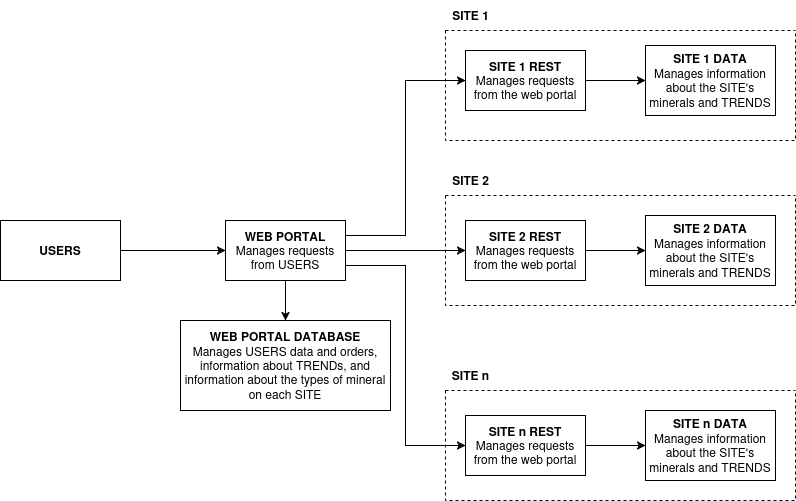
1)



2) The external services can be linked to the structure by using REST requests.

3) The token could be passed around with the use of HTTP Sessions.

4) One safe way to update the database can be the use of SQL Transactions and EF migrations, but I’ve also heard of Liquibase.

5) One way to handle persistence cases would be to again use SQL Transactions and EF migrations. If data loss due to device failure could be a problem, a load balancing solution could be implemented.

6) I would create a new table that represents the type of a mineral (with the fields Id, Name, Traits) and that is linked to the main table through a foreign key.

7) There are two possible solutions: the first is to use shared tables, where an update to one also cascades to the other, but could leave inconsistencies between the tables; the second would be to use an external database to which they are linked, but that would increase the costs. The first solution is best when there is a small numbers of shared entities, but for a large number the second solution is best.

8) No.

9) I don’t have experience with cloud, so I can’t answer this question.

10) Yes, I have used both JUnit and NUnit. I believe that test driven development is the best solution as it forces developer to think of the feature they are creating as a black box, where they don’t have to think about the implementation but only what to expect of it.

11) As I think that test driven development is best, I would plan Unit Tests in a test driven fashion.