**PROJECT PROPOSAL**

*Dungeon RPG*

Table of Contents

[Project Abstract 3](#_Toc113300636)

[Conceptual Design 3](#_Toc113300637)

[Proof of Concept 4](#_Toc113300638)

[Background 4](#_Toc113300639)

[Required Resources 5](#_Toc113300640)

## Project Abstract

What the product does and how it does it from a user point of view – at a high level. (You can include screenshot mockup of the interface)

*This document proposes a Discord bot called Dungeon RPG that enables users to play different dungeons and have fun on the Discord platform. Any Discord server can easily add the bot because it is integrated with the Discord API. Users of Dungeon RPG can access dungeons according to their Hunter rank. For instance, Hunters of the D rank can access D rank dungeons, Hunters of the C rank can access C rank dungeons, and possibly B rank dungeons, etc. The dungeon is basically a turn-based game that could possibly utilize randomness/luck. Anyone may start playing straight away thanks to the bot's intuitive user interface. By successfully clearing dungeons, players can level up and gain access to new Hunter ranks. Users can greatly improve their Discord experience and meet new people by playing Dungeon RPG.*

*This bot is mainly targeted towards children ages 8-16 since most children use discord now and days due to school and games. For this targeted audience, they can use this bot as a small side game. Of course, this game can also appeal to young adults and older if interested.*

*Graphical user interface, text, application

Description automatically generated*

## 

## Conceptual Design

Describe with text (and maybe UML diagrams) the initial design concept. Include hardware and software architecture, operating system, programming language, framework, libraries, APIs, etc.

*The initial design concept for the discord bot is centered around providing an engaging and user-friendly experience for Discord users. To achieve this goal, we can utilize the following:*

* *Hardware: Although the bot is designed to run on a cloud-based virtual machine that offers users the maximum level of availability and dependability, for the time being it will probably just use my PC due to the limited offers accessible.*
  + *To expand, due to most Discord bots being active 24/7, it will take up a tremendous number of resources which is why it needs to run on a cloud-based virtual machine. Since this is only, for now, just a project, we will use our PC since the number of users will be kept to a minimum of 1-5.*
* *Operating System: The bot will most likely function on a Windows-based operating system.*
* *Programming Language: Python*
* *Framework: Discord API since it will give us access to a wide range of tools and features for interacting with the Discord platform.*
* *Libraries: The bot will most likely utilize libraries like "asyncio" and "discord.py"*
  + *The asyncio library will most likely be needed to give the bot the ability to manage numerous tasks at once without impeding or delaying the completion of other tasks. This makes it possible to guarantee that the bot offers users a quick and responsive experience.*
  + *With discord.py, developers can easily build and deploy bots that can send messages, listen to, and respond to events, and perform various other actions within the Discord platform.*

## Proof of Concept

Include a link to a public git repository. This repository include code using the same programming language, framework, libraries, and APIs needed to demonstrate the tools are going to work together. The code does not need to be extensive, but it needs to compile and run. It is ok if the code is straight from a tutorial as long it is compiling and running.

In this document or a readme file in the repository, include instructions on how to compile and run the code. Specify the operating system version and the version of the compiler to used build the project.

If your project is as contribution to an open source, you need to show you can modify, compile and run the source code of the project.

*Link to public git repository:* [*https://github.com/alansaji2347/DungeonRPG*](https://github.com/alansaji2347/DungeonRPG)

## Background

The background will contain a more detailed description of the product and a comparison to existing similar projects/products. Proper citation of sources is required. If there are similar open-source products, you should state whether existing source will be used and to what extent. If there are similar closed-source/proprietary products, you should state how the proposed product will be similar and different.

*Most of the information regarding the bot has been stated within the project abstract. To go more in depth, all users will begin with the lowest rank as most games do. The users, or Hunters, will have to level up through dungeon completions. After a considerable amount of levels has been accumulated, the Hunter will then move on to the next rank, i.e., F🡪D🡪C🡪B🡪A🡪S. Each rank will also have access to its designated dungeon in which users can gain greater rewards.*

*In terms of similar products, I was able to find two similar discord bots:*

* “AniGame.” *Add AniGame Discord Bot | The #1 Discord Bot List*, <https://top.gg/bot/571027211407196161>.
  + *Trading and player vs. player experiences are more emphasized in AniGame. There are many more features, such as trade systems, shop systems, random drops, etc., in it. It is more focused on the PvP part, whereas my bot is more focused on player vs environment, or more story-driven gameplay in a sense.*
* “Solo Leveling Simulator.” *Discord Bots*, <https://discord.bots.gg/bots/703043558483034223>.
  + *In terms of RPG, this bot is comparable to my bot. Like "AniGame," it differs in that it offers additional features. It offers a system of social interaction that includes guilds, daily quests that users can perform, random drops, etc. As I mentioned previously, the bot I'm using is more focused on the narrative. PvP is the primary focus of the majority of the bots*.

*Between the two bots, the “Solo Leveling Simulator” is probably the most similar, but that is mainly because both bots utilize the RPG genre.*

## 

## Required Resources

Discuss what you need to develop this project. This includes background information you will need to acquire, hardware resources, and software resources (Raspberry Pi, data for training models, …). If these are not part of the standard Computer Science Department lab resources, these must be identified early and discussed with the instructor.

* *Hardware: Although the bot is designed to run on a cloud-based virtual machine that offers users the maximum level of availability and dependability, for the time being it will probably just use my PC due to the limited offers accessible.*
  + *To expand, due to most Discord bots being active 24/7, it will take up a tremendous number of resources which is why it needs to run on a cloud-based virtual machine. Since this is only, for now, just a project, we will use our PC since the number of users will be kept to a minimum of 1-5.*
* *Operating System: The bot will most likely function on a Windows-based operating system.*
* *Programming Language: Python*
* *Framework: Discord API since it will give us access to a wide range of tools and features for interacting with the Discord platform.*
* *Libraries: The bot will most likely utilize libraries like "asyncio" and "discord.py"*
  + *The asyncio library will most likely be needed to give the bot the ability to manage numerous tasks at once without impeding or delaying the completion of other tasks. This makes it possible to guarantee that the bot offers users a quick and responsive experience.*
  + *With discord.py, developers can easily build and deploy bots that can send messages, listen to, and respond to events, and perform various other actions within the Discord platform.*
* *Issue: The bot will most likely be a text-to-text responsive bot. Due to this, it can lead to just a wall of text history which could be confusing as the responses continue to get longer. A solution to this is simply to make an interactive interface where users can visually see the amount of damage dealt and taken in a singular box without having to see such a wall of responses. However, this implementation will be a bit more complicated as developers would need to be familiar with HTML, CSS, JavaScript, etc.*