset Store*, https://assetstore.unity.com/packages/2d/gui/icons/ui-button-pack-2-1200-button-130422.

"Terrain Sample Asset Pack: 3D Landscapes." *Unity Asset Store*, https://assetstore.unity.com/packages/3d/environments/landscapes/terrain-sample-asset-pack-145808.

Z-Craft. “Fortress Gate: 3D Fantasy.” *Unity Asset Store*, https://assetstore.unity.com/packages/3d/environments/fantasy/fortress-gate-46895.

Icons used in UI

“Dollar Icons” *Icons*, https://icons8.com/icons/set/dollar.

“I Icons.” *Icons*, https://icons8.com/icons/set/i.

“Mail Icons.” *Icons*, https://icons8.com/icons/set/mail.

“Cross Icons.” *Icons*, https://icons8.com/icons/set/cross.

Fort George

Paul. “Model of Fort George in Niagara-on-the-Lake.” *Sketchup*, https://3dwarehouse.sketchup.com/model/446207e0365bbd6f17beee0900b442d6/Model-Of-Fort-George-in-Niagara-on-the-Lake?hl=en.

3D Model

Woodhead, A. (2020). Waterloo British Infantry [Brown Bess Gun]. Sketchfab Retrieved from https://sketchfab.com/3d-models/waterloo-british-infantry-for-animation-85a1a8205b2c447fb8d0419978f17b75

General Brock statue - Borrowed from makerspace at Brock University

Camera Movement & Tracking Scripts

Punni, Dinesh, director. *AR Foundation & Unity 03: Image Tracking*. *YouTube*, YouTube, 19 Aug. 2019, https://www.youtube.com/watch?v=o_z_Eb8Yh2g. Accessed 30 Apr. 2023.

Blender

Aschlauch, director. *YouTube*, YouTube, 16 Sep. 2021, https://www.youtube.com/watch?v=izfyIBqVv14. Accessed 30 Apr. 2023.

Metashape (maybe makerspace if needed)

Vasilyan, Grisha, director. *Create a 3D Model from a Picture Using Agisoft Metashape*. *YouTube*, YouTube, 31 Oct. 2019, https://www.youtube.com/watch?v=wR1W1q_gnH0. Accessed 30 Apr. 2023.

Unity Scripts:

Omar, Ather. “Unity Tutorials: Drag to Rotate Object.” *YouTube*, YouTube, 2 July 2018, https://www.youtube.com/watch?v=kplusZYqBok&t=324s.

Dev Enabled, director. *AR Foundation Improved Image Tracking - Multiple Objects/Images - Unity Augmented Reality/AR*. *YouTube*, YouTube, 5 Apr. 2020, https://www.youtube.com/watch?v=I9j3MD7gS5Y&t=490s. Accessed 30 Feb. 2023.

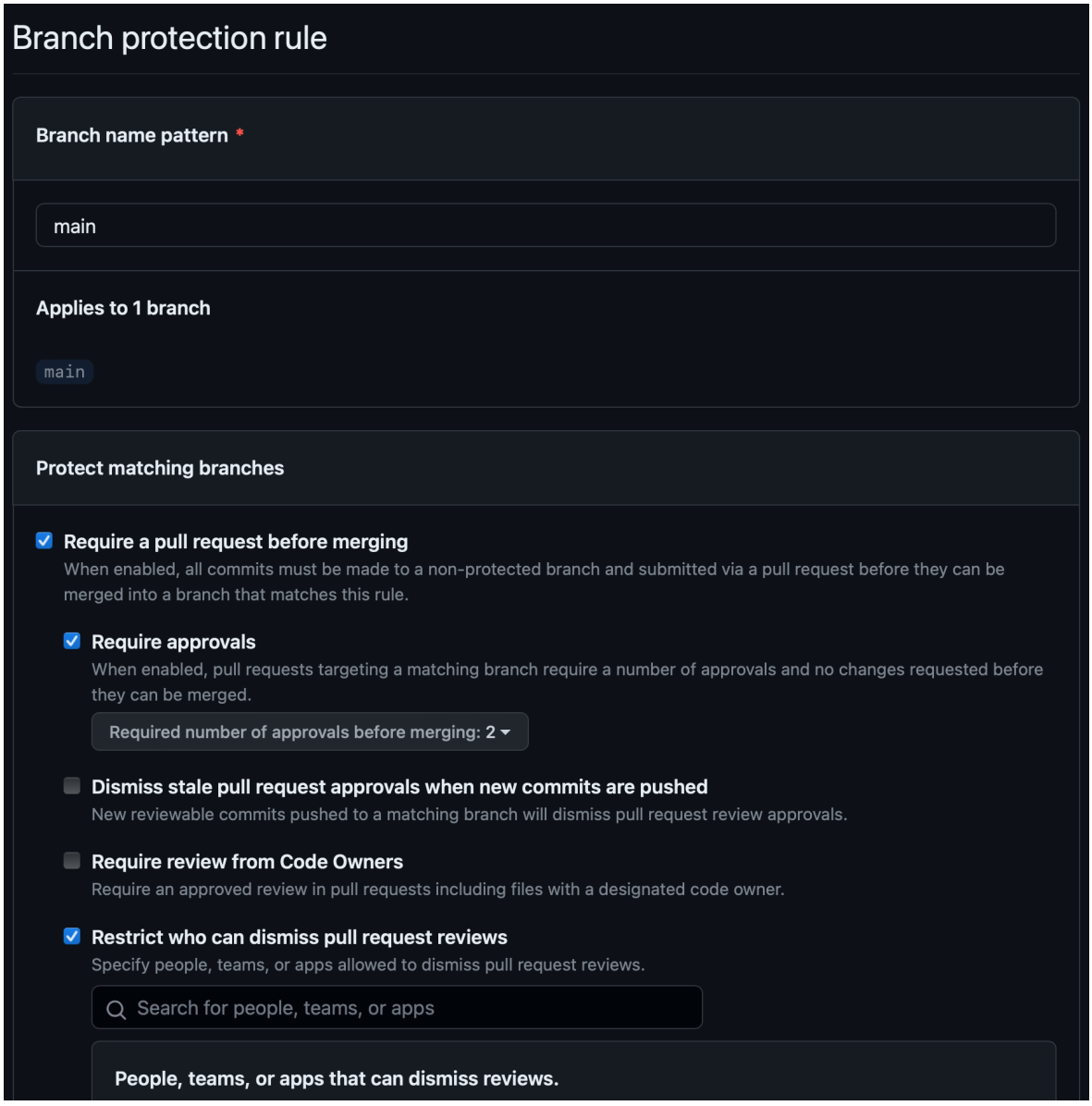
Technologies, Unity. “Unity User Manual 2021.3 (LTS).” Unity, https://docs.unity3d.com/Manual/index.html.

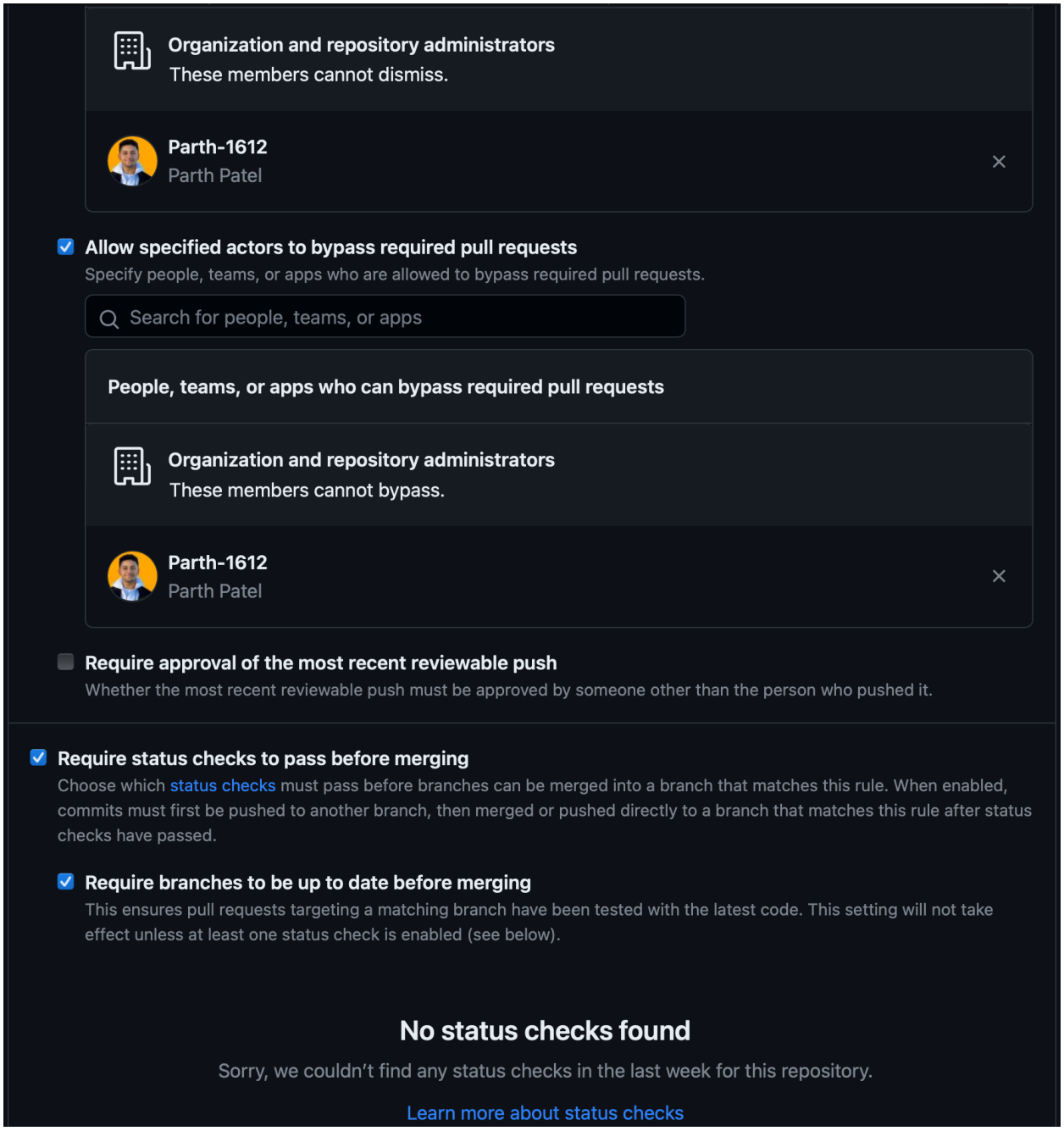
Test Documents/Result

While we do not have specific test documents or results to showcase, we have implemented several checks and safeguards throughout our development process to ensure the quality and stability of our app. One key measure we've taken is the protection of our master branch, which helps prevent accidental or unauthorized changes from being merged.

Additionally, we have structured our app by isolating different scenes in individual units. This modular approach allows us to quickly identify any scenes that could potentially break the app and pinpoint points of failure. By doing so, we can address issues more efficiently, resulting in a more stable and reliable application.

In summary, even though we don't have test documents or results to present, our development process has been designed with quality and stability in mind. By implementing branch protection and isolating scenes in separate units, we can ensure that our app remains robust and easily maintainable.


Branch protection rule
Branch name pattern *
main
Applies to 1 branch
main
Protect matching branches
Require a pull request before merging
When enabled, all commits must be made to a non-protected branch and submitted via a pull request before they can be merged into a branch that matches this rule.
Require approvals
When enabled, pull requests targeting a matching branch require a number of approvals and no changes requested before they can be merged.
Required number of approvals before merging: 2
Dismiss stale pull request approvals when new commits are pushed
New reviewable commits pushed to a matching branch will dismiss pull request review approvals.
Require review from Code Owners
Require an approved review in pull requests including files with a designated code owner.
Restrict who can dismiss pull request reviews
Specify people, teams, or apps allowed to dismiss pull request reviews.
Search for people, teams, or apps
People, teams, or apps that can dismiss reviews.


Organization and repository administrators
These members cannot dismiss.
Parth-1612
Parth Patel
Allow specified actors to bypass required pull requests
Specify people, teams, or apps who are allowed to bypass required pull requests.
Search for people, teams, or apps
People, teams, or apps who can bypass required pull requests
Organization and repository administrators
These members cannot bypass.
Parth-1612
Parth Patel
Require approval of the most recent reviewable push
Whether the most recent reviewable push must be approved by someone other than the person who pushed it.
Require status checks to pass before merging
Choose which status checks must pass before branches can be merged into a branch that matches this rule. When enabled, commits must first be pushed to another branch, then merged or pushed directly to a branch that matches this rule after status checks have passed.
Require branches to be up to date before merging
This ensures pull requests targeting a matching branch have been tested with the latest code. This setting will not take effect unless at least one status check is enabled (see below).
No status checks found
Sorry, we couldn't find any status checks in the last week for this repository.
Learn more about status checks

- [x] **Require conversation resolution before merging**
  When enabled, all conversations on code must be resolved before a pull request can be merged into a branch that matches this rule. Learn more.

- [ ] **Require signed commits**
  Commits pushed to matching branches must have verified signatures.

- [x] **Require linear history**
  Prevent merge commits from being pushed to matching branches.

- [ ] **Require merge queue** Beta
  Merges to matching branches must be performed via a merge queue.

- [ ] **Require deployments to succeed before merging**
  Choose which environments must be successfully deployed to before branches can be merged into a branch that matches this rule.

- [ ] **Lock branch**
  Branch is read-only. Users cannot push to the branch.

- [x] **Do not allow bypassing the above settings**
  The above settings will apply to administrators and custom roles with the "bypass branch protections" permission.

- [ ] **Restrict who can push to matching branches**
  Specify people, teams, or apps allowed to push to matching branches. Required status checks will still prevent these people, teams, and apps from merging if the checks fail.

Rules applied to everyone including administrators

- [x] **Allow force pushes**
  Permit force pushes for all users with push access.

  - ( ) **Everyone**
    Permit force pushes for all users with push access.
  - (•) **Specify who can force push**

Specify people, teams, or apps allowed to push to matching branches. Required status checks will still prevent these people, teams, and apps from merging if the checks fail.

Rules applied to everyone including administrators

- [x] **Allow force pushes**
  Permit force pushes for all users with push access.

  - ( ) **Everyone**
    Permit force pushes for all users with push access.
  - (•) **Specify who can force push**
    Only these people, teams, or apps are allowed to force push.

    Search for people, teams, or apps

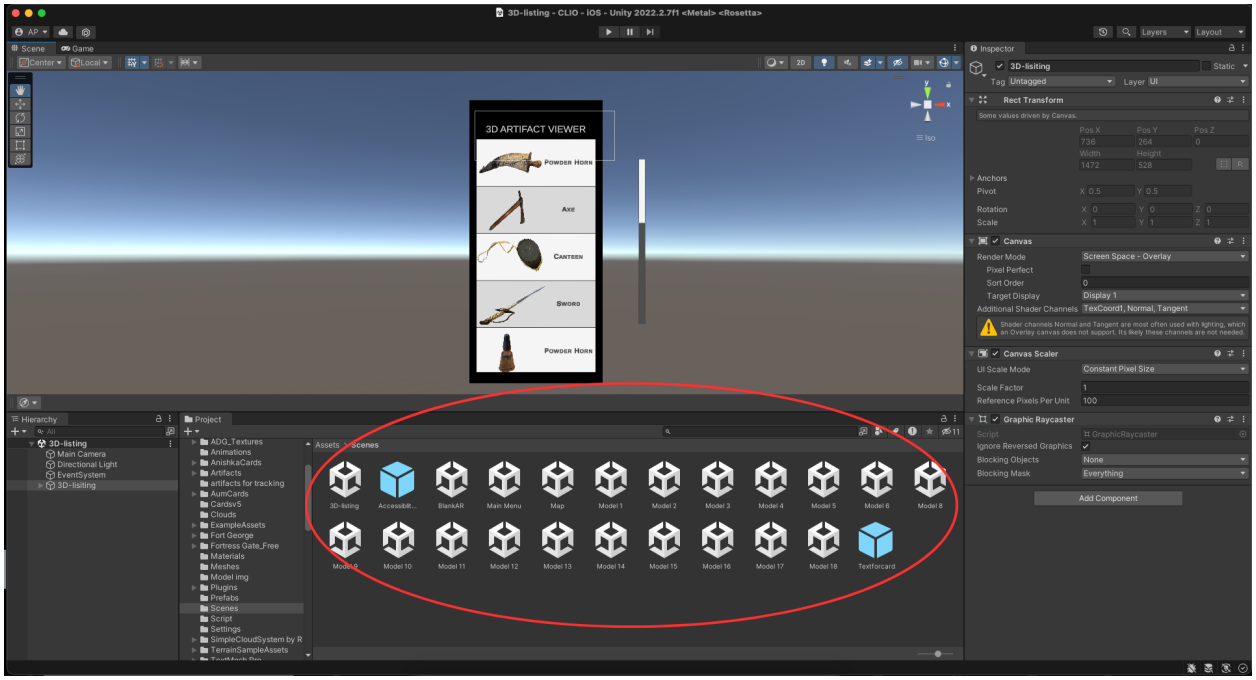
    People, teams, or apps who can force push

    Organization and repository administrators (automatic)

    
    Parth-1612
    Parth Patel ×

- [x] **Allow deletions**
  Allow users with push access to delete matching branches.

Save changes



As shown above, all of our scenes were isolated and tested separately so in any event that one scene might misbehave, all other scenes will remain unaffected. That way, we avoided problems in our already developed scenes due to addition of new scenes.

GitHub Logs

Note from Parth - Product Owner: I want to add that Git logs, which are essentially a record of commits, can provide valuable information on our project's development history. However, relying solely on git logs to evaluate contributions can lead to a flawed understanding of a team member's impact.

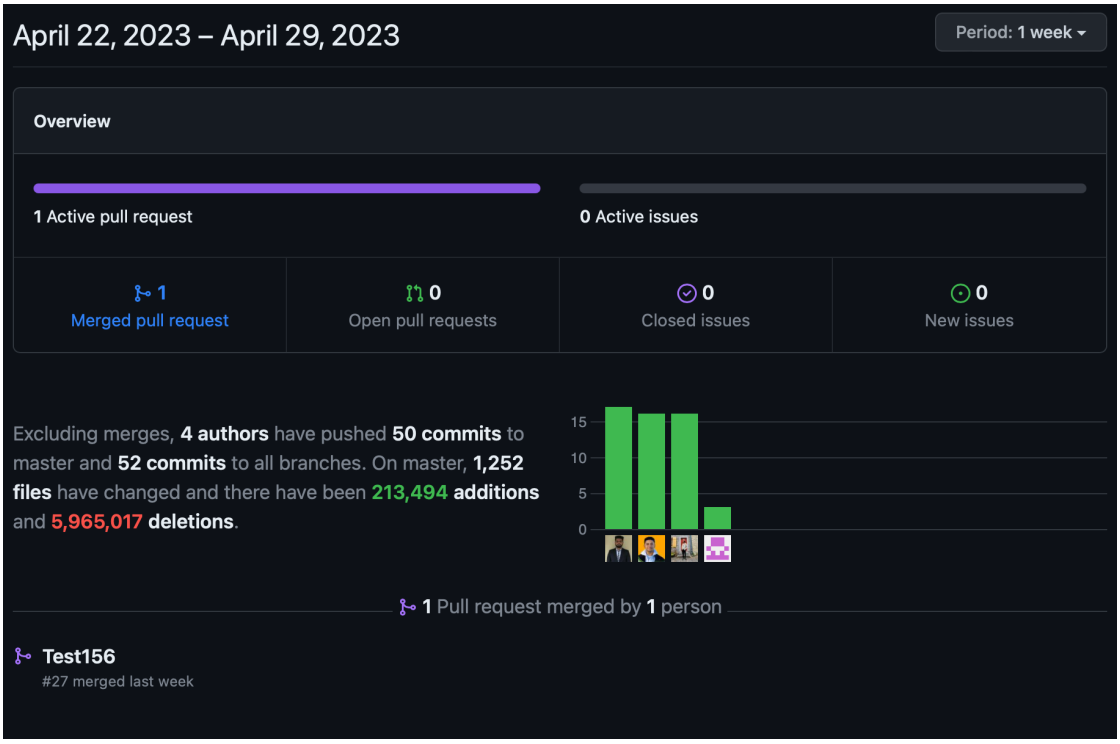
Commits do not always reflect the quality or complexity of the work. A single commit may contain a substantial, intricate change or a minor tweak, and it would be unfair to judge contributions based on quantity alone. Git logs do not account for non-code contributions, such as design, documentation, testing, or mentoring. These essential tasks contribute significantly to a project's success but might not be visible in commit history.

This is True for over team, where team members such Sneh and Darshak didn't have any commits to be seen on github but their work was really valuable and crucial for the App's success.

We also unfortunately had to cut off features that were no longer compatible with the free licenses we were using.Therefore a lot of work was never pushed to master or on github. But the time and effort invested in the app was worth everything.

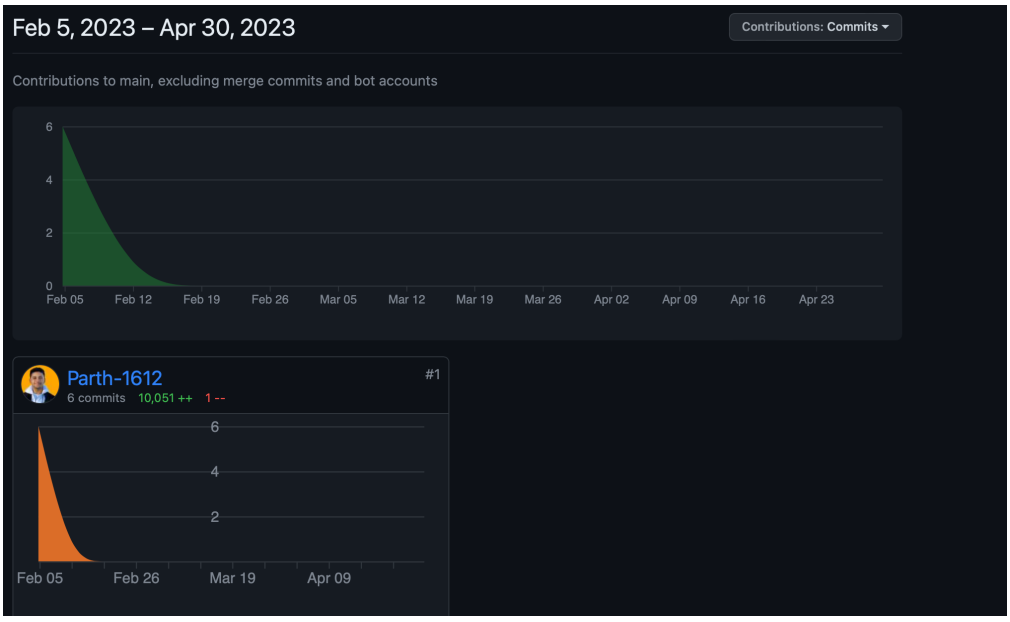
Moving forward I would love to have resources that are needed allocated to the teams, so that we can use industry standard tools without having to worry about compatibility or have to waste time implementing them ourselves.

CLIO - Our Main Unity Repo:



CLIO-React:

This was our initial approach but we had to trash this due to compatibility issues



Documents:

We store all our documents and resources for 3D models in this repo.