جامعة قطر
QATAR UNIVERSITY

SID: 202104114 STUDENT NAME: Anas Madkoor Effort given Task1 33%

SID: 202007437 STUDENT NAME: Abdullah al naemi Effort given Task2 33%

SID: 201803368 STUDENT NAME: Mansoor afeer Effort given Task 3 33%

- Course number: CMPS310
- Submission date: October 28th
- Theory Class section: L02

جامعة قطر
QATAR UNIVERSITY

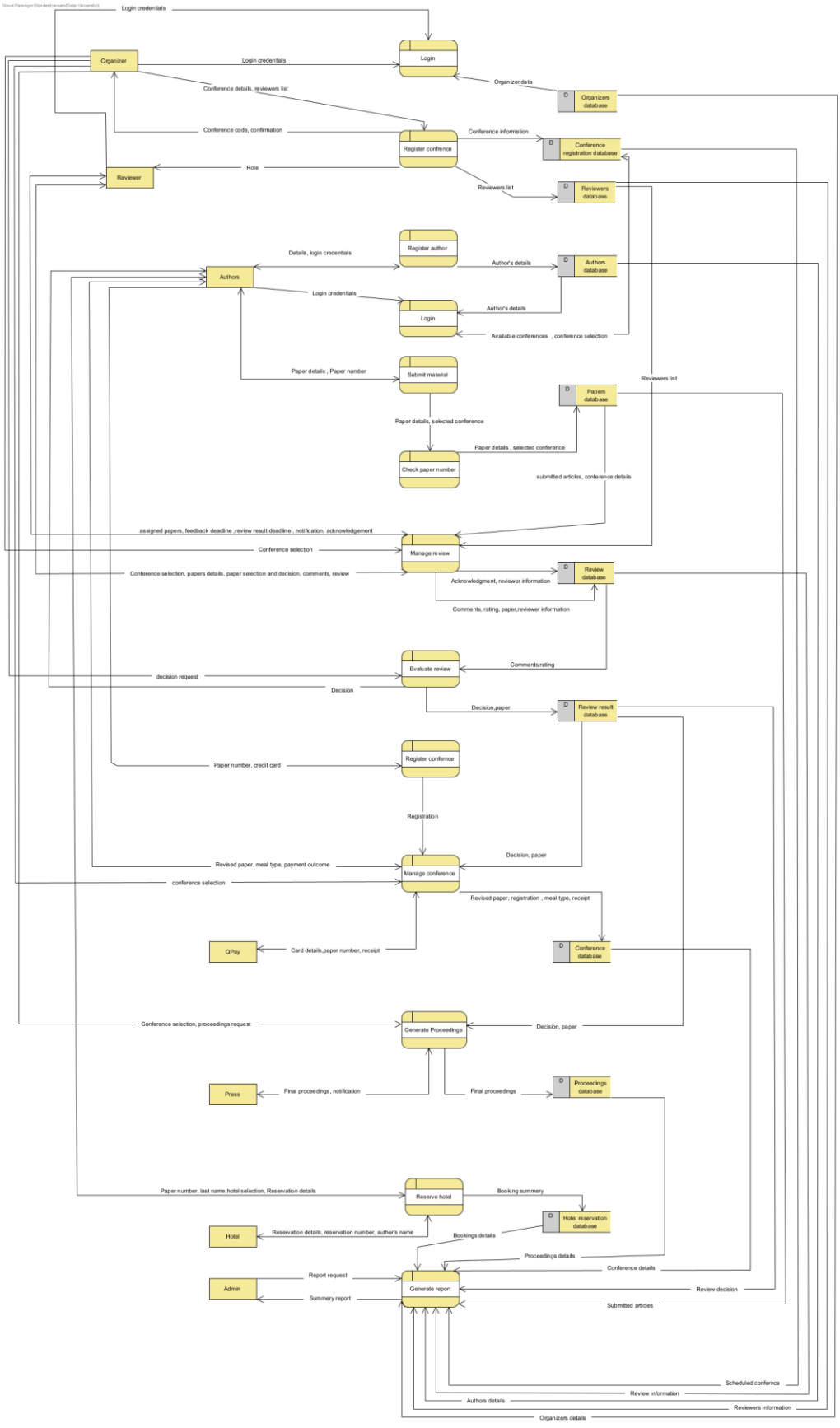
Task 1:

1.Data flow diagram

Explanations:

- Deliberately named “Submit materials” process that way, the author could submit paper/ article
- There’s no payment process as it’s done by external entity (Qpay)

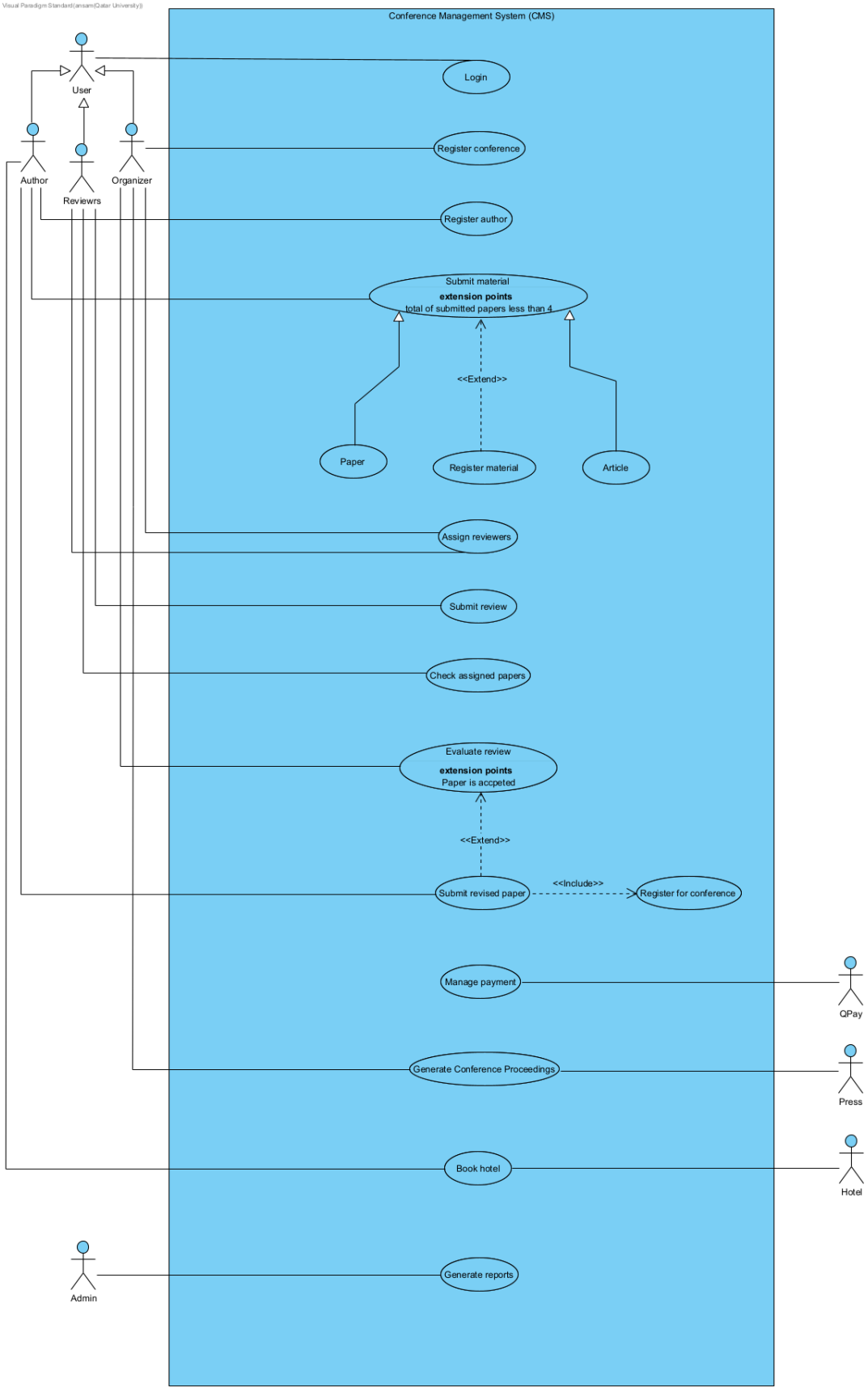
جامعة قطر
QATAR UNIVERSITY



2.Use case diagram

Explanations:

- The payment process is done by QPay, thus there's no payment process in CMS but, the system still interacts with QPay by sending and receiving details, I intentionally called the use case "manage payment" to indicate that the transaction itself does not happen in CMS but an actor is responsible for handling it (QPay) then send the outcome of the transaction to the system
- The admin could be generalized under user, logically the admin have to login but since theres not much detail I left as its



3.Explanation

Use case diagram:

Use Case Diagrams primarily focus on the functional requirements and user interactions within a system. These diagrams are excellent for providing an overview of the system's functionality from the user's perspective. However, their limitations become evident when attempting to represent the internal data flow and process intricacies of a CMS.

Advantages:

- Excellent for illustrating user-system interactions.
- Useful for identifying actors, use cases, and their relationships.
- Communicates high-level system behavior effectively.

Limitations:

- Lacks the detail required to represent data flow and internal processes.
- Doesn't explicitly show data storage or data transformations.
- Less suitable for technical stakeholders requiring a deeper understanding of data flow.

Data Flow Diagrams (DFDs):

DFDs are a modeling technique that excels at representing data flows, data processing, and internal system processes. In the context of a CMS, DFDs offer a range of advantages:

Advantages:

- Detailed data modeling: DFDs explicitly illustrate how data is processed, transformed, and stored within the system.
- Process modeling: Complex workflows and processes within a CMS are represented with clarity, including sub-processes and interactions.
- Data store representation: DFDs show data stores, providing a clear understanding of where and how data is stored.
- Data transformation: DFDs depict how data is transformed as it moves through various processes.
- Integration with other diagrams: DFDs can be used in conjunction with other diagrams, such as entity-relationship diagrams, to provide a holistic view of the CMS.

Conclusion:

While Use Case Diagrams are valuable for capturing high-level functionality and user interactions, they are insufficient for representing the internal workings of a CMS. In contrast, DFDs provide a more detailed, technical, and precise view of data flows and processes. This level of detail is indispensable for system design, development, and maintenance.

For a CMS, where data management, processing, and data integrity are paramount, DFDs are the superior choice for representation. Its ability to depict data flow, data storage, and process intricacies makes them the preferred modeling technique when a comprehensive understanding of a system's inner workings is required.

In conclusion, when representing a complex system like a CMS, the choice between Use Case Diagrams and DFDs should be driven by the need for detailing data flow, process, and internal system dynamics. For a CMS, where such depth is essential, DFDs emerge as the best-suited option.

جامعة قطر
QATAR UNIVERSITY

Task 2:

Use case Id:	<Register Conference>
Brief Description	This use case allows organizers to register a new conference or update details of an existing conference.
Primary actors	Organizer
Trigger(s)	Organizer logs in and initiates the conference registration/update process.
Preconditions: Organizer must be logged in, and the conference name must not already exist in the system.	
Post-conditions: Conference details are registered/updated in the system, and the organizer receives a confirmation message.	
Normal Scenario	
Actor Action	System Response
1. Organizer logs in	2. System validates login credentials.
3. Organizer selects "Register Conference" option.	4. System displays a form for entering conference details.
5. Organizer enters conference details.	6. System checks if the conference name already exists.
	7. If the conference name does not exist, system records conference details.
	8. System generates a unique conference code and confirms registration to the organizer.
Alternative flows: **5.a** If the conference name exists, system displays an error message and request new details	

جامعة قطر
QATAR UNIVERSITY

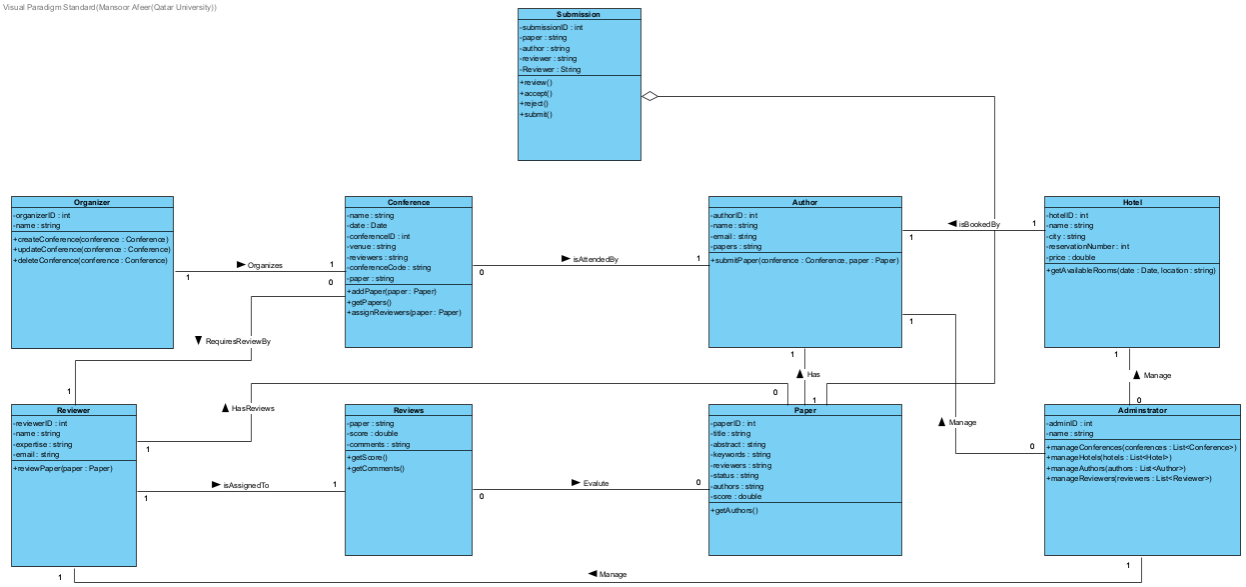
Use case Id:	**<Manage Review>**
Brief Description	This use case allows organizers and reviewers to manage the review process for submitted papers.
Primary actors	Organizer, Reviewer
Trigger(s)	Organizer selects a conference
Preconditions: Organizer is logged in and reviewers are available for assignment	
Post-conditions: Reviews are completed for all submitted papers, and the organizer can make a decision on each paper.	
Normal Scenario	
Actor Action	System Response
1. Organizer selects a conference and initiate in assigning reviews.	2.System retrieves conference details and list of submitted papers.
	3. System checks reviewer availability and assigns reviewers to papers.
	4. System notifies reviewers about their tasks and deadlines
5. Reviewer logs in.	6. System displays paper abstracts and reviewer deadline.
7. Reviewer accepts a paper to review.	8. System records reviewer acceptance and allows submission of comments and rating.
9. Reviewer submits comments and rating.	10. System records reviewer feedback and marks the paper as reviewed.
Alternative flows: • **7.a** If a reviewer declines to review a paper, the system assigns an alternative reviewer.	

جامعة قطر
QATAR UNIVERSITY

Use case Id:	**<Submit Articles >**
Brief Description	This use case allows authors to submit papers/articles to a selected conference.
Primary actors	Author
Trigger(s)	Author logs in and initiates the paper submission process.
Preconditions: Author must be logged in, and the selected conference must be open for submissions.	
Post-conditions: Paper details are recorded, and authors receive a paper number upon successful submission.	
Normal Scenario	
Actor Action	System Response
1. Author logs in.	2.System validates login credentials..
3. Author selects "Submit Article" option.	4. System displays a list of available conferences.
5. Author selects a conference.	6. System records the selection and prompts for paper details..
7. Author enters paper details.	8. System validates the information and allows file upload.
9. Author uploads the paper.	10. System stores paper details, assigns a paper number, and confirms submission to the author.
Alternative flows: **7.a** If an author attempts to submit more than three papers to a single conference, the system displays an error message and terminates the submission process.	

Use case Id:	**<Evaluate review >**
Brief Description	This use case allows authors to submit revised paper and register for conference based on the feedback from the reviewrs.
Primary actors	Author
Trigger(s)	Author receives notification about the acceptance of their paper.
Preconditions: Author must have an accepted paper and be logged in.	
Post-conditions: Author's registration and hotel reservation details are stored, and payment confirmation is received.	
Normal Scenario	
Actor Action	System Response
	1. system sends notification to the author about the decision of the paper
	2. System asks to submit revised paper and to register for conference
3. Author submits revised paper and provides paper number and credit card details.	4. System validates if the paper is accepted.
5. Author selects meal preference.	6. System stores the choice of the meal
	7. System forwards card details and paper number to QPay.
	8. System stores the outcome of the payment and notify the author
Alternative flows: • 3.a if the paper number does not exist the system displays error message and asks the author for valid paper details	

Task 3:



Application of design principles in the conference management system class diagram:

Design Principles

The following design principles were applied in the class diagram of the conference management system:

- Single responsibility principle: Each class is responsible for a specific set of tasks, making it easier to modify and update. For example, the Conference class is responsible for managing conferences, while the Paper class is responsible for managing papers, and the Author class is responsible for managing authors.
- Independence principle: The classes are designed to be independently variable from each other, making it easier to add new features. For example, the Conference class does not depend on the Paper class to function correctly, so new features can be added to the Paper class without having to modify the Conference class.
- Abstraction principle: The classes are designed to be as abstract as possible, making them easier to reuse in other contexts. For example, the Reviewer class is abstract about any specific type of review, so it can be used to review different types of papers.

Unit principle: The system is divided into independent classes, each responsible for a specific set of tasks. For example, the Conference class is independent of the Paper class.

Consistency principle: The classes are designed in a consistent way with each other, making it easier to understand the system. For example, the Conference class is associated with the Paper class using the aggregation relationship.

Extendibility principle: The classes are designed in a way that their functionality can be easily extended. For example, the Reviewer class is associated with the Paper class using the association relationship details.

Here are some specific examples from the system class diagram of how these principles were applied:

- Single responsibility principle:
    - The Conference class is responsible for creating conferences, managing papers, reviews, events, and attendance.
    - The Paper class is responsible for managing the paper content, authors, and reviews.
    - The Author class is responsible for managing author data and papers.
- Independency principle:
    - The Conference class does not depend on the Paper class to function correctly.
    - The Paper class does not depend on the Author class to function correctly.
- Abstraction principle:
    - The Reviewer class is abstract about any specific type of review.

- Unit principle:
    - The Conference class is independent of the Paper class.
- Consistency principle:
    - The Conference class is associated with the Paper class using the aggregation relationship.
- Extendibility principle:
    - The Reviewer class is associated with the Paper class using the association relationship.

Conclusion

The application of these design principles resulted in a class diagram that is easy to understand, modify, and extend.