CS496 Software Project:
VR Puzzle Game

Joe Fielding and Kyle Richards

2026-02-25

1 Client Information

By sharing this client information and the rest of this document, you are stating that this client has provided this project as something they want (not something you created and asked if they wanted), and that they are interested in having you complete this project for your capstone.

- Client name: Hoang Bui
- Client title: Associate Professor
- Client email address: hdbui@loyola.edu
- Client employer: Loyola University Maryland
- How you know the client: We had him as a professor, have work with him, and he is our Competitive Programming Coach

2 Project Description

2.1 Overview

The goal of this project is to create a virtual reality (VR) puzzle game that can be played using the Meta Quest headset. This app will allow the user to take pictures of real-life objects, and then the app will generate a puzzle by breaking down the real-life object into pieces. The pieces could either be normal jigsaw puzzle shaped pieces or different shapes of the user's choosing. The user should also be able to adjust the difficulty of the puzzle by choosing the number of pieces. Finally, the user can import the puzzle into the VR world and attempt to solve it by fitting the pieces together by using their hand motions.

The problem that this project is trying to solve is that the issue with real jigsaw puzzles is that you must buy another one every time you want to solve a new one. This is impractical for Dr. Bui because he enjoys solving puzzles but does not have the time to keep going out and buying new ones. He wants to be able to generate new puzzles quickly and then solve them using his Quest headset. This app will solve this issue because Dr. Bui can take a picture of any real object of his choosing, and then the app will generate a puzzle for him.

Another issue with real life puzzles is that they become less interesting if you are to do them a second time because you already know the solution. Since the app can generate new puzzles on the fly, it will keep Dr. Bui engaged by giving him a new challenge.

2.2 Key Features

- Scan picture of real object into puzzle, Solve the puzzle in VR world
- Ability to choose the number and shape of the puzzle pieces

- Since the puzzles may be challenging and take some time, Dr. Bui wants to be able to keep track of his progress on a puzzle so he can come back to it later
- Keep track of how long it takes to solve a particular puzzle. This allows for competition as you can challenge someone else to solve the same puzzle
- Leaderboard displaying the best solving times for a particular puzzle
- Providing hints for where the pieces fit
- Additionally, Dr. Bui wanted to have the ability to be able to take a picture of a real puzzle and then import it into the VR world. This would still use the same pieces, so essentially just solving the same puzzle but it is instead while being in the VR world.
- This app would also have to figure out how to solve the real puzzle in order to provide hints. Giving hints for the real life objects is much easier as you know the solution before you break it into a puzzle.

2.3 Why this Project is Interesting

This project is interesting because it is unique compared to many other senior capstone projects. There is also learning to be done to figure out how to make this app work, which we are excited about as we have never done a project related to VR. Additionally, it seems there is not much existing support for scanning real life objects and converting them into puzzles. Most mainstream puzzle apps use premade objects (such as the one linked below). Some of them did have an uploading feature, but it was a 2D image rather than taking a picture of a 3D object in real life.

Existing Puzzle Game Link: `https://www.meta.com/experiences/art-puzzle/5920812271341547/`

2.4 Areas of CS required

- Human Computer Interaction
- Game Development
- Algorithms
- Computer Vision

2.5 Potential Concerns and Questions

We think this project should fit the requirement of a senior capstone as there are several different areas of CS involved and the project will certainly be a lot of work. The only concern we would have for ourselves is that a lot of this will be new to us. We are not sure yet about how to use the hardware to accomplish all of this. Despite this, we are good at learning new technologies and ideas, so we should be able to manage well.

2.6 Summary of Efforts to Find a Project

We talked to Dr. Bui in person about this project, and he gave us the requirements and explained why he wanted this project. We have another project that we are currently considering, which is a SWE project for Blue Raven. We are currently a little concerned that the Blue Raven project may not have enough to do for a senior capstone which is why we are continuing to pursue multiple projects in parallel.

2.7 Comparison to Draft

This is not the same project that either of us proposed. We had concerns about the original project that we were trying to do because it seemed that it might be too basic. The original project was the one with Blue Raven. We have had struggles with getting in contact with the people who would help us refine the requirements of the project. This project definitely seems to be more promising. We decided to pursue both a completely different project and client due to the same concerns above. We were considering asking for a new project, but we were afraid of having the same issues of communication with them.

3 Requirements

3.1 Non-Functional Requirements

[Non-functional requirements are just as important as functional requirements. Dont forget to specify them.]

ID	NFR Title	Category	Description
NFR1	Puzzle Generation Speed	Performance	Puzzle should be generated quickly in less than 3 seconds
NFR2	Framerate	Performance	Game should run at at least 72fps (equal to minimum refresh rate) on the meta quest headset
NFR3	Understandable Menus	Usability	Good UI/UX for menus, easy to understand what menus are for
NFR4	Understandable Interactions	Usability	Pieces are easy to interact with and interactions are intuitive
NFR5	Game Server	Reliability	Server should be almost always up so that users can attempt uploaded puzzles and view the leaderboards at any time

Table 1: Non-Functional requirements

3.2 Functional Requirements (User Stories)

Table 2: Functional requirements as User Stories.

ID	Story Title	Points	Description
U0	Piece Movement	1	As a user I want to be able to pick up and move around puzzle pieces using my real hand motions so that I can figure out how to solve the puzzle.
U1	Piece Connection	5	As a user, I want to be able to connect puzzle pieces together so that I can make progress towards solving the overall puzzle.
U2	2D Picture Upload	1	As a user, I want to be able to upload a picture file contains a 2D object so that I can solve a puzzle consisting of pieces of that object.
U3	Jigsaw Piece Generation	8	As a user, I want to be able to generate jigsaw puzzle pieces out of an object so that I can reconstruct the object by fitting the pieces back together.
U4	Local Puzzle Creation	2	As a user I want to be able to create a puzzle locally so that I can solve it on my headset.
U5	Local Puzzle Delete	2	As a user I want to be able to remove a locally created puzzle so that I can get rid of puzzle that I don't want to play anymore.
U6	Move Puzzle	2	As a user I want to be able to pick up my partially solved puzzle and move it so that I can reorganize my workspace while solving the puzzle.
U9	Solve Time	2	As a user I want to be able to see how long it takes me to solve a puzzle so that I can put my puzzle solving skills to the test and challenge my friends.
U10	Puzzle Upload	2	As a user I want to be able to upload puzzles that I create so that other people can also attempt to solve them.
U11	Uploaded Puzzle Deletion	2	As a user I want to be able to delete puzzles that I uploaded so that I change my mind about which of my puzzles I want publicly available.

Continued on next page

Table 2 – *Continued from previous page*

ID	Story Title	Points	Description
U12	View Other Puzzle Uploads	2	As a user I want to be able to view puzzles created by other people so that I can decide which ones I want to try to solve.
U13	Download Puzzles	2	As a user I want to be able to download puzzles created by other people so that I can solve them myself.
U14	Save Progress	2	As a user I want to be able to save my current progress on a puzzle so that I can come back and finish it later.
U15	Provide Piece Hint	2	As a user I want to be able to receive a hint as to where one of the pieces that I have not yet connected fits into the puzzle so that I can get help when I am stuck.
U17	Leader Board Creation	2	As a user I want to have a leader board generated after I upload a puzzle so that I can see who is the best at solving my puzzle.
U18	View Puzzle Leader Board	2	As a user I want to be able to view the leader boards of uploaded puzzles so that I can see where my time ranks compared to others.
U19	Leader Board Update	2	As a user I want the leader board to update my I or someone else solves the puzzle so that I can see the most up to date standings.
U20	Take Picture with Headset	2	As a user I want to be able to take a picture of an object using the headset so that I can generate a puzzle from that image to solve.
U21	Crop Image on Headset	2	As a user I want to be able to crop an image that I take with the headset so that I can generate the puzzle out of only parts of the image I want to keep.
U22	Load Progress	2	As a user I want to be able to load my current progress on a puzzle so that I can come back and finish it.
U23	Piece Rotation	1	As a user I can rotate the puzzle pieces using my real hand motions so that I can orient them correctly when solving the puzzle.
U24	Real Puzzle Piece Detection	8	As a user I want the pieces of a real puzzle to be detected from an image so that I can solve the same puzzle but instead in VR.
U25	Real Puzzle Piece Detection Report	2	As a user I want the application to display how many pieces and the shapes of the pieces it detected so that I can know if the pieces are detected correctly.
U26	Real Puzzle Piece Render	8	As a user I want the puzzle pieces in the image to render the same in VR game so that I can solve the same puzzle expect in VR.

4 System Design

4.1 Architecture

We will be using the MVC architecture with a Spring Boot backend, and Unity will handle all the views on the frontend. We chose to use the Web MVC architecture because we need to handle online puzzles which will involve uploading puzzles, and playing puzzles made by others.

The main modules of our software on the backend will be the config, controller, model, repository, security, service, and puzzle generation modules.

The purpose of the config module will simply be for configuration. For example, we are planning on using sockets for communication between the backend and frontend when a user is solving an online puzzle.

The purpose of the controller module is to define all the endpoints that will be called from the frontend, and it will additionally delegate tasks to the appropriate service module. For example, when a player requests to download a puzzle, after that request has been authenticated it will pass it off to the PuzzleController. The PuzzleController will then pass responsibility off to the Puzzle Service which will later return what needs to be sent back to the client. The Controller can then format this into a http response to send back.

The service module handles all of the business logic for requests. Using the example of downloading a puzzle, the Puzzle service will know that it needs to download the file associated with that puzzle and it will then delegate that task to the Content service.

The model module is split up into the dto, entity, and request modules. The dto represent the responses sent from the server to the client, the entity represents what is stored in the database, such as the Puzzle metadata, and the request module has all the formats of the requests sent from the client to the server.

The repository module piggybacks off of JPA and hibernate and handles all the interactions with the database.

The security module will define who is allowed to access what endpoints. It is also responsible for defining the filter that requests will go through before being passed to the controller for authorization purposes.

Finally, the puzzle generation module will do all the heavy lifting for both breaking down a puzzle into pieces, and figuring out how a puzzle fits back together in the case of a real world import. This is where the PuzzleGame class will live and it will store an array of the generated pieces after breaking down the object.

The main modules on the frontend will at least for now be a game representation module and a scripts module.

The game representation will store the low level representation of the puzzle game. This module will correspond to the puzzle generation module on the backend in terms of it having the same level of abstraction as they are both the lower level representation. This is where the logic of keeping track of which pieces are connected and which are still unconnected as the use works to solve the puzzle. We are thinking at this stage that this will involve making sets of connected pieces and when the user connects a piece, it would call the set union operation.

The scripts module will contain everything unity specific such as the scripts for handling the physics of moving and rotating puzzle pieces. This will also involve managing creating and deleting unity objects and managing parent and child relationships between them as pieces are connected.

4.2 Diagrams

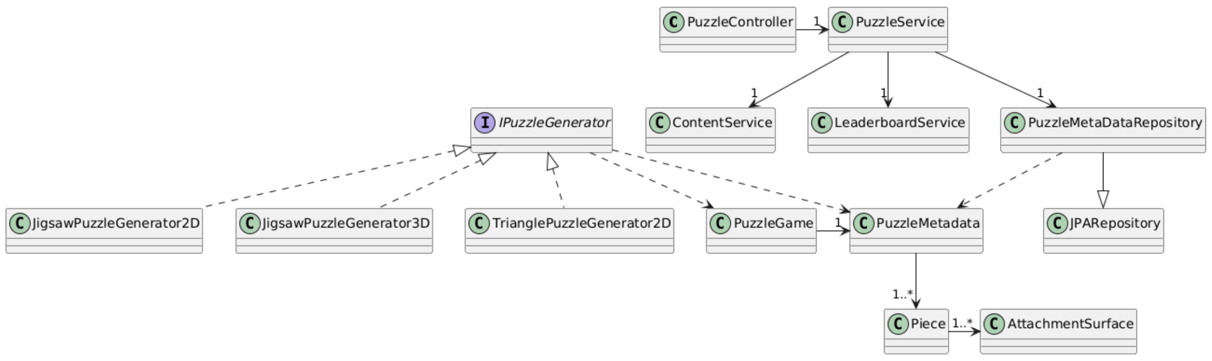


Figure 1: Intended Backend Simplified Class Diagram

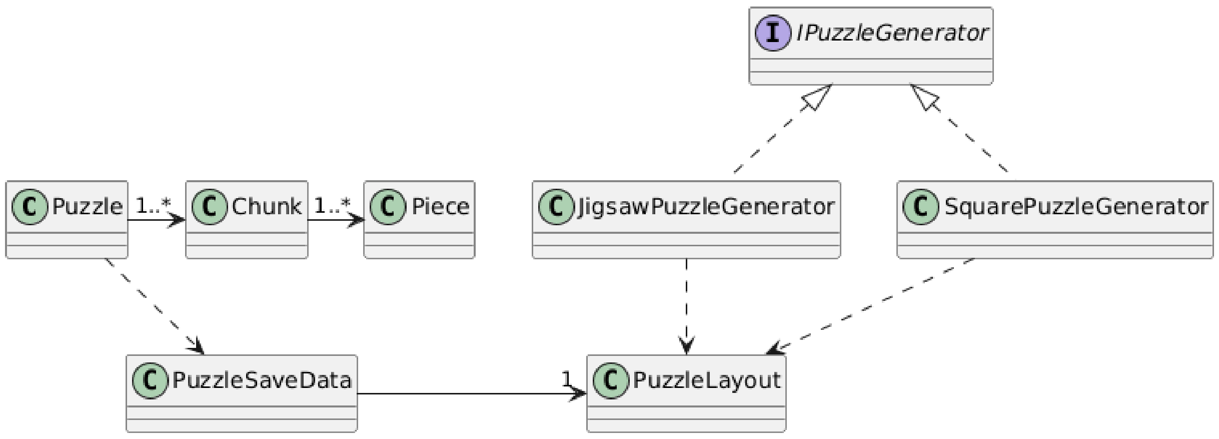


Figure 2: Current Frontend Simplified Class Diagram

4.3 Technology

Backend

- MongoDB
- Kotlin
- Spring Framework
- Spring Testing Framework

Frontend

- C#
- Unity 6
- Unity Testing Framework
- OpenXR
- Meta Quest XR All-In-One SDK, including:
  - Meta XR Core SDK
  - Meta XR Audio SDK
  - Meta XR Haptics SDK
  - Meta XR Interaction SDK Essentials
  - Meta XR Interaction SDK
  - Meta XR Platform SDK
  - Meta XR Simulator
  - Meta Mixed Reality Utility Kit

4.4 Coding Standards

- Database fields have snake case
- Database table names have Pascal Case
- C# codebase will use the official C# naming standards defined here:
  - https://learn.microsoft.com/en-us/dotnet/csharp/fundamentals/coding-style/identifier-names
- Kotlin codebase will use camel case for identifiers and Pascal case for classes
- Code needs at least 60% statement coverage to be merged into the main branch

4.5 Data

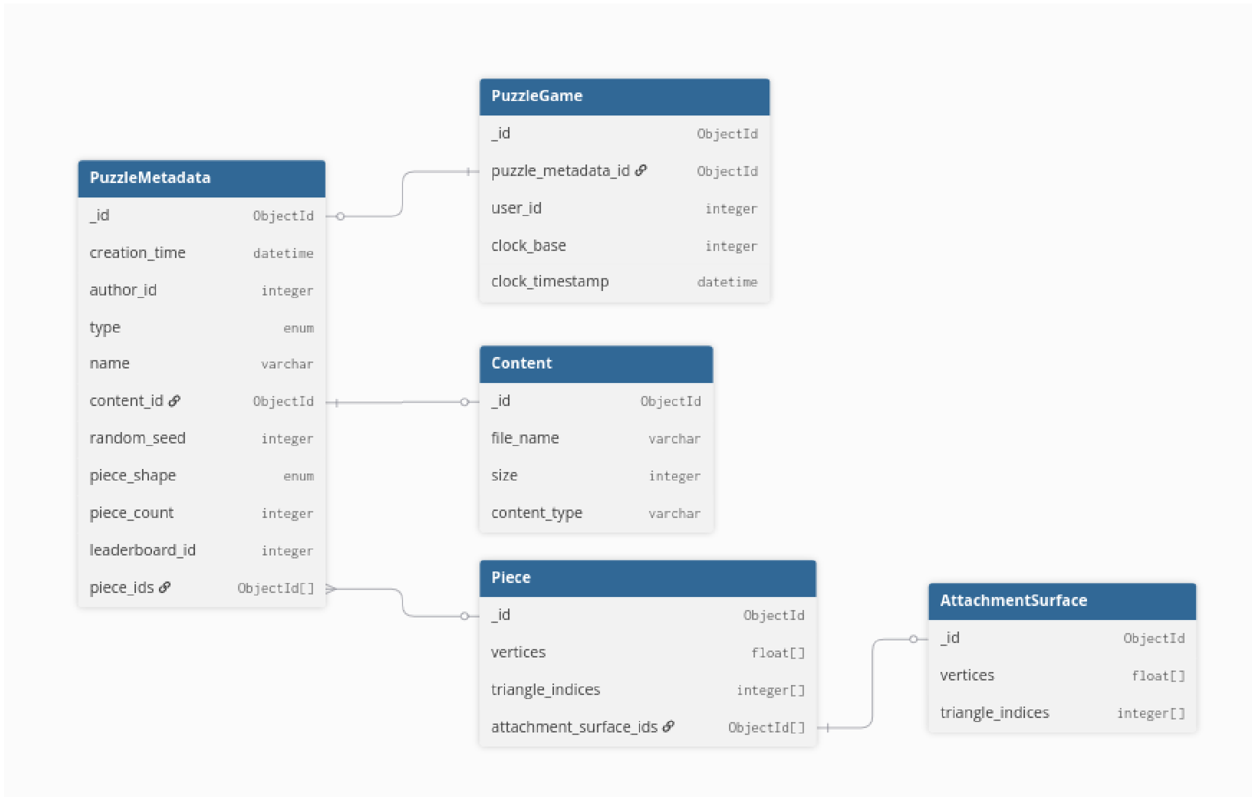


Figure 3: The reason that we do not have a Leaderboard table is that we found that Meta Quest has an API specifically for you and Meta can store these leaderboards.

4.6 UI Mocks

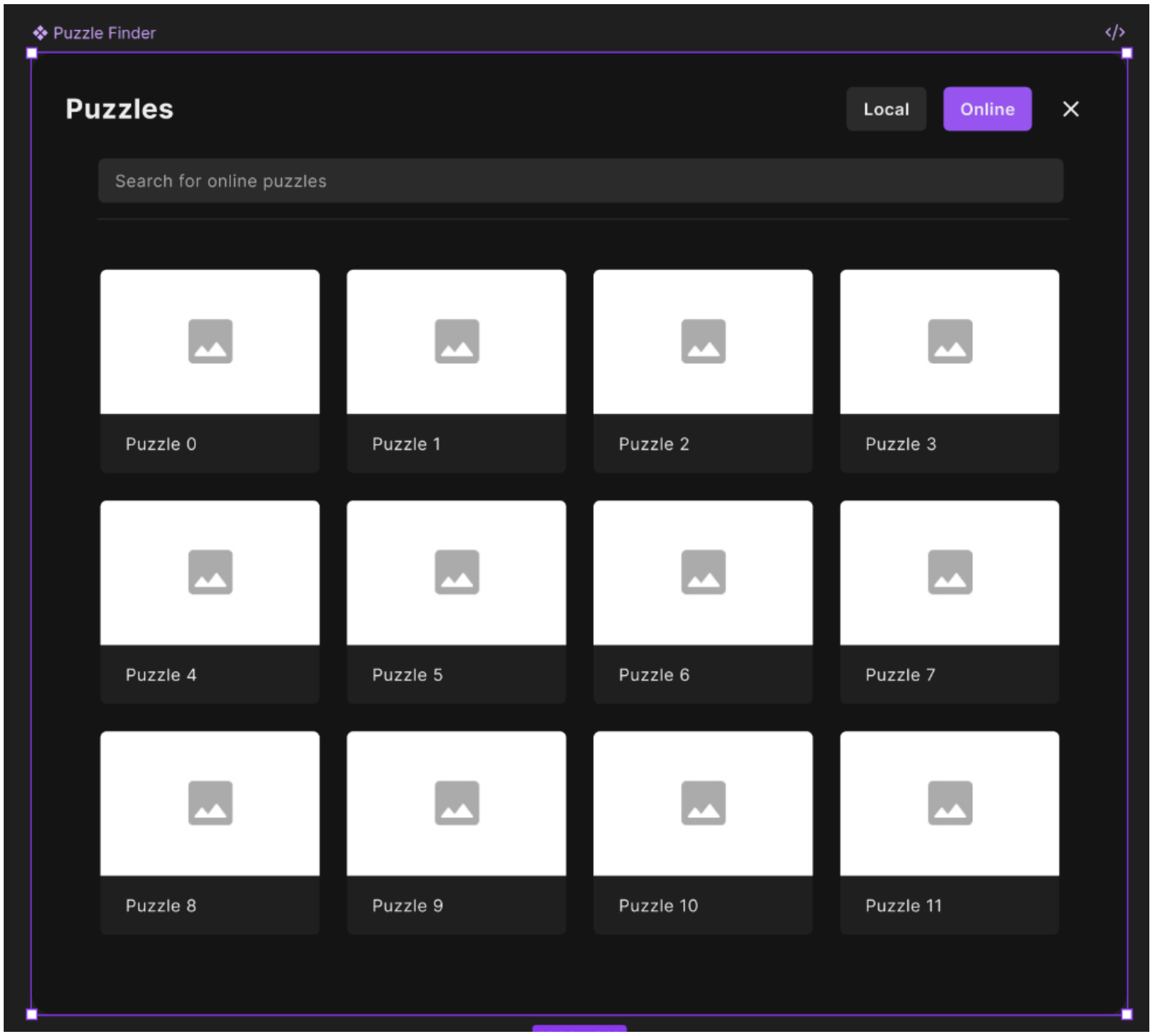
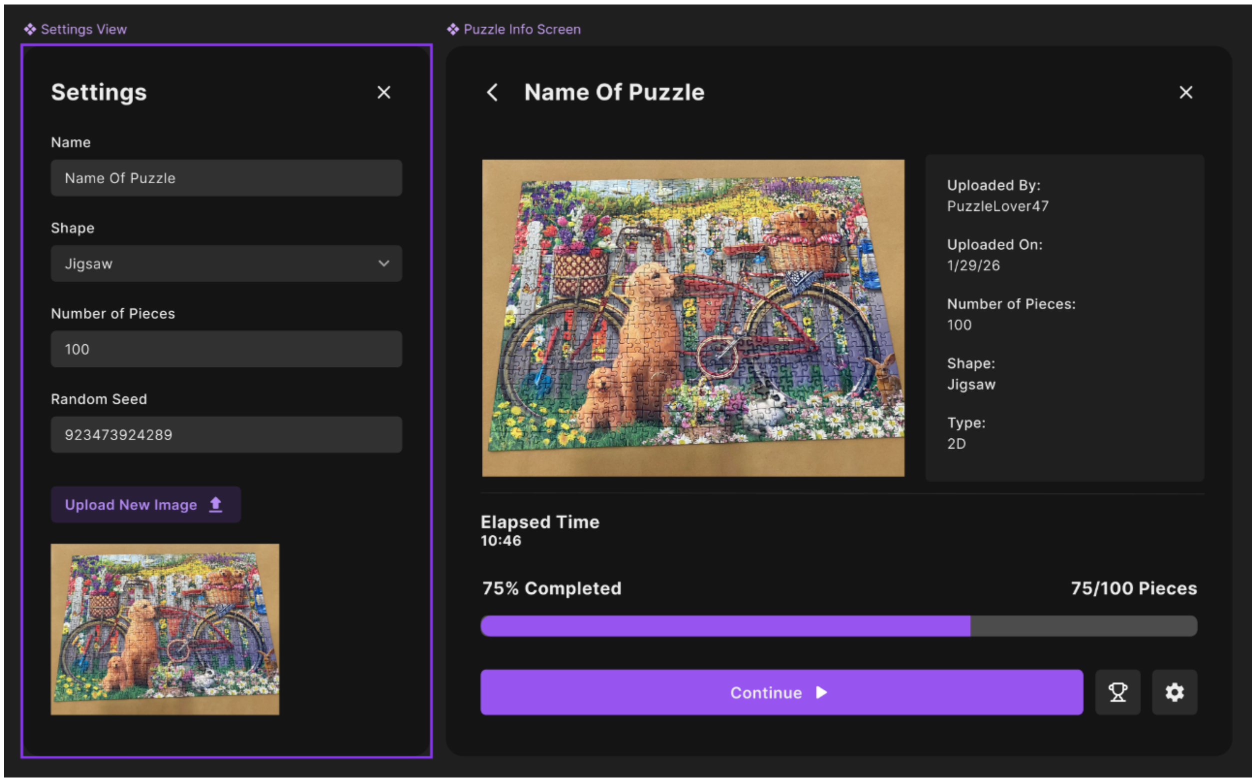


Figure 4: View of Puzzle List, with tabs for online and local


Settings View

Settings

Name: Name Of Puzzle

Shape: Jigsaw

Number of Pieces: 100

Random Seed: 923473924289

Puzzle Info Screen

Name Of Puzzle

Uploaded By: PuzzleLover47

Uploaded On: 1/29/26

Number of Pieces: 100

Shape: Jigsaw

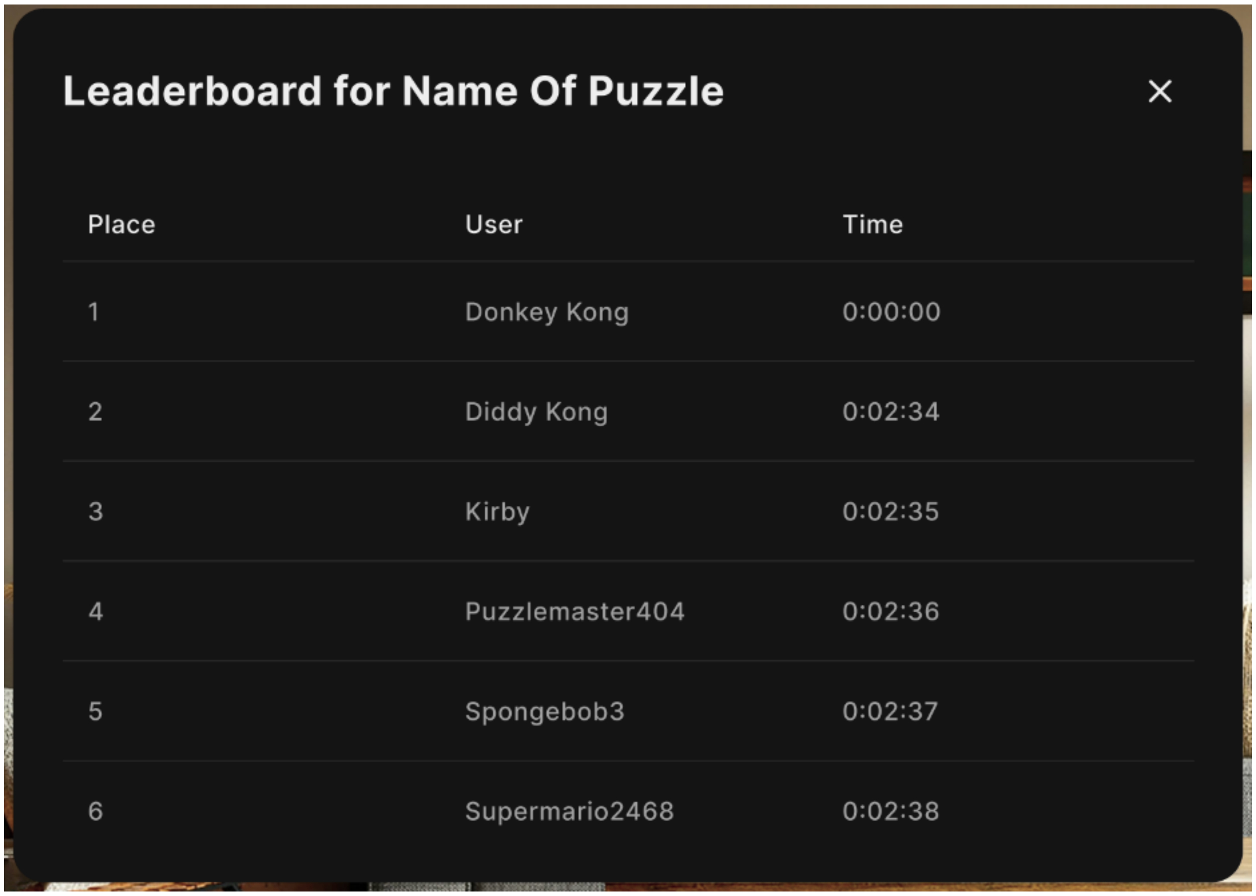
Type: 2D

Elapsed Time
10:46

75% Completed

75/100 Pieces

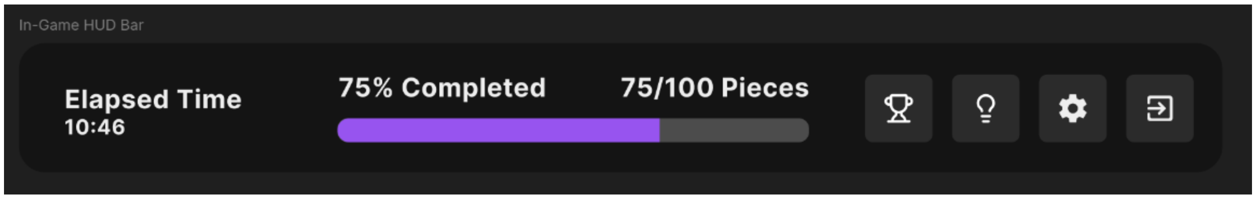
Figure 5: Puzzle Settings View



Leaderboard for Name Of Puzzle

Place	User	Time
1	Donkey Kong	0:00:00
2	Diddy Kong	0:02:34
3	Kirby	0:02:35
4	Puzzlemaster404	0:02:36
5	Spongebob3	0:02:37
6	Supermario2468	0:02:38

Figure 6: Puzzle Leaderboard View



In-Game HUD Bar

Elapsed Time
10:46

75% Completed

75/100 Pieces

Figure 7: Hud that shows up on the top of the screen as the user is solving the puzzle

5 Iterations

5.1 Iteration Planning

Iteration	Dates	Stories	Points
1	01/27 - 02/10	U0 Piece Movement (1), U23 Piece Rotation (1), U6 Move Puzzle (1), U1 Piece Connection (5), U2 2D Picture Upload (1)	9
2	02/10 - 02/24	U3 Jigsaw Piece Generation (8) U4 Local Puzzle Creation (2), U5 Local Puzzle Delete (2), U14 Save Progress (2), U22 Load Progress (2)	16
3	02/24 - 03/17	U24 Real Puzzle Piece Detection (8), U25 Real Puzzle Piece Detection Report (2), U26 Real Puzzle Piece Render (8)	18
4	03/17 - 03/31	U10 Upload the puzzle (2) U11 Uploaded Puzzle Deletion (2) U12 View Other Puzzle Uploads (2) U13 Download puzzle (2) User auth(2) U15 Provide Piece Hint (3) Support generating other pieces besides just jigsaw (5)	18
5	03/31 - 04/14	U9 Solve Time (2), U17 Leader Board Creation (2), U18 View Puzzle Leader Board (2), U19 Leader Board Update (2), U20 Take Picture with Headset (2), U21 Crop Image on Headset (2)	12
		Total:	73

Table 3: Iteration Planning for Incremental Deliveries

5.2 Iteration/Sprint 1

5.2.1 Planning

Team Member	Story	Story Description	Points
Joe	U1	Piece Connection (Pair Programming)	2.5
	U23	Piece Rotation	1
	U6	Move Puzzle	1
		Joe Total	**4.5**
Kyle	U0	Piece Movement	1
	U1	Piece Connection (Pair Programming)	2.5
	U2	2D Picture Upload	1
		Kyle Total	**4.5**
		Total	**9**

Table 4: Iteration 1 Planning

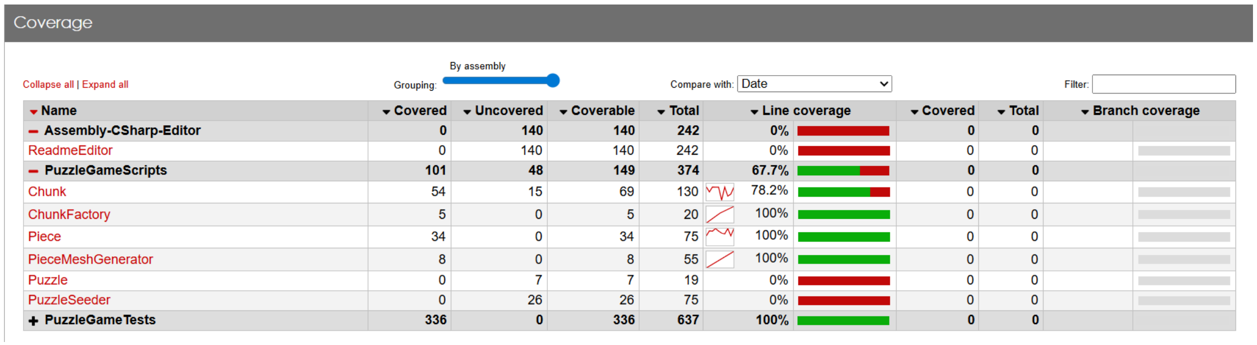
5.2.2 Work Done

Team Member	Story	Story Description	Points
Joe	U0	Piece Movement (Pair Programming)	0.5
	U1	Piece Connection (Pair Programming)	2.5
	U23	Piece Rotation (Pair Programming)	0.5
	U6	Move Puzzle (Pair Programming)	0.5
		Joe Total	4
Kyle	U0	Piece Movement (Pair Programming)	0.5
	U1	Piece Connection (Pair Programming)	2.5
	U23	Piece Rotation (Pair Programming)	0.5
	U6	Move Puzzle (Pair Programming)	0.5
		Kyle Total	4
		Total	8

Table 5: Iteration 1 Work Done

We completed all of the stories listed except for the file upload. This was mainly due to the fact that we were learning the Unity workflow with VR along with the Meta SDK. We worked in pair programming, as it was best for us to just sit down and knock each of the features out while also figuring out the proper design.

5.2.3 Testing Coverage


Coverage

Collapse all | Expand all   Grouping: By assembly   Compare with: Date   Filter:

Name	Covered	Uncovered	Coverable	Total	Line coverage		Covered	Total	Branch coverage
Assembly-CSharp-Editor	**0**	**140**	**140**	**242**	**0%**	🟥	**0**	**0**	
ReadmeEditor	0	140	140	242	0%	🟥	0	0	
PuzzleGameScripts	**101**	**48**	**149**	**374**	**67.7%**	🟩🟥	**0**	**0**	
Chunk	54	15	69	130	78.2%	🟩🟥	0	0	
ChunkFactory	5	0	5	20	100%	🟩	0	0	
Piece	34	0	34	75	100%	🟩	0	0	
PieceMeshGenerator	8	0	8	55	100%	🟩	0	0	
Puzzle	0	7	7	19	0%	🟥	0	0	
PuzzleSeeder	0	26	26	75	0%	🟥	0	0	
PuzzleGameTests	**336**	**0**	**336**	**637**	**100%**	🟩	**0**	**0**	

Figure 8: Iteration 1 Testing Coverage

We think our testing is good, but would like to test some of the extra logic on the Chunk class. This was made difficult by the shorter iteration and we think with the extra time it would be good to make a few more unit tests for that class to be extra confident in its behavior. We did not test Puzzle or Puzzle Seeder because they are only used for manual testing on the actual VR headset, so we have at least 78% for each script that matters.

5.2.4 Retroespective & Reflection

Overall, we believe we did quite well on this iteration, especially considering the shorter timeframe. We learned a lot about Unity and how VR works. We also found that the Meta SDK is quite convenient because they have some good prototyping ̈building blocks." The only feature we had trouble with was the file upload, as we learned that it isn't as simple as it would be on other devices. The Meta Quest has a modified version Android as its OS. Because of this, it has the filesystem a bit locked down, and apps are sandboxed. We have to figure out whether it is even possible to upload a file from the local filesystem. We were exploring alternatives such as fetching an image via URL. It may be possible to find a way to upload local files from the system, since it seems that we might be able to grant the app permissions. In total we spent a little more hours in the second week. The first was about 9, but the second was probably more like 12-15.

5.3 Iteration/Sprint 2

5.3.1 Planning

Team Member	**Story**	**Story Description**	**Points**
	U3	Jigsaw Piece Generation	8
Joe		**Joe Total**	**8**
	U4	Local Puzzle Creation	2
Kyle	U5	Local Puzzle Delete	2
	U14	Save Progress	2
	U22	Load Progress	2
		Kyle Total	**8**
		Total	**16**

Table 6: Iteration 2 Planning

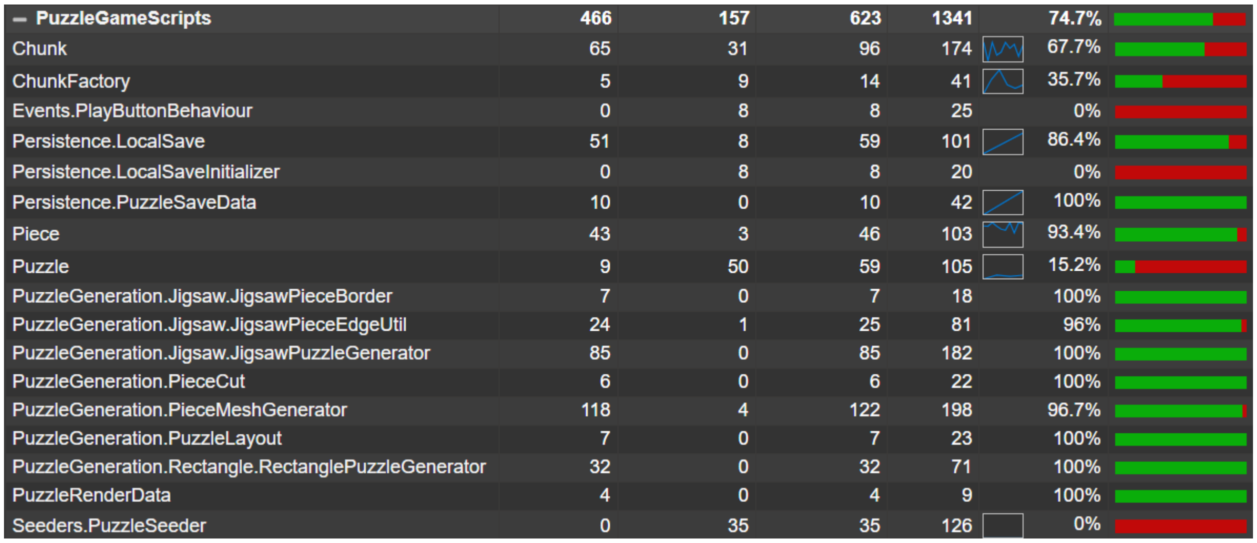
5.3.2 Work Done

Team Member	**Story**	**Story Description**	**Points**
	U3	Jigsaw Piece Generation	6/8
Joe		**Joe Total**	**6/8**
	U4	Local Puzzle Creation	1.5/2
Kyle	U5	Local Puzzle Delete	1.5/2
	U14	Save Progress	1.5/2
	U22	Load Progress	1.5/2
		Kyle Total	**6/8**
		Total	**12/16**

Table 7: Iteration 2 Work Done

We have made progress on all the stories that we planned for iteration 2, and got them working in all other aspects except for the UIs, which is why we estimate them at 75% complete each. The UIs were more annoying than we initially thought, and we will need a little extra time to figure them out. Once we figure them all out though we should ideally get them done much faster in later iterations.

5.3.3 Testing Coverage



− PuzzleGameScripts	**466**	**157**	**623**	**1341**	**74.7%**
Chunk	65	31	96	174	67.7%
ChunkFactory	5	9	14	41	35.7%
Events.PlayButtonBehaviour	0	8	8	25	0%
Persistence.LocalSave	51	8	59	101	86.4%
Persistence.LocalSaveInitializer	0	8	8	20	0%
Persistence.PuzzleSaveData	10	0	10	42	100%
Piece	43	3	46	103	93.4%
Puzzle	9	50	59	105	15.2%
PuzzleGeneration.Jigsaw.JigsawPieceBorder	7	0	7	18	100%
PuzzleGeneration.Jigsaw.JigsawPieceEdgeUtil	24	1	25	81	96%
PuzzleGeneration.Jigsaw.JigsawPuzzleGenerator	85	0	85	182	100%
PuzzleGeneration.PieceCut	6	0	6	22	100%
PuzzleGeneration.PieceMeshGenerator	118	4	122	198	96.7%
PuzzleGeneration.PuzzleLayout	7	0	7	23	100%
PuzzleGeneration.Rectangle.RectanglePuzzleGenerator	32	0	32	71	100%
PuzzleRenderData	4	0	4	9	100%
Seeders.PuzzleSeeder	0	35	35	126	0%

Figure 9: Iteration 2 Testing Coverage

We think that our testing coverage is good so far. The PuzzleSeeder is for manual testing, which is why we did not test it. Also, The Puzzle class is likely to change a lot in the future as its current interface is unstable. This class is for storing the current state of a puzzle as the user is solving it. We may decide to add and remove more fields to store in the future and since unit tests are tightly coupled to what they test, we would likely have to repeatedly change both the code and unit tests. Once the Puzzle becomes more stable in the future, then we will start writing unit tests. We are also finding that writing unit tests for Chunk is difficult, and it may signify that we need to rethink design. Part of this difficulty is due to the fact that it currently depends on both Piece and Puzzle. The dependency with Piece is natural, as Chunks are made up of Pieces. The dependency on Puzzle, which is its parent, feels more awkward. This is because when Chunks collide and combine, they need to let the Puzzle know this so that it can remove the destroyed chunk from its list of Chunks. There are alternatives such as using the Observer design pattern but this feels less honest as the only observer would be the single parent puzzle which makes use think that it is overkill. Unity obviously deals a lot with parent/child relationships so we think there should be a solution idiomatic to unity for this exact problem which we should look into.

5.3.4 Retroespective & Reflection

Overall, we think we did well in this iteration besides the challenges with the UIs. We probably should have looked more into how creating UIs worked early in the first week of this iteration. This would have been very beneficial as we could have realized how annoying they were earlier, which would have helped us make a better estimate on when we could finish them by. Also, the local save had some challenges as we want to store unity-specific objects in the database like Vector2. However, this is not directly supported by DBLite so we currently have a workaround solution that first uses Unity's serializer and then stores that serialized string in the BDLite database. We learned a lot about how rendering works in computer graphics. We learned about both the Bézier curve algorithms and Ear clipping algorithms, which are extremely common in computer graphics. The Bézier curve seemed complicated at first, but once we understood how it works and why it was used, it made a lot of sense to use it in this project. These curves are calculated from a list of control points, and are extremely common in computer graphics due to how easily scalable they are. For now, we just put the jigsaw socket in the middle of the piece's edge, and it is always the same size, but for future polishing, we can easily adjust the control points to move the socket along the edge or increase the size of it. Additionally, the ear clipping algorithm was very interesting and convenient because it works for any outline of a simple polygon that does not overlap itself. For now, we just have the jigsaw and rectangle, which for the rectangle we just triangulate manually, as a rectangle face only needs 2 triangles, but the jigsaw shape would be extremely time-consuming and error-prone to triangulate manually, so the ear clipping was good here. Additionally, since it is so versatile, we can reuse it for other piece shapes in the future. However, there is the tradeoff of it being slow which may become an issue for larger puzzles but also possibly not since each puzzle piece really doesn't need too many vertices.

5.4 Iteration/Sprint 3

5.4.1 Planning

5.4.2 Work Done

[Which stories did you complete in this iteration/sprint. Which ones did you partially complete? Who worked on which story? You may elaborate in paragraph(s) to add more detail about the work done.]

5.4.3 Testing Coverage

[Testing is very important. Show your coverage here. Is this coverage good enough? Explain why you think so. Is it not good enough? Explain a plan to increase the coverage. You may also elaborate on why some artifacts do not undergo much testing. If the testing changed from the last iteration, explain the reasons.]

5.4.4 Retroespective & Reflection

[What were the pitfalls, challenges, and issues you had in this iteration? How can you address them to improve the process in the next iteration? Did anything not go according to plan? Why so and how to avoid the same mistake? Write a personal reflection on what you learned in this iteration (even if a small technical thing like Database storage).]

5.5 Iteration/Sprint 4

[CS496 has 5 sprints. CS482 only has only 3 sprints (remove Iterations 4 and 5 from this doc if you are writing a doc for 482]

5.5.1 Planning

5.5.2 Work Done

[Which stories did you complete in this iteration/sprint. Which ones did you partially complete? Who worked on which story? You may elaborate in paragraph(s) to add more detail about the work done.]

5.5.3 Testing Coverage

[Testing is very important. Show your coverage here. Is this coverage good enough? Explain why you think so. Is it not good enough? Explain a plan to increase the coverage. You may also elaborate on why some artifacts do not undergo much testing. If the testing changed from the last iteration, explain the reasons.]

5.5.4 Retroespective & Reflection

[What were the pitfalls, challenges, and issues you had in this iteration? How can you address them to improve the process in the next iteration? Did anything not go according to plan? Why so and how to avoid the same mistake? Write a personal reflection on what you learned in this iteration (even if a small technical thing like Database storage).]

5.6 Iteration/Sprint 5

5.6.1 Planning

5.6.2 Work Done

[Which stories did you complete in this iteration/sprint. Which ones did you partially complete? Who worked on which story? You may elaborate in paragraph(s) to add more detail about the work done.]

5.6.3 Testing Coverage

[Testing is very important. Show your coverage here. Is this coverage good enough? Explain why you think so. Is it not good enough? Explain a plan to increase the coverage. You may also elaborate on why some artifacts do not undergo much testing. If the testing changed from the last iteration, explain the reasons.]

5.6.4 Retroespective & Reflection

[What were the pitfalls, challenges, and issues you had in this iteration? How can you address them to improve the process in the next iteration? Did anything not go according to plan? Why so and how to avoid the same mistake? Write a personal reflection on what you learned in this iteration (even if a small technical thing like Database storage).]

6 Final Remarks

6.1 Overall Progress

[Have you completed everything? If so, present evidence on how you brought value to your client, and the overall client satisfaction. Otherwise, estimate how much progress you done and how long it would take to finish this project.]

6.2 Project Reflection

[Your personal reflection on the project. What lessons did you learned. What would you have done differently. How can you do better work in future projects? You may write this as a team or per person (or both)]

Appendix

[Appendix section if needed]