Ministry of Education, Culture and Research of the Republic of Moldova

Technical University of Moldova

Department of Software and Automation Engineering

**REPORT**

Laboratory work No. 6

Discipline: AMS

Topic: Study and analysis of OO abstractions and classes in UML (class diagrams).

Analysis and modeling of an online store (e-commerce)

Done by: Lupascu Felicia

st.gr. FAF212

Verified by: univ.lect.  
 Sava Nina  
Melnic Radu

Chișinău 2022

**Objective**: studying the notions of state, compound state with concurrent, submitted and disjoint substates, activity, guard condition, synchronization bar (fork, join), decision block.

**Task:** to make 3 state diagrams and 3 activity diagrams for the chosen information system

**Theoretical considerations:**

A **state diagram** consists of states, transitions, events, and activities. You use state diagrams to illustrate the dynamic view of a system. They are especially important in modeling the behavior of an interface, class, or collaboration. State diagrams emphasize the event-ordered behavior of an object, which is especially useful in modeling reactive systems. You use state machines to model the behavior of any modeling element, although, most commonly, that will be a class, a use case, or an entire system which focuses on the event-ordered behavior of an object, which is especially useful in modeling reactive systems.

A **state machine** is a behavior that specifies the sequences of states an object goes through during its lifetime in response to events, together with its responses to those events.

A **state** is a condition or situation during the life of an object during which it satisfies some condition, performs some activity, or waits for some event.

An **event** is the specification of a significant occurrence that has a location in time and space. In the context of state machines, an event is an occurrence of a stimulus that can trigger a state transition.

A **guard condition** is evaluated after the trigger event for the transition to occur. It is possible to have multiple transitions from the same source state and with the same event trigger, as long as the guard conditions do not overlap. A guard condition is evaluated just once for the transition at the time the event occurs. The boolean expression may refer to the state of the object.

A **transition** is a relationship between two states indicating that an object in the first state will perform certain actions and enter the second state when a specified event occurs and specified conditions are satisfied. Activity is an ongoing non-atomic execution within a state machine.

An **action** is an executable atomic computation that results in a change in the state of the model or the return of a value.

The **activity diagram** is another important behavioral diagram in UML diagram to describe dynamic aspects of the system. Activity diagram is essentially an advanced version of a flow chart that models the flow from one activity to another activity.

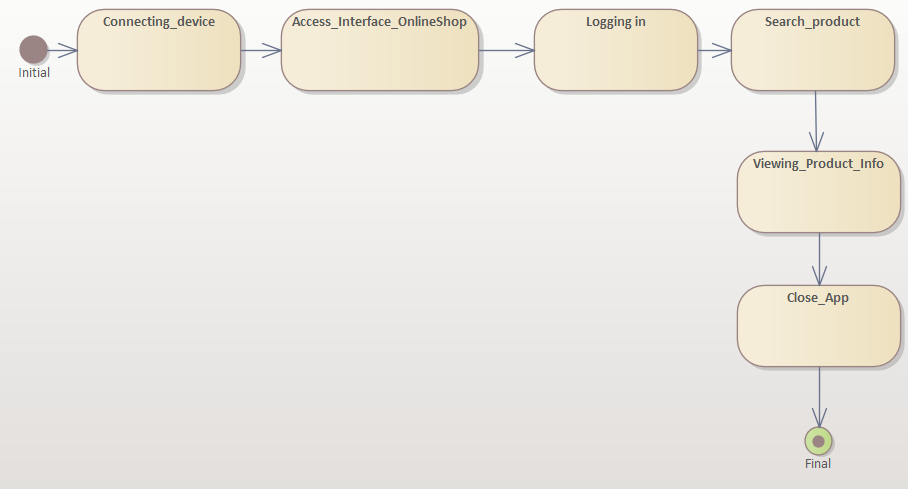
Activity Diagrams describe how activities are coordinated to provide a service which can be at different levels of abstraction. Typically, an event needs to be achieved by some operations, particularly where the operation is intended to achieve a number of different things that require coordination, or how the events in a single use case relate to one another, in particular, use cases where activities may overlap and require coordination. It is also suitable for modeling how a collection of use cases coordinate to represent business workflows.

Activity diagrams are constructed from a limited number of shapes, connected with arrows.[4] The most important shape types:

* ellipses represent actions.
* diamonds represent decisions
* bars represent the start (split) or end (join) of concurrent activities
* a black circle represents the start (initial node) of the workflow
* an encircled black circle represents the end (final node).

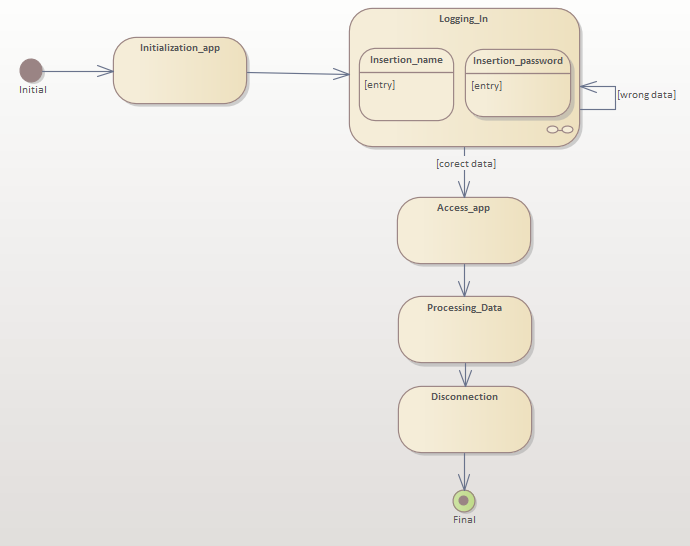
Arrows run from the start towards the end and represent the order in which activities happen.  
Activity diagram is suitable for modeling the activity flow of the system. An application can have multiple systems. Activity diagram also captures these systems and describes the flow from one system to another. This specific usage is not available in other diagrams. These systems can be databases, external queues, or any other system.

The chosen topic is **Analysis and modeling of an online store (e-commerce**). In recent times, the advancement of wireless technology and the growth of market potentials have led to an increase in the number of mobile device users. The emergence of this technology gave rise to the rapid development of mobile e-commerce technologies. It brings on-the-go Internet access to the general online market, without geographical and time constraints.

**Implementation, practical results:**  


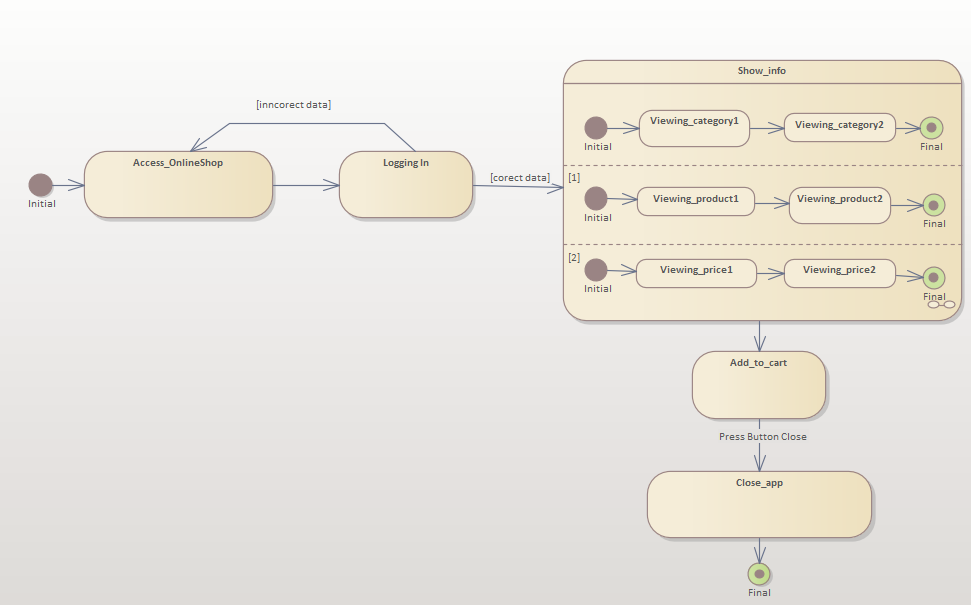
***Figure 1.*** *System Online Shop Diagram (state diagram)*

State diagram no. 1 represents some of the activities carried out within the Online Shop application. The diagram starts with the initial point, continues with several states that represent stages in the work process in the system. The relationship between the states being a transition relationship. The diagram ends with the final point.



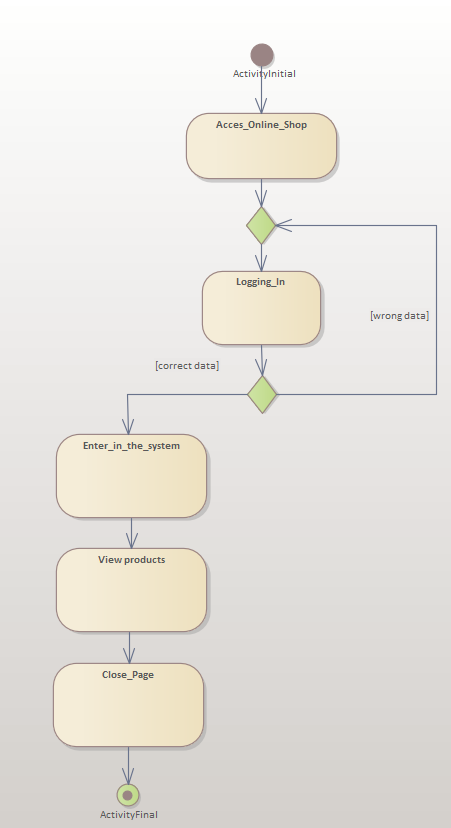
***Figure 2.*** *Logging In Diagram (state diagram- substari depuse)*

State diagram no. 2 represents the logging process in more detail. The diagram starts with the initial point, continues with 1 state (Initialization\_app), and then with the State Machine (Logging In). Where the user enters the name and password. If they are wrong, the user returns to entering the data. Then other states follow (Access\_app, Processing\_Data, Disconnection) and the diagram closes with the final point.



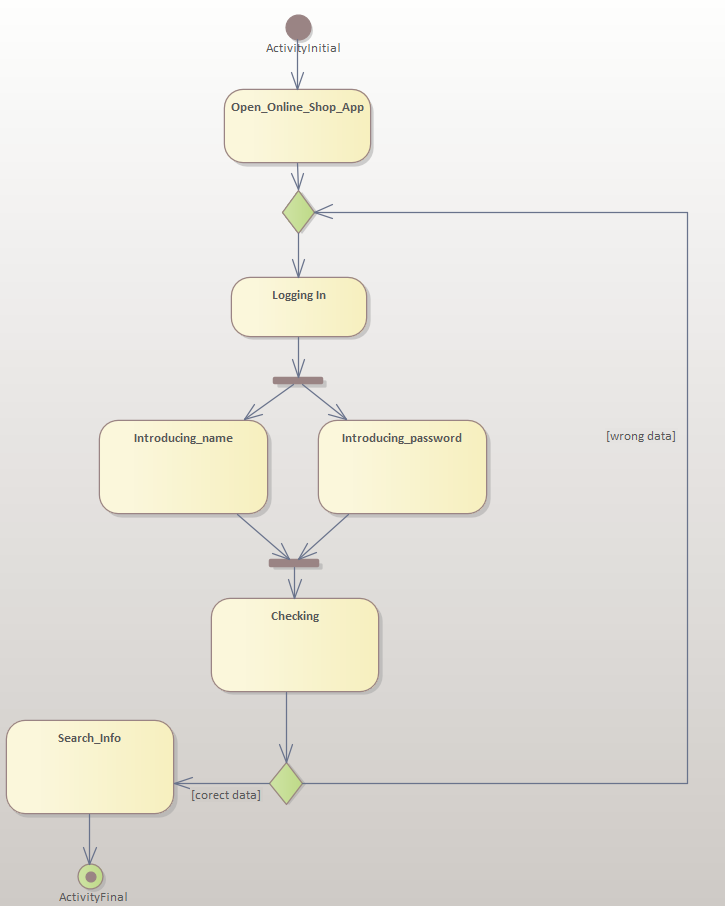
***Figure 3.***  *View product info Diagram (parallel substates, state diagram)*

State diagram no. 3 represents the State Machine with a substate composed of parallel (concurrent) substates. The user can search for categories, products, prices. The next action is that the user can add the product to the card, and in the end to close the app. I also used guard conditions ( [inncorect data] ) and comments (Press Button Close). **Guard conditions** is information written between square brackets. When the condition is met, the system continues its work process, otherwise either the process is repeated or the work of the system (application) is interrupted. A comment is a message written on the transition relation. It aims to explain in more detail the process of system transition from one state to another.



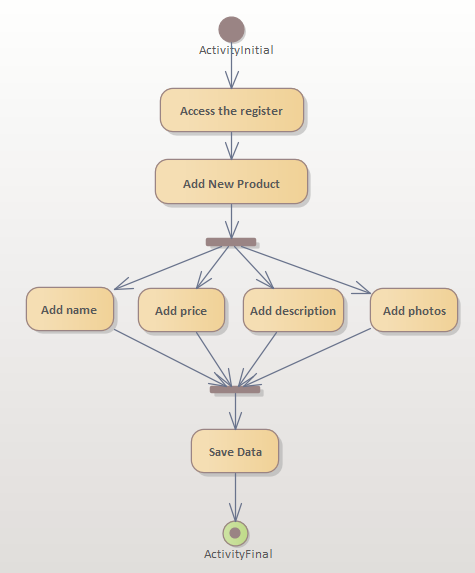
***Figure 4.*** *Logging In Diagram (activity diagram)*

Figure 4 Loggin In Diagram shows us some different types of actions in Online Shopping System from initial activity to Final Activity. I used the Guard Condition. **Guard condition** - is information written between square brackets. When the condition is met, the system continues its work process, otherwise either the process is repeated or the work of the system (application) is interrupted. **Brancher** – used to connect multiple transitions together. The brancher will NEVER have guard conditions noted.



***Figure 5.*** *Logging In Diagram (activity diagram)*

Activity diagram no. 5 represents the login process in the Online Shopping application. I use branching, guard condition, transition relation, join, fork , initial point, and final activity. **Join** represents the timing bar that waits for two or more tasks to complete and connects them together. **Fork** - represents the synchronization bar that branches two or more activities that take place in parallel.

**

***Figure 6.*** *Add New Product Diagram (activity diagram)*

Activity diagram no. 6 represents the process of adding a new product to the product catalog. Fork - represents the synchronization bar that branches two or more activities that take place in parallel. Join - represents the synchronization bar that waits for the fulfillment of two or more activities and connects them together.

**Conclusions:** The purpose of this lab work is to understand and familiarize ourselves with State and Activitiy diagrams, the connections between them and the states. In this lab work I created some 6 diagrams. State diagrams enable you to describe the behaviour of objects during their entire life span. In addition, the different states and state changes as well as events causing transitions can be described. On other words: State diagrams make the system behaviour visible. The activity diagram is used to demonstrate the flow of control within the system rather than the implementation. It models concurrent and sequential activities. The activity diagram helps in envisioning the workflow from one activity to another. In conclusion I can say that these diagrams are very important and useful.

**Bibliography**

1. **Melnic R., Sava N.** Indrumar metodic “Analiza si modelarea sistemelor informationale”.
2. **Моделирование бизнес процессов|CASE средства|Rational Rose**, [Электронный ресурс].-Режим доступа: <http://www.kpms.ru/Automatization/Rational_Rose.htm>