Technical Specification Document

CECS 491
TheBoiZ

Lead:	014073642	Fa Fu
Members:	015188925	Brian Nguyen
	013122991	Louie Yonzon
	015774900	Kevin Simon
	015938089	Dylan Thorin
	015260958	Kevin Phan

Date: November 12, 2019

Table of Contents

1. OVERVIEW **1**
1.1 Purpose 1
1.2 Background Information 2
2. BUSINESS CASE **2**
3. DEVELOPMENT ENVIRONMENT **2**
4. Servers (cloud services) **4**
5. Project Management Tools **4**

1. OVERVIEW

1.1 Purpose

The purpose of this document is to provide information about the technologies used in the **Kraken Bracket** project

Note: **This version of the document is not finalized.** Any information provided in this document is subject to change.

1.2 Background Information

- When it comes to managing large-scale video game tournaments, it is likely that there will be confusion among staff and competitors
- Other video game tournament apps on the market do not provide real-time updates once the information is inputted into the system. This will cause delays and inconveniences for those who are keeping track of updates regarding tournament matches, competitors, etc.
- The **Kraken Bracket** project will provide solutions to these problems. The most important aspect of the project is to provide all users with real-time information on events, tournaments, and other aspects of the app.

2. BUSINESS CASE

- Potential major platform for a gaming audience
- Provides a better solution for updating tournament bracket information
- Provides a social aspect for tournaments to better engage viewers and participants

3. DEVELOPMENT ENVIRONMENT

Our choices for what we will use for development need to be able to work on all of the developers devices, but also make it easy to accurately test our application from where we may be located. Also our devs need to easily develop with these with little to no fuss.

- Operating System for backend
    - Linux (Ubuntu 16.01 or above)
        - Pros: does not restrict administrative functionality out of the box
        - Cons: requires a lot of manual setup
    - Windows 10 (any variant)
        - Pros: Familiar with the whole team
        - Cons: restricts administrative functionality, and manages it in a confusing manner.
- OS Testing Environment
    - Real computer hardware (whatever the dev has on hand)
        - Pros: authentic testing using real hardware.
        - Cons: not readily available for the whole team. It is also not even necessary to do.
    - VirtualBox (latest version)
        - Pros: widely used pc emulation software
        - Cons: does not run natively on Mac or Linux.
    - VMWare (latest version)
        - Pros: DOES work on Windows, Mac, and Linux.
        - Cons: none known.
- Development IDE
    - Eclipse (latest version)
        - Pros: good syntax highlighting for both c++ and Java
        - Cons: interface is mildly confusing, and may not support .NET or ASP.NET.
    - Notepad++
        - Pros: can edit any text file with ease, great for coding in obscure languages or for changing weird config file types.
        - Cons: very poor syntax highlighting, and it requires that the developer knows what they are doing. It also only runs on Windows (the default text editors for Mac and linux have this same functionality so they don't really need Notepad++).

    - Visual Studio 2017
        - Pros: can develop .NET app in any standard.
        - Cons: only runs on Windows.
- Front end Testing
    - Google Chrome (and other Chromium based browsers)
        - Pros: the browser architecture is used very widely. And runs on most consumer devices.
        - Cons: possible over usage of RAM.
    - FireFox (latest version)
        - Pros: the popular alternative to Google Chrome
        - Cons: possible memory leaks. Also might not run on iOS phones.
    - Java (latest version)
        - Pros: works on nearly any device
        - Cons: not optimised for speed; however, this is negligible for non CPU intensive tasks.
- Back end Testing
    - SQL Server 2017 Developer Edition (Database Engine)
    - SQL Server Management Studio (Database Client)
- Database Management
    - mySQL (any version)
        - Pros: works on just about any server configuration and is supported by most IaaS and PaaS services.
        - Cons: not as feature-rich compared to other database management systems
    - MongoDB
        - Pros: Popular choice due to its asset compliance. It does relational databases.
        - Cons: less flexibility when querying
    - NodeSQL
        - Pros: efficient for data sorting
        - Cons: none known

4. SERVERS (CLOUD SERVICES)

Our backend server will need to be on a cloud service and allow us to run anything we need to. Namely a dynamic website and a way to manage a data store.

- Cloud Services
    - AWS
        - Pros: Geared toward VM based server hosting. The EC2 instances can be SSH'ed into with ease. AWS also has really good scalability for computer resources and only charges for what you need.
        - Cons: requires SOME training to use.
    - Google Cloud
        - Pros: our class got this service at a discount. It can do VMs with rdp support.
        - Cons: Unlike AWS, we don't know the inner working of this service.
    - Microsoft Azure
        - Pros: good for anything that works on IIS
        - Cons: does not allow for VM based hosting.

5. PROJECT MANAGEMENT TOOLS

- Document sharing
    - Google Drive: used for sharing documents between team members.
- Code repository
    - Github: Syncing development code with multiple team members.
- Team communication
    - Discord: Allows for IM over the internet with the ability to share files with the whole team. We can even post links that all members can access. This can be used to notify specific team members and also works on mobile.
    - Trello: very similar in function to Discord, however our team already is used to communicating on Discord.