**Ticketing Domain – Persistence Level (SQL, NoSQL) + DAL**

1. What are your steps to start designing database?

To start designing database, it’s necessary to

- determine the purpose of the database: the kind information to be stored, its consumers (people who will use that information) and the way they will interact with it

- find and organize the information required: gather all of the types of information you might want to record in the database, create a description of the data type, size, and format

- start designing with division the information into subjects (or entities), each one then becomes a table.

1. When can we say that our database is modeled correctly?

A modeled-correctly database divides the information into subject-based tables (it helps to reduce redundant data), provides with the necessary details to gather the information from various tables as required, aids in maintaining and assuring the precision and consistency of the stored data and accommodates the stored data processing and reporting needs.

1. What is a Data Access Layer (DAL), and how does it simplify database interactions?

A Data Access Layer (DAL) is a part of the software application (module, project or any else), which is responsible for interaction with the database. In a three-tier layered application together with Business Logic (BL) and Presentation layers, forms a well-supported modular system, each part of which has its own responsibilities. DAL provides the simplified and consistent methods for accessing, retrieving and manipulating the stored data.

The simplification of the interactions is about:

- encapsulating of details of operations, leaving only a single interface

- single interface provision allows the BL layer’s services to be independent on the type of database used, so any changes to the DB structure (adding a new table, changing a data type) don’t affect the business logic. Same to complete change of DB (e.g. from MySQL to PostgreSQL)

- increased security to validate queries for SQL injections.

1. You need to implement a new service for a customer. How would you select database (SQL or NoSQL)?

The choice depends on

- budget bounds

- the way data should be structured (NoSQL will be more flexible and will handle any structural changes of data or storing the unstructured one)

- the speed/consistency needs (SQL databases follow ACID principles and its data is always consistent, but NoSQL doesn’t follow with having a speed and simplified scalability)

- needs to handle complex queries (SQL databases have powerful querying capabilities, when NoSQL are not good at dealing with complex queries, but cope with quick processing of large amounts of high-velocity, variable data).

**Ticketing Domain – Introduction**

1. What diagram types do you have on your current project?

There is a component diagram and a state diagram on my current project.

1. What are the pros and cons of UML diagrams?   
   – learning of UML may be a long process (due to variety of standard details and diagram types)

– usually take much time for creation  
– take time for post-creation support (keeping them up to date)   
– it’s important to keep the diagrams simple – that influences on their readability and understanding

+ standardization

+ don’t require technical knowledge to be reviewed by customers, clear for clients, designers and developers

+ allow to decompose complex systems into smaller pieces

1. What is the difference between a structural diagram and a behavioral diagram in UML?

As their names say, structural diagram shows the structure of the system components and their relationship & interaction.

A behavioral diagram describes the way of processes held between the components, how they work over time, the messages and events between the components and the way they are handled.

1. Name the relationship types in a use case diagram

Association, Extend, Include, Use case or actor generalization

1. What is the sequence diagram?

A type of UML diagram that shows the way and the order of objects operate with one another over time. The objects involved in the sequence are represented as a vertical dotted “lifelines”. The interaction between those lifelines is shown with horizontal arrows in the same order (from up to down) the actions are performed