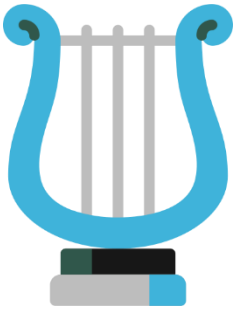


Designmodell

Designbeschreibung der ersten Iteration

Version 4.2

Team Apollon:

Paul-Benedict Burkard

Florian Albert

Leon Jerke

Daniel Kröker

Alfred Rustemi

Etienne Zink

Projektbezeichnung	APOLLON
Projektleiter	Paul-Benedict Burkard
Verantwortlich	Etienne Zink
Versionsdatum	16.05.2021

Inhaltsverzeichnis

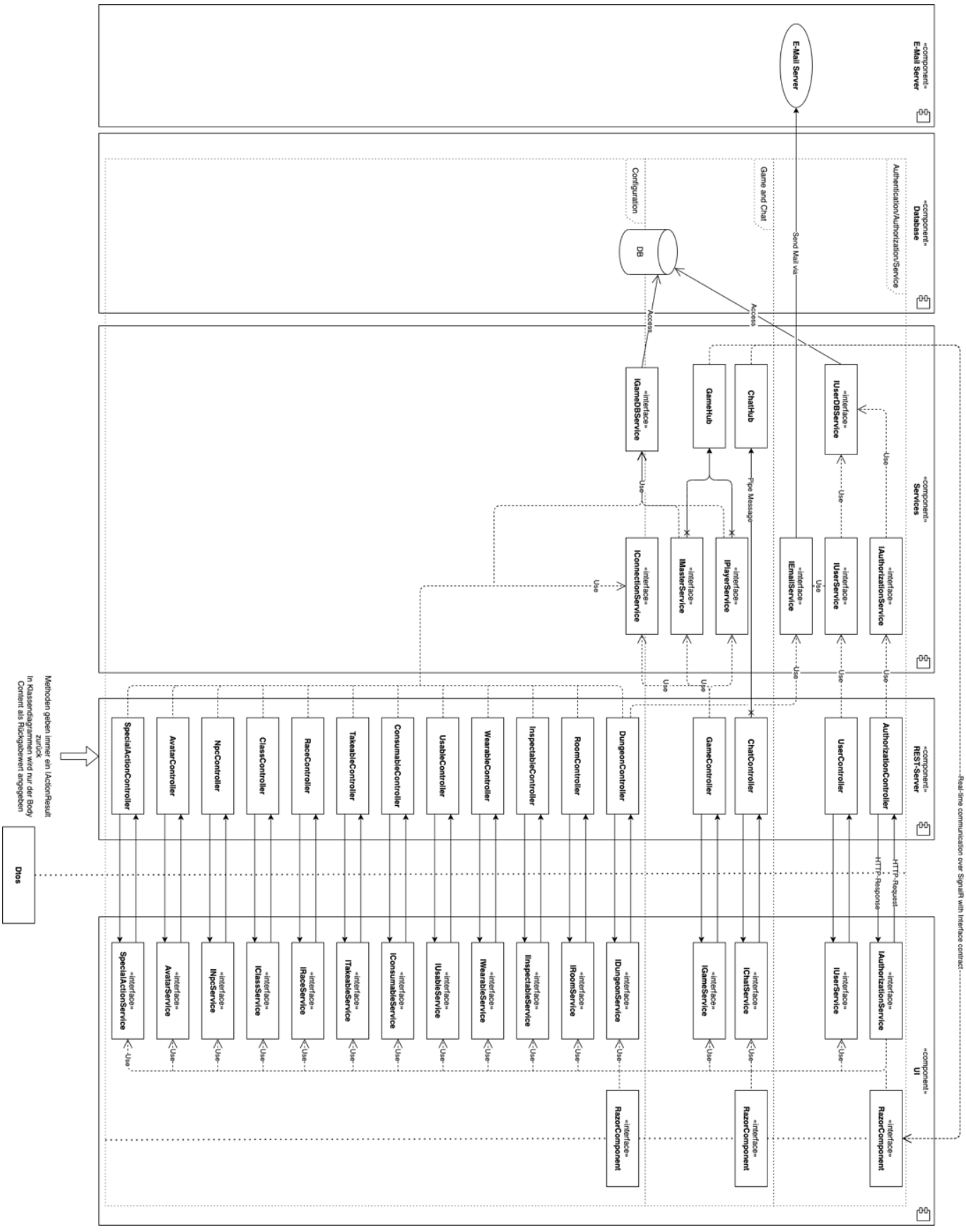
1. Zielsetzung

Ziel des Projekts ist die Entwicklung einer Software für einen Kunden. Der Kunde soll den Server übernehmen und selbstständig betreiben. Durch die Software sollen Spieler über den Web Browser textbasierte Multi User Dungeons selbst konfigurieren und an von anderen Spielern konfigurierten Dungeons, mit einem erstellten Avatar, teilnehmen können.

Somit ist das Ziel dieses Dokuments der erste Designentwurf dieser Software genauer zu erläutern. Dieser Entwurf enthält sowohl die geplante Gesamtarchitektur als auch die einzelnen Komponenten und deren erste Planung. Hierbei wurden die Ansichten des Modells nach statischem und dynamischem Design unterschieden. Außerdem enthält das Dokument einen Entwurf der genutzten Datenbank, als auch des User-Interfaces.

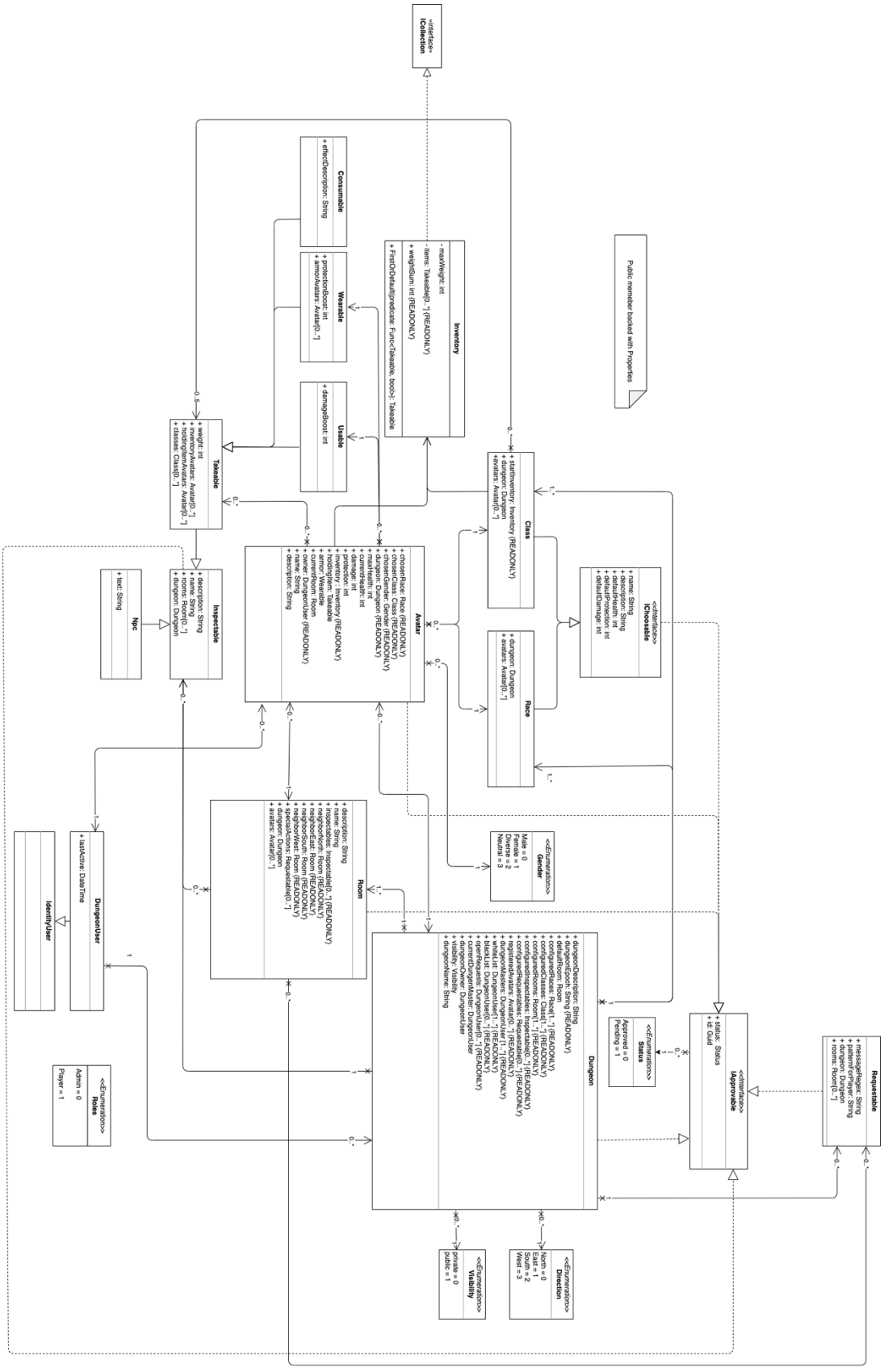
2. Gesamtarchitektur

Die Gesamtarchitektur setzt sich aus den Komponenten des „E-Mail-Server“, „Database“, „Services“, „REST-Server“ und „UI“ zusammen. Über die Komponenten hinaus wird die Software außerdem in unterschiedliche Schichten unterteilt. Diese sind die folgenden Schichten: „Authentication/Authorization/Service“, „Game and Chat“ und „Configuration“. Dargestellt sind hauptsächlich die benutzten Interfaces, welche durch Dependency-Injection durch die dedizierten Klassen zur Laufzeit ersetzt werden.


«component»
E-Mail Server
E-Mail Server
Send Mail via
«component»
Database
Authentication/Authorization/Service
Configuration
DB
Game and Chat
Access
Access
«component»
Services
«interface»
IGameDBService
GameHub
ChatHub
«interface»
IUserDBService
Use
Pipe Message
«interface»
IConnectionService
«interface»
IMasterService
«interface»
IPlayerService
«interface»
IEmailService
«interface»
IUserService
«interface»
IAuthorizationService
«component»
REST-Server
SpecialActionController
AvatarController
NpcController
ClassController
RaceController
TakeableController
ConsumableController
UsableController
WearableController
InspectableController
RoomController
DungeonController
GameController
ChatController
UserController
AuthorizationController
HTTP-Request
HTTP-Response
Real-time communication over SignalR with Interface contract
Methoden geben immer ein IActionResult zurück
In Klassendiagrammen wird nur der Body Content als Rückgabewert angegeben
Dtos
«component»
UI
«interface»
ISpecialActionService
«interface»
IAvatarService
«interface»
INpcService
«interface»
IClassService
«interface»
IRaceService
«interface»
ITakeableService
«interface»
IConsumableService
«interface»
IUsableService
«interface»
IWearableService
«interface»
IInspectableService
«interface»
IRoomService
«interface»
IDungeonService
«interface»
IGameService
«interface»
IChatService
«interface»
IUserService
«interface»
IAuthorizationService
«interface»
RazorComponent
«interface»
RazorComponent
«interface»
RazorComponent
Use

3. Statisches Modell

3.1Darstellung der Datenmodelle

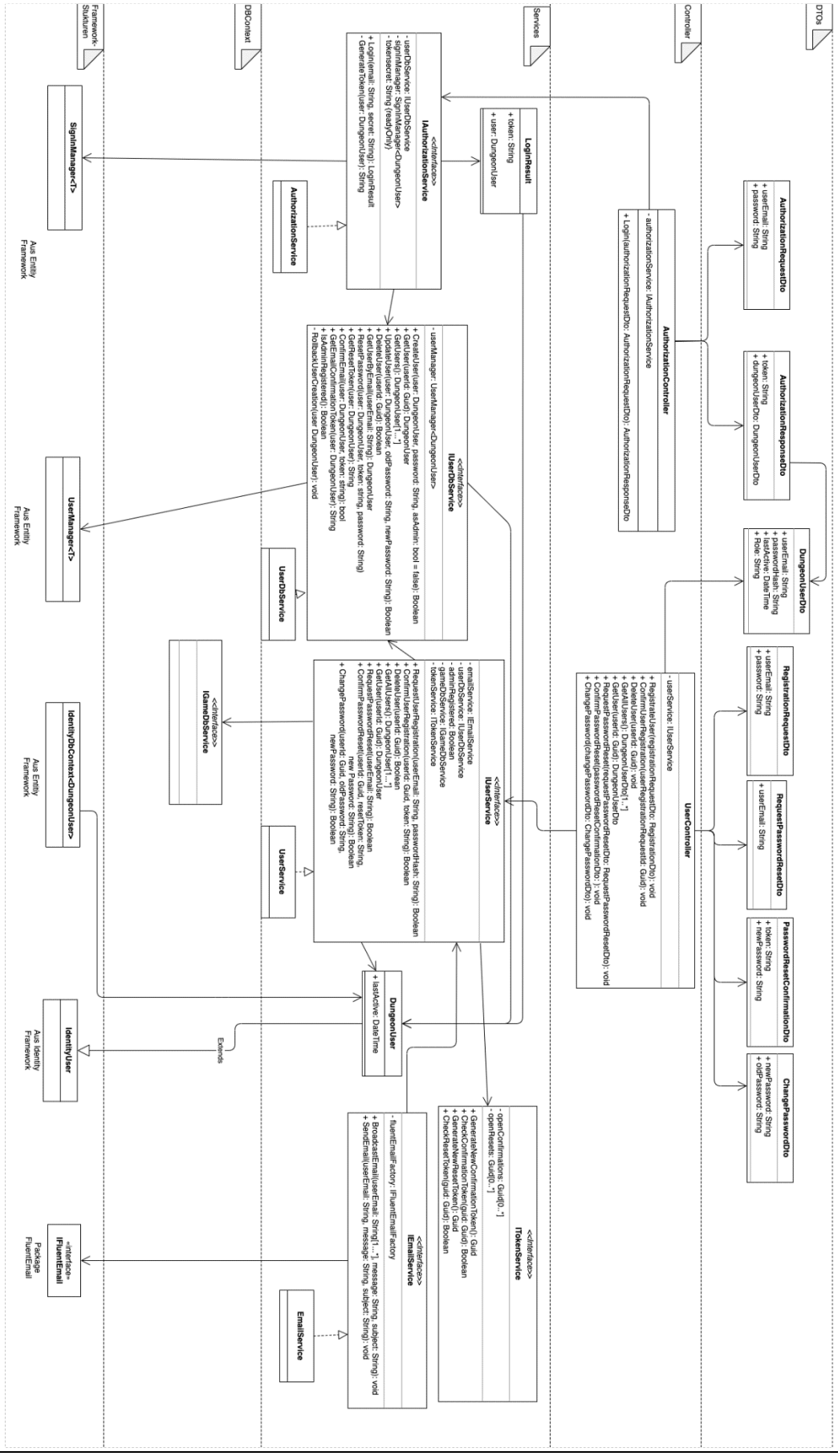


Die Klassen werden hierbei wie in der Darstellung durch Interfaces abstrahiert. Dies bietet in C# die Möglichkeit, die Attribute trotzdem mittels Properties zu generalisieren. Die Interfaces bzw. deren konkreten Klassen werden außerdem nicht direkt über den Rest-Server versendet. Hierzu werden sogenannte Dto's generiert. Ein Dto einer Klasse beinhaltet alle Attribute, welche intrinsische Datentypen oder Strings sind.

Änderungen:

- Darstellung nicht mehr als Interfaces sondern Klassen, da unser Datenbankkontext dies benötigt
- Avatar erbt nicht mehr von Inspectable
- Room hat dadurch eine Liste an Avataren
- Avatar hat eigenes Objekt der Klasse Inventory statt Liste an Takeables

3.2 User Management



Das Usermanagment erfolgt über zwei verschiedene REST-Controller, den Authorization- und den UserController um administrative von anwendungsbezogenen Aufgaben zu trennen.

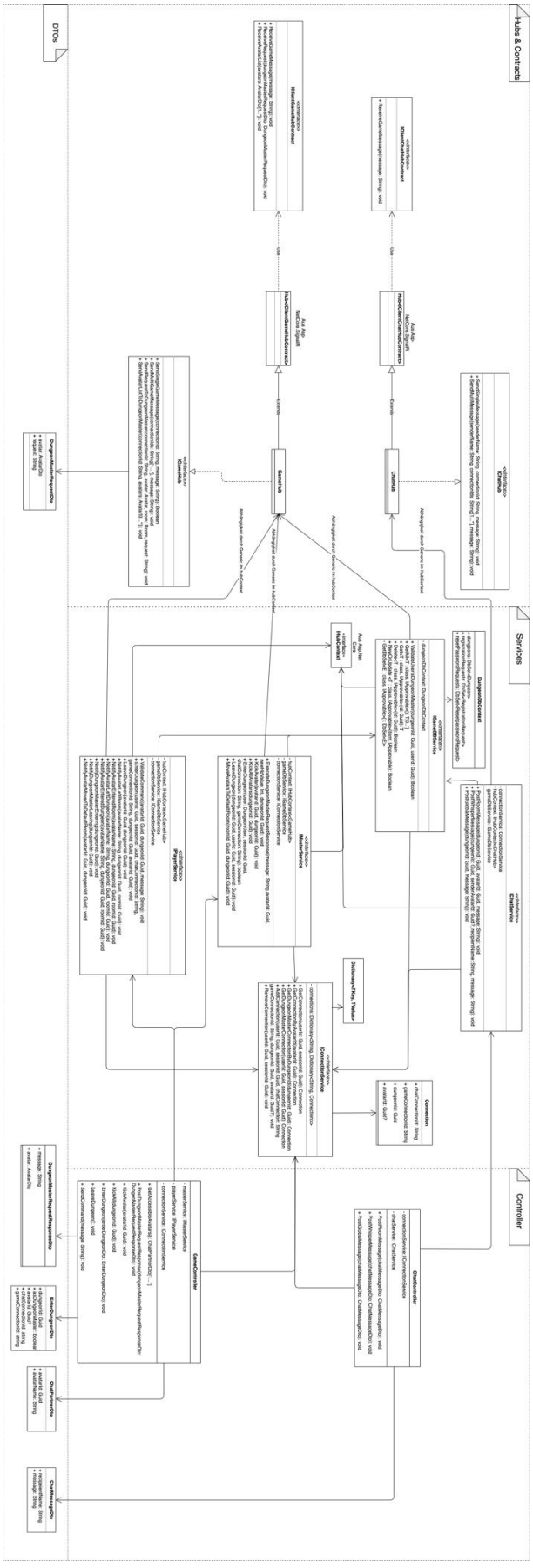
Die Autorisierung erfolgt dabei mithilfe von JSON Web Tokens, die bei jeglichen Anfragen an geschützte Routen entscheidende Informationen mitliefern, sowie die Identität des anfragenden Clients bestätigen. Ein solches Token wird über den AuthorizationController erzeugt und dem Client zurückgeliefert.

Sonstige administrative Funktionen zu Usern werden über den UserController geregelt, wie etwa die Registrierung, Löschung von Benutzern, oder „Passwort vergessen“-Funktion.

Änderungen:

- Neue Methoden zu UserDB Service hinzugefügt
- DungeonDBContext entfernt
- IGameDBService hinzugefügt
- TokenService hinzugefügt

3.3 Game Logik



Die Game-Logik basiert auf dem Chat- und GameController. Der GameController ist dabei für alles um das Spielgeschehen zuständig, während der ChatController nur Anfragen für ausgehende Nachrichten annimmt.

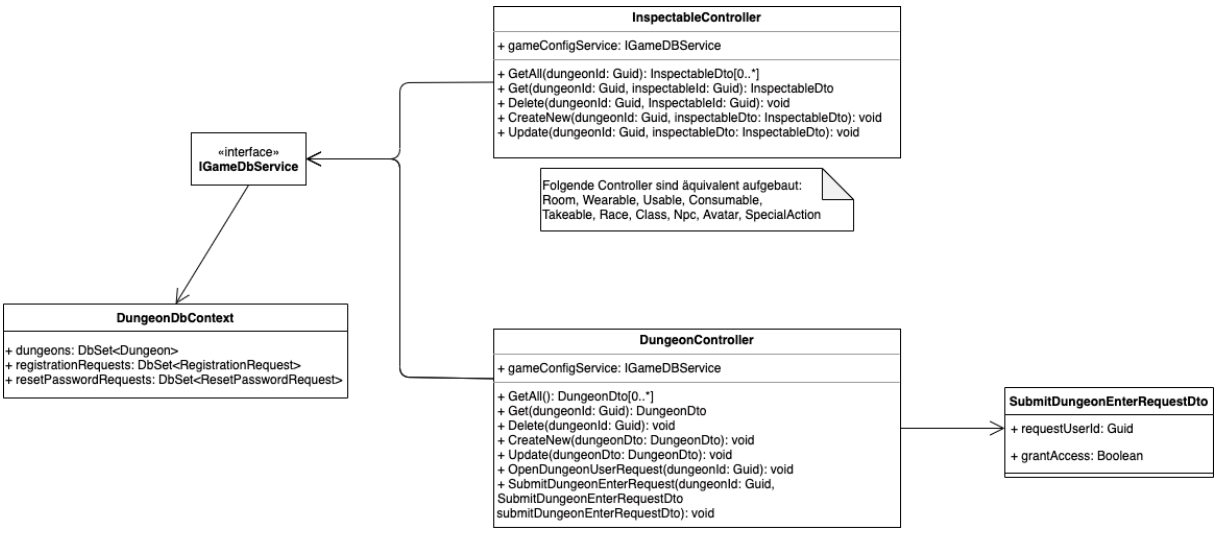
Gameanfragen werden über verschiedene Services validiert und deren Konsequenzen in die Datenbank durchgepiped. Speziell konfigurierte Texteingaben eines Spielers werden über den GameHub über eine SignalR-Verbindung an den diensthabenden Dungeon-Master geschickt, welcher wieder über den GameController seine Antworten posten kann. Über sämtliches Spielgeschehen werden nach der Auswertung die betroffenen Spieler über den GameHub benachrichtigt, sodass ihre Konsole serverseitig aktualisiert werden kann.

Chatanfragen werden über den Chatcontroller geposted. Die Empfänger entweder über den konkreten Avatarnamen aufgelöst oder über die Referenz über den aktuellen Raum des Senders ermittelt.

Änderungen:

- Neue Methoden zu Master- und Playerservice hinzugefügt
- GameController neue Methoden hinzugefügt
- Chat Service Anpassung an PostRoomMessage Methode

3.4 Konfiguration



Die Konfiguration eines Dungeons erfolgt über verschiedenste Controller. Je nachdem welche Bestandteile geändert werden sollen, gibt es dafür verschiedene Controller, um möglichst granulare Änderungen vornehmen zu können.

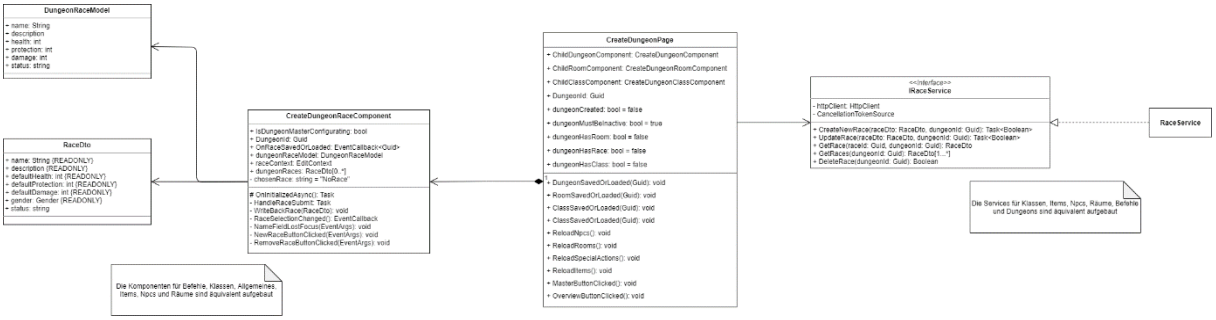
Sie greifen alle auf einen generischen Datenbank-Service zu, der abhängig vom generischen Typ auf den unterschiedlichen Bestandteilen des Dungeons Veränderungen ausführt. Sollten diese Änderungen das Spielgeschehen beeinflussen, kann der Service über den GameHub sämtliche betroffenen Spieler erreichen und diese über Änderungen informieren.

Dies wurde in der Übersicht exemplarisch anhand des InspectableContollers und des DungeonControllers gezeigt. Alle weiteren Controller, die der Gesamtübersicht zu entnehmen sind, funktionieren auf äquivalente Weise.

Änderungen:

- Connection Service entfernt, da die angedachten „Querverbindungen“ (bspw. Dungeon-Master zum zugehörigen Dungeon) über den IGameDBService realisiert werden konnten.

3.5 Frontend: Erstellen eines Dungeon



Die Konfiguration eines Dungeon im Frontend erfolgt zentral über die entsprechende Razor Komponente. In dieser sind die nötigen Datenstrukturen zum Darstellen der Daten auf Clientseite abgelegt und die Methoden zum Verarbeiten nutzerabhängiger Events implementiert. Zum Weitergeben der Daten werden diese an den jeweiligen Frontendservice weitergereicht. Hierfür werden die angegebenen Data-Transfer-Objects verwendet.

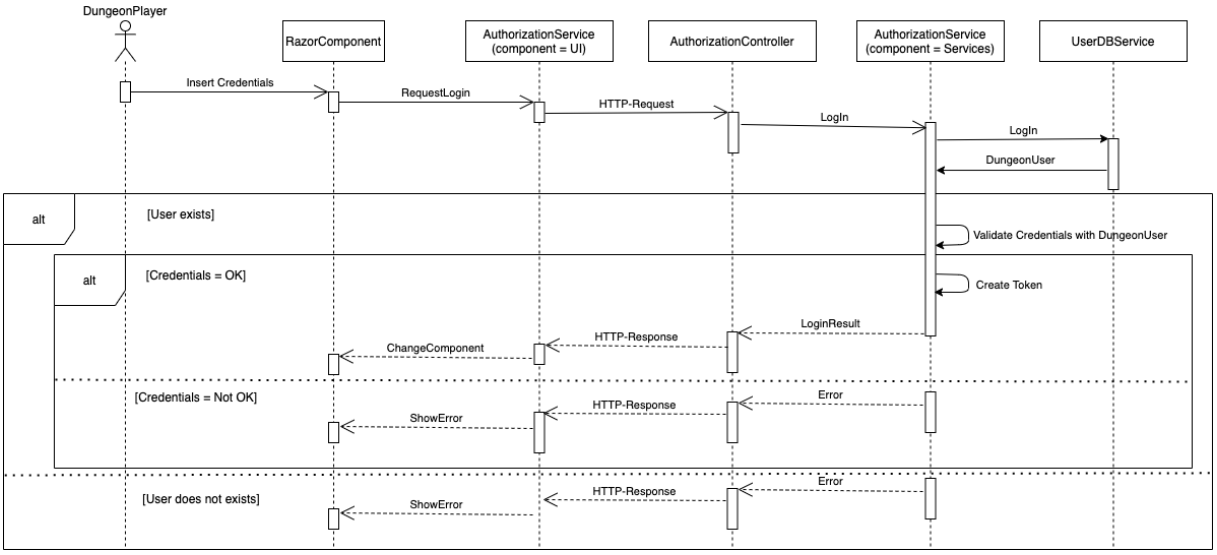
Dieses spezifische Modell zur Erstellung eines Dungeons gilt beispielhaft für die weiteren Razor Komponenten, da die zugrundeliegende, statische Funktionsweise sich bei diesen nur minimal unterscheidet.

Änderungen:

- Komplette Überarbeitung: Aufteilung in einzelne, kategorische Komponenten, als Kompositionen einer Page
- Einfügen der Model-Klassen zur Validierung

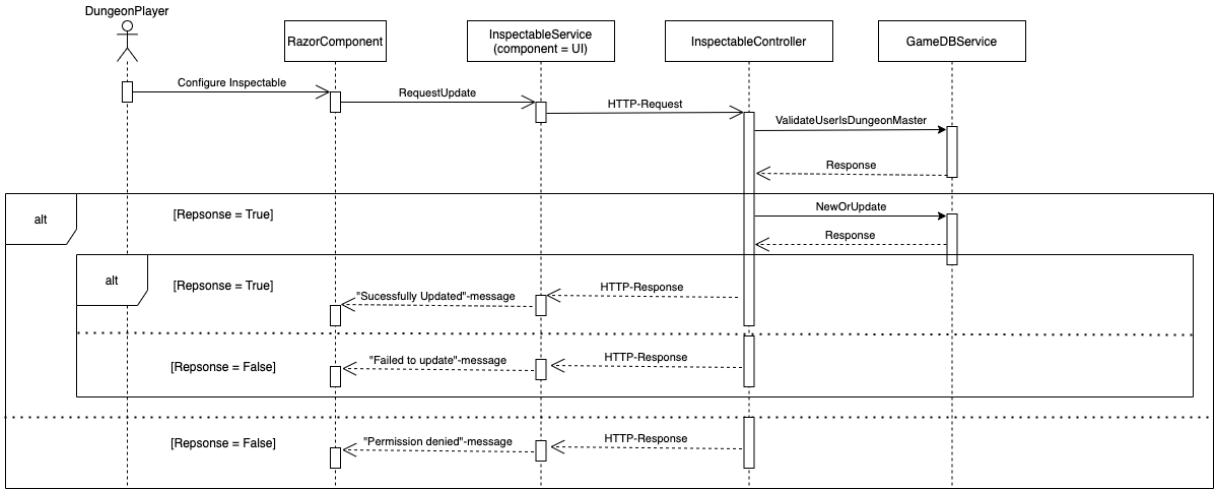
4. Dynamisches Modell

4.1 Autorisierung



Obenstehend ist ein schematischer Ablauf einer Autorisierung in Form eines Sequenzdiagramms dargestellt. Hierbei ist sowohl das Verhalten des Systems bei einem falschen Benutzer oder Passwort als auch bei einer erfolgreichen Autorisierung dargestellt. Der Benutzer gibt hierbei über seine Benutzeroberfläche, welche als RazorComponent dargestellt ist, seine Anmeldedaten ein. Hiernach werden diese durch das System wie dargestellt versendet und anschließend validiert. Auf Grundlage dessen, erfolgt dann die Antwort des Systems über eine HTTP-Response des AuthorizationControllers.

4.2 Konfiguration eines bereits bestehenden Inspectables



5. Datenbank

Aufgrund der Größe ist diese extern dargestellt. Siehe Anhang Apollon_DBModell

6. User-Interface

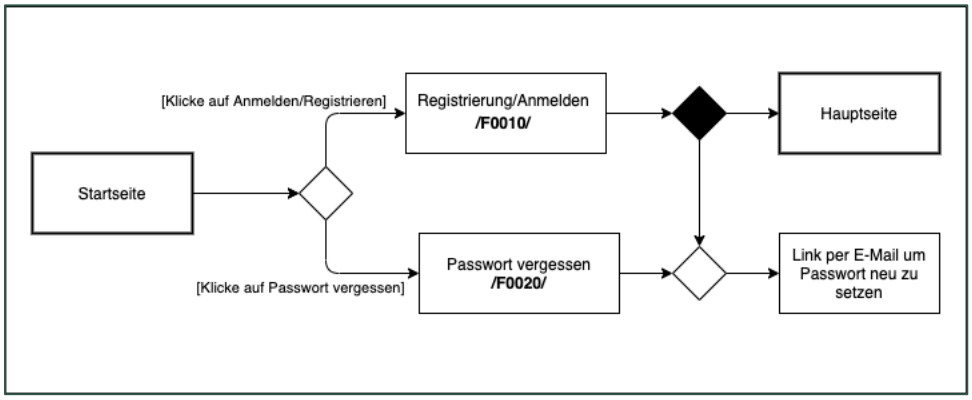
Standardmäßig ist eine menüorientierte Benutzung der Software vorzusehen. Diese wird mit Maus und Tastatur bedient. Aufgrund diverser Funktionen der Software, kann auf die Tastatur zur Bedienung nicht verzichtet werden.

Nachfolgend wird die grobe Dialogstruktur der Software aufgeführt. Diese gilt für eine fehlerfreie oder konfliktfreie Benutzung der Software als Spieler der Software. Tritt während der Benutzung ein Fehler auf, so wird die dargestellte Struktur verlassen und in einer separaten Seite abgearbeitet.

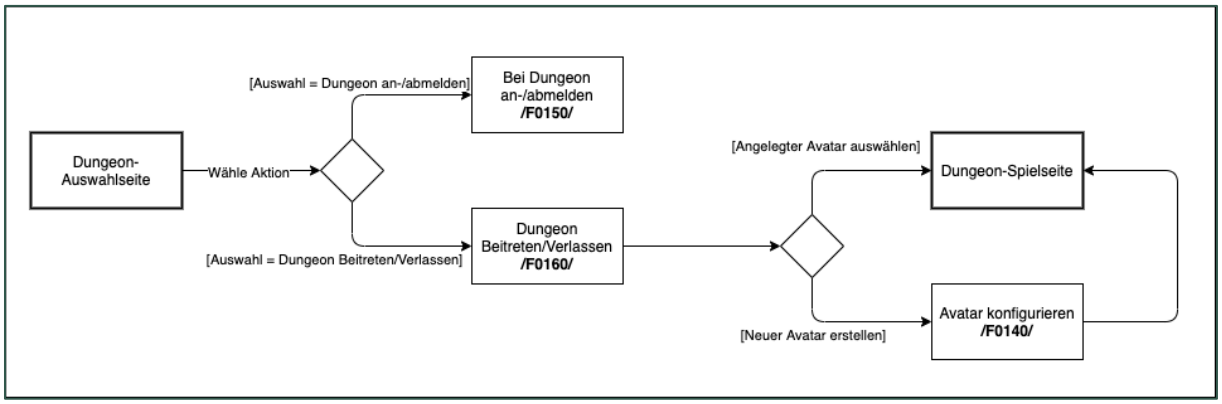
Außerdem ist zu beachten, dass die nachfolgenden Bildschirmskizzen/Wireframes ein erstes prototypisches Design darstellen. Somit wird sich eine Änderung im Verlauf des Projekts vorbehalten.

6.1 Hauptseite

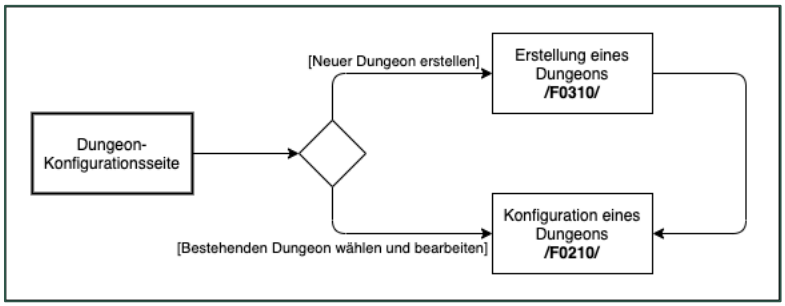
Die Hauptseite ist die Startseite eines angemeldeten Benutzers. Durch das hier zur Verfügung stehende Menü, kann der Benutzer jede weitere Seite erreichen, bis auf die Fehlerseite. Des Weiteren kann durch das Menü zwischen den einzelnen Seiten hin und her gewechselt werden. Dies wird in den folgenden Seiten als vorausgesetzt betrachtet und nicht erneut modelliert.



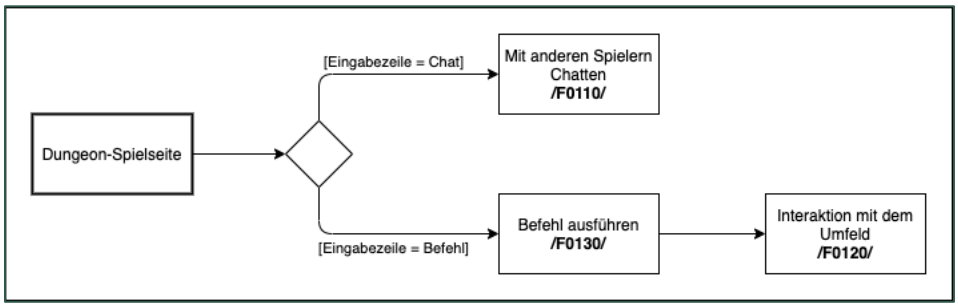
6.2 Dungeon-Auswahlseite



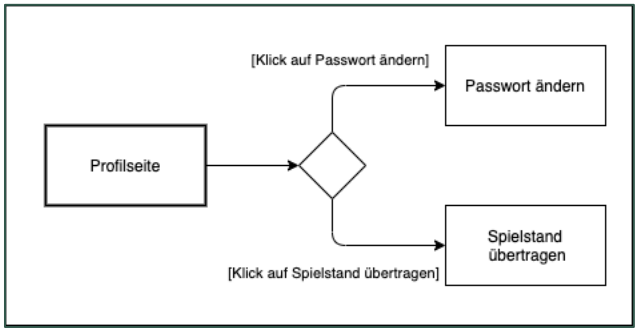
6.3 Dungeon-Konfigurationsseite



6.4 Dungeon-Spielseite



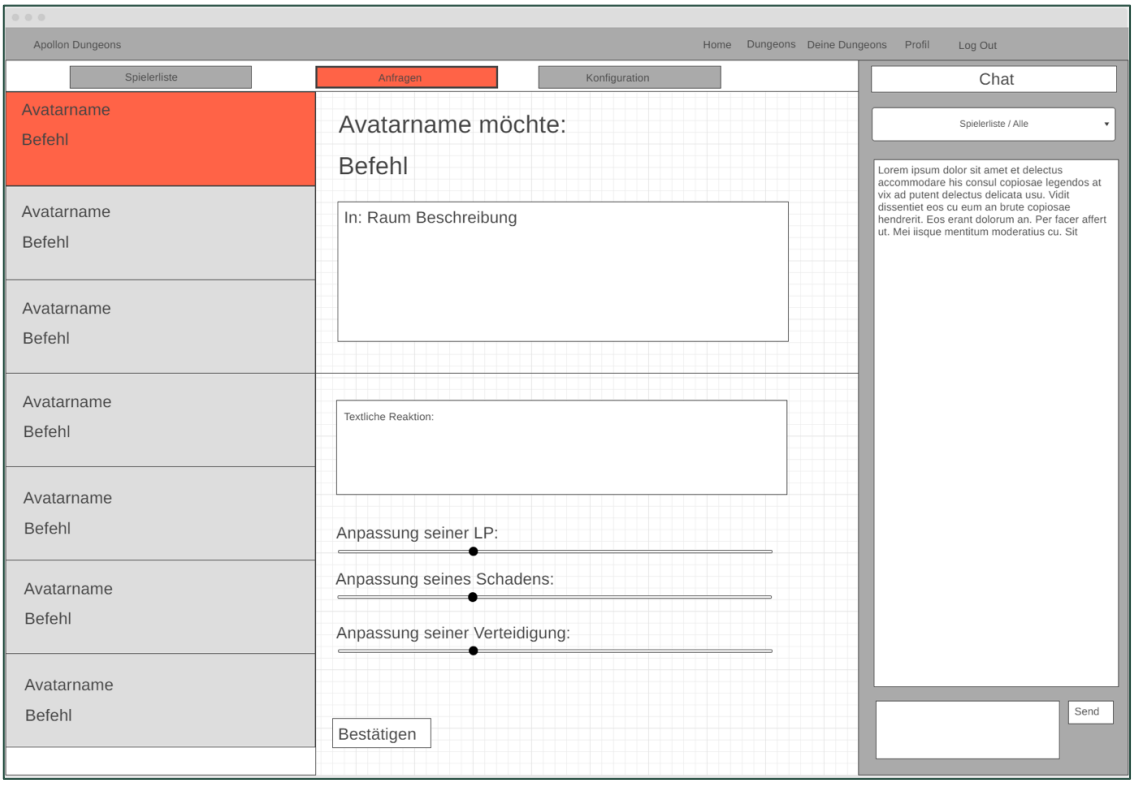
6.5 Profilseite



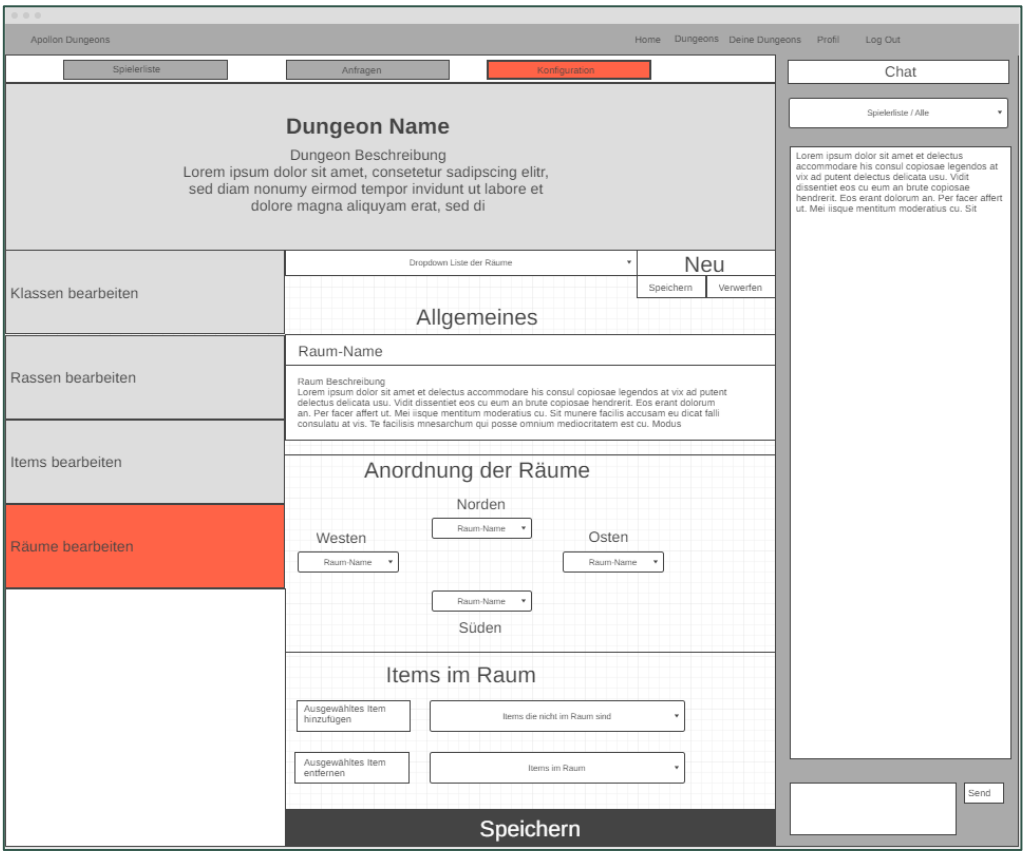
6.6 Dungeon Master Ansicht

Der Dungeon Master besitzt die Möglichkeit bestimmte Attribute eines Avatars im Spiel zu ändern. Eine Änderung der Attribute eines Avatars kann nur als Reaktion auf eine sogenannte „Special-Action“ (welche ein besonderer Befehl ist, der dem Dungeon Master zugestellt wird) erfolgen. Die Anpassung seines Verteidigungswertes kann nur als Reaktion stattfinden, wenn der Avatar eine neue Rüstung anzieht. Somit sind die Reaktionen kontextbezogen. Damit der Dungeon Master eine Anpassung vornehmen kann, benötigt er Name, Position sowie Spielgeschehen des Avatars. Die Informationen dienen als Prävention, damit der Dungeon Master nicht zufällig einem anderen Avatar eine falsche Anpassung übergibt.

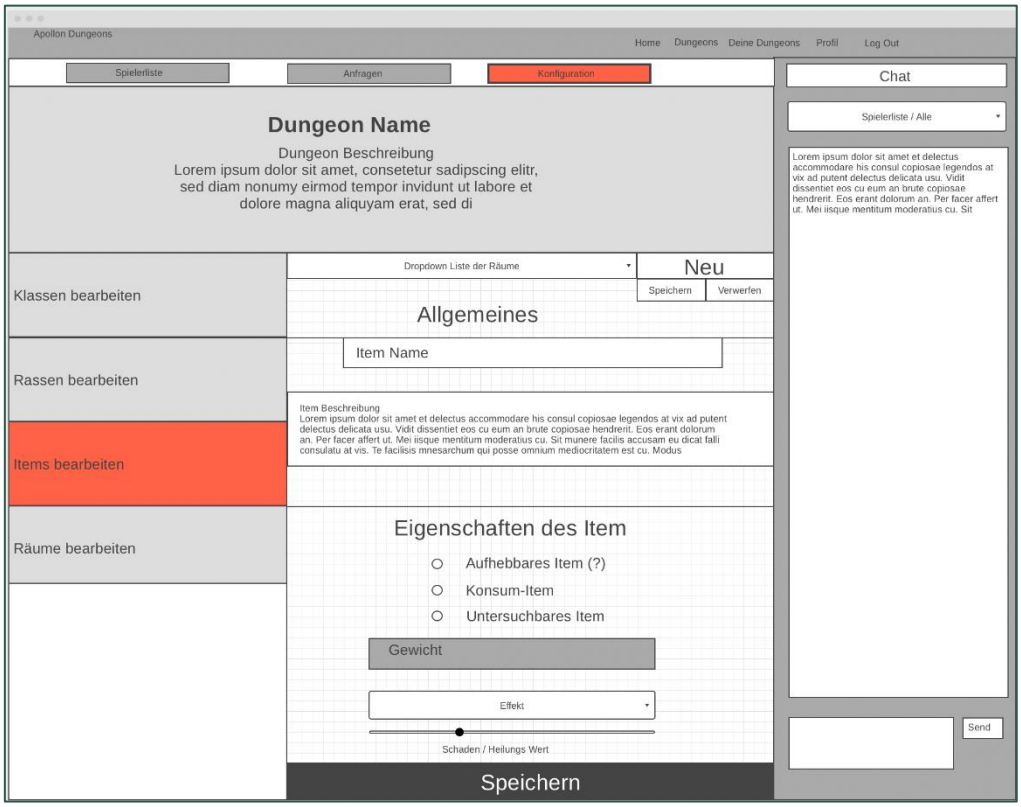
Auf der rechten Seite ist ein Chatfenster. Der Dungeon Master hat jederzeit die Möglichkeit, sich mit den Spielern in Verbindung zu setzen. Die Spieler können sich auf diesem Weg auch mit dem Dungeon Master verständigen.



In der Ansicht werden die möglichen Konfigurationen in den Räumen dargestellt. Dem Dungeon Master ist es möglich einen neuen Raum zu erstellen. Bei der Erstellung des Raumes können Nachbarn dessen festgelegt werden. Diese Nachbarschaft orientiert sich an den Raum-Namen der schon vorhandenen Räumen. Der Raum-Name dient dazu, Räume eindeutig definieren zu können. Mithilfe der schon vorhandenen Räume und den dazugehörigen Raum-Namen, kann gezielt bestimmt werden an welcher Position der neue Raum erstellt wird. Zusätzlich zu der Position und Raum-Namen ist die Beschreibung und Item Auswahl des Raumes von entsprechender Relevanz. Die Raumbeschreibung zeigt den semantischen Aspekt des Raumes wieder. Vorhandene Items im Raum sind entfernbar. Nicht vorhandene Items im Raum sind implementierbar. Abschließend wird die Konfiguration des Raumes abgespeichert. Dazu dient der Button „Speichern“.

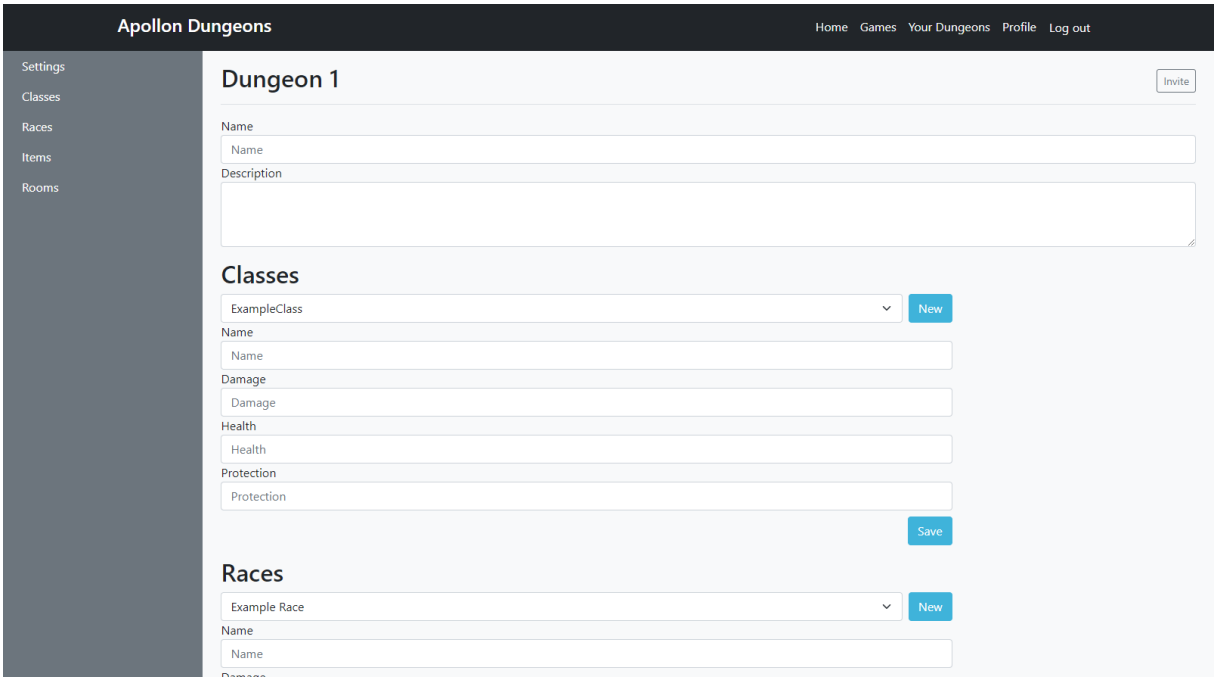


Der Dungeon Master kann auf der Seite „Items bearbeiten“ ein Item konfigurieren. Die Konfiguration beginnt mit der Namensvergabe des Items. Nachdem der Name vergeben wurde, ist es nötig dem Item eine Beschreibung zu geben. Die Beschreibung gibt recht oberflächlich das Aussehen und die Funktionsweise des Items wieder. Die Items sind in drei Kategorien aufgeteilt. Die erste Kategorie ist der Items, nennt sich „aufhebbare Items“. In der Kategorie „aufhebbare Items“ gehören alle Items, die aufgehoben werden können. Als zweites befinden sich die Konsumgüter. Konsumgüter sind konsumierbar. Das Konsum-Item hat einen Effekt auf den Konsumenten des Items. Der dazugehörige Effekt wird ebenfalls in der Konfiguration erstellt und ist somit individuell. Die dritte Kategorie umfasst die untersuchbaren Items. Jedes Item besitzt ein Gewicht als Eigenschaft. Mit einem Regler bestimmt der Dungeon Master einen Schadenseffekt oder einen Heilungseffekt. Abschließend wird das neu erstellte Item mit „Speichern“ abgespeichert und steht dem MUD zur Verfügung. Da sich im Laufe des Projekts herausgestellt hat, dass mehr als drei Kategorien an Items benötigt werden, stellt diese Darstellung keinen Anspruch auf Vollständigkeit dar und ist wie bereits beschrieben, ein prototypischer Entwurf, welcher im Allgemeinen einen ersten Eindruck der Oberfläche zur Konfiguration eines Dungeons vermitteln soll.

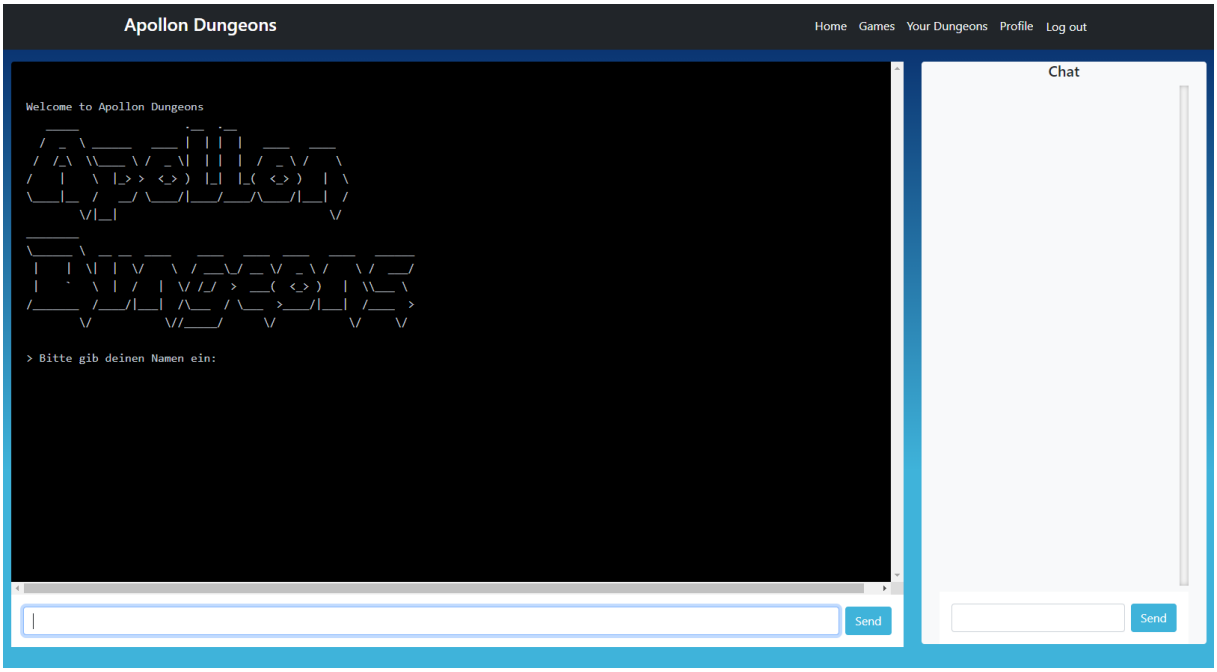


6.7 Bildschirmskizzen

In der nächsten Bildschirmskizze ist die Seite der Erstellung eines Dungeons dargestellt. Die Skizze ist von dem Prototyp entnommen, da Sie für die Umsetzung in dieser Form geplant ist, aber Änderungen eintreten können.



Folglich ist die Dungeon-Spielseite dargestellt. Die Abbildung entstammt ebenso dem Prototyp und setzt keine sonderlichen Veränderungen voraus, diese werden sich aber im Verlaufe des Projektes vorbehalten, falls sie als notwendig erscheinen.



Versions-Historie

Version	Datum	Bearbeiter	Art der Änderung	Stand
1.0	12.04.2021	Etienne Zink, Daniel Kröker	Erstes Aufsetzen des Dokuments	Fertiggestellt
2.0	15.04.2021	Team Apollon	Besprechung des Entwurfs	Fertiggestellt
2.1	18.04.2021	Daniel Kröker, Etienne Zink, Alfred Rustemi	Einfügen der Diagramme und Texte für DB, Dynamisch und Frames	Fertiggestellt
2.2	18.04.2021	Daniel Kröker, Etienne Zink	Qualitätskontrolle des Dokuments ohne Punkte 3.2 bis 3.5	Fertiggestellt
3.0	18.04.2021	Paul Burkard, Florian Albert	Einfügen der Punkte 3.2 bis 3.5	Fertiggestellt
3.1	19.04.2021	Paul Burkard, Alfred Rustemi	Fixen des Inhaltsverzeichnisses Ausbessern von Schreibfehlern	Fertiggestellt
4.0	09.05.21	Etienne Zink	Einfügen der neuen Diagramme der Game Logik, User Management, Datenmodelle und Konfiguration	Fertiggestellt
4.1	16.05.21	Etienne Zink	Überarbeitung Game Logik und Datenmodelle	Fertiggestellt
4.2		Paul Burkard	Überarbeitung des Modells zur Dungeon Erstellung	Fertiggestellt