Requirements Analysis Document (RAD)

Introduzione

Il progetto consiste nello sviluppo di un bot Discord in grado di automatizzare la gestione e l'organizzazione di eventi Capture The Flag (CTF) all'interno di un server Discord, basandosi sui dati forniti dall'API di CTFTime. Questo bot permetterà agli amministratori del server di creare facilmente canali, ruoli, e notifiche per ogni evento CTF, semplificando la comunicazione e il coinvolgimento dei membri della community.

Obiettivi del progetto

- **Automatizzare la gestione delle CTF** all'interno dei server Discord, riducendo il carico di lavoro per gli amministratori e aumentando l'efficienza nell'organizzazione di eventi CTF.
- **Creare e gestire canali, ruoli e eventi Discord** specifici per ogni CTF, facilitando la comunicazione e il coordinamento tra i partecipanti.
- **Integrare con la piattaforma CTFTime** per recuperare e utilizzare informazioni accurate e aggiornate sugli eventi CTF.

Requisiti funzionali

1. **Creazione automatica di canali dedicati**:
    - Dato un URL di un evento CTF su CTFTime, il bot crea un canale dedicato per l'evento nella categoria "CTF Attive" precedentemente configurata.
    - Il bot invia un messaggio con embed nel canale, contenente informazioni dettagliate sull'evento CTF, inclusi titolo, descrizione, date di inizio e fine, logo e link al sito di CTFTime.
2. **Gestione degli eventi Discord**:
    - Il bot crea un evento Discord per ogni CTF con le date di inizio e fine recuperate da CTFTime.
    - All'inizio e alla fine dell'evento, il bot invia automaticamente notifiche ai partecipanti, aggiornando lo stato dell'evento nel server.
3. **Gestione dei ruoli**:
    - Creazione di un ruolo associato alla CTF per i partecipanti, con nome e permessi personalizzabili.
    - Assegnazione automatica del ruolo ai membri che reagiscono all'embed relativo all'evento CTF nel canale dedicato, semplificando il processo di adesione.
4. **Notifiche per l'inizio e la fine di ogni evento**:
    - Quando l'evento inizia, il bot tagga automaticamente i membri con il ruolo associato nel canale della CTF, inviando un messaggio di avvio.
    - Quando l'evento termina, il bot invia un messaggio di avviso e sposta automaticamente il canale nella categoria "CTF Archiviate", organizzando gli eventi passati.
5. **Configurazione delle categorie e dei ruoli**:

- Comando di configurazione per definire la categoria delle CTF attive, la categoria delle CTF archiviate e il ruolo minimo richiesto per gestire le CTF (basato sulla posizione dei ruoli).
- Solo gli utenti con il ruolo minimo o superiore possono creare nuovi eventi CTF.

Requisiti non funzionali

1. **Scalabilità**:
   - Il bot deve essere in grado di gestire più CTF in contemporanea su diversi server, permettendo l'uso su larga scala.
2. **Affidabilità**:
   - Gestione robusta degli errori durante la creazione di canali, eventi e ruoli. Eventuali problemi devono essere segnalati in tempo reale all'amministratore del server con messaggi di errore appropriati.
3. **Sicurezza**:
   - Accesso limitato ai comandi di gestione delle CTF, configurabile tramite ruoli. Solo gli utenti con il ruolo minimo (o superiore) possono eseguire comandi per la creazione e gestione delle CTF.
   - Gestione sicura dei dati ricevuti da CTFTime per prevenire possibili exploit.
4. **Manutenibilità e facilità di configurazione**:
   - Documentazione chiara per gli amministratori su come configurare e utilizzare il bot.
   - Architettura del bot che permette facilmente modifiche e aggiornamenti, garantendo compatibilità con futuri cambiamenti dell'API di CTFTime o di Discord.
5. **Tempo di risposta**:
   - Il bot deve rispondere ai comandi in modo rapido, minimizzando i tempi di attesa per gli utenti, specialmente durante la creazione di canali ed eventi.

Definizione delle feature (comandi)

Comando `/init`

- **Descrizione**: Comando di inizializzazione che configura le impostazioni di base per la gestione delle CTF nel server.
- **Parametri**:
  - `categoria_ctf_attive`: Categoria in cui saranno inseriti i canali delle CTF attive.
  - `categoria_ctf_archiviate`: Categoria in cui saranno spostati i canali delle CTF archiviate.
  - `ruolo_minimo`: Ruolo minimo richiesto per poter creare e gestire le CTF, basato sulla posizione dei ruoli nel server.
- **Funzionalità**:
  - Imposta le categorie e il ruolo minimo richiesti per le future creazioni di CTF.
  - Se la configurazione è completata correttamente, il bot conferma con un messaggio; in caso di errore, segnala il problema all'utente.

Comando `/create_ctf`

- **Descrizione**: Comando per creare un evento CTF all'interno del server.
- **Parametri**:
  - `url`: URL dell'evento su CTFTime nel formato `https://ctftime.org/event/<id>`.
- **Funzionalità**:
  - Recupera le informazioni sulla CTF dall'API di CTFTime, verificando la validità dell'URL.
  - Crea un canale dedicato nella categoria "CTF Attive", con un embed contenente i dettagli dell'evento.
  - Crea un evento Discord per la CTF con date di inizio e fine, notificando i partecipanti all'inizio e alla fine dell'evento.
  - Crea un ruolo dedicato alla CTF e lo assegna ai membri che reagiscono all'embed.
  - Se l'evento è già presente nel server, invia un messaggio di avviso informando che la CTF esiste già.

Diagramma di flusso delle operazioni principali

1. **Configurazione (`/init`)**:
   - Un utente con il permesso amministratore in qualsiasi canale invia il comando `/init` con i parametri necessari.
   - Il bot salva le categorie e il ruolo minimo per gestire le CTF.
   - Il bot conferma con un messaggio o segnala errori in caso di parametri mancanti.
2. **Creazione Evento CTF (`/create_ctf`)**:
   - Un utente con i permessi minimi definiti nel comando `/init` invia il comando `/create_ctf` in qualsiasi canale con l'URL dell'evento CTFTime.
   - Il bot verifica l'URL e recupera i dati dell'evento.
   - Il bot crea il canale nella categoria "CTF Attive".
   - Il bot crea l'evento Discord e il ruolo per i partecipanti.
   - Il bot invia un embed nel canale, permette l'assegnazione del ruolo con reazione, e notifica all'inizio e alla fine dell'evento.

Casi d'Uso

Caso d'Uso 1: Configurazione iniziale del bot

- **Attori**: Amministratore del server Discord
- **Descrizione**: L'amministratore del server esegue il comando `/init` per configurare le impostazioni di base per la gestione delle CTF.
- **Flusso principale**:
  1. L'amministratore esegue il comando `/init` specificando le categorie e il ruolo minimo per gestire le CTF.
  2. Il bot salva le impostazioni e conferma la configurazione.
- **Estensioni**:
  - Se il comando `/init` non include tutti i parametri necessari, il bot invia un messaggio d'errore e richiede le informazioni mancanti.
- **Pre-condizioni**: Colui che invia il messaggio deve avere i permessi necessari per eseguire il comando.
- **Post-condizioni**: La configurazione del bot è completata, pronta per l'uso.

Caso d'Uso 2: Creazione di un evento CTF

- **Attori**: Amministratore del server Discord
- **Descrizione**: L'amministratore crea un nuovo evento CTF utilizzando il comando `/create_ctf`.
- **Flusso principale**:
  1. L'amministratore invia il comando `/create_ctf` con l'URL dell'evento su CTFTime.
  2. Il bot recupera le informazioni sull'evento dall'API di CTFTime.
  3. Il bot crea un canale nella categoria "CTF Attive" e pubblica un embed con i dettagli dell'evento.
  4. Il bot crea un evento Discord per il CTF, definendo le date di inizio e fine.
  5. Il bot crea un ruolo per i partecipanti e assegna il ruolo agli utenti che reagiscono all'embed nel canale.
  6. Il bot programma le notifiche per l'inizio e la fine dell'evento.
- **Estensioni**:
  - Se l'URL dell'evento non è valido o l'API di CTFTime non risponde, il bot invia un messaggio d'errore all'amministratore.
  - Se l'evento CTF esiste già nel server, il bot invia un avviso senza duplicare i canali o ruoli.
- **Pre-condizioni**: Il bot deve essere stato configurato con il comando `/init`.
- **Post-condizioni**: L'evento CTF è creato con canale, embed informativo, evento Discord e ruolo partecipanti.

Caso d'Uso 3: Partecipazione dei membri a una CTF

- **Attori**: Membro del server Discord
- **Descrizione**: Un membro del server si unisce all'evento CTF reagendo all'embed informativo nel canale dedicato.
- **Flusso principale**:
  1. Il membro del server visualizza l'embed con le informazioni della CTF nel canale.
  2. Il membro reagisce all'embed con l'emoji appropriata.
  3. Il bot assegna automaticamente al membro il ruolo creato per la CTF.
- **Estensioni**:
  - Se l'utente non ha i permessi necessari per accedere al canale o reagire all'embed, il bot segnala l'errore e non assegna il ruolo.
- **Pre-condizioni**: L'evento CTF deve essere stato creato e il ruolo configurato per le reazioni.
- **Post-condizioni**: Il membro ottiene il ruolo della CTF e accesso completo al canale dedicato.

Caso d'Uso 4: Notifica di inizio evento

- **Attori**: Bot Discord
- **Descrizione**: Quando la data di inizio di un evento CTF è raggiunta, il bot invia automaticamente una notifica nel canale dedicato.
- **Flusso principale**:
  1. All'ora di inizio dell'evento, il bot invia un messaggio nel canale della CTF, taggando i membri con il ruolo associato per informarli dell'inizio dell'evento.
- **Pre-condizioni**: Il canale e il ruolo della CTF devono essere stati creati, e il bot deve avere i permessi per inviare notifiche.
- **Post-condizioni**: I partecipanti vengono notificati dell'inizio della CTF.

Caso d'Uso 5: Conclusione e archiviazione dell'evento

- **Attori**: Bot Discord
- **Descrizione**: Alla fine di un evento CTF, il bot notifica i partecipanti e archivia il canale.
- **Flusso principale**:
    1. Alla data di fine dell'evento, il bot invia un messaggio nel canale della CTF informando i partecipanti della conclusione dell'evento.
    2. Il bot sposta automaticamente il canale della CTF nella categoria "CTF Archiviate" per organizzare gli eventi passati.
- **Pre-condizioni**: L'evento deve essere attivo e il bot configurato con le categorie per l'archiviazione.
- **Post-condizioni**: Il canale viene archiviato, i membri vengono informati del termine dell'evento e il ruolo viene reso grigio chiaro (tutti i ruoli creati dopo dal bot verrano messi sopra il ruolo precedente in modo che sia sempre messo in primo piano il ruolo della ctf attiva).

Considerazioni aggiuntive

- **Integrazione API**: Assicurarsi che l'API di CTFTime sia accessibile e gestire eventuali limiti di rate per evitare blocchi.

Conclusione

Il bot Discord proposto automatizzerà la gestione delle CTF, semplificando la creazione e la gestione di eventi complessi e migliorando l’esperienza degli utenti del server. Grazie alla sua integrazione con CTFTime, il bot fornirà informazioni sempre aggiornate sulle CTF, mentre l'automazione dei canali, eventi e ruoli renderà l'organizzazione efficiente e professionale.

Requirements Analysis Document (RAD)

Introduction

This project consists of the development of a Discord bot that can automate the management and organization of Capture The Flag (CTF) events within a Discord server, relying on data provided by the CTFTime API. This bot will allow server administrators to easily create channels, roles, and notifications for each CTF event, simplifying communication and engagement with community members.

Project goals

- **Automatize CTF management** within Discord servers, reducing the workload for administrators and increasing efficiency in organizing CTF events.
- **Create and manage channels, roles, and Discord** events specific to each CTF, facilitating communication and coordination among participants.
- **Integrate with the CTFTime** platform to retrieve and use accurate and up-to-date information about CTF events.

Functional requirements

1. **Automatic creation of dedicated channels**:
    - Given a CTF event URL on CTFTime, the bot creates a dedicated channel for the event in the previously configured “CTF Active” category.

    - The bot sends an embed message to the channel, containing detailed information about the CTF event, including title, description, start and end dates, logo, and link to the CTFTime site.
2. **Discord event management**:
    - The bot creates a Discord event for each CTF with start and end dates retrieved from CTFTime.
    - At the beginning and end of the event, the bot automatically sends notifications to participants, updating the status of the event in the server.
3. **Role Management**:
    - Creating a role associated with CTF for participants, with customizable name and permissions.
    - Automatic role assignment to members who react to the embed related to the CTF event in the dedicated channel, simplifying the joining process.
4. **Notifications for the start and end of each event**:
    - When the event starts, the bot automatically tags members with the associated role in the CTF channel, sending a start message.
    - When the event ends, the bot sends an alert message and automatically moves the channel to the “CTF Archived” category, organizing past events.
5. **Configuration of categories and roles**:
    - Configuration command to define the category of active CTFs, the category of archived CTFs, and the minimum role required to manage CTFs (based on role position).
    - Only users with the minimum role or higher can create new CTF events.

Non-functional requirements

1. **Scalability**:
    - The bot must be able to handle multiple CTFs simultaneously on different servers, allowing for large-scale use.
2. **Reliability**:
    - Robust error handling during channel, event and role creation. Any problems should be reported in real time to the server administrator with appropriate error messages.
3. **Security**:
    - Limited access to CTF management commands, configurable via roles. Only users with the minimum role (or higher) can execute commands for creating and managing CTFs.
    - Secure management of data received from CTFTime to prevent possible exploits.
4. **Maintenance and ease of configuration**:
    - Clear documentation for administrators on how to configure and use the bot.
    - Bot architecture that easily allows changes and updates, ensuring compatibility with future CTFTime or Discord API changes.
5. **Response time**:
    - The bot must respond to commands quickly, minimizing wait time for users, especially when creating channels and events.

Feature definition (commands)

Command `/init`

- **Description**: Initialization command that configures basic settings for managing CTFs in the server.
- **Parameters**:
    - `category_ctf_active`: Category in which the channels of active CTFs will be placed.

    - `category_ctf_archived`: Category into which the channels of archived CTFs will be moved.
    - `minimum_role`: Minimum role required to be able to create and manage CTFs, based on the location of roles in the server.
- **Functionality**:
    - Sets the categories and minimum role required for future CTF creations.
    - If the configuration is completed correctly, the bot confirms with a message; if there is an error, it reports the problem to the user.

Command `/create_ctf`

- **Description**: Command to create a CTF event within the server.
- **Parameters**:
    - `url`: URL of the event on CTFTime in the format `https://ctftime.org/event/<id>`.
- **Functionality**:
    - Retrieves CTF information from the CTFTime API, checking the validity of the URL.
    - Creates a dedicated channel in the "CTF Active" category, with an embed containing event details.
    - Create a Discord event for the CTF with start and end dates, notifying participants at the beginning and end of the event.
    - Creates a dedicated role for the CTF and assigns it to members who react to the embed.
    - If the event already exists in the server, sends an alert message informing that the CTF already exists.

Flowchart of main operations

1. **Configuration (`/init`)**:
    - A user with administrator permission in any channel sends the `/init` command with the necessary parameters.
    - The bot saves categories and minimum role to manage CTFs.
    - The bot confirms with a message or reports errors in case of missing parameters.
2. **Create CTF Event (`/create_ctf`)**:
    - A user with the minimum permissions defined in the `/init` command sends the `/create_ctf` command in any channel with the CTFTime event URL.
    - The bot verifies the URL and retrieves the event data.
    - The bot creates the channel in the "CTF Active" category.
    - The bot creates the Discord event and role for participants.
    - The bot sends an embed into the channel, allows role assignment with reaction, and notifies at the beginning and end of the event.

Use Cases

Use Case 1: Initial setup of the bot

- **Actors**: Discord server administrator.
- **Description**: The server administrator runs the `/init` command to configure basic settings for handling CTFs.
- **Main Flow**:

    1. The administrator runs the `/init` command specifying the categories and minimum role to manage CTFs.
    2. The bot saves the settings and confirms the configuration.
- **Extensions**:
    - If the `/init` command does not include all the necessary parameters, the bot sends an error message and requests the missing information.
- **Pre-conditions**: The one sending the message must have the necessary permissions to execute the command.
- **Post-conditions**: The bot configuration is completed, ready for use.

Use Case 2: Creating a CTF event

- **Actors**: Administrator of the Discord server.
- **Description**: Administrator creates a new CTF event using the `/create_ctf` command.
- **Main Flow**:
    1. The administrator sends the `/create_ctf` command with the event URL to CTFTime.
    2. The bot retrieves the event information from the CTFTime API.
    3. The bot creates a channel in the “CTF Active” category and publishes an embed with the event details.
    4. The bot creates a Discord event for the CTF, defining the start and end dates.
    5. The bot creates a role for participants and assigns the role to users who react to the embed in the channel.
    6. The bot schedules notifications for the start and end of the event.
- **Extensions**:
    - If the event URL is invalid or the CTFTime API is not responding, the bot sends an error message to the administrator.
    - If the CTF event already exists in the server, the bot sends an alert without duplicate channels or roles.
- **Pre-conditions**: The bot must have been configured with the `/init` command.
- **Post-conditions**: CTF event is created with channel, information embed, Discord event and participant role.

Use Case 3: Member participation in a CTF

- **Actors**: Member of the Discord server.
- **Description**: A server member joins the CTF event by reacting to the information embed in the dedicated channel.
- **Main stream**:
    1. The server member displays the embed with the CTF information in the channel.
    2. The member reacts to the embed with the appropriate emoji.
    3. The bot automatically assigns the role created for the CTF to the member.
- **Extensions**:
    - If the member does not have the necessary permissions to access the channel or react to the embed, the bot reports the error and does not assign the role.
- **Pre-conditions**: The CTF event must have been created and the role configured for reactions.
- **Post-conditions**: Member gets the CTF role and full access to the dedicated channel.

Use Case 4: Event Start Notification

- **Actors**: Discord bot.
- **Description**: When the start date of a CTF event is reached, the bot automatically sends a notification in the dedicated channel.
- **Main Stream**:
    1. At the start time of the event, the bot sends a message in the CTF channel, tagging members with the associated role to inform them of the start of the event.
- **Pre-conditions**: The CTF channel and role must have been created, and the bot must have permissions to send notifications.
- ◦ **Post-conditions**: Participants are notified of the start of the CTF.

Use Case 5: Conclusion and archiving of the event

- **Actors**: Discord bot.
- **Description**: At the end of a CTF event, the bot notifies participants and archives the channel.
- **Main Flow**:
    1. On the end date of the event, the bot sends a message in the CTF channel informing participants that the event has ended.
    2. The bot automatically moves the CTF channel to the “CTF Archived” category to organize past events.
- **Pre-conditions**: The event must be active and the bot configured with categories for archiving.
- **Post-conditions**: The channel is archived, members are notified of the end of the event, and the role is made light gray (all roles created later by the bot will be placed on top of the previous role so that the role of the active ctf is always foregrounded).

Additional considerations

- **Integration API**: Ensure that the CTFTime API is accessible and manage any rate limits to avoid blockages.

Conclusion

The proposed Discord bot will automate the management of CTFs, simplifying the creation and management of complex events and improving the experience of server users. Through its integration with CTFTime, the bot will provide up-to-date information on CTFs, while the automation of channels, events, and roles will make the organization efficient and professional.