

# City Builder Game

---

team #31 Code Connoisseurs  
partner - Project: Human City

# Introducing the Problem

- The Problem: How do we get people to go outside and interact with their city?
- The Solution: Make a game that rewards players for interacting with their city.
  - Finding different kinds of real world objects
  - Maintaining a game city that utilizes principles of sustainable development
  - Teach players about pollution and environmentally friendly solutions

# About The Game

- Web-based city-building game
- Start all from scratch
- Acheived the MVP of the game
- Future development: Link with AR camera for real-world resource gathering, diverse building options.

# Partner Introduction

## Project: Human City



- Non-profit organization
- Focus on human inequality, social injustice and basic needs
- Initiatives for a more inclusive and equitable world

## Organization Leader:

- James Rhule
- jamesrhule@projecthumancity.com
- Leading initiatives for global impact and fostering collaborations for social change.

## Collaborative Members:

- Cheng-Ming Hsu: Spotstitch AR Camera Lead
- Dushyant Mehul Lunechiya: Frontend Product Lead
- Anupama Kadambi: Backend Development Lead
- Ali Hassan Amin: Spotstitch Frontend Developer

# Target Users – Understanding Diverse Player Types

Entertainment Enthusiasts: Enjoy building cities from scratch for fun. (Current Focus)

Professional City Designers: Utilize the game as a professional design tool.

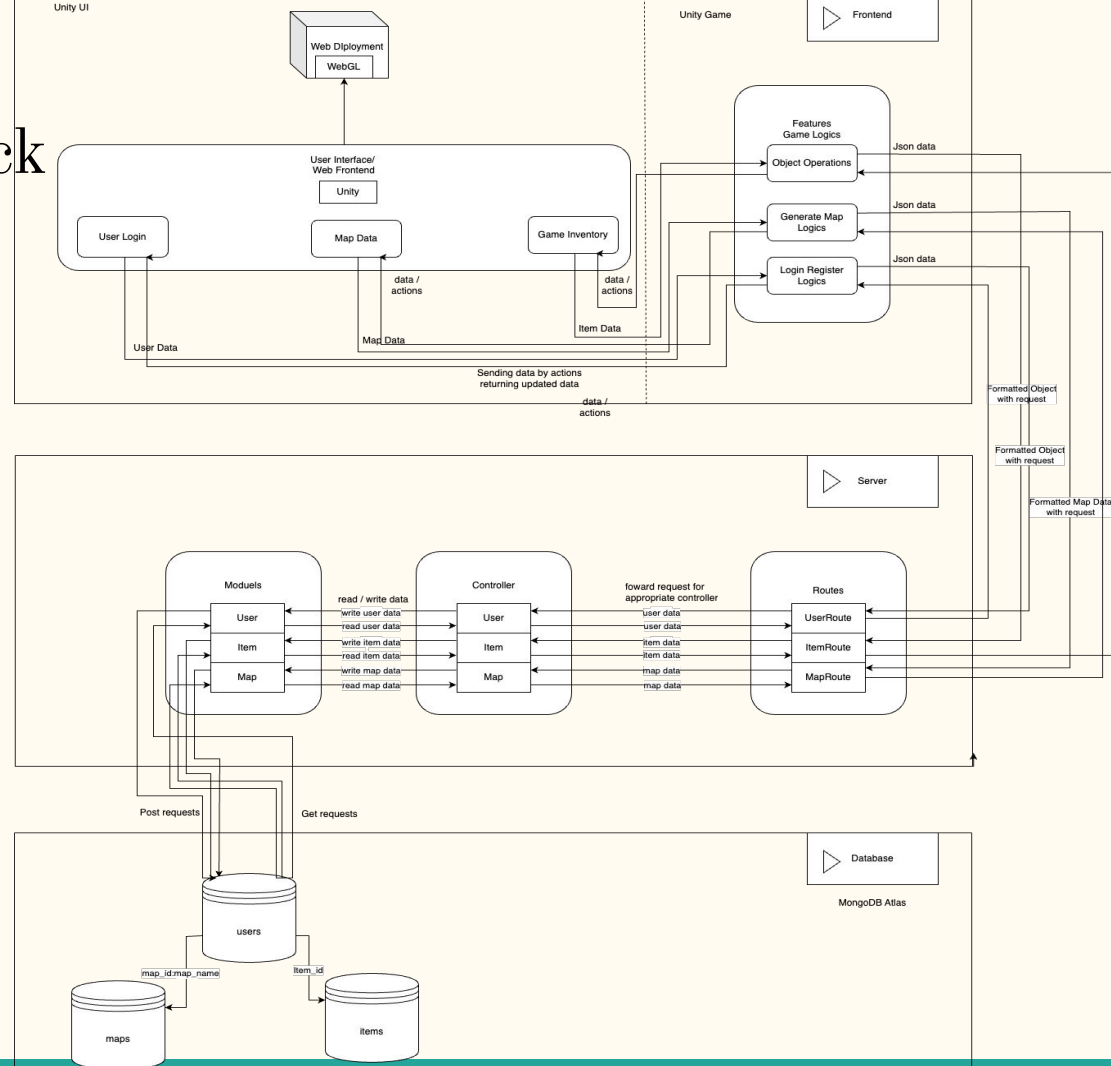


Spotstich App (App already developed by our partner) Users: Engage for socializing within the Spotstich community

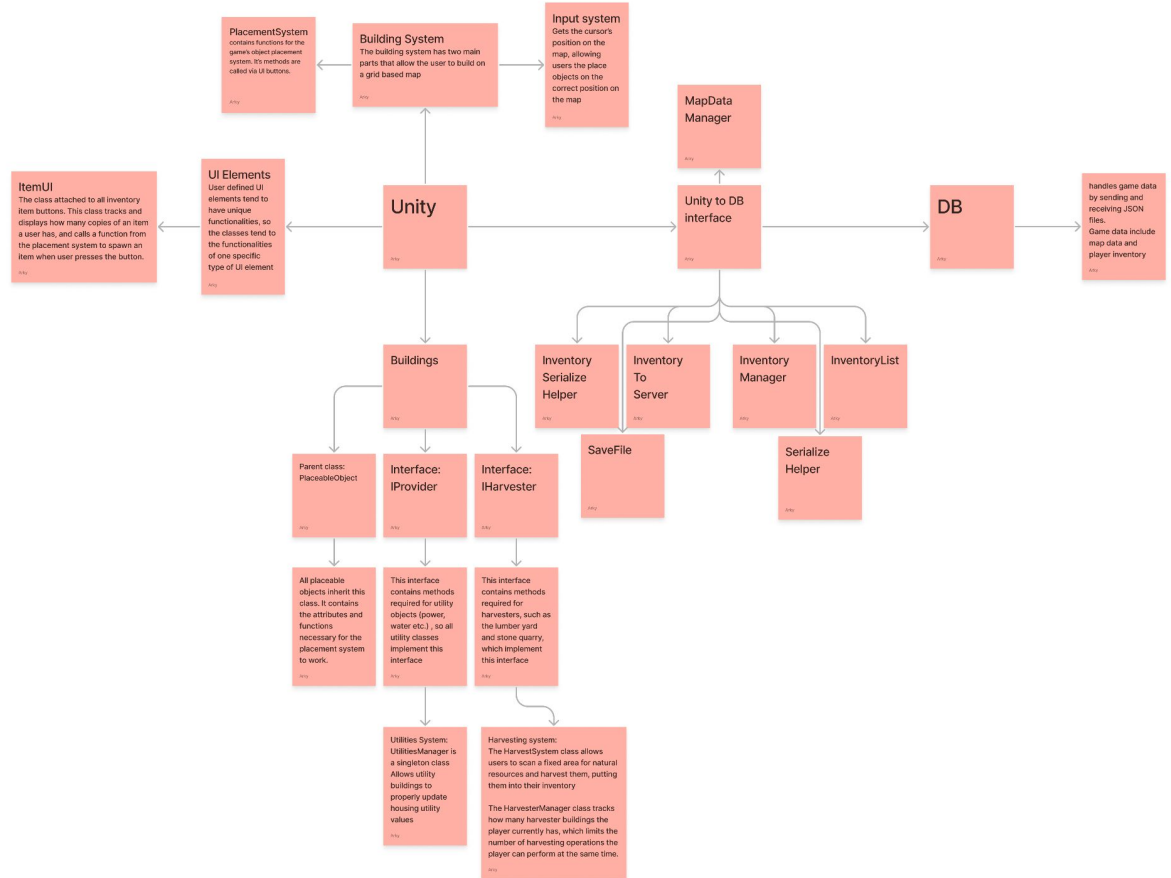
Demo

# Architecture & Tech Stack

- Frontend
  - Unity
- Server
  - Node.js
  - React
- Database
  - Mongo DB Atlas



# Unity Architecture Diagram





# Why Use MongoDB, Node.js, and Unity?

## Mongodb:

- Partner Collaboration
- Uniformity and Intergration



## Nodejs:

- Speed and Scalability
- Efficient Server Management
- Seamless Integration with MongoDB



## Unity:

- Partner Requirement
- Support for High-End Graphics
- Experienced Team Members



# Technical Discussion

## Coding styles & Clean Code Practices:

- coding and naming conventions
  - Google C# Style Guide, descriptive naming

```
7 public class PlacementSystem : MonoBehaviour
8 {
9     public int testInt = 10;
10    [SerializeField] private GameObject pointer;
11    [SerializeField] private InputManager inputManager;
12    [SerializeField] private GameObject[] placeableObjects;
13    public GameObject road;
14    public GameObject currentlyPlacing;
15    public GameObject currentlySelecting;
16    public GameObject currentlyHovering;
17    public GameObject gameCanvas;
18    public GameObject objectMenu;
19    public GameObject objectMenuPrefab;
20    public GameObject objectMenuHousingPrefab;
21    public CameraController cameraController;
22    public MenuManager menuManager;
23    public InventoryManager inventoryManager;
24    Vector3 currentRotation;
25    Vector3 oldPosition;
26    Vector3 oldRotation;
```

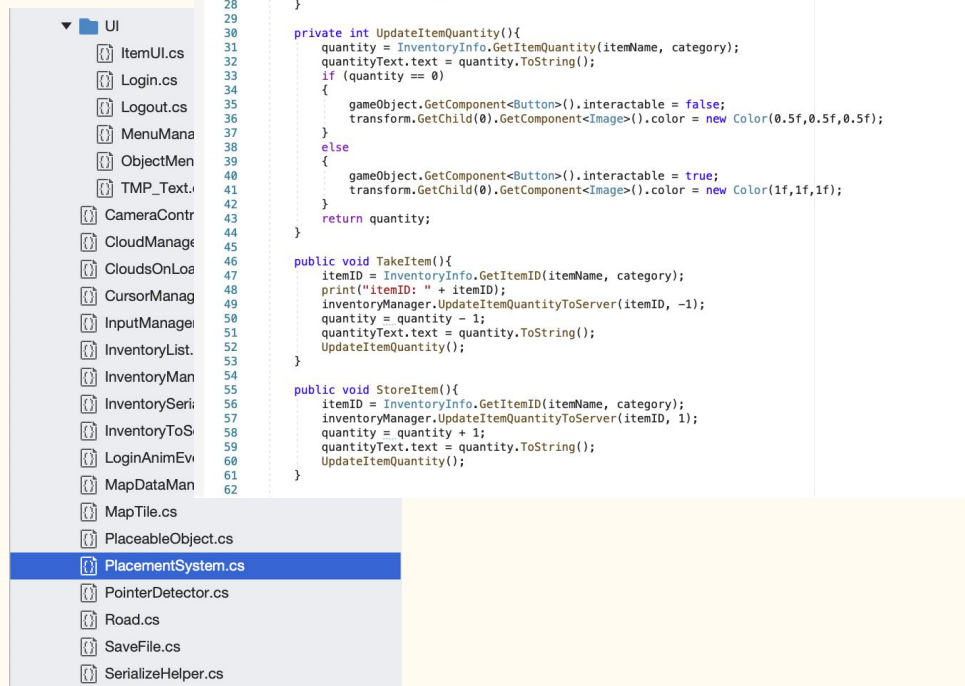
```
274 public void DrawStructureObjects(MapSerialization mapObjs)
275 {
276     // Redraw buildings and roads
277     foreach (var structure in mapObjs.structureObjData)
278     {
279         foreach (var placeableObj in inventory.inventoryList)
280         {
281             if (structure.name.IndexOf(placeableObj.name) != -1)
282             {
283                 if (structure.name.IndexOf("Road") != -1)
284                 {
285                     // Create road object
286                     GameObject roadObj = Instantiate(placeableObj);
287                     roadObj.transform.position = structure.position.GetValue();
288                 }
289                 else
290                 {
291                     // Create building object
292                     GameObject building = Instantiate(
293                         placeableObj,
294                         structure.position.GetValue(),
295                         Quaternion.Euler(structure.rotation.GetValue()));
296                     // Update colliding tiles and building state
297                     building.transform.parent = null;
298                     foreach (
299                         GameObject tile in building
300                         .GetComponent<PlaceableObject>()
301                         .GetCollidingTiles())
302                     {
303                         tile.GetComponent<MapTile>().isOccupied = true;
304                         tile.GetComponent<MapTile>().placedObject = building;
305                     }
306                     inputManager.placementLayerMask =
307                         LayerMask.GetMask("Ground") | LayerMask.GetMask("Foreground");
308                     building.GetComponent<PlaceableObject>().isHovering = false;
309                     building.GetComponent<PlaceableObject>().hasBeenPlaced = true;
310                 }
311             }
312         }
313     }
314 }
315 }
316 }
317 }
```

# Technical Discussion

## Coding styles & Clean Code Practices:

### - Modularity

- breakdown into modular files & classes
- clear responsibilities for each functions



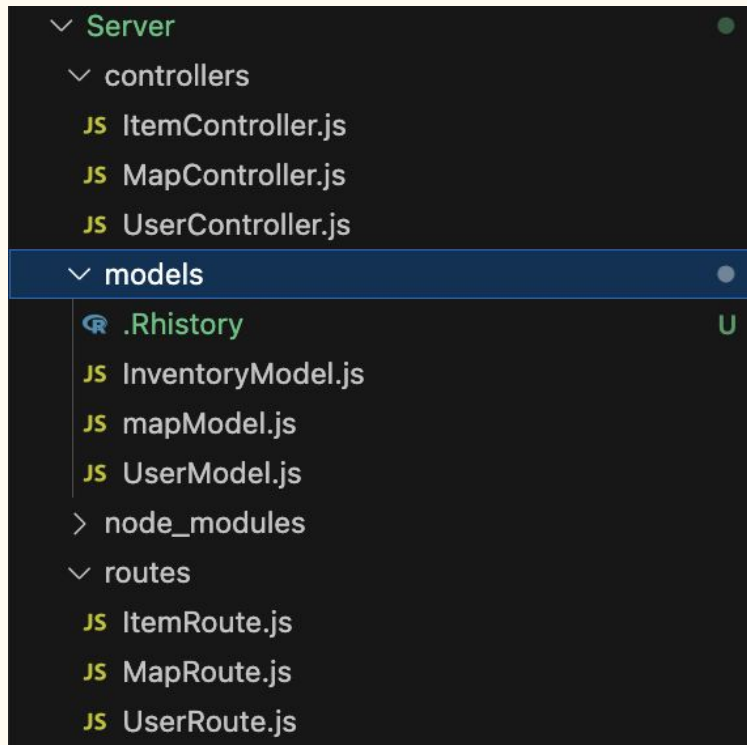
# Technical Discussion

## Coding styles & Clean Code Practices:

- **Modularity**
  - breakdown into modular files & classes
  - clear responsibilities for each functions

e.g: For server part using Node.js

- models: define schema for mongodb
  - map, user, inventory
- controllers: functions that handle different requests
  - map, user, inventory
- routes: provides API endpoint to trigger functions in controller
  - map, user, inventory



# Technical Discussion

## Coding styles & Clean Code Practices:

- Error Handling:

```
47  ✓  userSchema.statics.findByCredentials = async function (email, password) {  
48      const user = await User.findOne({ email });  
49      if (!user) throw new Error('invalid email or password');  
50  
51      const isMatch = bcrypt.compare(password, user.password);  
52      if (!isMatch) throw new Error('invalid email or password');  
53      return user  
54  }  
55  
56  
57  const User = mongoose.model('User', userSchema);  
58  
59  
60  module.exports = User;
```

# Technical Discussion

## Coding styles & Clean Code Practices:

- Optimization

- e.g.

- expand inventory easily

```
4 public class inventoryItem{
5     public string name { get; set; }
6     public int quantity { get; set; }
7     public string itemID { get; set; }
8 }
9
10 public static class InventoryInfo
11 {
12     public static Dictionary<string, Dictionary<string, inventoryItem>> itemInventoryDict = new Dictionary<string, Dictionary<string, inventoryItem>>();
13     public static bool isNew = true;
14
15     public static int GetItemQuantity(string itemName, string category)
16     {
17         if (InventoryInfo.itemInventoryDict.ContainsKey(category))
18         {
19             if (InventoryInfo.itemInventoryDict[category].ContainsKey(itemName))
20             {
21                 //Debug.Log("get quantity of item: " + itemName + "in category: " + category);
22                 return itemInventoryDict[category][itemName].quantity;
23             }
24         }
25     }
26 }
```

```
const user = await User.create({name, email, password});
const item1 = await Item.create({userID: user._id, quantity:2, category: "housing", name: "singleHouse"});
const item2 = await Item.create({userID: user._id, quantity:2, category: "energy", name: "coalPlant"});
const item3 = await Item.create({userID: user._id, quantity:2, category: "energy", name: "windTurbineGenerator"});
const item4 = await Item.create({userID: user._id, quantity:2, category: "energy", name: "solarEnergyPlant"});
const item5 = await Item.create({userID: user._id, quantity:2, category: "water", name: "waterTower"});
const item6 = await Item.create({userID: user._id, quantity:2, category: "sewage", name: "sewageTreatment"});
```

```
const item7 = await Item.create({userID: user._id, quantity:0, category: "resource", name: "wood"});
const item8 = await Item.create({userID: user._id, quantity:0, category: "resource", name: "stone"});
const item9 = await Item.create({userID: user._id, quantity:0, category: "resource", name: "metal"});
```

```
user.items.push(item1._id);
user.items.push(item2._id);
user.items.push(item3._id);
user.items.push(item4._id);
user.items.push(item5._id);
user.items.push(item6._id);
```

```
user.items.push(item7._id);
user.items.push(item8._id);
user.items.push(item9._id);
```

# Technical Discussion

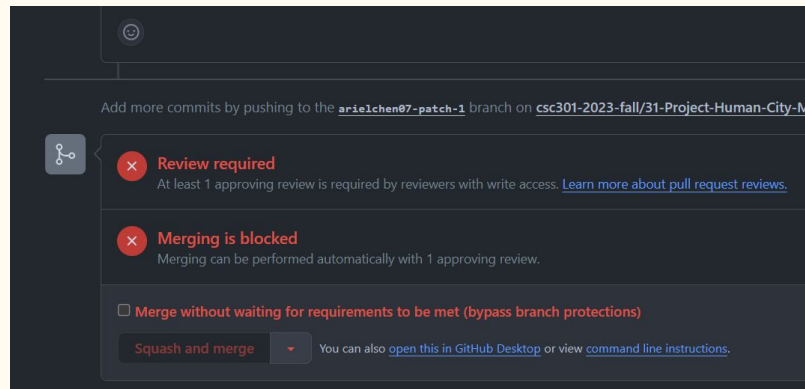
## Coding styles & Clean Code Practices:

- **Documentations:**
  - **README**
  - Game Design Document
  - Technical Document

# Process

Worked well:

- Pull request needs review from other



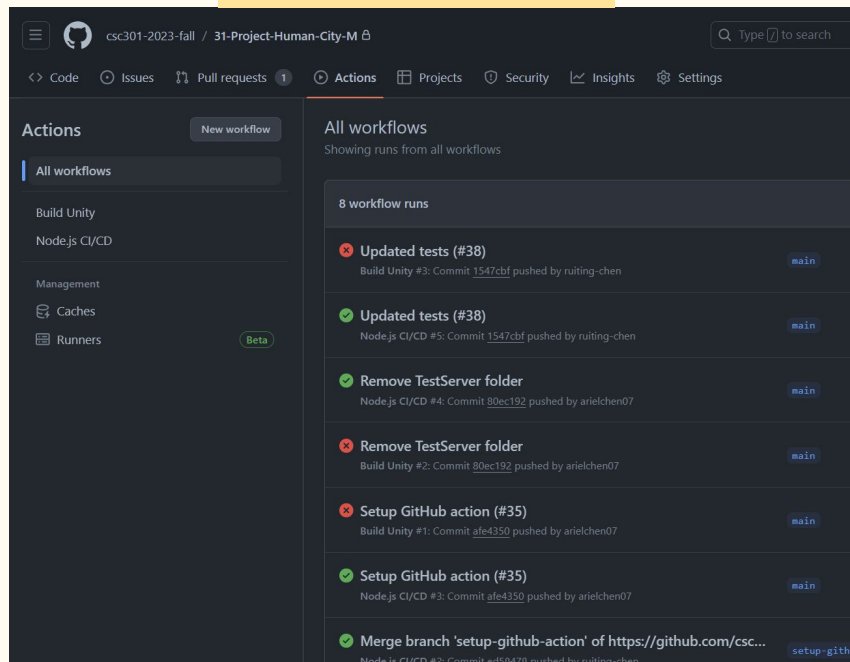
Improvements:

- Ambiguous deadlines for tasks
  - Solution: Use project management tools: ie. Jira, setup sprints, estimation of deadline



# Deployment

## GitHub Actions



- Triggered on each commit onto Main branch (ignore some files ie. README updates)
- Automated testing, then deployment on success test
- Frontend (Unity): desktop exe
- Backend (server): Render

# Server Deployment Process

31-Project-Human-City-M / 31-Project-Human-City-M

<> Code Issues Pull requests **Actions** Projects Security Insights Settings

← Node.js CI/CD

✓ Setup GitHub action (#35) #3

Summary

Jobs

- ✓ Test (20x)
- ✓ Deploy

Run details

Usage

Workflow file

Triggered via push 1 hour ago

arielchen07 pushed · afe4350 main Success 53s

server\_deploy.yaml

on: push

Matrix: Test

- ✓ 1 job completed
- ✓ Deploy 5s

Show all jobs

Auto Test → Auto Deploy

31-Project-Human-City-M / .github / workflows / server\_deploy.yaml

Code Blame 51 lines (43 loc) · 1.27 KB Your organization can pay for GitHub Copilot

```
1 name: Node.js CI/CD
2
3 on:
4   push:
5     branches: [ "main" ]
6   paths-ignore:
7     - 'README.md'
8     - 'deliverables/**'
9     - 'Unity/**'
10    - 'Builds/**'
11    - '.github/workflows/unity_build.yml'
12
13 jobs:
14   test:
15     name: Test
16     runs-on: ubuntu-latest
17
18     strategy:
19       matrix:
20         node-version: [20.x]
21
22     steps:
23       - name: Checkout
24         uses: actions/checkout@v3 # was "v2" before
25         with:
26           fetch-depth: 0
27
28       - name: Use Node.js ${{ matrix.node-version }}
29         uses: actions/setup-node@v3 # was "v2" before
30         with:
31           node-version: ${{ matrix.node-version }}
32
33       - name: NPM install, build and test
34         working-directory: ./Server
35         run: |
36           npm install
37           npm run build --if-present
38           npm test
39
40 # Heavily modified deploy job to fit render.com
41 deploy:
42   name: Deploy
43   needs: [test] # Our tests must pass in order to run the deploy job
44   runs-on: ubuntu-latest
45
46   steps:
47     - name: Deploy to production
48       uses: johnbeynon/render-deploy-action@v0.0.8
49       with:
50         service-id: ${{ secrets.SERVICE_ID }} # Can be found as part of the
51         api-key: ${{ secrets.RENDER_API_KEY }} # Create your API key in Render
```

# Server Deployment Process

## Auto Test

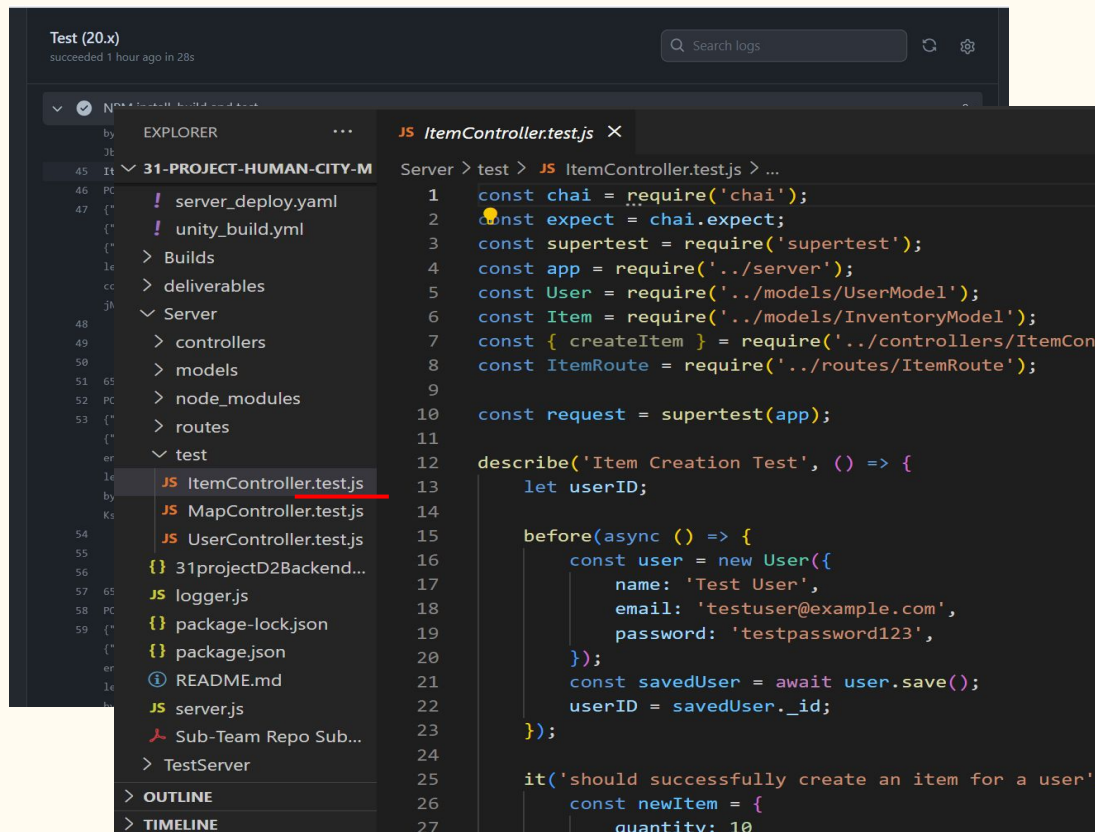
```
test:
  name: Test
  runs-on: ubuntu-latest

  strategy:
    matrix:
      node-version: [20.x]

  steps:
    - name: Checkout
      uses: actions/checkout@v3 # was "v2" before
      with:
        fetch-depth: 0

    - name: Use Node.JS ${ matrix.node-version }
      uses: actions/setup-node@v3 # was "v2" before
      with:
        node-version: ${ matrix.node-version }

    - name: NPM install, build and test
      working-directory: ./Server
      run: |
        npm install
        npm run build --if-present
        npm test
```



# Server Deployment Process

## Auto Deploy

```
deploy:
  name: Deploy
  needs: [test] # Our tests must pass in order to run
  runs-on: ubuntu-latest

steps:
  - name: Deploy to production
    uses: johnbeynon/render-deploy-action@v0.0.8
    with:
      service-id: ${ secrets.SERVICE_ID } # Can be found in the Render dashboard
      api-key: ${ secrets.RENDER_API_KEY } # Create a new API key in the Render dashboard
```

The screenshot displays the Render dashboard for a service named 'unity-game-server'. The top navigation bar includes links for Dashboard, Blueprints, Env Groups, Docs, Community, and Help, along with a 'New +' button and a user profile for 'Ruiting Chen'. The main content area shows the service details, including its URL 'https://unity-game-server.onrender.com', a 'Node' environment, and a 'Free' instance type. A 'Connect' button and a 'Manual Deploy' button are visible. Below this, a sidebar on the left lists various management options: Log, Events, Logs, Disks, Environment, Shell, Previews, Jobs, Metrics, Scaling, and Settings. The 'Logs' tab is currently selected, displaying a list of logs with a search bar and a 'Live tail' button. The log entries show HTTP GET requests to the service, with details such as response time, status code, and headers. A warning message at the top of the logs section states: 'Free instance types will spin down with inactivity. Upgrade to a paid instance type to prevent this behavior. Learn more.'

# Unity Deployment Process

```
Code Blame 63 lines (60 loc) · 1.8 KB Your organization can pay for GitHub Copilot

1  name: Build Unity
2
3  on:
4    push:
5      branches: [ "main" ]
6      paths-ignore:
7        - 'README.md'
8        - 'deliverables/**'
9        - 'Server/**'
10       - 'Builds/**'
11       - '.github/workflows/server_
12
13  env:
14    UNITY_EMAIL: ${ secrets.UNITY_EMA
15    UNITY_PASSWORD: ${ secrets.UNITY_
16    UNITY_SERIAL: ${ secrets.UNITY_SE
17
18  jobs:
19    buildForAllSupportedPlatforms:
20      name: Build for ${ matrix.target
21      runs-on: ubuntu-latest
22      # This steps builds that are in_
23      concurrency:
24        group: unity-build
25        cancel-in-progress: true
26      strategy:
27        fail-fast: false
28        matrix:
29          projectPath:
30            - './Unity/CityBuilder
31          targetPlatform:
32            - StandaloneWindows64 # Bu
33          testMode:
34            - './Unity/
35      steps:
36        - name: Checkout
37          uses: actions/checkout@v4
```

Triggered via push 2 days ago Status Success Total duration 4m 47s

ruiting-chen pushed · 1118a5a main

EXPLORE

31-PRO.

unity\_build.yml

on: push

Matrix: buildForAllSupportedPlat...

1 job completed

Show all jobs

Artifacts

Produced during runtime

Name	Size
Build-StandaloneWindows64	78.1 MB

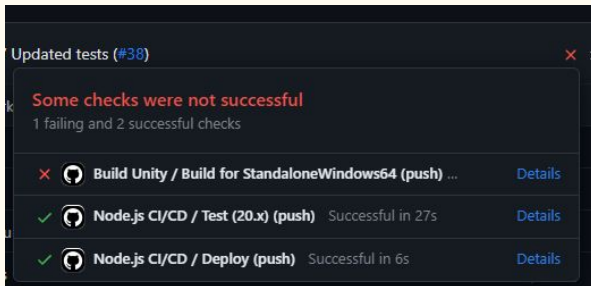
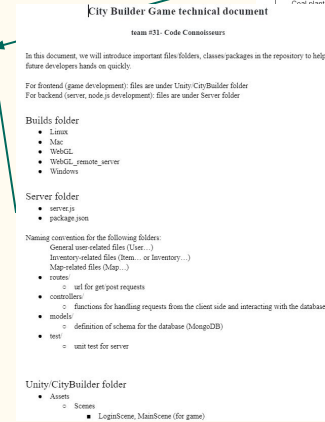
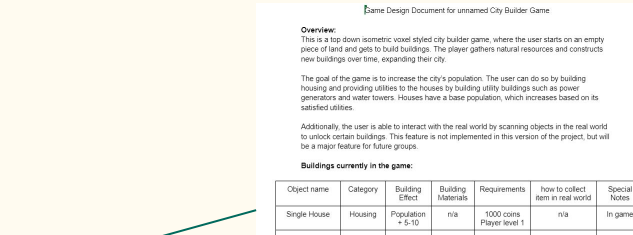
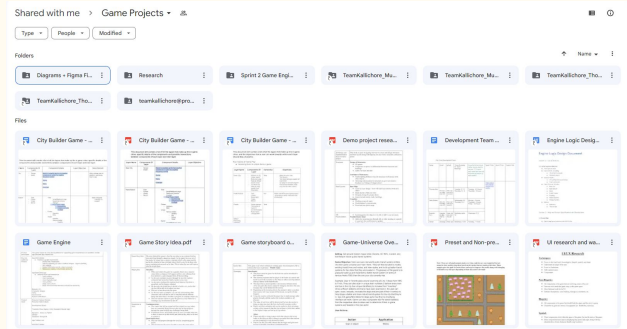
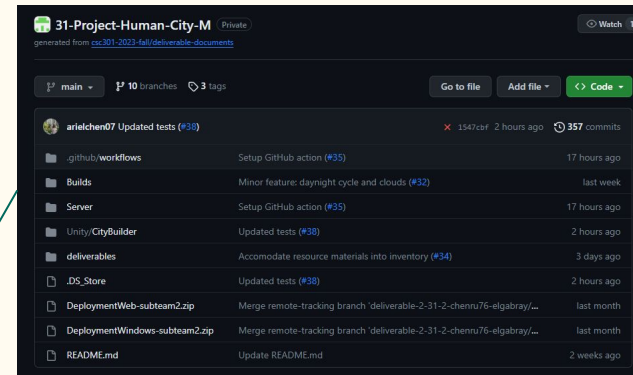
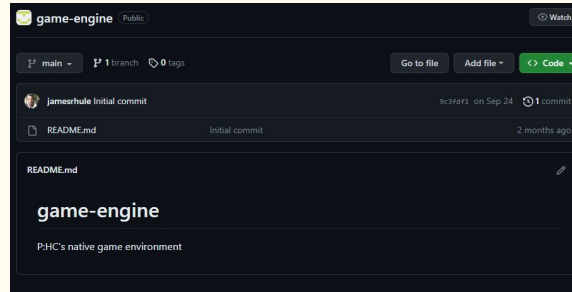
# Deployment & Accessing application

- Access the project
  - Access through GitHub: Clone '31-Project-Human-City-M' repository.
  - Executables for different OS in 'Builds' folder for easy access.
- Readme Guide & Partner Instructions
  - One-stop guide for deployment and development.
  - Covers user login/signup, map generation, city design.
  - Outlines automatic save/load system and future AR features.
  - Detailed server running instructions
  - Access game via webpage with supporting demos and guides
- Development Requirements
  - Technology setup guide for macOS and Windows in Readme
- Deployment and Workflow
  - current deployment via itch.io
  - structured GitHub workflow with guidelines for contributions
- Next Step
  - Ongoing communication with partners

# Accessing application

## Hand-off plan:

- Clone repo to organization:
  - Code follows guidelines
  - Simplified deployment
  - README
- Upload documentation to drive:
  - Game Design Document
  - Technical Document
  - Architectural Diagrams
  - Demo video
- Deployment accounts by email





# Working Process

## Monday meeting

- ref
- ass
- testing
- que

For Server

For Datab

## ▼ D3

Ricky

☐ Inventory

☐ Item mer

## Updates to partner:

- meeting
- Slack message

Ricky: road placement and login screen revamp #31

Merged YahyaElgabra merged 6 commits into main from ricky-road-placement last week

Conversation 2 Commits 6 Checks 6 Files changed 34

PyroPara commented last week

Previously, road placement was done via pressing the R key. Implemented road placement via button to be consistent with the rest of the game, and removed the R key functionality. Revamped the login screen. All functionalities remain the same, only UI changes.

PyroPara added 3 commits last week

- Updated login scene
- Revert "Revert "Implemented road placement and deletion""
- Merged Login screen changes

PyroPara commented last week

@ariel @ruiting-chen @YahyaElgabra @Krystal-Miao @Tommy-Gong This is the new PR. The old one has been deleted. Would appreciate if someone could review this.

PyroPara added 3 commits last week

- R1non UI change
- Revert "R1non UI change"
- R1non UI change

YahyaElgabra approved these changes last week

login screen looks much better, and UI changes are great as well, well done!

## During week:

Slackbot 9:00 AM

Reminder: group meeting from 8-9 pm Monday in breakout room 31 of tutorial Zoom  
<https://utoronto.zoom.us/j/83875450377>

Pinned by Sirui (Ariel) Chen

@channel TO DO:

1. Refine game design document:  
<https://docs.google.com/document/d/1ZVos2YGsZgIYH5dUNQezWZ9ePwc4>
2. Write tech document  
<https://docs.google.com/document/d/18tfmNHeAGg8C2FyqJkVoZzyBlvluCO>
3. Readme update - Auto deploy + auto testing + organizing structure
4. Presentation slides + prepare presentation draft for your own part (please finish after noon)  
[https://docs.google.com/presentation/d/1DJP\\_iaWAKommQnX8bN=id.p](https://docs.google.com/presentation/d/1DJP_iaWAKommQnX8bN=id.p)
5. Presentation recording (will record at 6pm, due 11.30 11:59pm)

(edited)

## Task completion:

- Pull Request reviews, at least one other member to approve to merge



# Reflection

- **What did we learn?**

- **technical skills**

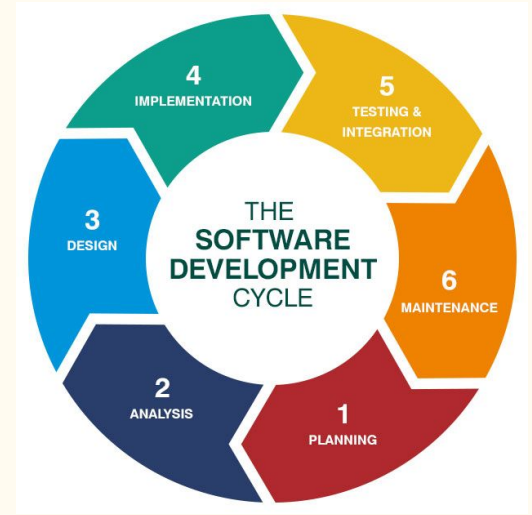
- Unity
    - Node.js,

- **soft skill**

- communication with partner
    - team organization

- **Takeaway message:**

- Software engineering is more than just coding; it's a holistic approach that involves user experience considerations, the art of seamlessly connecting diverse software components and deployment strategies.



# Reflection

- **What would we do differently next time?**
  - **Unity version control or alternatives**
    - Consistent merging problem in github throughout the whole project
  - **Team work**
    - use project management tools: e.g. Jira
    - setup sprints
    - estimation of deadline



**Unity DevOps**  
Version Control

# Individual Contribution

Sirui Chen - Team Organizer, Software developer (Unity): Inventory Manager scripts and connections to UI display, resource inventory manager, setup automated testing Github actions for Unity

Ruiting Chen - Save/load map from Unity to server, Update/Load inventory and resource item from Unity to Server, Setup deployment GitHub actions for server and Unity, Setup Render service for server deployment

Krystal Miao- Backend User registration&login , map creation&deletion. Unity login&register system(including UI and connection to server)

Tommy Gong - Backend Inventory, item creation, increment & decreament. Unity login loadmap logic( requestes between Unity and server)

Yahya Elgabra - Save/load map and decorations between Unity and server, Generate new maps, Integrate decorations into maps, Implement harvest system

Ricky Wen- Partner Liason, Unity Developer, Game Designer: All models and UI sprites, building system, inventory UI, all building prefabs and UI buttons. Login transition, settings UI, population logic and utilities system, login scene background, clouds and day-night cycle system