# CS2209 INFORMATION STORAGE AND MANAGEMENT II

*2023/2024*

**Assignment**

**Due: 2359hrs 16th April 2024**

**The assignment is worth 10% of the final module score**

**All work must be your own**

You are supposed to be creative with your thinking and make your own assumptions if needed. Grading is based on the demonstration of domain understanding, technical correctness, design comprehensiveness, and innovativeness.

**Project:** 2. CVAT - Computer Vision Data Annotation Platform for AI

**Question 1**. Propose your ideas to enhance/improve the assigned open-source software. It is important to avoid overlaps in ideas/concepts among your group members.

**Part A)** Describe your features/ideas/concepts for the open-source software enhancement. You MUST declare any external sources if you reference them in your description (e.g., Website URLs, Books or Articles).

## Group Member 1: (Min: 100 words – Max: 300 words)

My idea for an improvement that can be made to CVAT AI is creating customisable annotation templates that allows users to create, save, and reuse annotation templates for their different projects. The goal of the creation of annotation templates is to speed up the setup process in CVAT. This will be beneficial to users that have the same annotation needs between their different projects. The template could have some predefined labels, default configurations for AI-assisted annotations, some guidelines and instructions, which would help a group of programmers working on a project have clear and concise instructions, and some keyboard shortcuts. The creation of templates can speed up users' annotation projects, especially for users working on a number of projects that all have similar requirements. The templates themselves can be shared among team members, and even throughout the greater CVAT AI community. Doing all this will significantly reduce configuring new projects from scratch, which will ensure the consistency of projects and help save time.

**Part B)** Describe data structures, objects/entities/documents and types involved in your ideas/concepts.

## Group Member 1: (Min: 100 words – Max: 300 words)

|  |
| --- |
| My idea of annotation templates will require the creation of four different entities:   1. Annotation Template entity, which will be our main entity, and represent the customisable templates that users can create and apply to their projects. It will contain settings such as AI models to be used, guidelines, and some keyboard shortcuts; all of these will be specific for the project type. It will also include template specific information such as creation and modified date. It will have the following attributes: a unique ID, template name, guidelines, AI Model, creation date, and last modified date. 2. Label entity, which will define a label that can be used for the annotation project. This entity will reference the previous entity and will contain the following attributes: a unique ID, a reference to the template ID, label name, and colour of label. 3. Attribute entity, that will specialise and add to the label entity if need be, for example to describe the colour or size of the object in the project being annotated. This entity will reference the label entity, and will contain the following attributes: unique ID, a reference to label ID, attribute name, attribute type, default attribute value. 4. Keyboard shortcut entity, that will store common keyboard actions that would be specific to projects. This entity will also reference the template entity, and will contain the following attributes: a unique ID, reference to template ID, action performed by the shortcut, and the key combination needed for the shortcut.   All relationships are many-to-many relationships, as a template can have many labels, and a label can belong to many templates, a label attribute can be found in many labels, and a label can have many label attributes, and a template can have many shortcuts, and a shortcut can belong to many templates. |

**Question 2**. Discuss and select a relevant database management system for the proposed ideas (e.g., relational, document-based, graph-based databases).

***(Min: 500 words – Max: 2000 words)***

Before selecting a database management system, it is important to focus on each idea in isolation and determine for each what the best options are. After doing this, we will be better equipped to choose one system to proceed with.

The first idea that was examined in relation to improvements that could be made to CVAT AI is the addition of annotation templates. For this particular task it can be decided that relational databases, such as SQl, are the best one to use, and the following points should help demonstrate this. Document based databases such as MongoDB excel in flexible data storage, as our templates need different settings, MongoDB can handle these variations easily as it does not require a fixed structure. However they are not the best at managing relationships between data, like linking a particular label to many templates. Graph based databases such as Neo4j are excellent when it comes to dealing with data where relationships are the most important factor. If we are trying to figure out how or why a template might relate to different labels, Neo4j makes this easy. However graph based databases are a highly specialised tool, and setting them up for a project which deals with mainly regular tasks, might be more complicated than it is worth. The use of SQL solves the problems of the previous two, being easy to use for regular tasks while being able to handle relationships effectively. The annotation template idea involves the need for complex data structures due to the many-to-many relationships that can be found between the different templates, for example between the templates and labels entities, and SQL allows an efficient and simple implementation of this. SQl databases are good at organising data in structured ways and making sure everything stays consistent and accurate, especially when dealing with multiple interconnected items. SQL allows for asking highly complex and specific questions about your data, like retrieving all the labels used in a specific template. SQL allows for easy addition and modification of the different tables in the database too, which is greatly needed, due to the highly customisable nature of the annotation templates. So for managing detailed and connected data for our templates, a relational database is definitely the best choice.

The next idea that was examined was the addition of a version history to CVAT. A version history system requires the ability to relate past versions to current versions. This can best be done using a relational database, as look up times will be quick to display all past versions of an annotation. Each iteration can link to its previous iteration using foreign keys in the entity. MySQL’s indexing and lookup times make it a clear choice for this feature of version history as querying will be incredibly fast. Using foreign keys to link data provides clear organisation and allows the system to separate current and past versions. This separation allows for much faster access to the current version, rather than other database systems which would require a linear search or more indexing. The constraints of a relational database provide great reliability. Relational databases also enforce ACID properties when making transactions. The use of transactions can be incredibly useful for rolling back changes in annotations.

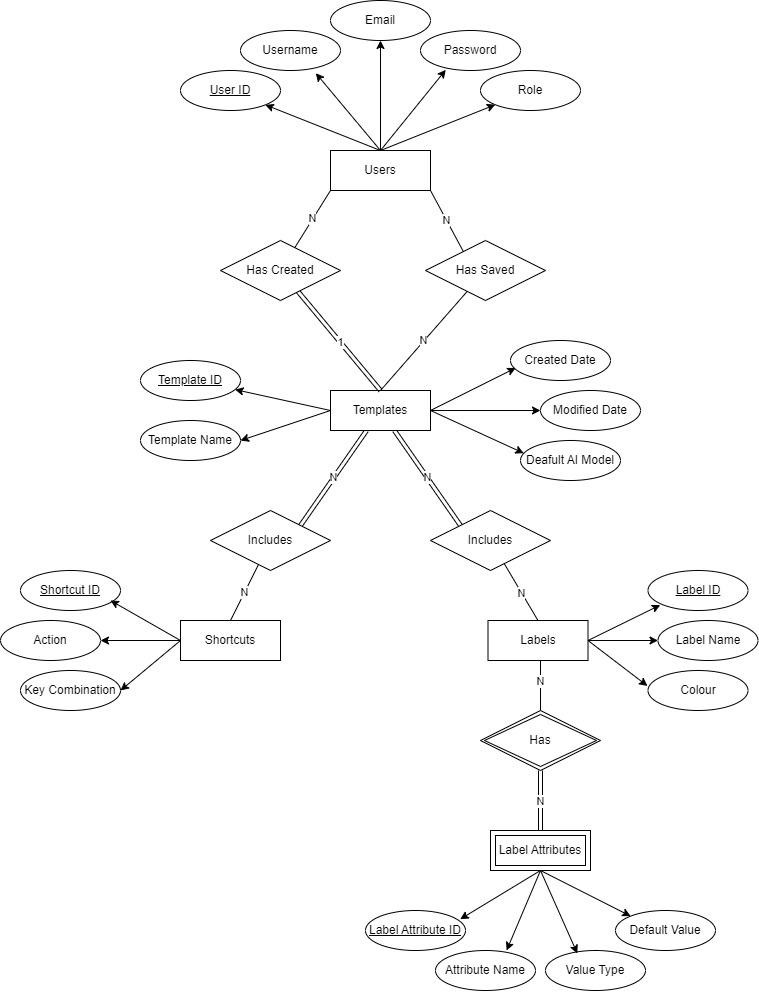
|  |
| --- |
| Next we looked at adding Distributed Storage and Processing for our next improvement to CVAT AI. A relational database management system such as MySQL is the most logical choice for implementing distributed storage and processing. MySQL is a well-known RDBMS that supports features such as replication and sharding, which are essential for the distributed storage and processing setup.  MySQL's support of replication and sharding enables efficient data distribution and parallel processing, which are essential for managing large annotation tasks. Additionally, MySQL's ACID properties ensure data consistency and transactional reliability, which are vital for managing annotation data in a distributed system.  Apache Spark, which is used for parallel processing, integrates seamlessly with MySQL using JDBC connectors. This integration enables efficient data retrieval and processing across a distributed infrastructure, significantly improving the speed and scalability of annotation tasks. By using MySQL and Apache Spark together, CVAT can achieve distributed storage and processing capabilities, fault tolerance, parallel processing, resource optimization, data consistency, and proactive management, resulting in improved efficiency and reliability in annotating tasks.  The next idea for improvement to CVAT AI is the introduction of reliability coefficients. Creating a reliability coefficient is not an overly complex idea and therefore could undoubtedly be achieved using any database management system; relational, document based, or graph based. Given that it is a relatively simple task, a relational database would be the ideal choice for generating and displaying a reliability coefficient to the user on CVAT. Different numbers of annotations of a particular label will correspond to different values of the reliability coefficient. MySQL provides a simple, fast way to access these different values when provided with the number of annotations. An index can be created on the number of instances annotated which allows for efficient look up time to find the corresponding reliability coefficient. SQL’s ‘count’ and ‘group by’ functions also make it easy to find the current number of annotations for each label. In summary, with regards to the ‘reliability coefficient’ enhancement idea, any database management system could be used but the simplicity and efficiency of MySQL combined with our knowledge and experience of the language would make it the perfect option for this idea.  We finally looked at the introduction of API keys as our final improvement to CVAT AI. For the creation of the API keys one of the reasons that we chose MySQL was because it provides a really structured environment to store data. It can use multiple tables like was suggested above to split the data. As mentioned earlier API keys are an extra layer of security for the user MySQL is able to offer encrypted connections and advanced user access controls which can aid in preventing any unwanted access and changing of any data which is crucial when trying to manage any API keys that restrict access to any files or applications. MySQL also offers very high performance query capabilities which would be incredibly helpful when it comes to retrieving data that may be based on multiple different attributes. The indexing features of MySQL also ensure that the data that relates to API keys are managed so that they can be accessed in both a quick and efficient way. |

Taking all enhancement ideas into consideration, it seems that a relational database is the most compatible option. Using MySQL, all five ideas can be implemented in a concise and efficient manner. Use of MySQL indexing is common throughout all ideas and is an essential part of the DBMS.

**Question 3**. Design and explain your data models.

## Group Member 1: (Min: ½ page – Max: 2 pages)

|  |
| --- |
| My data model will have the ‘Template’ entity as the main entity, with the various other entities either directly or indirectly relating to it. As discussed earlier there is a relationship between the template entity, and the labels and keyboard shortcut entities. There is also a relationship between the label entity and the label attribute entity, which exists to specialise and extend the labels of the annotation template if need be. As discussed earlier, all relationships found here are many-to-many relationships.  How does it relate to the other data models found within the project belonging to my teammates?  - These entities will be connected to a Users entity in the following way: users may have certain templates that they have created or saved to favourites, and two additional relationship entities will be included to describe these relationships. The first being a created entity, that will include the following attributes: a reference to User ID, and a reference to template ID. A template can be created by only one user, but a user can create many templates, so one-to-many from the created entity to the user and template entities, with the many sides being at the relationship entity. There is also a saved entity, that is a relationship entity with the following attributes: reference to User ID and reference to Template ID. A user can save many templates and a template can be saved by many users, so many-to-many relationships on both sides of this relationship entity. |



**Question 4**. Write queries to create database objects/records/documents with dummy data (310 records/objects/documents). Moreover, illustrate the use of these objects/records/documents in your ideas (3-5 queries/operations).

## Group Member 1: (Min: 300 words – Max: 1000 words)

|  |
| --- |
| Top of Form  --First lets create the various tables needed--  -----------------------------------------------------  CREATE TABLE 'Users' (  UserID INT PRIMARY KEY,  'Username' VARCHAR(50),  'Email' VARCHAR (50),  'Password' VARCHAR(50),  'Role' VARCHAR(50)  );  CREATE TABLE 'Templates' (  TemplateID INT PRIMARY KEY,  'TemplateName' VARCHAR(50),  'Guidelines' VARCHAR(1000),  'CreationDate' DATETIME,  'ModifiedDate' DATETIME,  'DefaultAIModel' VARCHAR(50)  );  CREATE TABLE 'TemplateCreation' (  UserID INT,  TemplateID INT,  Primary KEY (UserID, TemplateID),  FOREIGN KEY (UserID) REFERENCES 'Users'(UserID),  FOREIGN KEY (TemplateID) REFERENCES 'Templates'(TemplateID)  );  CREATE TABLE 'TemplateSaves' (  UserID INT,  TemplateID INT,  Primary KEY (UserID, TemplateID),  FOREIGN KEY (UserID) REFERENCES 'Users'(UserID),  FOREIGN KEY (TemplateID) REFERENCES 'Templates'(TemplateID)  );  CREATE TABLE 'Shortcuts' (  ShortcutID INT PRIMARY KEY,  TemplateID INT,  'Action' VARCHAR(250),  'KeyCombination' VARCHAR(50),  FOREIGN KEY (TemplateID) REFERENCES 'Templates'(TemplateID)  );  CREATE TABLE 'Labels' (  LabelID INT PRIMARY KEY,  TemplateID INT,  'LabelName' VARCHAR(255),  'Colour' VARCHAR(255),  FOREIGN KEY (TemplateID) REFERENCES 'Templates'(TemplateID)  );  CREATE TABLE 'LabelAttributes' (  LabelAttributeID INT PRIMARY KEY,  LabelID INT,  'AttributeName' VARCHAR(255),  'ValueType' VARCHAR(255),  'DefaultValue' VARCHAR(255),  FOREIGN KEY (LabelID) REFERENCES 'Labels'(LabelID)  );  --Now lets insert some dummy data into these tables--  -----------------------------------------------------------------  INSERT INTO 'Users' (UserID, 'Username', 'Email', 'Password', 'Role')  VALUES  (1, 'John Doe', 'johndoe@gmail.com', 'password1', 'Admin'),  (2, 'Jane Doe', 'janedoe@gmail.com', 'password2', 'Admin'), (3, 'Sean Doe', 'seandoe@gmail.com', 'password3', 'User');  INSERT INTO 'Templates' (TemplateID, 'TemplateName', 'Guidelines', 'CreationDate', 'ModifiedDate', 'DefaultAIModel')  VALUES  (1, 'Vehicle Detection', 'Use boxes when annotating vehicles. Pay special attention to bicycles and motorcycles, which might be less visible', '2023-02-01 14:22:56', '2024-04-12 23:43:11', 'YOLOv4'),  (2, 'Customer Tracking', 'Use ovals when annotating all the people entering and leaving the shop, and tag each which actions such as entering, leaving, or browsing. Ensure the right timestamp is used for actions', '2025-09-30 07:55:56', '2029-10-07 11:09:11', 'DeepLabv3+'), (3, 'Park Animal Tracking', 'Labels should be tagged by the species of the animal, and any interactions between animals should also be noted. If part of a group of the same species, note group size.', '2024-11-23 19:57:57', '2024-11-23 19:57:57', 'Mask R-CNN');  INSERT INTO 'TemplateCreation' (UserID, TemplateID)  VALUES  (1, 3),  (2, 2),  (1, 1);  INSERT INTO 'TemplateSaves' (UserID, TemplateID)  VALUES  (1, 1),  (1, 2),  (1, 3),  (2, 2),  (3, 1),  (3, 2);  INSERT INTO Shortcuts (ShortcutID, TemplateID, 'Action', 'KeyCombination')  VALUES  (1, 1, 'Continue to the next frame', 'Ctrl+Right'),  (2, 1, 'Go back to the last frame', 'Ctrl+Left'),  (3, 1, 'Mark this frame as important', 'Ctrl+I'),  (4, 2, 'Continue to the next frame', 'Ctrl+Right'),  (5, 2, 'Go back to the last frame', 'Ctrl+Left'),  (6, 2, 'Mark this frame for deletion', 'Ctrl+D+L'),  (7, 2, 'Return to this frame later', 'Ctrl+L'),  (8, 3, 'Continue to the next frame', 'Ctrl+Right'), (9, 3, 'Go back to the last frame', 'Ctrl+Left');  INSERT INTO Labels (labelID, TemplateID, 'LabelName', 'Colour')  VALUES  (1, 1, 'Car', '#000000'),  (2, 1, 'Bicycle', '#ff0000'),  (3, 2, 'Employee', '#000000'),  (4, 2, 'Customer', '#0000ff'),  (5, 3, 'Owl', '#000000');  INSERT INTO LabelAttributes (LabelAttributeID, LabelID, 'AttributeName', 'ValueType', 'DefaultValue')  VALUES  (1, 1, 'Vehicle Type', 'String', 'Car'),  (2, 2, 'Vehicle Type', 'String', 'Bicycle'),  (3, 1, 'Condition', 'String', 'Used'),  (4, 2, 'Colour', 'String', 'Red'),  (5, 3, 'Activity', 'String', 'Folding T-shirts'),  (6, 4, 'Age Group', 'String', 'Teen'),  (7, 5, 'Species', 'String', 'Bird'),  (8, 5, 'Group Size', 'Integer', '1');  --Finally let's use some SQL queries to illustrate my ideas-------------------------------------------------------------------------  --Find all the templates created by a specific user, and also note the number of times each has been saved to a users 'Favourites'--  SELECT t.TemplateName, COUNT(ts.UserID) as TimesSaved  FROM Templates as t JOIN TemplateCreation as tc ON t.TemplateID = tc.TemplateID LEFT JOIN TemplateSaves as ts ON t.TemplateID = ts.TemplateID  WHERE tc.UserID = 1  GROUP BY t.TemplateID, t.TemplateName;  --Get all the details related to a specific template, including all labels and the labels's default attributes--  SELECT t.TemplateName, t.Guidelines, l.LabelName, l.Colour, la.AttributeName, la.DefaultValue  FROM Templates as t JOIN Labels as l ON t.TemplateID = l.TemplateID JOIN LabelAttributes as la ON l.LabelID = la.LabelID  WHERE t.TemplateID = 1;  --Get all the shortcuts related to a specific template--  SELECT t.TemplateName, s.Action, s.KeyCombination  FROM Templates as t JOIN Shortcuts as s ON t.TemplateID = s.TemplateID WHERE t.TemplateID = 1;  --Display the stats related to the templates, including who created it, how many times it has been saved, and the most recent occasion it was modified--  SELECT t.TemplateName, u.Username as Creator, COUNT(ts.UserID) as TimesSaved,  MAX(t.ModifiedDate) as LastModified  FROM Templates as t JOIN TemplateCreation as tc ON T.TemplateID = tc.TemplateID  JOIN Users as U ON tc.UserID = u.UserID  LEFT JOIN TemplateSaves as ts ON t.TemplateID = ts.TemplateID  GROUP BY t.TemplateID, t.TemplateName, u.Username;  FROM ReliabilityCoefficient AS rc JOIN AnnotationCount AS ac  WHERE rc.num\_instances = ac.num\_instances AND  ac.label = ‘car’;  /\* returns 0.60, so more annotations will need to be done manually to increase the reliability coefficient \*/  //get the current reliability coefficient for small vans  SELECT ac.label, coefficient\_value  FROM ReliabilityCoefficient AS rc JOIN AnnotationCount AS ac  WHERE rc.num\_instances = ac.num\_instances AND  ac.label = ‘small van;  //returns 0.20, again lots more annotations will need to be done manually. Bottom of Form |
|  |

**Question 5**. Write your reflections on contributions and lessons learnt.

Group Member 1:

***(Min: 100 words – Max: 300 words)***

|  |  |
| --- | --- |
| Now that our assignment has ended, it is time for reflection on my contributions and the lessons learned during the assignment.  Much of the learning, at-least for myself came, from the topic of CVAT AI itself, how it is used, and its need/purpose. CVAT AI is a piece of software that I have personally never encountered myself, and trying to produce improvements and enhancements was challenging at the start, especially without any experience in using this software before. I believe through my experimentation with this software, I produced a topic that could greatly improve CVAT AI, especially for beginners such as myself - the introduction of templates, which will help streamline projects and improve consistency and efficiency.  The assignment was mostly an individual project, with five different avenues for improvement explored by my teammates and I, and much of the teamwork came during the brainstorming stage, and during our collaborative effort in writing Q2. As most of the duration of this assignment occurred during Easter recess, meetings had to occur virtually. This is the first time I have collaborated on a project virtually, and I learned a number of teamwork skills that will be beneficial to me when I go out and begin my first professional career.  I am happy with my contribution to this assignment and feel as though if I were to have completed this assignment again, there are not many things I would have changed. A look at my time management skills may be needed, and maybe a more challenging CVAT AI improvement could be explored, but besides these few titbits, I feel confident that the 8/10 score I have given myself is more than deserved.   |  | | --- | | **Self-rating: 8 / 10** | |