# List of methods

## Frontend

* Ability Modifier
  + Ability Scores + Custom Modifier + Proficiency
* Proficiency Calculations
  + Based on player level
* Hit Calculation
* Damage Calculations
* SortQueue
  + Sorts queue based on initiative result in a list [player, initiative score].
* Update queue change

## Backend

### Player

* GetPlayerInformation
  + Id
  + Name
  + Level
  + Class
  + Armour Class
  + Attribute
  + Skills
  + Defences
    - Resistance
    - Conditions
    - Defences
  + Notes
* GetPlayerInfromationBrief
  + Id
  + Name
  + Level
  + Class
* UpdatePlayerInformation
* Player Information Saved

### Character

* UpdateCustomModifers
  + Saved as -1/-2/-3 or +1/+2/+3 so when it’s
  + Properties that can only be changed
    - ~~Attribute -~~ The update attribute will just override the attribute number to avoid future complications when using it in other components in the code
    - Skills - Override local calculation of skills. Otherwise, it is calculated
    - Initiative change - Override initiative modifier locally. Otherwise, it is calculated
* UpdateEncounterQueue

### Monster

* GetMonsterInformation
  + Id
  + Name
  + Level
  + Class
  + Armour Class
  + Attribute
  + Skills
  + Defences
    - Resistance
    - Conditions
    - Defences
  + Notes
* CreateMonster

### Turn

* **Reset Turn**
  + Action
  + Weapon
  + Target
  + custom
  + HitDiceRoll
  + Damagediceroll
  + bonusAction
  + Reaction

### Encounter

* **CreateEncounter**
* **UpdateEncounter**
* **ResetEncounterTurns**

**Referenced document**

<https://docs.google.com/document/d/1owj6X3lDuwFTmgCp1bkV4Js8VPKmkc23KPMbYcUj9h0/edit?usp=sharing>

# Keys

**BOLD**: Database calls

*Italics*: Handled in front end

Green: Implemented

Yellow: In progress

Red: Blocker

None: I haven’t started

# 3. Home

* 1. **GetUserName** - DM’s Username
* Username
  1. **GetUserCampaigns** - Sorted by the most recent time it was created. From database
* Campaign id
* Campaign names
* The campaign's last updated dates
  1. *Switch pages from home -> selected campaign dashboard*
     1. **GetCampaignInformation** - Specific campaign from its id
* [Campaign information](#_v1ck4tx60pqt)
  1. [**CreateCampaign**](#_a8ezdy4gfioi)
     1. *Open a model for creating a Campaign*
  2. *Sign out method*

## Create Campaign

1. **Campaign name - CreateCampaign**
2. *Close the model without saving*
3. *Create a new campaign in the home*
   1. *Greyed out when campaign name is null*
   2. *Return to Campaign List in Home*

# 4. Campaign Dashboard

1. **GetCampaignInformation**

* Name
* Notes
* Player list
  + Name
  + Level
  + Class
* NPC

1. *The create button opens the character screen:*

* [Create Character](#_59v0qakx5zki)

1. *Open import character model*

* [Import Character](#_59v0qakx5zki)

1. *Close without import*

## Import Character

1. Handled by AI data collected

* **CreatePlayer**
* Current HP is updated to Max HP if it is blank
* Temp HP is 0
* Saving throws for successful and fail are converted to 1 boolean list
  + Saving throws are added first then failed throws

# 5. Character Sheet

1. **Player Information Saved**

* Character name
* Class
* Level
* Ability Score
  + Strength
  + Dexterity
  + Constitution
  + Intelligence
  + Wisdom
  + Charisma
* Armor Class
* Proficiency - (Specific Skill/Attribute)
* Max HP
* Current HP
* Temp HP
* Conditions
* Defences
  + Resistances
  + Immunities
  + Vulnerability
* Weapons
  + Add ID to the list to save in the database
* Equipment list
* Custom changes to Ability Score/Skill/Initiative

1. **Weapons**

* Hit Modifier
* Amount and Damage Dice
  + [Amount, Dice Types]
* Damage Modifier
  + Extra damage on top of dice
* Ability Score Type
  + What the weapon scales off
* Notes

1. *Information Calculated*

* *Ability Modifiers*
  + When updating update the ability in the database to save changes.
* *Skills*
  + When updating add to the “**CustomModifiers**” list in the database to save changes. This list adds/minus values for modifications
* *Initiative*
  + When updating add to the “**CustomModifiers**” list in the database to save changes. This list adds/minus values for modifications
* *Proficiency Bonus*

1. Save changes

* **CreatePlayerStats/UpdatePlayerStats**

1. *Go back to the campaign dashboard*

# 6. Encounter

## Initiative Queue

* Initial Queue
  + Position in initiative queue
    - *SortQueue* - Sorts queue based on initiative result in a list [player, initiative score].
  + Can be modified in the encounter queue
    - *Update queue position*
    - **Update turn’s list information**
  + *The player in the queue* (maybe get all information from player/enemies from the database when launching the encounter screen)
    - Current HP (Current + temp)
    - Max HP
    - Armor Class
    - Action
    - Bonus
    - Reaction
    - Death Save
  + If a player has 0 HP they have to do saving throws
* Reset all character turns
  + **Reset Characters Turn**
    - It goes through the list and resets the player or enemy turn in MongoDB
  + *Reset Characters*
    - Reset Action, Bonus Action and Reactions buttons

## Add Character - Modal

* Player
  + Get Player (NPC in the same database) information:
    - Name
    - Level
    - Class
* Bestiary
  + Get Enemy information
    - Name
    - Challenge Rating
* Generate Encounter
  + **Add enemy information to the database**
  + **Add enemy into encounter list**
  + **Add enemy into current Initiative Queue**

## Current Turn

1. Character information in turn (Information was received at the start of the encounter)

* **Player**
  + Current HP
  + Max HP
  + Temp HP
  + Armor Class
  + Death saves
  + Conditions
  + Defences
  + Note
* **Turn**
  + *Action*
    - Attack, Dash, Disengage, Dodge, Grapple, Help, Hide, Improvise, Influence, Magic, Ready, Search, Shove, Study, Utilize
  + *Target*
    - Select multiple characters (ObjectID)
  + *Roll / Auto Roll*
    - Manual or automatically rolls but it’s an int input
* **Execute Action (Has weapon)**
  + *CalculateCharacterHit*
    - Hit value = Roll + Attack modifier
    - AC of the target
    - AC > Hit value
  + Generate Attack Log

1. Conditions (Information was received at the start of the encounter)

* *Display character’s defences:*
  + Resistances
  + Immunities
  + Vulnerability

1. Target

* *Display target information*
  + Current HP
  + Max HP
  + Temp HP
  + Armor Class
  + Death Saves
  + Conditions
  + Defences

1. Combat log (Combat log is included in GetEncounterInformation)

* **GetCombatLog**
* *Create combat log*
  + Preset structure for weapons
    - Justin swings a short sword at a goblin for 5 damage
    - Justin swings a short sword at a goblin and missed
  + Add target and player
    - If multiple targets it prints separate lines
  + Add “\n” at the start of the log before anything else