CHALMERS

EXAMINATION / TENTAMEN

Course code/kurskod	Course name/kursnamn			
DIT 345	Fundamental of software architecture			
Anonymous code Anonym kod		Examination date Tentamensdatum	Number of pages Antal blad	Grade Betyg
DIT 345-0059-EJJ		2023-10-27	9	

* I confirm that I've no mobile or other similar electronic equipment available during the examination.
Jag intygar att jag inte har mobiltelefon eller annan liknande elektronisk utrustning tillgänglig under eximinationen.

Solved task Behandlade uppgifter No/nr		Points per task Poäng på uppgiften	Observe: Areas with bold contour are to completed by the teacher. Anmärkning: Rutor inom bred kontur ifylles av lärare.
1	X	16	
2	X	24	
3	X	36	
4	X	22	
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
15			
16			
17			
Bonus poäng			
Total examination points Summa poäng på tentamen		98	

[redacted]

CHALMERS	Anonymous code Anonym kod	Points for question (to be filled in by teacher) Poäng på uppgiften (fylles av lärare)	Consecutive page no. Löpande sid nr
	DIT345-0059-EJJ	16p	1
			Question no. Uppgift nr 1

Question P1

(A) Technical constraint:

The system shall only operate on Android phones. 2.5p

Business constraint:

The system shall be launched before March 2024. 2.5p

(B) Functional requirements:

1) The system shall allow users to sign up and sign in. 2.5p

2) The system shall allow users to access statistics about the company, including comleted trips, cash flow, % of booked vehicles. 2.5p

(C) Quality attribute requirements:

1) Availability: The system shall ~~be~~ have a minimum uptime of 99,75%. 3p

2) Security: The system shall encrypt all of the data about cargo and its real time location. 3p

2

CHALMERS

Anonymous code Anonym kod DIT 345-0059-EJJ	Points for question (to be filled in by teacher) Poäng på uppgiften (fylles av lärare)	Consecutive page no. Löpande sid nr 2 Question no. Uppgift nr 2

Question P2

(A) 1) QAS for performance 6/6

Source : 1000 users from 3 different continents ✓ (but not Europe)

Stimulus : Sign up to the system ✓

Artefact : User service ✓

Environment : normal load ✓

Response : Sign up for all of the users is processed & completed ✓

Response measure : within 2 seconds ✓

2) QAS for security 6/6

Source : Hacker ✓

~~Stimulus : attacks the user se~~

Stimulus : Attacks the system to leak user passwords ✓

Artefact : User service / User DB ✓

Environment : normal operations ✓

~~Response : The user service shuts down and an audit trail is kept~~

~~Response measure : within 5 minutes~~

Response : An audit trail is kept and hacker is discovered ✓

Response measure : within 2 minutes ✓

Answer only one question on this page. Do not write on the back of this paper
Behandla endast en uppgift på detta blad. Skriv ej på baksidan

CHALMERS

Anonymous code / Anonym kod: DIT 345-0059-EJJ

Points for question (to be filled in by teacher) / Poäng på uppgiften (ifylles av lärare)

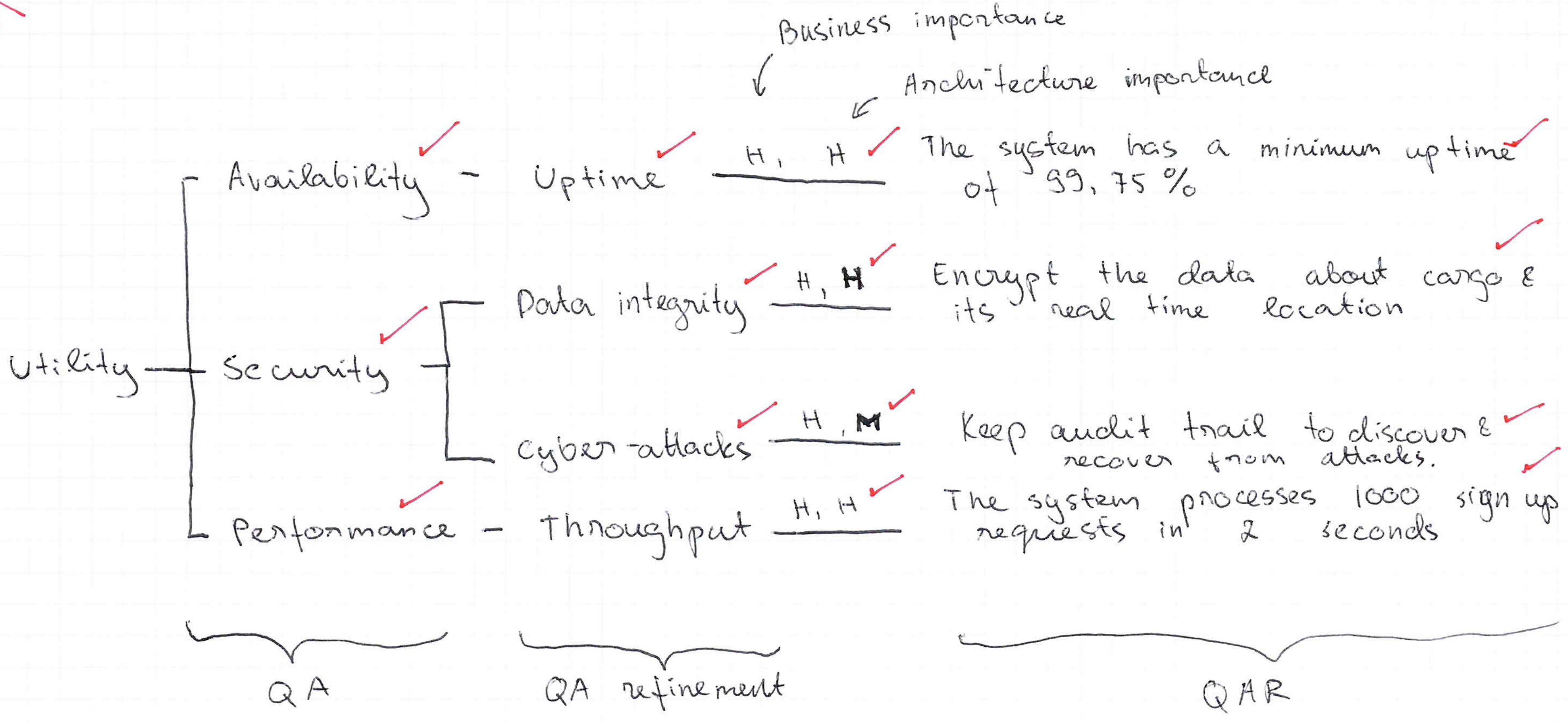
Consecutive page no. / Löpande sid nr: 3

Question no. / Uppgift nr: 2

Question P2

(B)

12/12



Business importance
Architecture importance
Utility
Availability
Uptime
H, H
The system has a minimum uptime of 99,75 %
Security
Data integrity
H, H
Encrypt the data about cargo & its real time location
Cyber-attacks
H, M
Keep audit trail to discover & recover from attacks.
Performance
Throughput
H, H
The system processes 1000 sign up requests in 2 seconds
QA
QA refinement
QAR

~~Since~~ It is not mentioned in the task, but I added the importance for business and architecture since it is usually a part of a utility tree

Question P3

1p

(A) This diagram shows that the architecture for the system is monolithic. The diagram has mostly modules. which are not even grouped ~~groped~~ together as components. Even though there is a separation between the database and the rest of the application, I would not call this architecture layered. The MLAV Application component does not separate the modules inside in any way. There is direct communication between the UI modules and the "management" modules. In this case we can clearly see that the diagram tries to use layered style, however, it is not done successfully, since the "UI layer" and "controller layer" are not grouped together and could not be easily replaced. ~~to this~~ Thus, it is monolithic, since all modules are pretty coupled. A layer should have one responsibility but here the MLAV Application component serves for UI and the "controller" functionalities.

1p

(B) This is not a very good solution because, it is not very modifiable nor scalable. ~~All the ser~~ One of our main requirements was that the system shall have a high uptime. In this diagram, all modules use the same database. In case something happens, the whole database (and thus system) is down. In addition, our system wants to work in multiple continents which assumes that the system shall be scalable. In this architecture, it is very difficult to add any services. since the changes should be made also in the database and there are a lot of dependencies. This is also the reason why this architecture is not very modifiable. In case of a new features, changes have to be made in most likely at last 3 modules/components.

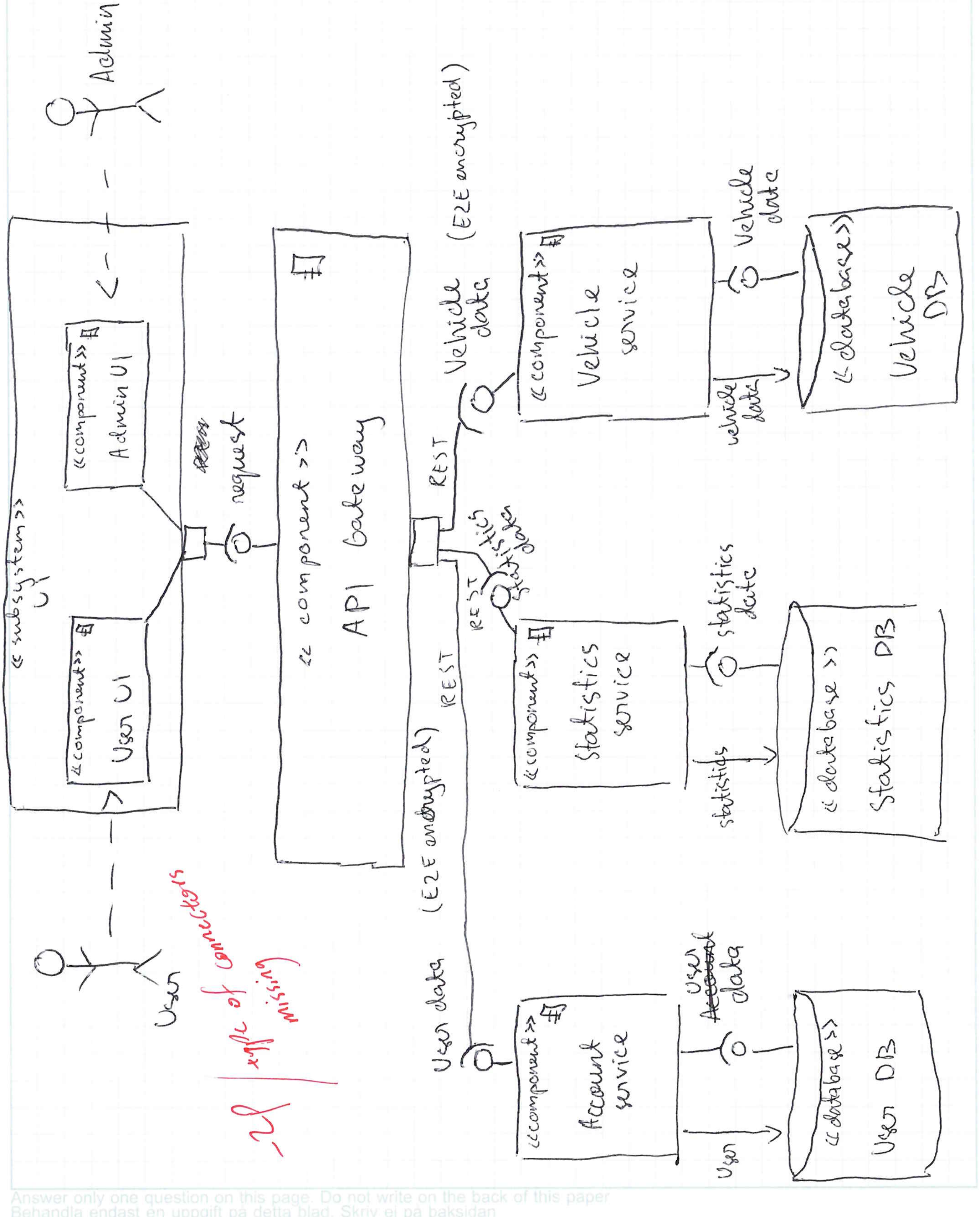
This architecture might be good for performance ~~on some cases~~ ~~security~~, however ~~for a~~ in the long run it would cause technical debt due to not being modifiable nor scalable. Also not good for security since all data is in a single database. If a hacker gets access to vehicle data, hes also access to user data.

Anonymous code Anonym kod	Points for question (to be filled in by teacher) Poäng på uppgiften (ifylles av läraren)	Consecutive page no. Löpande sid nr	5
DIT345-0059-EJJ		Question no. Uppgift nr	3

Question P3

Continues on next page ->

Ⓒ * There was no requirement stating that users can place orders on vehicles / rent them. If there was (seems logical), I would add another service "Order service" with a separate DB and also a component "payment service" with a separate DB and a component Stripe API that would send and request paymend data with the Payment service.



-2 | Lot of connectors missing

2

CHALMERS

Anonymous code Anonym kod DIT345-0059-EJJ	Points for question (to be filled in by teacher) Poäng på uppgiften (ifylles av lärare)	Consecutive page no. Löpande sid nr	6
		Question no. Uppgift nr	3

Question P3

15p

(c) Look at the diagram on previous page.

I used microservices architecture ~~because~~, I have a separate component for the UI. The ~~UI~~ [illegible] UI component interacts with the API Gate Way, which handles the requests, sends them to the appropriate service, it acts as an orchestrator/mediator. There are 3 different services with each their own database.

The tactics I used:

1) Because the microservices are not very good for performance, I would introduce concurrency so that ~~some of~~ these services could be used at the same time.

2) For availability, I used the tactic active redundancy. It is not shown in the diagram because of room issues :(but I would have a replica of the most important components, for example the ~~Account service and~~ Account DB.

If the Account DB goes down, there would be an extra copy, which would be all the time synchronized with the actual DB and in case of failure, could take over.

~~3) For security~~
~~But microservices promotes~~

3) For security, I would use multiple tactics, e.g:

3.1) Authenticate users

3.2) Encryption of data that is related to user passwords or vehicle real time location.

4) For performance I would additionally:

Increase computational efficiency and

Decrease computational overhead, however this ~~can~~ more relies on the developers and the quality of their code.

Answer only one question on this page. Do not write on the back of this paper
Behandla endast en uppgift på detta blad. Skriv ej på baksidan

Question P3

1p

(1) My solution (using microservices architecture) is better because:

1) Incresed availability: Microservices is very decoupled. In case of a failure in one of the services, the others are fully unaffected. This can be even further promoted by using circuit breakers. In addition, I use the active redundancy tactic, making the mean time ~~betw~~ to repair very little, since there is an exact replica of a component that can always take over.

2) Increased security & privacy:

Microservices are very secure since the services are so uncoupled. In the diagram done by my coworker, there is one single database, meaning that in case of a cyber-attack, all the data could be accessed from one place. In my design, all services use different databases, thus, even if the vehicle DB gets hacked, no user data can be accessed and vice-verse.

In addition, I introduced two tactics for authenticating users and encrypting data. I would also keep an audit trail of all actions to easily detect hackers.

3) My solution is also better for scalability and modificability since it is very easy to add a new service to my architecture; however in my coworker's solution, adding a new "service" would require changes to the existing database and most likely other modules as well.

Question P4 Normal

I participated in the workshop & I remember the tradeoffs.

(A) Tradeoffs 8/8

1) Performance ————○———— Security ✓

We chose to use microservices architecture which promoter security since the services are very decoupled and thus data is also in separate databases and not accessible from one point. However, microservices ~~does~~ not do good for performance since network calls take longer to make. Also, encrypting data ~~adds to the~~ decreases the computation efficiency. Thus, having a very secure system often means that performance is not as good. On the other hand, sending real time data with good performance does not give time to encrypt.

2) Cost ————○———— Scalability ✓

In our group, we decided that the Bästraffik system would have an increasing user base and would most likely want to move to more cities etc. Thus, we chose an architectural style that would promote this.

However, designing a system that would be scalable & modifiable is expensive. The cheapest would be to build a monolith, but that is not scalable. We decided to avoid technical debt and prioritize later changes than make a cheap system.

(B) My group chose Microservices style. Because we 8/8 thought it was best for scalability, modifiability, security and availability. Other groups had mostly the same choice as us. which is why I would still choose microservices. From the presentations we got good tactics to deal with performance issues in microservices (such as introduce concurrency, decrease computational overhead, scheduling)

So I think the performance issues could be solved. In addition. Bästraffik was funded by tax payer so money was not really a problem.

I think microservices offer the best quality of service for such an app. It is better to design a more costley app that will have better modifiability and scalability and thus a larger user base. Microservices are also very secure since the date is distributed between databases & services are decoupled.

2

CHALMERS	Anonymous code Anonym kod DIT 345-0059-EJJ	Points for question (to be filled in by teacher) Poäng på uppgiften (ifylles av lärare)	Consecutive page no. Löpande sid nr 9 Question no. Uppgift nr 4

Question P4 Normal

(c) I got more insight about how all of the quality attributes are important. For instance, how a huge percentage of users leave the app if it takes more than 5 seconds to load the page. It was very interesting to see which architectural styles promote which quality attributes and which QA are hindered. In the beginning it might seem like one style is a good option (e.g. layered) but then discussing why security or scalability is so important for such a system, it might turn out that it is not the best fit. I also gained insight how tactics can be used to promote the QA that are usually hindered by a certain style. In addition, it was very insightful to know how the systems that we use every day (Västtrafik) are architecturally built. I learnt that there is never a perfect solution where all QA are promoted by an architectural style, however there are pros and cons to each one.

6/6