

Draw it or Lose It

# **CS 230 Project Software Design Template**

Version 1.0

## Table of Contents

[**CS 230 Project Software Design Template** 1](#_Toc107173257)

[Table of Contents 2](#_Toc107173258)

[Document Revision History 2](#_Toc107173259)

[Executive Summary 3](#_Toc107173260)

[Design Constraints 3](#_Toc107173261)

[System Architecture View 3](#_Toc107173262)

[Domain Model 3](#_Toc107173263)

[Evaluation 4](#_Toc107173264)

[Recommendations 7](#_Toc107173265)

## [Document Revision History](#_grjogdjh5fi8)

| Version | Date | Author | Comments |
| --- | --- | --- | --- |
| 1.0 | 05/22/2022 | Jacob Hershberger | Initial design template creation |
| 2. | 06/22/2022 | Jacob Hershberger | Maybe getting something right for once this term 😉 |
| 3. | 6/26/2022 | Jacob Hershberger | Re-submit for project 2 |

**Instructions**

Fill in all bracketed information on page one (the cover page), in the Document Revision History table, and below each header. Under each header, remove the bracketed prompt and write your own paragraph response covering the indicated information.

## [Executive Summary](#_sbfa50wo7nsh)

The client is wanting a web-based game based on the 1980s television game “*Win, Lose or Draw”* called Draw it or Lose it. The client is unsure how to set up the environment required for the web-based application. Using Java I have setup the ability to have one or more teams involved in the game with unlimited players per team – as of now. I have also enabled the code to only allow one user, instance, and team of a given name in addition to only one instance of the game can exist at a given point in time.

## [Design Constraints](#_2et92p0)

* Each game will have the ability to have one or more teams involved
* Each team will have multiple players assigned to it – no limit currently
* Game and teams must be unique in name
* Only one instance of the game can exist in memory at a given point in time

## [System Architecture View](#_ilbxbyevv6b6)

Please note: There is nothing required here for these projects, but this section serves as a reminder that describing the system and subsystem architecture present in the application, including physical components or tiers, may be required for other projects. A logical topology of the communication and storage aspects is also necessary to understand the overall architecture and should be provided.

## [Domain Model](#_8h2ehzxfam4o)

You have your main “entity” class that acts as a sort of parent controller to the game/gameService, team, and player classes. Each entity is unique to the system memory as a result of the singleton design pattern.

Inside the entity is the game instance, team(s) instance, and player(s) instance. Each are connected to one gameService entity. However, multiple games, players and teams, are allowed to be connected to a game service.

**"The Gaming Room UML diagram. The top of the diagram is labeled as com dot gamingroom. Test boxes are placed in two layers. The first layer has three text boxes and the second layer has four of them. In the first layer, the 'ProgramDriver' textbox points to 'SingletonTester' textbox. The 'ProgramDriver' textbox contains the text 'asterisk main round brackets.' The 'SingletonTester' textbox contains the text 'asterisk testSingleton round brackets.' The arrow between these two text boxes are labeled 'open two angle brackets uses close two angle brackets'. In the second layer, there are 'GameService', 'Game', 'Team', and 'Player' text boxes. The 'GameService' textbox has texts arranged in two layers. The first layer contains games colon List open angle bracket Game close angle bracket, nextGamesId colon long, nextPlayer Id colon long, nextTeamId colon long, and service colon GameService. The second layer contains GameService round brackets, getinstance round brackets colon GameService, addGame open parenthesis name colon String close parenthesis colon Game, getGame open parenthesis id colon long close open parenthesis colon Game, getGame open open parenthesis name colon String close open parenthesis colon Game, getGameCount round brackets colon int, getNextPlayerID round brackets colon long, and getNextTeamId round brackets colon long. The 'GameService' box is connected with the 'Game' textbox with a line labeled 'zero dot dt dot asterisk'.  The 'Game' textbox also contains text in two layers. The first layers contains the text teams colon List open angle bracket Team close angle bracket. The second layer has Game open round bracket id colon long comma name colon String close parenthesis, addTeam open parenthesis name colon String close parenthesis Team, toString round brackets colon String. The 'Game' textbox is connected with the 'Team' textbox with a line labeled 'zero dot dt dot asterisk'. The 'Team' textbox also contains text in two layers. The first layers contains the text players colon List open angle bracket Player close angle bracket. The second layer has Team open parenthesis id colon long comma name colon String close parenthesis, addPlayer open parenthesis name colon String close parenthesis colon Player, and toString round brackets colon String. The 'Team' textbox is connected with the 'Player' textbox with a line labeled 'zero dot dt dot asterisk'. It contains the text Player open parenthesis id colon long comma name colon String close parenthesis and toString round brackets colon String. The 'Game', the 'Team, and the 'Player' boxes point to the 'Entity' textbox in first layer. The 'Entity' textbox contains text in two layers. The first layer has the text id colon long and name colon String. The second layer has Entity round brackets, Entity open parenthesis id colon long comma name colon String close parenthesis, getId round brackets colon long, getName round brackets colon String, toString round brackets colon String.**

## [Evaluation](#_2o15spng8stw)

Using your experience to evaluate the characteristics, advantages, and weaknesses of each operating platform (Linux, Mac, and Windows) as well as mobile devices, consider the requirements outlined below and articulate your findings for each. As you complete the table, keep in mind your client’s requirements and look at the situation holistically, as it all has to work together.

In each cell, remove the bracketed prompt and write your own paragraph response covering the indicated information.

| **Development Requirements** | **Mac** | **Linux** | **Windows** | **Mobile Devices** |
| --- | --- | --- | --- | --- |
| **Server Side** | Mac is generally not known for hosting so software and support could be limited should they decide to go this route | Linux or Windows would be ideal due to compatibility with other software and hardware for monitoring, alerting, security, etc. ease of use, and overall supportability. Linux and Windows also allow hosting of an API that would be accessible from all other mobile platforms should they build a user interface for it. Potential costs would be comparable should they deploy on Linux or windows. Though Linux is open source whereas windows operating systems do have fees to purchase newer versions. | Linux or Windows would be ideal due to compatibility with other software and hardware for monitoring, alerting, security, etc. ease of use, and overall supportability. Linux and Windows also allow hosting of an API that would be accessible from all other mobile platforms should they build a user interface for it. Potential costs would be comparable should they deploy on Linux or windows. Though Linux is open source whereas windows operating systems do have fees to purchase newer versions. | Mobile devices would not be recommended as a server side host for a web based application especially when options such as Windows Server and Linux are far more readily available. I am not aware of a mobile device even having the ability to allow server side hosting. |
| **Client Side** | Windows and Mac web browsers are on par with each other and don’t really need any special requirements outside of good use of programming practices | Linux really suffers when it comes to a UI for the client side. Unless the game is strictly CLI based, There is not a lot of value in ensuring linux users being able to use it outside of good industry standard development practices | Windows and Mac web browsers are on par with each other and don’t really need any special requirements unless you are targeting a very specific version of Internet Explorer which is NOT recommended as Microsoft has stopped supporting IE and discourages it’s use. | Mobile web browsers are at the mercy of the data connection and tend to be far slower than their mac/windows counterparts. I’d suggest using a native application for mobile use. In that there is an Android app already available, I'd keep it as it stands and maybe port a iOS version. |
| Building software that is movable across devices is not impossible, in fact it’s quite a bit cheaper and more accessible than one would expect. Using front end frameworks such as React Native would enable the user to “build once and deploy everywhere” which is the entire mindset of using React Native as a user interface. The same code could be used for Android, iOS, Web, Windows phone, amazon fire tablet, and more. The only extra costs would be the costs of deploying an application to the respective phone app stores as well as having the physical device – or web tool that can emulate it – for validation testing. It would be a bit longer to develop this way for one single system, but it would be far shorter to do this than rebuilding the app for every unique system. | | | |
| **Development Tools** | Languages, Tools, and deployment methodologies are going to be the same across Mac, Linux, and Windows. Because the major languages are generally cross compatible. | You can use tools like VIM or VI but you lose out on capabilities such as compilers to run your code from Java, or a C language. That being said you can easily install compilers to do that via CLI. | Languages, Tools, and deployment methodologies are going to be the same across Mac, Linux, and Windows. Because the major languages are generally cross compatible. | I would not recommend developing on a mobile application as your code editor. They do not have the software nor processing capabilities for a successful development experience |
| I’d recommend the following development tools: An IDE such as IntelliJ for Java API development as well as Visual Studio Code for React Native UI development. Both IDEs offer a free tier for the client to use without increasing costs. I would recommend a small team for API development as well as a team for UI development this would enable both front and backend to be developed at the same time and with dedicated focus, though it would be possible for a single individual or two to do both API and UI. I would recommend a physical device that connects to each app store they intend to release the app on for validation, such as an iPhone, iPad, Android phone and tablet, etc. if costs are tight, you can skip the tablet and just obtain a physical iPhone and Android phone. The web has capabilities that can accommodate the tablet screen resolutions. | | | |

## Recommendations

Analyze the characteristics of and techniques specific to various systems architectures and make a recommendation to The Gaming Room. Specifically, address the following:

**Operating Platform**: My recommendation would be to operate on a Windows server. One of the main reasons for this is it’s among the most widely used platforms for hosting applications. You can connect a seemingly endless number of applications and services to help monitor security, performance, application health and uptime metrics. Windows operating systems also are compatible with Java API which would enable the back-end code to be accessible via other platforms such as iOS, Android, etc.

**Operating Systems Architectures**: The Windows operating system has two kinds of system processes. One is the system process, and the other is the user process. System process has unrestricted access to manipulate hardware and operating system data. The user process can only access operating system data through sending requests through the system process. Numerous base processes such as system support, services, user applications, and environment subsystems get pushed to subsystem DLL’s. From there, system support services, services, and user applications are processed through Executive and then further moved through kernel and device drivers to end at HAL. Executive, kernel, device drivers, and HAL are all in kernel mode. The environment subsystems in user mode moves to the windowing and graphics part of kernel mode.

**Storage Management**: Included in the Windows operating system is disk management along with Storage Sense. Additionally, Windows has a disk cleanup tool. Disk management is a utility mostly used for advanced storage tasks. Disk cleanup and Storage Sense are useful in helping delete files that are unnecessary and that take up storage space.

**Memory Management**: Memory management is built into windows operating systems as a system utility. For the game images to be easily accessible by the application a logical folder structure with file locations stored in a database for quick and easy access.

**Distributed Systems and Networks**: A client-server distributing system would be the recommended choice as it would be easiest for the client to edit, add, and remove images onto the game and have them apply across the different instances of the application. Additionally, with different nodes (users) joining and leaving the network the client-server system would allow for a more stable connection and user experience with the application. The risk with the client-server system is when it comes to outages. Should the client-server system fail, unless there is redundancy and a readily available failover switch the application will suffer and go offline until the outage is resolved.

**Security**: In addition to tools such as Windows Defender, Windows Operating systems are compatible with countless security applications that range from encryption, NGINX, data processing and monitoring and more. There is no shortage of security configurations available on Windows Operating systems.