

03-T2 Minecraft

Software Design Document

CS 4850 - Section 01 – Spring 2026

February 8, 2026

 Matthew Pflieger Team Leader Developer	 Andrew Burgos Developer	 John McDowell Developer	 Hezekiah Holloman Documentation	 Chase Holt Documentation
---	---	---	--	--

Team Members:

Name	Role
Matthew Pflieger (Team leader)	Developer
Chase Holt	Documentation
Andrew Burgos	Developer
Hezekiah Holloman	Documentation
John McDowell	Developer
Sharon Perry	Project Owner or Advisor

Table of Contents

1. Introduction and Overview.....	3
2. Design Considerations.....	3
2.1. Assumptions and Dependencies.....	3
2.2. General Constraints.....	4
2.3. Development Methods.....	5
3. Architectural Strategies.....	5
Programming Language.....	5
PaperMC.....	5
Future Plans.....	5
4. System Architecture.....	5
5. Detailed System Design.....	6
5.1. Classification.....	6
5.3. Constraints.....	6
5.4. Resources.....	6
6. Glossary.....	7

1. Introduction and Overview

Our team is developing a minecraft gamemode known as Turf Wars. Turf Wars is a 2-team gamemode where both teams compete to control the arena. Each kill with a bow earns you more control over the area to build and fight. This gamemode is for use in KSU E-sports events so it needs to hold a large player number. We will need to use Paper API as well as command blocks to achieve the gamemodes functionality.

2. Design Considerations

This gamemode is for use in Kennesaw State E-sports events. That means it must accommodate various low-high end hardware, as well as handle a large capacity of players. It also must be easy to understand and play for first time attendees of these events.

3. Architectural Strategies

Programming Language

- **Language:** Java 21+

PaperMC

- The industry standard software for running high performance Minecraft servers.

Future Plans

- Future game mode implementations for the Minecraft server

4. System Architecture

The System will be divided into six subsystems:

- An area manager that will handle the creation, loading, and saving of the areas as well as player spawn locations.
- A match engine which will allow for control over each match's lifecycle and timing.
- A team manager which will assign players into one of two teams and make sure the teams are balanced.
- A turf manager to calculate territory ownership and will update when conditions are met.
- A rules manager which will set arena boundaries and gameplay restrictions.
- A UI manager which will display game information like the scoreboard, time remaining for each phase, and turf percentage.

5. Detailed System Design

The implementation of the turf wars minigame will largely be in the form of a Paper API plugin. The final deliverable will be a portable .jar file, created through Paper's Gradle build system, to be placed in the KSU Minecraft server's plugins directory. Existing infrastructure and the Paper API will take our code and host it as a playable minigame from there.

5.1. Classification

The Paper API's concept of a "plugin" is a modular Java package meant to be loaded on top of an existing Minecraft server. Plugins do not redefine any of the basic game logic associated with Minecraft, as that processing is handled by the game server. They use the Paper API to communicate with the game server to change game data, such as items and blocks in the world and players' inventories.

5.2. Constraints

The minigame will be online at only certain event times, as instructed by the people running the server. It should only need to store and gather state for as long as the minigame is online - any more persistent data like usernames will be handled by the API and the existing server. We may want to store state such as score records, which can be made to persist on server storage, but

permanent storage will not be needed or considered as a functional requirement. The plugin component will likely have its own txt or yaml file for storing data, with I/O access moderated by the Paper API.

5.3. Resources

In order to function, the plugin will need the KSU Minecraft server to be up and running and ready to receive it. This implies a certain amount of CPU and RAM already allocated to the server, which is usually a minimum of 4GB and multiple processing cores. Given that the plugin will be running at times where server load is higher, the server will likely need more than this in order to provide adequate performance. The plugin itself will require minimal overhead in terms of resources, needing only enough storage and memory to load the lobby and arena worlds.

6. Glossary

Turf Wars - A minigame design within Minecraft involving two opposing teams that try and gain territory by getting eliminations using a bow and arrow.

Paper API (PaperMC) - High-performance Minecraft server API (application performance interface) used to create plugins and manage gameplay features.

Plugin - A Java component loaded by a Minecraft server to extend or modify the game functionality without changing core game code.

CPU (Central Processing Unit) - Fetches, decodes, and executes instructions to process data and control system operations.

RAM (Random Access Memory) - High-Speed, short-term memory used to temporarily store active data.

YAML File - A human-readable configuration file format used to store plugin settings and temporary game data.

