Major Goals (Completed and Not Completed)

Completed

Major Goal 1: Establish a basic webpage that can talk to the server. Also establish a way to be able to create a hunt and display it.

Setting up the website is the main goal for the sprint and establishes the look of the overall website. Also, the website should just display a simple list of tasks for the hunt.

Goal 1 Tasks:

- Establish the look and feel of the website.
- Assign the name of the hunt.
- Come up with the 10 tasks (Location, Question and Answer)
- We need some kind of file to store the tasks and info for them.
- Create the website (homepage (login, registration), hunt page)

NOT Completed

Major Goal 2: Capture Player entries and track/display progress

Main goal for this sprint is to make sure that a user can update the task list. The list should allow for a user to submit valid answers for each task that task should have a location, question, and answer.

Tasks:

- Making sure email and phone numbers are valid and are not duplicated.
- Figure out how to track individual user progress on tasks.
- Store the locations/questions within the database.
- Validation on answers, display if answer is wrong or correct.
- After each answer a check mark or a way to show that it was the correct answer button.

Major Goal 3: Create Admin console on the server to CRUD Player information and send out invites.

Tasks:

- Establish a way to send an email to players to tell them they have access to the hunt.
- Allow users to sign into the hunt with their access code.
- The task list then would populate with the user's information via their access code.

The rest of the main goals did not list the tasks, but only the user stories so I listed those here instead.

Major Goal 4: Add QR code-based answers.

High Priority:

1.) If the player has a camera enabled, they can hit "scan QR Code" to read in the QR code symbol located at the hunt location.
    - Figure out how to setup the ability to QR scan from a device.
    - Establish what type of device can be used to scan a QR code.
    - Attach a QR code to each task.

- Come up with a unique QR code.
- Print the QR code once it is created.
- What happens if a random person scans a QR code that's not attached to a hunt?

2.) Alternatively, the player can enter the text that appears below the QR code symbol located at the hunt location.
    - Add the box that would take the text below the QR text.
    - Make unique codes for each QR Code
    - Attach each unique code to each task.

Major Goal 5: Add Lat/Long to location sets, add a map to the web site, show locations on the map.

High Priority:

1. If the player has location enabled, they can hit "I am here" to compare their location to the lat/long associated with the list of locations in the hunt (must be within 50 feet)

2. As a Player, I want to be able to see, on a map, which task locations I have visited and which ones I haven't so that I can determine where to go next.

Server Setup (General Guide with some instructions)

ETSU provides a windows server 2019.

Setting up and distributing access to the server:

We had 3 different VM's (servers one for each team) from there if you want you can add different accounts for different users to access the server.

- Windows start > computer management > (left side) local users and Groups.
- From here you can right-click and add users and groups.
- Make sure that you add users to the proper other groups they need in order to perform the proper tasks. For example, PM and Scrum Master might need to be administrators, and all users that have access will probably need to be added to the remote user's group as well.
- This is done by clicking on the user > selecting the members of tab > select add > and enter the name of the group you want to add them to > click apply.
- Also, by clicking on the group > add > entering members to add.
- If a user is unable to access the remote server, they may need to modify the firewall settings on their user device.
- If there is NO static IP for the server then all end users will NEED to be connected to ETSU Wi-Fi in order to access the server.
- To access the server from the end user device:
    - Start > search RDC (remote desktop connection) from here enter either the server's name (provided to you by ETSU), or if a static IP exists the static IP can be entered.
    - Enter the username and password for the account on the server that will be accessed, admin/user/ (these are set up in the previous step).

- You should have access.

Source control to deploy code to the server. (Linking w/ GitHub and other pcs for file transfer)

This can be handled in several different ways.

- A shared folder can be created between a user's computer and the server. This is essentially a cloud folder that both can access.
- Click Server manager > files and store devices > shares > right-click in the white space and click new share.
- Walk through the process of creating the share folder and adding permissions (permissions will allow read, write, and execute) to different users.
- If you have issues writing to the folder from the user device navigate to the folder > right-click folder > properties > sharing > advanced sharing > permissions > change the access to everyone based on how you want to do it, I allowed full access to anyone.
- Downloading visual studio to the server itself.
- Once this is done you can link your GitHub account to visual studio and pull from the branch you wish to pull from (this is the much easier way we have found).

Once this is complete you can deploy the code in several different ways all done through IIS in the server.

This video helped greatly to walk through the process of deploying via IIS.

https://www.youtube.com/watch?v=dHBQqqIwTAI

Class Notes (How the pervious team managed their workflow and general note-taking)

Comms: Discord

Tasks: Trello

UI Standards:

Coding standards:

- Document attached.

Repository - GitHub

Def of Done

- Code review
- Happy path and unhappy.

Define UI standards to follow (e.g. display must fit on phones or computers; displays must resize automatically)

Define coding standards to follow.

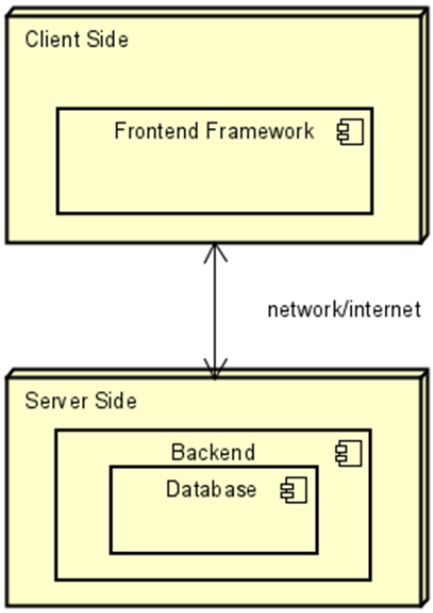
What is the architecture?

What is the production environment? Blazor visual studio

What is the development environment (GitHub, Trello? ide?)? GitHub

What is the language? C#

What is the technology stack? Visual Studio, SQL Express 2019



Define a Definition of Done (e.g. designed{e.g. sequence or interaction diagram}, built, reviewed {meets coding standards, fully implements design, follows good coding practices}, tested {happy path and all known unhappy paths}, committed {the code and the tests})

Define the team's sprint velocity (the duration is 1 week, 2 class periods plus at least 2 hours so that's a minimum of 6 hours each person on the development team each week; eg. 9*6=54).

Steps for each sprint:

- PO adds the user stories to the product backlog and the whole team grooms them based on priority.
- Sprint duration is 1 week; ends the last hour of the last class session of that calendar week.
- Sprint planning: Recurring 1st day of each sprint, time box for 15 minutes the first day of the sprint
    - Revisit sprint goal; update as necessary.
    - Select the Product Backlog user stories to be delivered in the sprint (includes a review/discussion of the stories/estimation)
        - Break them down as necessary to less than 4-hour tasks.
- Daily Scrum/Standup: Recurring daily during sprint, time box for 15 minutes.
    - Add a 16th minute for additional collaboration, time box for 15 more minutes and dismiss any team members not necessary for discussion.
- The last hour of the last class day of a sprint

- Backlog grooming: max 30 minutes, (not limited to the scope of a single sprint.)
    - meant to ensure the highest priority backlog items are detailed enough and estimated so they can be used in Sprint Planning.
    - Ideally, enough stories are detailed for at least 2 sprint's worth of work, so you have enough for the next sprint.
    - plus more if you can pull additional work into the next sprint)
- Sprint review: max 15 minutes; this is akin to acceptance testing with the customer/product owner.
- Sprint retrospective: Remainder of the last hour (min 15 minutes).
    - team discusses changes to daily routines, definition of done, ways to improve (e.g. training and or pair programming participants).

Backlog Grooming (at the end of a sprint):

- The product backlog should have already been prioritized by the PO going into this exercise.
- Do you have enough stories detailed out for the next sprint?
    - If yes, do enough for one more sprint or till time runs out.
    - If no, detail out at least enough for the next sprint (even if you go over on time some). The PO and SM should take action to improve the backlog.
- Talk through each story: (also see INVEST https://www.knowledgehut.com/tutorials/scrum-tutorial/user-stories)
- What are the words behind the words?
- Risks/Constraints/relationship to other stories in list
- What are the acceptance criteria (may only need this for the main story but helpful to have it for the subtasks/substories as well)
- Can we estimate to be < 4 hours (if no, keep discussing or break into multiple stories)
- Estimate it (in hours based on definition of done)

Sprint Planning (at the start of the next sprint)

- Revisit the sprint goal and adjust as necessary based on the highest priority product backlog items.
- Make sure all stories associated with the sprint's goal are estimated,
- Prioritize them within the sprint.
- Add stories to the sprint backlog until the target performance estimate is reached (and no more)

Sprint Review -

- PO/Scrum Master should have a list of acceptance criteria available based on the stories selected for the sprint during sprint planning.
- Step through each acceptance criteria; any feedback can be captured as a new task/story in the product backlog for the PO to prioritize.
- These then get pulled into the next sprint's planning if high enough priority.
- Once approved, deploy the software.

Sprint Retrospective -

- Everyone writes down up to 3 things the team did well and then discusses them (eliminating duplicates)

- Everyone writes down up to 3 things the team could do better and then discusses them (eliminating duplicates)
- The Scrum Master and the PO do not get to contribute but they do attend (this varies among Scrum implementations)
- The SM then tracks these items and helps the team remember them going forward.

ProductBacklog.xlsx (This table was included along with the other Word Documents they left)

User Story	Priority	Done	Sprint	Brokedown?
Assign a title and theme	1	Y	1	Y
For the first release, all tasks are locations the player must go to	1	Y	1	Y
A player's access code is unique to the hunt they are invited into	1	Y	2 and 3	Y
The player enters that code on the hunt URL page in order to play	1	Y	2 and 3	Y
Create a new account using a person's email address and phone number	1	Y	4	Y
Assign the list of tasks each individual player is to perform	1	N	NA	Y
Set the invitation text that is included in player invitations	1	N	NA	Y
Invite someone to participate in a hunt by sending the hunt's URL and a invitat	1	N	NA	Y
If the player has a camera enabled, they can hit "scan QR Code" to read in the (	2	N	NA	N
Alternatively, the player can enter the text that appears below the QR code sy	2	N	NA	N
If the player has location enabled, they can hit "I am here" to compare their lo	2	N	NA	N
As a Player, I want to be able to see, on a map, which task locations I have visit	2	N	NA	N