**1. Projekt Setup**

**Unity Installation und Einrichtung**

* **Unity Hub herunterladen**: Installiert den Unity Hub von der offiziellen Unity-Website.
* **Unity Version auswählen**: Wählt die neueste LTS-Version (Long Term Support) von Unity aus und installiert sie über den Unity Hub.
* **Neues Projekt erstellen**: Öffnet den Unity Hub, klickt auf „Neues Projekt“ und wählt „2D“ als Vorlage aus. Gebt eurem Projekt einen Namen und wählt einen Speicherort.

**Assets importieren**

* **Grafiken und Sounds**: Sammelt alle benötigten Assets wie Charakter-Sprites, Tilemaps, Hintergrundmusik und Soundeffekte. Ihr könnt kostenlose Assets von Websites wie OpenGameArt oder Unity Asset Store verwenden.
* **Import in Unity**: Importiert die Assets in euer Unity-Projekt, indem ihr sie in den „Assets“-Ordner zieht oder über „Assets > Import New Asset“ im Menü.

**2. Grundlegendes Gameplay**

**Spielwelt erstellen**

* **Tilemap-Tool**: Nutzt das Tilemap-Tool von Unity, um eure Spielwelt zu gestalten. Erstellt verschiedene Tilemaps für den Boden, Hindernisse und Dekorationen.
* **Kollisionsabfrage**: Fügt Collider zu den Tilemaps hinzu, um Kollisionen zu erkennen und zu verhindern, dass der Spieler durch Wände läuft.

**Spielersteuerung**

* **Input-System**: Verwendet das neue Input-System von Unity, um die Steuerung des Spielercharakters zu implementieren. Erstellt ein Skript, das die Eingaben des Spielers verarbeitet und den Charakter entsprechend bewegt.
* **Animationen**: Erstellt Animationen für die verschiedenen Bewegungen des Charakters (z.B. Laufen, Stehen). Nutzt den Animator-Controller, um die Animationen zu verwalten und zu wechseln.

**3. Charakteranpassung**

**Klassen und Fähigkeiten**

* **Charakterklassen**: Definiert verschiedene Klassen wie Krieger, Magier und Schütze. Jede Klasse sollte einzigartige Fähigkeiten und Attribute haben.
* **Fähigkeiten-Skripte**: Erstellt Skripte für die verschiedenen Fähigkeiten. Nutzt Unity-Events, um die Fähigkeiten auszulösen und Effekte wie Schaden oder Heilung zu berechnen.

**4. Spielwährung und Händler**

**Währungssystem**

* **Münzen sammeln**: Implementiert ein System, bei dem der Spieler Münzen sammeln kann. Erstellt ein Skript, das die Anzahl der gesammelten Münzen speichert und anzeigt.
* **Währung ausgeben**: Fügt Funktionen hinzu, um Münzen für Gegenstände und Upgrades auszugeben. Erstellt ein Inventarsystem, um die gekauften Gegenstände zu verwalten.

**Händler-Interaktionen**

* **Händler-NPCs**: Erstellt NPCs, die als Händler fungieren. Nutzt Dialogsysteme, um Interaktionen zwischen dem Spieler und den Händlern zu ermöglichen.
* **Kauf- und Verkaufssystem**: Implementiert ein System, bei dem der Spieler Gegenstände kaufen und verkaufen kann. Nutzt Skripte, um die Transaktionen zu verwalten und die Inventare zu aktualisieren.

**5. SQL-Anbindung und Anmeldung**

**Datenbank einrichten**

* **SQLite**: Nutzt SQLite für eine lokale Datenbanklösung. Installiert das SQLite-Paket über den Unity Package Manager.
* **Datenbank-Schema**: Definiert das Schema für eure Datenbank, einschließlich Tabellen für Spielerprofile, Fortschritt und Inventar.

**Anmeldesystem**

* **Benutzerkonten**: Implementiert ein System zur Erstellung und Verwaltung von Benutzerkonten. Nutzt Hashing-Algorithmen, um Passwörter sicher zu speichern.
* **Fortschritt speichern und laden**: Erstellt Funktionen, um den Fortschritt des Spielers in der Datenbank zu speichern und auf verschiedenen Geräten zu laden. Nutzt Unity’s Networking-Tools oder externe Dienste wie Firebase für die Synchronisation.

**6. Weitere Überlegungen**

**Speichersystem**

* **Lokales Speichern**: Implementiert ein lokales Speichersystem, um den Fortschritt des Spielers auf dem Gerät zu speichern. Nutzt PlayerPrefs oder JSON-Dateien.
* **Cloud-Speichern**: Fügt eine Cloud-Speicheroption hinzu, um den Fortschritt online zu speichern und auf verschiedenen Geräten zugänglich zu machen.

**Testing**

* **Regelmäßige Tests**: Führt regelmäßige Tests durch, um sicherzustellen, dass alle Funktionen wie gewünscht funktionieren. Nutzt Unity’s Test-Framework für automatisierte Tests.
* **Bugfixing**: Seid bereit, auf Fehler zu reagieren und diese zu beheben. Nutzt Debugging-Tools und Log-Ausgaben, um Probleme zu identifizieren.

**Ressourcen**

* **Unity Learn**: Unity Learn bietet viele Tutorials und Ressourcen, die euch bei der Entwicklung helfen können.
* **YouTube-Tutorials**: Es gibt viele hilfreiche YouTube-Tutorials zu spezifischen Themen wie Spielersteuerung und Feindverhalten. Ein Beispiel ist dieses YouTube-Tutorial zur Spielersteuerung.