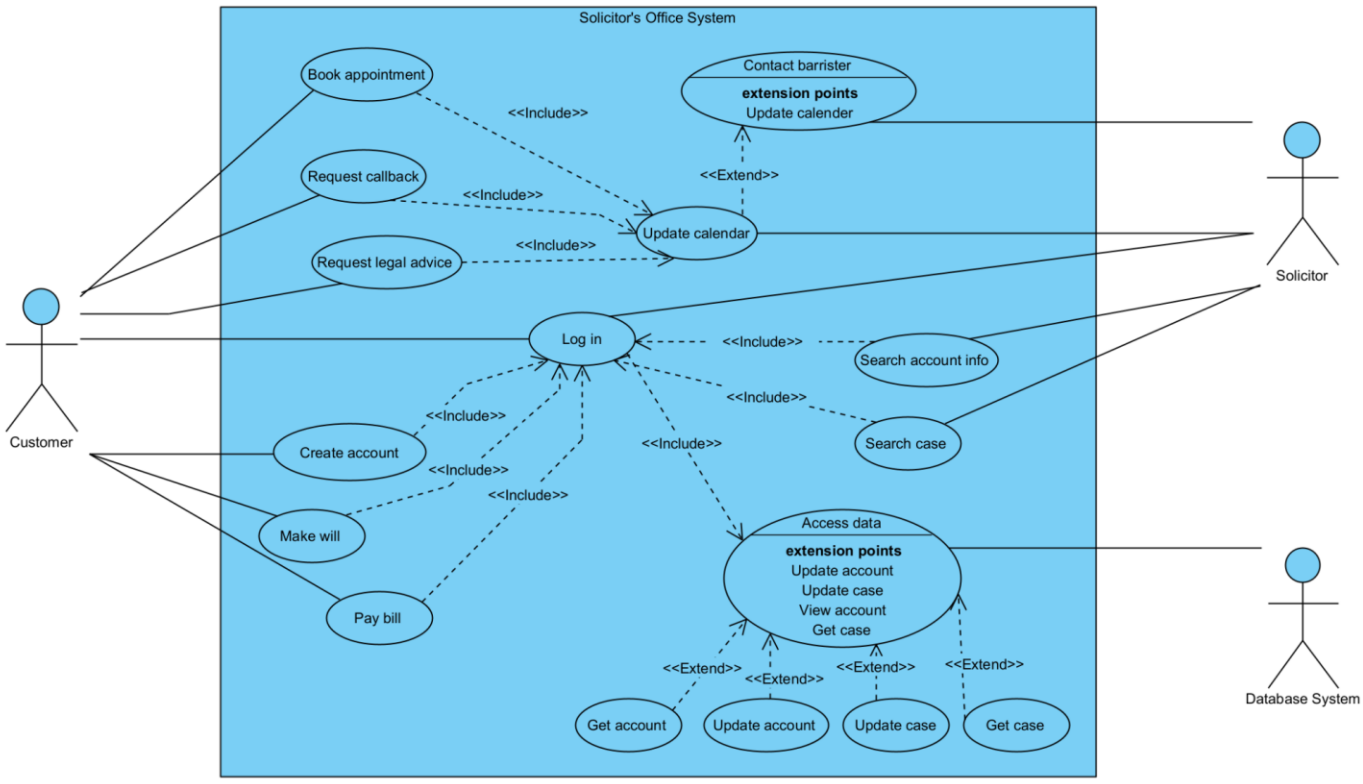
OO Software Engineering Project
Part A

Exercise 1 - ***Use Case Model***:
Denis Murray



Actors Explained:

- Customer: the customer is able to log into the system and perform some basic functions, such as paying an outstanding bill or making a quick will. For any legal advice or to speak with a solicitor, the customer requests such and the system updates a solicitors calendar for the day booked.
- Solicitor: the solicitor can log contact barristers and update their calendar through the system. They can also search by customer accounts and case files and update these.
- Database system: the database system handles any requests for data for the system. If the customer or solicitor requests data, for example the solicitor searching for a case to update or the customer wants to view their account, the database system pulls the correct data from the database and shows and alters the data depending on the user's authorization level.

Exercise 2 - ***Use Cases in detail:***

Use Case 1: Pay bill
Denis Murray

The customer can request to pay their bill through the solicitor's office system.The customer must log in to proceed. A successful flow has the customer requesting to pay the balance, the database management system returns the account balance, the system informs the customer of the balance, receives payment from the customer, and the database management system updates the account with the new balance.

An alternate flow involves following the same steps up until receiving payment from the customer. Instead, the customer does not pay immediately. The customer is informed of the balance and reminded of the timeframe to pay.

An exceptional flow is followed when the customer does not pay their balance before the set out time frame. In this scenario, the customer is informed of this and a legal document is provided to the customer with the information to pay.

Use Case 2: Book appointment
Miroslav Tadej

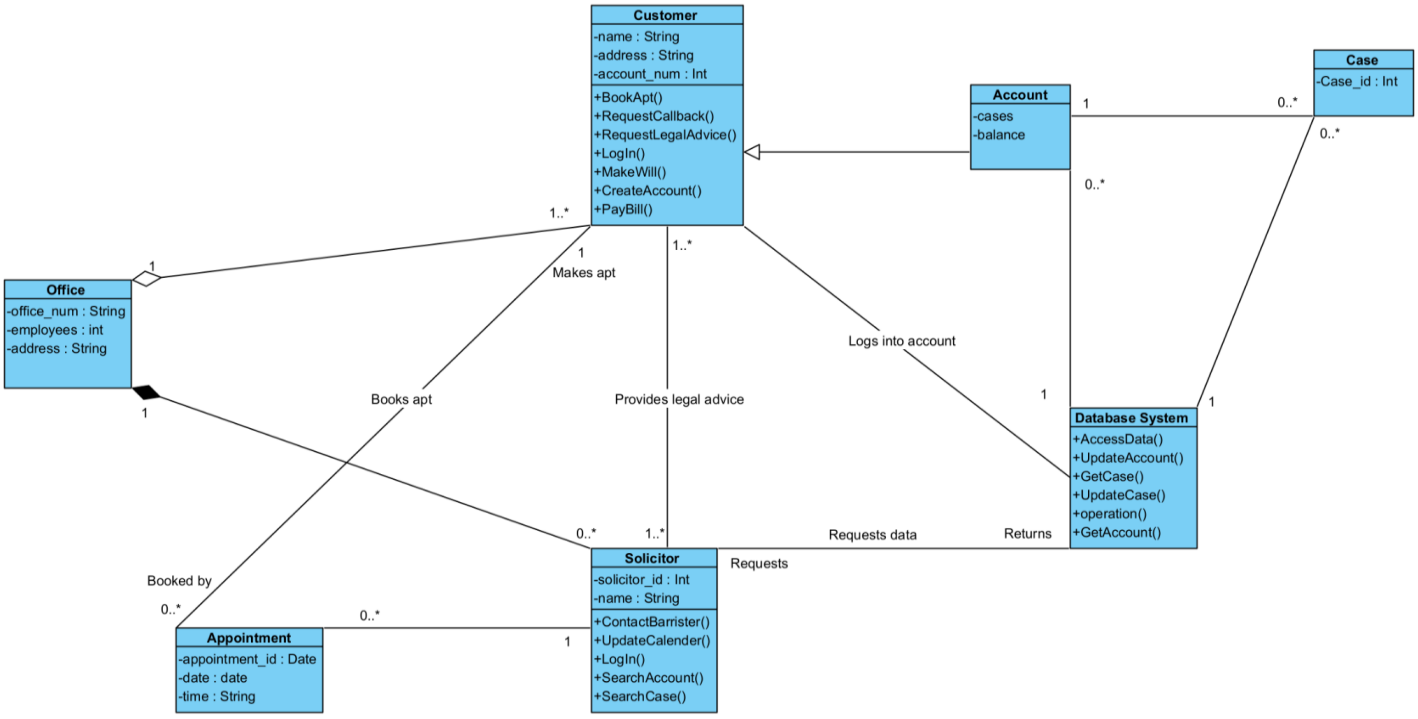
The customer can request to book an appointment with a solicitor through the system. The customer requests a day, the system checks for availability and returns whether that is available, and the system updates the solicitor's calendar with the appointment.

An alternative flow can be followed by following the same steps until the system checks availability. If the requested day is not available, the system will suggest and alternative day to the customer. The system will then update the solicitor's calendar.

An exceptional flow can be reached by following the same steps as the successful flow until the system checks for availability. If the requested day is unavailable, the system will attempt to find another suitable time. If there are no days available, the system will return an error message to the customer and inform them that there are no suitable days available.

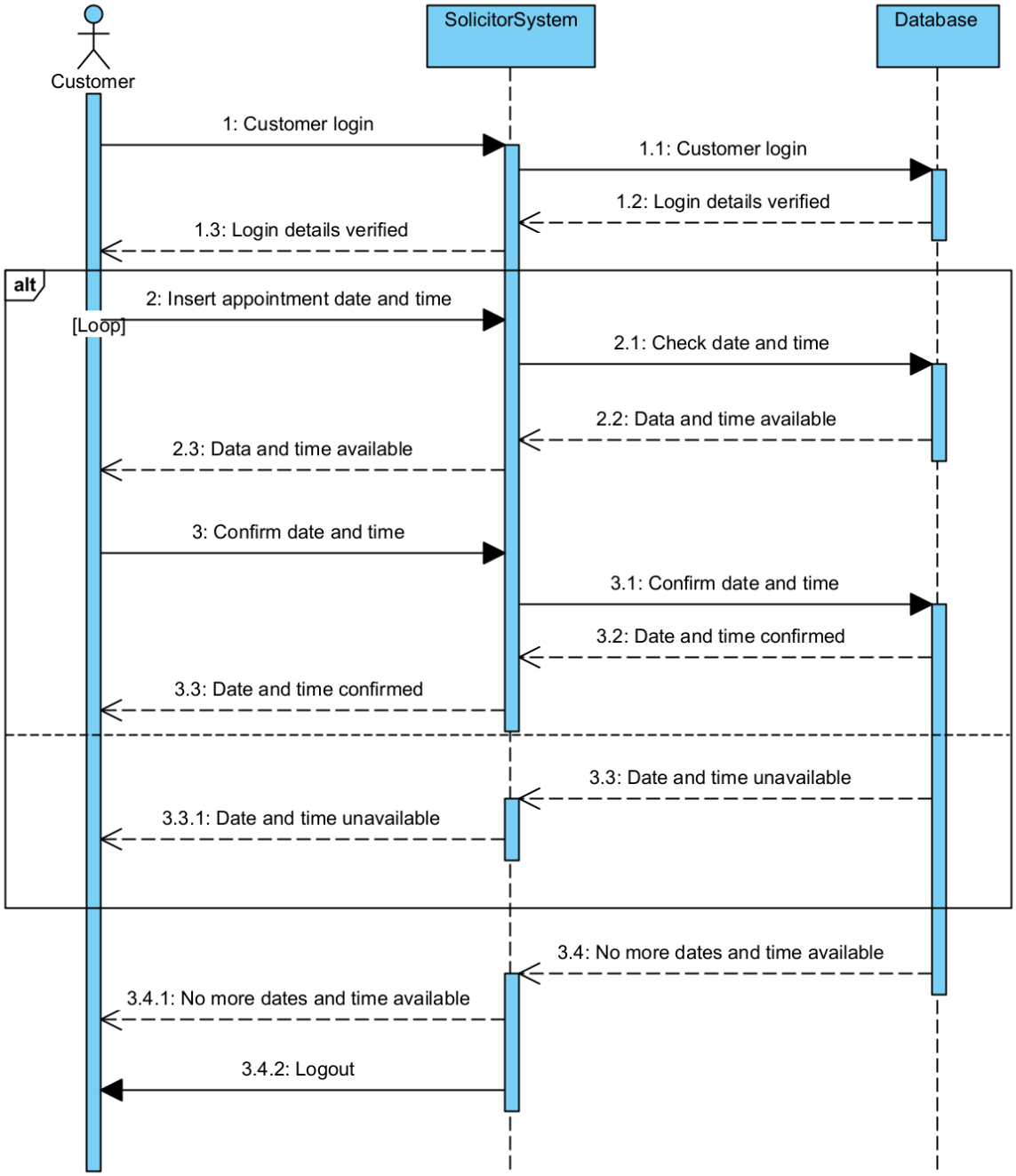
Exercise 4: - ***Class Diagram***

Denis Murray

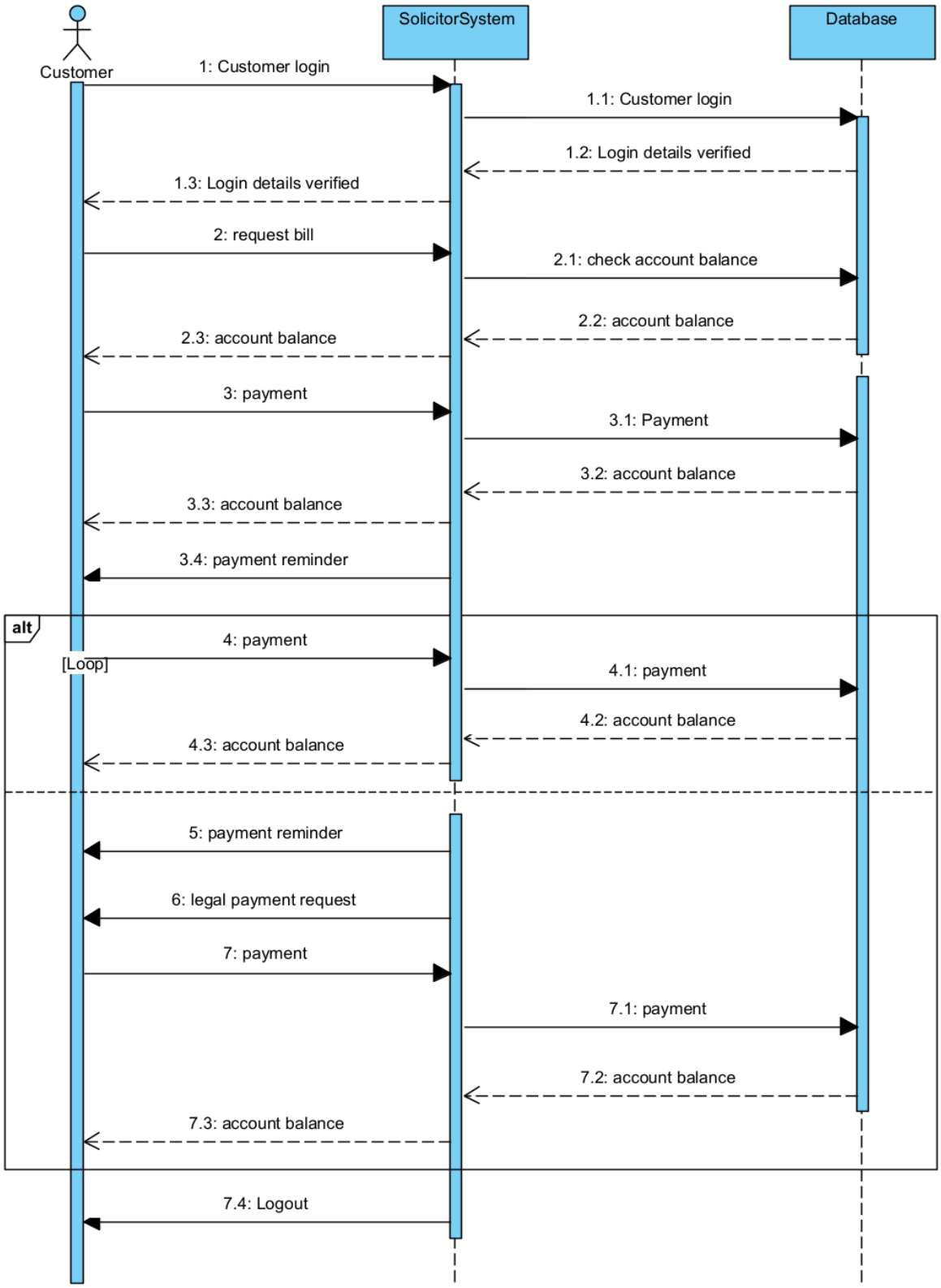


Exercise 5: - ***System Sequence Diagram***
Miroslav Tadej

- Book Appointment System Sequence Diagram



- Pay Bill System Sequence Diagram



Exercise 6: - ***Contracts***

Use Case 1: Pay bill
Denis Murray

Operation Contract for "payBill ()"	
Operation:	payBill (payBillId: Integer, value: Double)
Cross References:	Use Cases: Pay bill
Preconditions:	There is a payment process underway
Postconditions:	• A payBill instance was created. (Instance creation) • payBill was associated with the current customerRecord (id). (Association formed) • payBill was associated with an balance, based on a match. (Association formed)

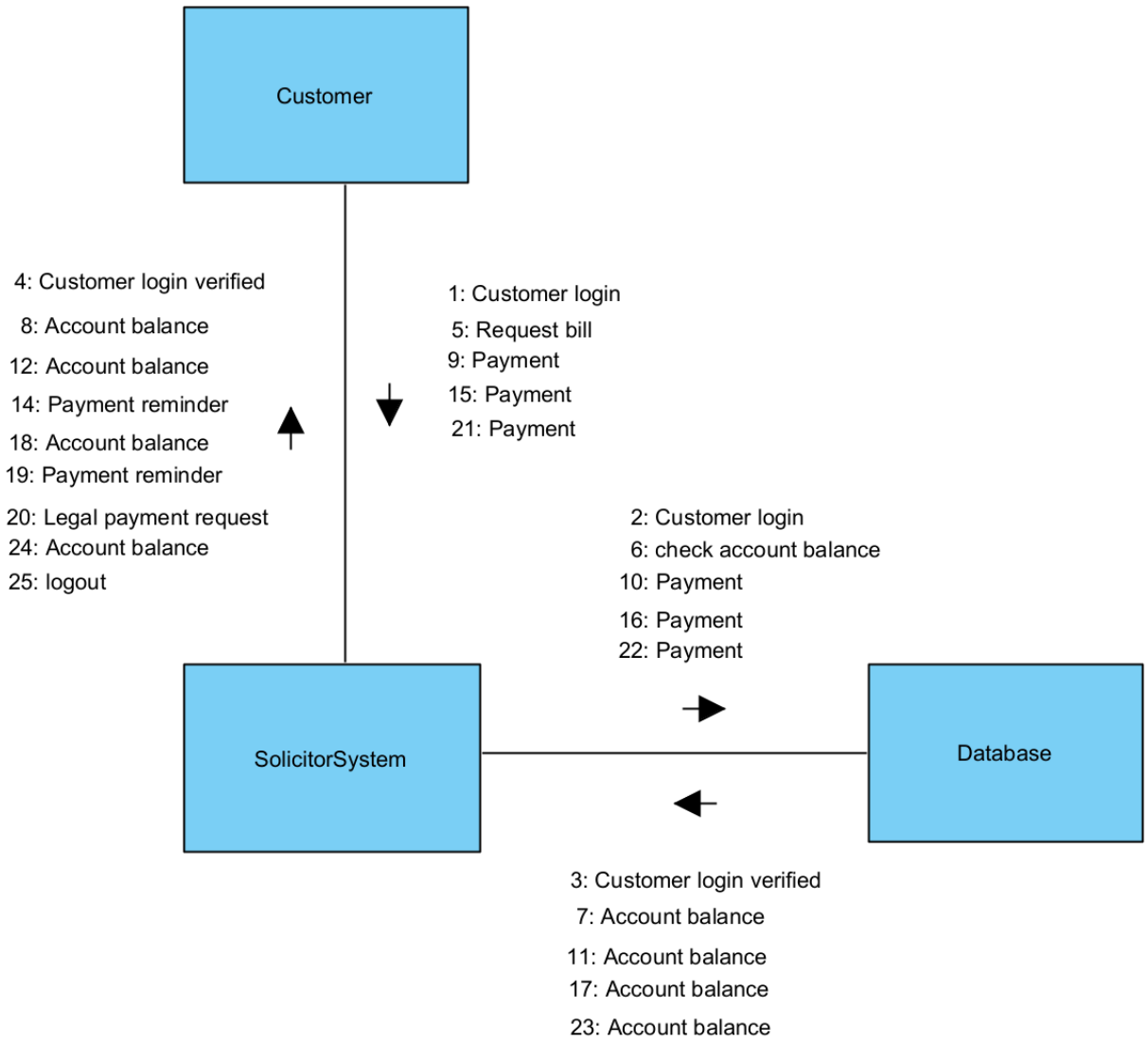
Use Case 2: Book appointment
Miroslav Tadej

Operation Contract for "bookAppointment ()"	
Operation:	bookAppointment (appointment: date, time)
Cross References:	Use Cases: Book appointment
Preconditions:	There is an appointment booking underway
Postconditions:	• An appointment was created. (Instance creation) • bookAppointment was associated with the customerRecord (id). (Association formed) • bookAppointment was associated with an appointment (date, time), based on match. (Association formed)

Exercise 7 - ***Communication Diagram***

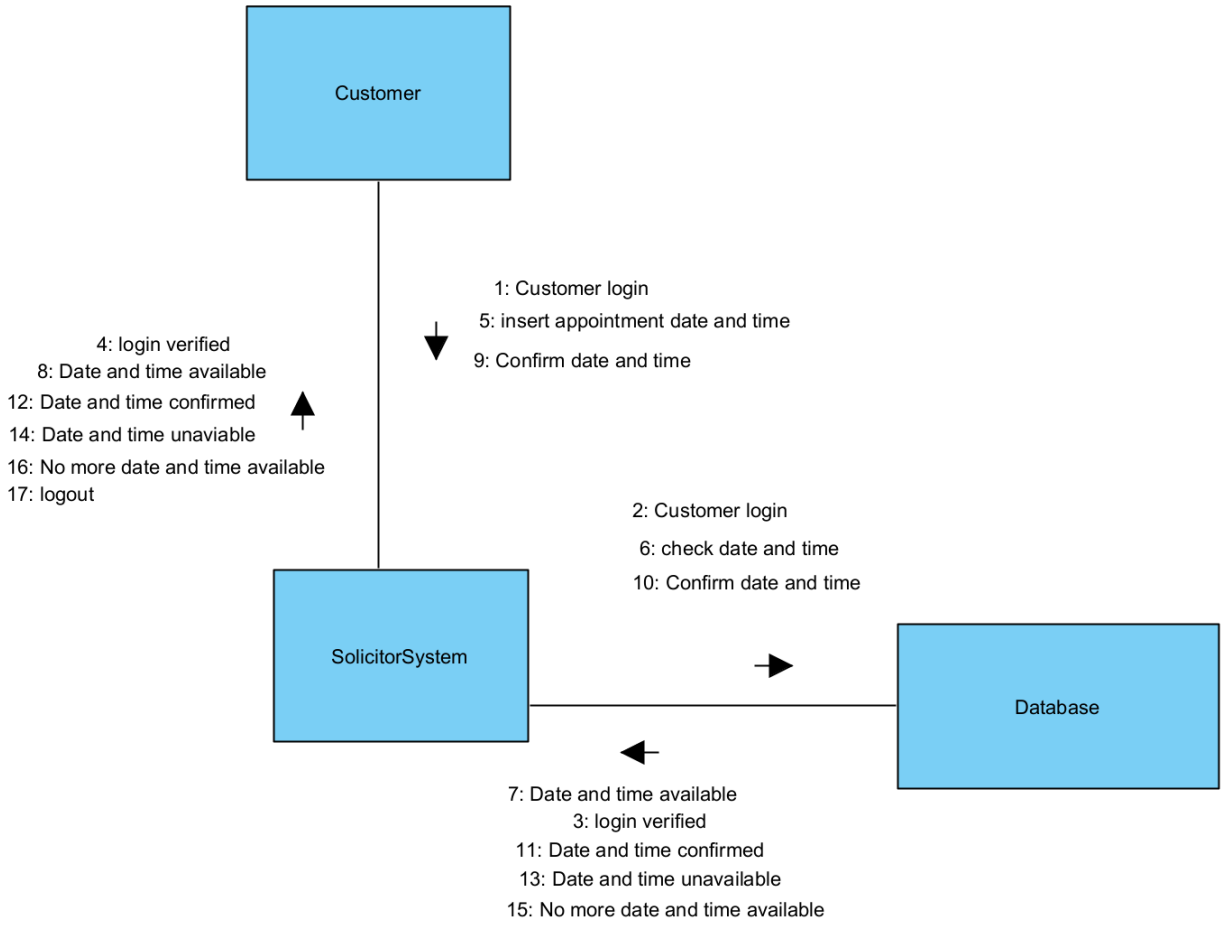
Use Case 1: Pay bill

Denis Murray



Use Case 2: Book appointment

Miroslav Tadej



Exercise 3 - ***Glossary***
Denis Murray

Term	Definition
Use case	“A Use Case instance is a sequence of transactions performed by the system, which is initiated by a particular actor and yields a meaningful, measurable result of value to a particular actor” (Andrews, Week 2)
Actor	“Represents anything outside the system that interacts with the system” (Andrews, Week 2)
Association Relationship	“An association is a relationship between classes that indicates some meaningful and interesting connection” (Andrews, Class Diagrams, Week 3)
Include relationship	Indicates that a use case is mandatory and part of the base use case (Kashif, 2022)
Exclude relationship	Indicates that a use case is optional and comes after the base use case (Kashif, 2022)
Alternate flow	“Step or a sequence of steps that achieves the use case’s goal following different steps than described in the main success scenario” (Lohmeyer, 2013)
Exceptional flow	“Anything that leads to not achieving the use case’s goal” (Lohmeyer, 2013)
Multiplicity	“The active logical association when the cardinality of a class in relation to another is being depicted.” (Nishadha, 2022)
Aggregation	“Aggregation implies a relationship where the child can exist independently of the parent” (Visual Paradigm, n.d.)
Composition	“Composition implies a relationship where the child cannot exist independent of the parent” (Visual Paradigm, n.d.)
Generalization	Shows that a subclass inherits from a superclass (Andrews, Class Diagrams, Week 3)
Contracts	“Describe detailed system behaviour, after a system operation has executed.” (Andrews, Operation Contracts,

Term	Definition
	Week 5)
Communication Diagram	Diagram that shows the interactions between elements at run-time… effective at visualizing processing over time. (Sparx Systems, n.d.)