netidee
PROJEKTE

GigapixelTutor Entwicklerinnen-Dokumentation

Dokumentation | Call 17 | Projekt ID 6384

Inhalt

Einleitung

Das GigapixelTutor-Projekt ist für Studenten und Lehrer gedacht, die mit riesigen Bildern (i.e. Gigapixel-Bilder), manchmal sogar mehrere Gigabyte groß, arbeiten. Diese Bilder werden in verschiedenen Bereichen wie Kunstgeschichte, Astronomie, Stadtplanung, Geographie und Medizin verwendet.

In der Medizin werden zum Beispiel Gewebeproben als Gigapixel-Bilder online bereitgestellt. Während der Pandemie hat sich die Lehre teilweise auf Online-Plattformen wie Moodle verlagert. Neue Lehrmethoden wie das "Inverted Classroom Model" zeigen die Vorteile des selbstgesteuerten Lernens in Verbindung mit Präsenzunterricht. Aber bisher fehlt es an Interaktivität, die beim Arbeiten mit echten Mikroskopen im Klassenzimmer gegeben ist.

Das Projekt GigapixelTutor will diese Lücke teilweise schließen und ist in zwei Teile aufgeteilt, die unabhängige Webapplikation GigapixelTutor sowie die davon unabhängige Moodle-Plugins um Gigapixel-Bilder direkt in Moodle zu verwenden. Wir nutzen vorhandene Open-Source-Tools, um diese Gigapixel-Bilder online anzubieten.

Standalone-Webapplikation

Architektur

Das Architekturkonzept basiert auf einem robusten Stack von Technologien, der verschiedene Komponenten umfasst, um eine effiziente und skalierbare Lösung zu gewährleisten. Zentral für diese Architektur ist die Verwendung von Docker zur Containerisierung, was eine konsistente Bereitstellung und Skalierung ermöglicht. Auf der Anwendungsebene wird Laravel, ein PHP-Framework, eingesetzt, um eine strukturierte und leistungsstarke Backend-Entwicklung zu ermöglichen. Die Datenbank wird mit MariaDB verwaltet, was eine zuverlässige und schnelle Speicherung von Daten ermöglicht. Für Echtzeitkommunikation wird Soketi genutzt, um eine bidirektionale Verbindung zwischen Client und Server zu ermöglichen. Insgesamt bietet diese Architektur ein solides Fundament für die Entwicklung und Bereitstellung von modernen Webanwendungen, die flexibel, skalierbar und robust sind.

Laravel ist ein beliebtes PHP-Framework, das für die Entwicklung von Webanwendungen verwendet wird. Eine Eigenschaften von Laravel ist seine Unterstützung des Model-View-Controller (MVC)-Architekturmusters. Bei der MVC-Aufteilung werden die verschiedenen Aspekte einer Anwendung - Datenmodell (Model), Präsentation (View) und Anwendungslogik (Controller) - in separate Komponenten aufgeteilt. Das Model ist verantwortlich für die Verwaltung der Datenbankinteraktionen und die Geschäftslogik der Anwendung. Die View ist für die Darstellung der Benutzeroberfläche und die Präsentation der Daten zuständig. Der Controller nimmt Benutzeranfragen entgegen, verarbeitet sie und koordiniert die Interaktion zwischen dem Model und der View. Durch die klare Trennung dieser Komponenten ermöglicht Laravel eine saubere und strukturierte Entwicklung von Webanwendungen, was die Wartbarkeit, Erweiterbarkeit und Wiederverwendbarkeit des Codes verbessert.

Komponenten

Die Software umfasst mehrere wichtige Komponenten, die das Benutzererlebnis verbessern und die Funktionalität erweitern.

Login und Registrierung: Diese Funktion ermöglicht es Benutzern, sich sicher anzumelden und neue Konten zu erstellen, um auf die Plattform zuzugreifen und ihre eigenen Daten zu verwalten.

My Datasets: Diese Funktion bietet Benutzern eine übersichtliche Liste aller Datensätze, die sie erstellt oder zugeordnet haben, und erleichtert somit den schnellen Zugriff auf ihre gespeicherten Daten.

Dataset Create Dataset Screen: Mit dieser Funktion können Benutzer neue Datensätze erstellen und anpassen. Es können momentan nur Bilder, die schon auf dem Server gespeichert sind, über diese Komponente zu einem Dataset hinzugefügt werden.

Dataset Overview: Die Dataset Overview bietet eine zusammenfassende Darstellung aller verfügbaren Datensätze, um Benutzern einen schnellen Überblick über ihre gespeicherten Daten zu geben und eine einfache Navigation zwischen verschiedenen Datensätzen zu ermöglichen. Wir unterscheiden zwischen öffentlichen Datasets, geteilten Datasets und den eigenen Datasets.

Dataset Share: Diese Funktion ermöglicht es Benutzern, ihre Datensätze mit anderen Benutzern zu teilen, um die Zusammenarbeit und den Wissensaustausch zu fördern.

Dataset Edit: Mit der Dataset Edit-Funktion können Benutzer ihre Datensätze bearbeiten und anpassen, indem sie vorhandene Daten aktualisieren oder neue Daten hinzufügen.

Dataset Admin-List: Die Dataset Admin-List bietet Administratoren einen Überblick über alle Gigapixel-Bilder in einem Datensatz und ermöglicht es diese in tabellarischer Form zu exportieren.

Dataset Download: Benutzer können mit dieser Funktion Datasets von der Plattform herunterladen, um sie lokal zu speichern.

Dataset Upload: Die Dataset Upload-Funktion ermöglicht es Benutzern, neue Gigapixel-Bilder direkt zu einem Dataset hinzuzufügen, um ihre Datenbank zu erweitern und zu aktualisieren.

Dataset Session: Mit dieser Funktion können Benutzer Live-Session mit anderen Benutzern teilen.

Detailansicht Einzelbild: Diese Funktion bietet eine detaillierte Ansicht jedes Bildes innerhalb eines Datensatzes, um Benutzern eine genauere Analyse und Bewertung ihrer Daten zu ermöglichen. Man kan leicht zwischen Bildern eines Datasets vor- und zurück gehen.

Detailansicht - Annotate: Die Detailansicht - Annotate ermöglicht es Benutzern, ihre Bilder mit verschiedenen Annotationen wie Audio-, Text- und SVG-Annotationen zu versehen, um wichtige Informationen zu markieren und zu dokumentieren.

Audio description, Text description, SVG-Annotation, Comments, ROI: Diese Funktionen bieten verschiedene Möglichkeiten zur Anreicherung und Kommentierung von Bildern innerhalb eines Datensatzes, um wichtige Details zu kennzeichnen und zu dokumentieren und die Zusammenarbeit zwischen Benutzern zu fördern.

Datenbank

Das ER-Diagramm (Entity-Relationship-Diagramm) in der Abbildung unten stellt die Datenbankstruktur von GigapixelTutor dar. Die blauen Tabellen repräsentieren die Basistabellen, die die Hauptdaten der Anwendung speichern. Die roten Tabellen sind für die Rechteverwaltung zuständig, wobei Informationen über Zugriffsrechte und Berechtigungen gespeichert werden. Die grünen Tabellen dienen dem Teilen von Inhalten und speichern Daten über geteilte Ressourcen und Zugriffslinks. Die gelben Tabellen sind für Annotationen vorgesehen und enthalten Informationen über Text- und Audioanmerkungen sowie andere Arten von Annotationen, die Benutzer zu den Gigapixel-Bildern hinzufügen können. Insgesamt bietet das ER-Diagramm einen detaillierten Überblick über die Datenbankstruktur von GigapixelTutor und zeigt, wie verschiedene Komponenten miteinander verbunden sind.

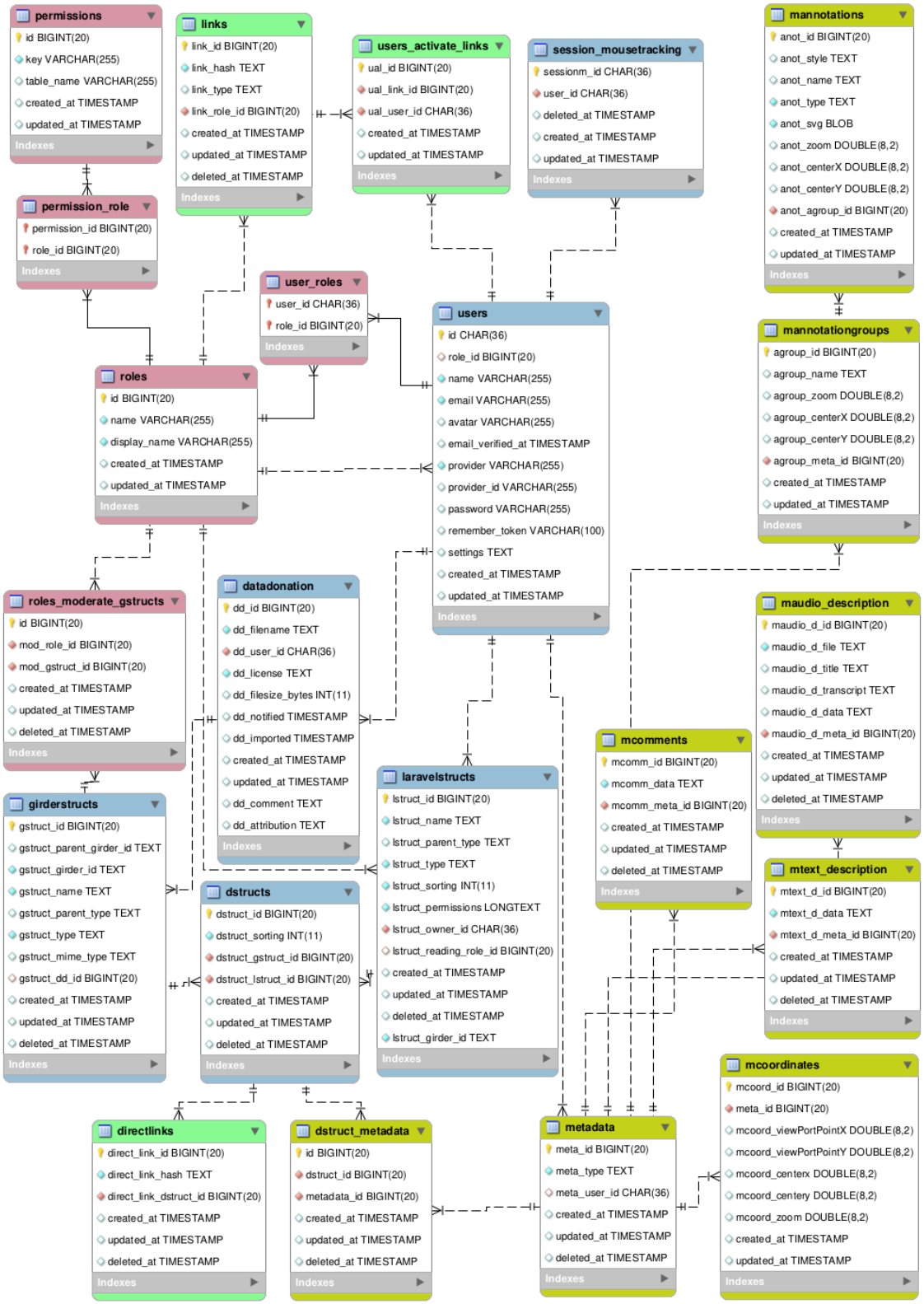

Abbildung: ER-Diagram der Datenbank von GigapixelTutor. Blau: Base-Tabellen, Rot: Rechteverwaltung, Grün: Sharing, Gelb: Annotationen

Berechtigungskonzept

Datasets in GigapixelTutor werden in der LaravelStructs-Tabelle gespeichert und verarbeitet. Jedes Dataset hat seine eigenen Berechtigungen, die in JSON im Lstruct-Permission-Spalte gespeichert sind. Berechtigungen werden pro Rolle definiert; Jede Berechtigung wird mit einem Gate in der AuthServiceProvider.php überprüft. Wenn ein Benutzer Teil einer Rolle ist, die eine Berechtigung auf "true" gesetzt hat, hat der Benutzer Zugriff auf die Funktion.

Ein öffentliches Dataset kann erstellt werden, indem die Berechtigung für den Benutzer auf "true" gesetzt wird. Benutzerdefinierte Berechtigungen können erstellt werden, indem eine benutzerdefinierte Rolle erstellt und Berechtigungen für diese Rolle hinzugefügt werden. Die Lstruct_type-Spalte in der laravelstructs-Tabelle bestimmt vorerst nur, welche Symbole auf der Hauptseite für die Ordner verwendet werden. Durch das Setzen der "shared"-Berechtigung auf einen Ordner werden die Berechtigungen für Benutzer verwaltet, mit denen dieser Ordner über die Freigabelink-Funktionalität geteilt wird.

Beispiel-JSON eines Standardordners in der Lstruct-Permission-Spalte:

```
{
  "admin": {
        "browse": true,
        "viewer": true,
        "delete": true,
        "share": true
  },
  "user": {},
  "owner": {
        "browse": true,
        "viewer": true,
        "share": true
  },
  "shared": {
        "browse": true
  }
}
```

Wenn Sie eine neue Berechtigung/Funktion hinzufügen möchten, erstellen Sie einfach ein neues Gate in der AuthServiceProvider.php. Hier ist ein Beispiel für ein Gate für die Funktion mit dem Namen 'browse_lstruct'. Die Berechtigung im Berechtigungs-JSON (Lstruct-Permission-Spalte) wäre 'browse':true

Konfigurationen

In Laravel können Konfigurationseinstellungen über sogenannte .env-Dateien verwaltet werden. Diese Dateien enthalten Umgebungsvariablen, die von der Laravel-Anwendung während ihrer Ausführung geladen werden. Die .env-Datei liegt im Stammverzeichnis des Laravel-Projekts und enthält Schlüssel-Wert-Paare für verschiedene Einstellungen, wie z.B. Datenbankverbindungen und Logging-Einstellungen. Indem man diese Variablen in der .env-Datei festlegt, können Entwickler die Konfiguration der Anwendung zentralisieren und sie je nach Umgebung (Entwicklung, Test, Produktion) oder individuellen Anforderungen anpassen. Diese Trennung von Konfigurationswerten von den eigentlichen Quellcodes ermöglicht eine sauberere und sicherere Verwaltung der Anwendungskonfiguration und erleichtert auch das Deployment in verschiedenen Umgebungen, ohne dass Änderungen im Quellcode vorgenommen werden müssen.

Es ist wichtig zu beachten, dass die .env-Datei für eine lokale Testumgebung vorkonfiguriert sind und noch entsprechend angepasst werden müssen, bevor die Anwendung in den Produktionsbetrieb überführt wird. Bevor die Anwendung live geschaltet wird, müssen Entwickler sicherstellen, dass alle Konfigurationseinstellungen in der .env-Datei entsprechend den Anforderungen der Produktionsumgebung aktualisiert werden.

Für Testzwecke steht ein Bash-Script (i.e. deploy.sh) zur Verfügung, das alle erforderlichen Konfigurationen für eine Testversion vornimmt und Testdaten generiert. Dieses Skript automatisiert den Prozess der Konfiguration und Bereitstellung, indem es die .env-Datei entsprechend den Anforderungen der Testumgebung aktualisiert und Testdaten in die Datenbank einfügt. Dadurch wird sichergestellt, dass Entwickler schnell und effizient Testversionen der Anwendung erstellen und testen können, ohne jedes Mal manuell Konfigurationen vornehmen oder Testdaten eingeben zu müssen. Dies erleichtert den Entwicklungsprozess und fördert eine kontinuierliche Integration und Bereitstellung von Updates und neuen Funktionen.

Das Testskript setzt die Anwendung auch auf die Ursprungsversion zurück, damit Entwickler einen sauberen Ausgangszustand für ihre Tests haben. Durch das Zurücksetzen auf die Ursprungsversion werden alle vorherigen Änderungen und Testdaten entfernt, was es den Entwicklern ermöglicht, Tests in einer konsistenten Umgebung durchzuführen und sicherzustellen, dass die Ergebnisse reproduzierbar sind. Dies fördert die Zuverlässigkeit und Konsistenz der Tests und ermöglicht es den Entwicklern, potenzielle Probleme oder Fehler in der Anwendung effektiv zu identifizieren und zu beheben.

Datenschnittstellen

In GigapixelTutor gibt es mehrere Datenschnittstellen, die es den Benutzern ermöglichen, auf Daten zuzugreifen und mit ihnen zu interagieren. Einige der wichtigsten Datenschnittstellen sind:

1. **API-Endpunkte**: GigapixelTutor bietet nativ eine REST-Schnittstelle über Laravel, die es ermöglicht, auf bestimmte Funktionen und Daten direkt zuzugreifen. Diese REST-API-Endpunkte können beispielsweise verwendet werden, um Informationen über Benutzerkonten abzurufen, Datensätze zu erstellen oder abzurufen oder verschiedene Anwendungsaktionen auszuführen.
2. **Datenbankzugriff**: Benutzer können direkt auf die Datenbank von GigapixelTutor zugreifen, um Informationen zu speichern, abzurufen oder zu aktualisieren. Dieser direkte Datenbankzugriff ermöglicht eine umfassende Kontrolle über die gespeicherten Daten und bietet Entwicklern die Flexibilität, komplexe Abfragen oder Datenmanipulationen durchzuführen.
3. **Dateisystemzugriff**: GigapixelTutor kann auch auf das Dateisystem zugreifen, um Dateien wie Bilder, Audioaufnahmen oder andere Medien zu speichern und abzurufen. Dieser Zugriff ermöglicht es Benutzern, multimediale Inhalte hochzuladen, herunterzuladen und zu verwalten, die in der Anwendung verwendet werden.

Source-Code

Der Source-Code ist unter https://gitlab.com/gigapixeltutor/gigapixeltutor zu finden, in der README.md Datei findet sich die Anleitung zum Starten und Konfigurieren des Systems.

Moodle Plug-ins

Architektur

Wie in Abbildung 1 zu sehen ist, ist Moodle in „Core“, „Annotation Service“ und „Tile Webservice“ unterteilt. Core umfasst die neu entwickelten Activity- und Block-Plug-ins, sowie alle weiteren Standard Moodle-Funktionalitäten, wie das Anlegen von Benutzern, Kursen, Rollenverwaltung usw., die nicht explizit angezeigt werden. Sie sind zusammengefasst, da die Plug-ins den gleichen Standard aufweisen wie alle anderen Moodle-Plug-ins. Das bedeutet, dass sie nach der Installation sofort in der Liste der Plug-ins verfügbar sind. Deshalb muss der/die Entwickler:in beispielsweise keine extra API erstellen, die z.B. die Erstellung einer neuen Gigapixel Activity Plug-in Instanz ermöglicht. Dies ist alles bereits vorhanden.

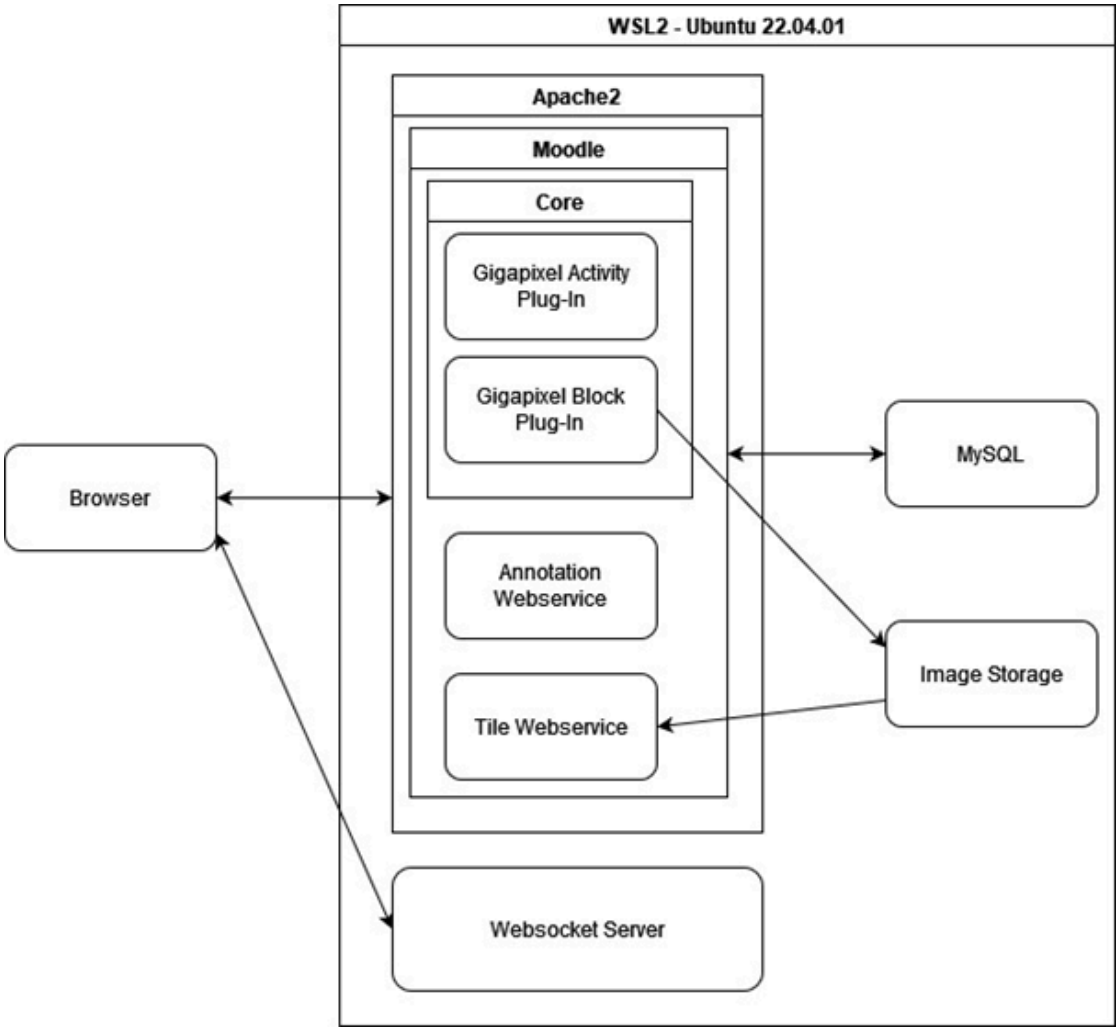

Abbildung 1: Die Komponenten und ihre Schnittstellen

Außerhalb von Core liegen noch der Annotation und Tile Webservice. Diese müssen ebenfalls den Standards von Moodle Webservices folgen, sind aber letztenendes zu verwenden wie andere Schnittstellen. Entwickler:innen haben die Möglichkeit GET, POST, PUT und DELETE HTTP-Anfragen zu senden, je nach Schnittstellendefinition.

Um die erstellen Resourcen, Benutzer-Accounts, Annotations, etc. zu speichern, muss noch eine Datenbank, in diesem Fall eine MySQL Datenbank, verfügbar sein. Die Gigapixel-Bilder müssen außerhalb der Datenbank abgelegt werden, weshalb noch eine extra Komponente “Image Storage” dargestellt ist. Dies muss ein Ort sein, auf den der Tile Webservice zugreifen kann, damit dieser die benötigten Tiles an den/die Benutzer:in zurückliefern kann.

Die letzte Komponente ist der Websocket Server. Moodle und PHP bieten keine Möglichkeit Websockets aufzubauen, weshalb noch ein Node Server erstellt wurde. Dieser ermöglicht das Erstellen einer Websocket-Verbindung zwischen dem Browser des/der Benutzers:in und dem Server. Über diese Verbindung werden die Bild-Koordinaten des/der Professesor:in mit den Studenten:innen geteilt. Die genaue Funktionsweise wird im folgenden Kapitel erläutert.

Komponenten

Block Plug-in

Block Plug-ins sind ein ergänzender Bestandteil von Moodle, der die Anzeige zusätzlicher Informationen und die Bereitstellung hilfreicher unterstützender Features und Funktionen ermöglicht. Es wurde ein Block Plug-in erstellt, das den berechtigten Benutzer:innen die Möglichkeit gibt, Gigapixel-Bilder auf Moodle hochzuladen. Nachdem der Upload erfolgreich war, zeigt das Block-Plug-in den Namen des Bildes sowie eine eindeutige ID an. Diese ID wird später beim Erstellen eines Aktivitäts-Plug-Ins benötigt, um auf das Bild zu verweisen und es anzuzeigen.

In Abbildung 2 ist ein normaler Moodle Kurs zu sehen und auf der rechten Seite das Gigapixel Block Plug-in. In diesem Fall wurde bereits ein Bild hochgeladen, weshalb auch dessen Name und eine generierte ID angezeigt wird. Zusätzlich können unter dem “add” Hyperlink noch weiter Bilder hochgeladen werden.

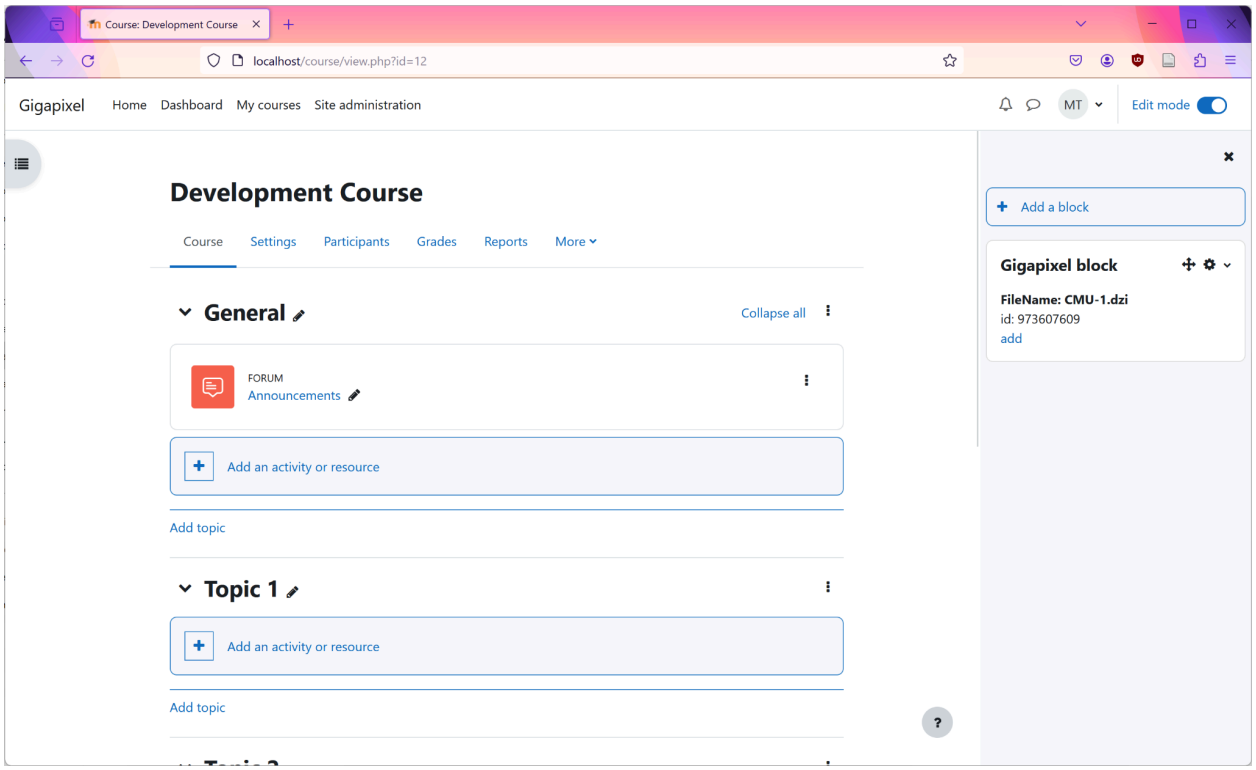

Abbildung 2: Ein Moodle Kurs mit dem Gigapixel Block Plug-in auf der rechten Seite

Activity Plug-in

Activity Plug-ins dienen zur Bereitstellung von Lerninhalten in Moodle. Mit dem Gigapixel Activity Plug-in kann der/die Lehrer:in in seinem/ihrem Kurs eine neue Gigapixel-Instanz erstellen. Sie müssen über die ID, die im Block-Plugin angezeigt wird, auf ein Gigapixel-Bild verweisen. Ansonsten wird nur ein Titel benötigt, um eine gültige Ressource zu erstellen. Optional kann dem Bild auch eine Beschreibung hinzugefügt werden, worin auch anklickbare Koordinaten eines Bildausschnitts enthalten sein können.

Nach dem Erstellen der Ressource kann jede:r mit den entsprechenden Rechten darauf zugreifen. Das Gigapixel Bild wird geladen und im OpenSeadragon (OSD) Viewer angezeigt. Sobald innerhalb des Bildes navigiert wird, löst der Viewer ein Ereignis aus, um die benötigten Bildkacheln (Tiles) via GET-Request vom Server anzufordern.

Abbildung 3 zeigt eine beispielhafte Gigapixel-Instanz, die mit dem Activity Plug-in erstellt wurde. Es ist das Gigapixel-Bild innerhalb des OSD Viewers zu sehen. Über dem Bild befinden sich zwei Symbole und unter dem Bild befindet sich ein kurzer Text mit einem eingebetteten Hyperlink. Bei den Symbolen handelt es sich um Schaltflächen zum Erstellen von Annotations in verschiedenen Formen, während der Text eine Beschreibung des Bildes mit Koordinaten-Hyperlinks zum genauen Bild-Abschnitt darstellt.

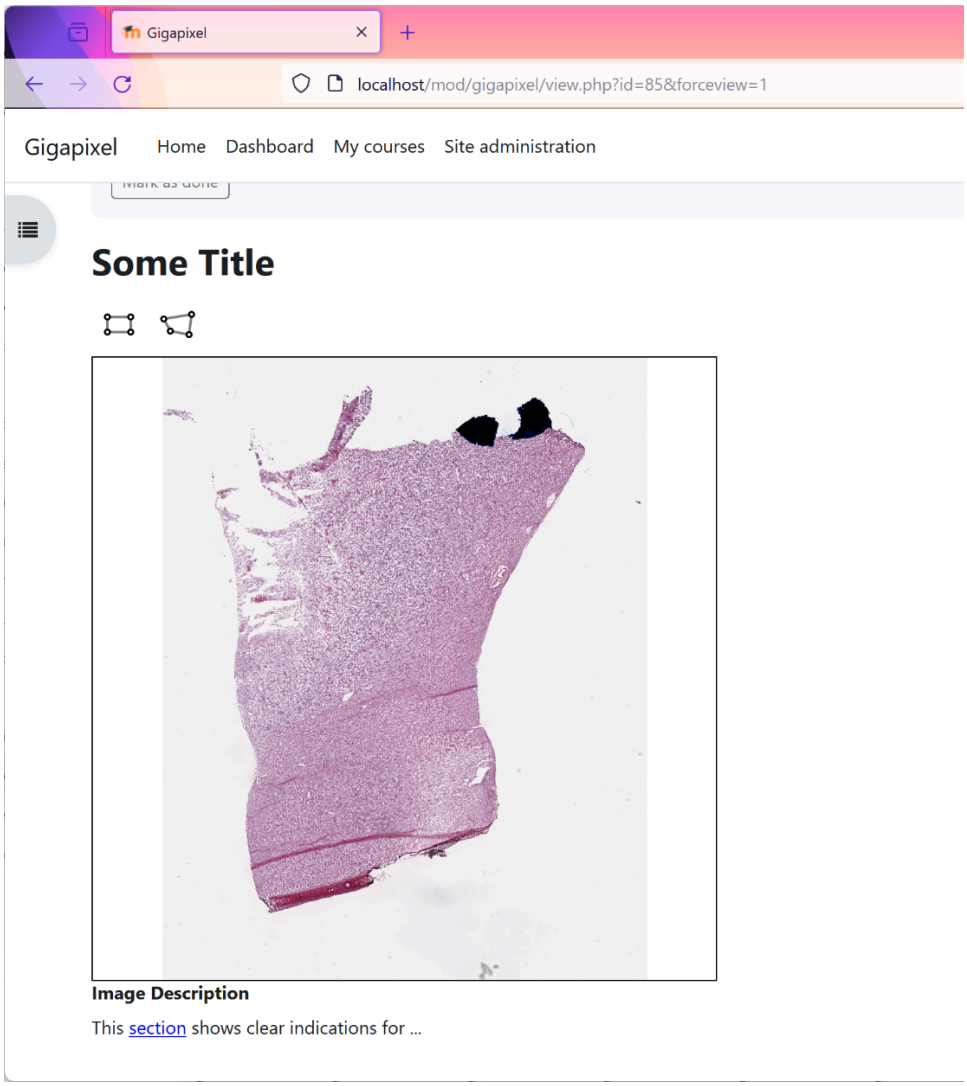

Abbildung 3: Instanz eines Gigapixel Activity Plug-ins

Annotation Webservice

Der Annotation Webservice enthält eine Schnittstelle für das Erstellen, Aktualisieren und Löschen von Annotations. Beim Laden des Activity Plug-ins werden bereits die Annotationen beigefügt, weshalb keine GET Schnittstelle benötigt wird.

Die Annotations werden mithilfe der Annotorious-Extension von OSD erstellt und dargestellt. Um die erstellten Annotations zu speichern, wird ein POST-Request an den Server gesendet. Dieser benötigt, neben den standardmäßig benötigten Parametern für Moodle Webservices, noch die ID des Activity Plug-ins, sowie die erstellte Annotation als String. Als letzter Parameter wird eine “ImageUrl” benötigt, welche aber leergelassen werden kann, da es sich hier um einen veralteten, nicht mehr verwendeten Parameter handelt.

Sollte bei einer Annotation beispielsweise ein Kommentar hinzugefügt werden, wird wieder ein POST-Request an den Server gesendet um ein Update durchzuführen. Dieser beinhaltet, wieder neben den Standardparametern, nur einen String, der alle Annotations dieser Resource enthält.

Sollte eine Annotation gelöscht werden, wird jene ID an den Server als DELETE-Request übermittelt, wonach sie danach aus der Datenbank entfernt wird.

Tile Webservice

Der Tile Webservice wird benötigt, wenn innerhalb von OSD navigiert oder gezoomt wird. Bei diesen Events werden die jeweiligen Bildkacheln via GET-Request vom Server angefordert. Dabei wird bei jedem Request die Bild ID und die benötigten Koordinaten (level, x, y) angehängt. Die Bildkachel wird dann von OpenSeadragon heruntergeladen und im Viewer angezeigt.

Websocket Server

Der Websocket Server wird benötigt, damit es dem/der Professor:in möglichst ist, seine/ihre Bildkoordinaten mit den Student:innen zu teilen. D.h. wenn der/die Professor:in im OSD-Viewer navigiert, wird ein Event ausgelöst, in dem die aktuellen Bildkoordinaten via Websocket an den Server gesendet werden. Student:innen, die sich auf derselben Gigapixel-Instanz befinden, erhalten über ihre Websocketverbindung diese Koordinaten. Diese werden dann dem Viewer übergeben, welcher daraufhin automatisch zu dem Bildausschnitt navigiert auf dem sich der/die Professor:in befindet.

Wenn ein:e Student:in eine Gigapixel-Instanz öffnet, wird automatisch eine Websocketverbindung geöffnet. Daraufhin wird ein Paket an den Server gesendet. Dieses Paket enthält einen Parameter “command” mit dem Wert “register” und einen Parameter “fileId”, welcher die Bild-ID beinhaltet. Daraufhin wird diese Verbindung mit anderen Verbindungen unter dieser Bild-ID gesammelt.

Auch für Professor:innen wird eine Verbindung geöffnet sobald die Gigapixel-Instanz besucht wird. Nur wird, wie oben bereits erklärt, über diese Verbindung nur Pakete versendet, wenn der/die Benutzer:in im OSD-Viewer navigiert. Diese Pakete haben ebenfalls den Parameter “fileId” und “command”, jedoch letzteres mit dem Wert “distributeCoordinates”. Weiters haben sie Feld “coordinates”, das aus x, y, z Koordinaten besteht. Auf dem Server kann dann mithilfe des “command”- und “fileId”-Parameters die Koordinaten an die Verbindungen weitergeleitet werden, welche sich auf jene fileId registriert haben.

Der Server hört auf Verbindungen auf Port 8081.

Konfigurationen

Unter dem folgendem Link kann das Gigapixel Block Plug-in heruntergeladen werden: https://gitlab.com/gigapixeltutor/moodle-block_gigapixel

Im Readme befinden sich die nötigen Schritte, um die Installation des Block Plug-ins abzuschließen.

Unter dem folgendem Link kann das Gigapixel Activity Plug-in heruntergeladen werden: https://gitlab.com/gigapixeltutor/moodle-mod_gigapixel

Die Schritte, damit das Activity Plug-in fertig konfiguriert werden kann, können im Readme im folgendem Link gefunden werden:

https://gitlab.com/gigapixeltutor/moodle-gigapixel?#setup-gigapixel