Engenharia de Serviços (MEI / MES)

2023/2024

Practical Project, Part 2

Applying service design techniques to model a real-world service

Deadline #1 (part 1, for feedback): 10 March 2024 (23h59m)
Deadline #2 (part 1+ part 2, for assessment): 19 May 2024 (23h59m)
Submission via Inforestudante

Note: Academic fraud is a serious ethical breach and is not admissible behavior for a student and future practitioner. Any attempt of fraud may lead to the cheater and their accomplices failing the course. Other sanctions may additionally apply.

Objectives

Apply the service design techniques to model a service, and, in doing so:

- Gain an understanding of the complexity of services and the need for the said techniques.
- Develop competences in using those techniques for diagnosing and evolving existing services and for designing new ones.
- Apply the cloud-based technologies you have learned, to create a simple, powerful, responsive, well-designed, and beautiful service.

Final Delivery

For part #1 (specification of the service) of this assignment, you must submit:

- Persona(s) - two distinct and well-defined personas are enough for the purpose of the assignment;
- Customer journey map(s);
- Stakeholder map(s);
- Expectation maps(as);
- Other elements that the groups deem relevant.
- By deadline #2 you must also re-submit an improved specification of the service, based on the feedback you received in part #1. You must **include a brief document stating the changes** that you made to your original specification to address the feedback.

The first three instruments are available in the Smaply software used in the course. Others require additional forms or tools. Further details are provided in class. A set of PDFs with the deliverables must be generated and submitted via Inforestudante by the deadline.

Pólo II · Pinhal de Marrocos · 3030-290 Coimbra · Portugal · Tel.: +351 239 290 000 · info@dei.uc.pt · http://www.uc.pt/fctuc/dei

For part #2 (revised specification + implementation) of this assignment, you must:

- Develop part of the assignment in the Amazon AWS cloud. This means that you should install and configure multiple services on this cloud.
- Keep your installation on Amazon until you present the assignment. Keep it untouched and remember that many services keep track of the dates they were updated.
- Make sure that you keep a comfortable margin from your budget to present the assignment in the final defense.
- You should bundle all the code and other elements that you do not keep online on Amazon and deliver them in Inforestudante before deadline #2. Please keep your file sizes as small as possible, by uploading only the source code. Do not upload compiled and public software libraries that you have running with the code. You do not need to deliver a report.

Overview

This project aims to develop and implement a modern health consultation service with a focus on physiotherapy and wellness.

When customers need a physiotherapy appointment, they visit the clinic's website to begin the appointment scheduling process. The user selects the type of specialty they require (e.g., evaluation appointment, musculoskeletal physiotherapy, therapeutic massage, shock waves, etc.) and may optionally specify the doctor's name. Available appointment dates are displayed, and the user chooses the most convenient date.

Upon completing the scheduling process, the user receives an SMS with a link to make the payment. After following the link, users select a payment method (MBWay or Multibanco) and provide invoice details (VAT number or identify as a final consumer without VAT). If MBWay is chosen, a valid phone number is required. If Multibanco is selected, payment reference numbers are generated and provided. Once payment is made, the invoice is sent to the user via email.

On the appointment consultation date, the user arrives at the clinic where face recognition technology is used at the entrance for identification. The user's name is displayed on the waiting room screen along with the appointment room number and the estimated waiting time. When the appointment time arrives, the user proceeds to the designated appointment room. The service finishes at the end of the appointment.

You may assume that the payment account has been previously topped up.

References

Researching facts and not making assumptions is part of the process of good service diagnosis and design. Feel free to investigate real services online. The instructors are also available to discuss your options.

Pólo II · Pinhal de Marrocos · 3030-290 Coimbra · Portugal · Tel.: +351 239 290 000 · info@dei.uc.pt · http://www.uc.pt/fctuc/dei

Important aspects (based on errors frequently made by students)

Regarding personas
The descriptions of personas must be rich and detailed. They must be credible as if we were describing real people. Only knowing people well enables you to design a service that suits them. Regarding the number of personas, it's not about being a lot or just a few, but how different and complete are the described profiles and needs. For instance, it does not contribute a lot to the service design if we have a lot of personas with basically the same needs; but we should not leave out important profiles.

Regarding customer journey maps
Being so rich, this is one of the most important tools in service design. It enables us to understand how the customer "travels" along our service. It's almost like a movie, where we have various scenes or snapshots in sequence. One of the most important aspects – see slides and book – is to make sure that we have the most adequate touchpoints (the moments of interaction). Journey maps are also very powerful in the sense that they enable us to relate what the customer sees and does with back-office actions and systems and the channels that are used for the interaction in touchpoints. If the customer receives a notification by SMS (channel), then there must have been a back-office system/person/process sending that message (back-office lane) — all these events and lanes must be consistent with each other. It is the proper synchronization of people, technology, and processes that ensures that the service flows smoothly. Pay close attention to how front-end systems and back-end systems interact across various channels. All must be consistent in the customer journey map. Remember that a channel is a "means for contact": email, phone, SMS, face-to-face encounter, land mail, etc. Product or money are not channels.

Regarding the number of maps, check the slides and book. It all depends on the level of abstraction and detail that you decide is adequate. You may have "happy path" scenarios, exception scenarios, different maps for different ways to use the service, etc. Please also remember that your maps must be understandable. Avoid too much clutter in one map (e.g., lots of personas).

It is frequent for people to forget touchpoints when modeling. Remember that confirmation emails/SMS are touchpoints, email/SMS warnings of the impending arrival of the order at your home are touchpoints, and the physical interaction with the delivery person is a touchpoint.

Regarding stakeholder maps
It is key to identify the different importance of the various involved stakeholders. Keep things clear, so that someone else can understand the exchanges between the various actors. The number of maps to create depends on the different scenarios of exchanges that you want to explain.

Regarding expectation maps
Expectation maps should be consistent with the profiles and needs of your personas. It does not make sense to have several different personas, with different motivations, and then just the same expectation map for all of them. Indeed, some expectations may be

Pólo II · Pinhal de Marrocos · 3030-290 Coimbra · Portugal · Tel.: +351 239 290 000 · info@dei.uc.pt · http://www.uc.pt/fctuc/dei

common, but others will be different. For instance, someone with a lot of money and little time has different expectations than someone short on cash. The expectations of a young active person are different from those of a senior or handicapped person.

Implementation

Students should implement the system as depicted in Figure 1, which uses multiple AWS services. Students should write the frontend using React (Svelte is also allowed).

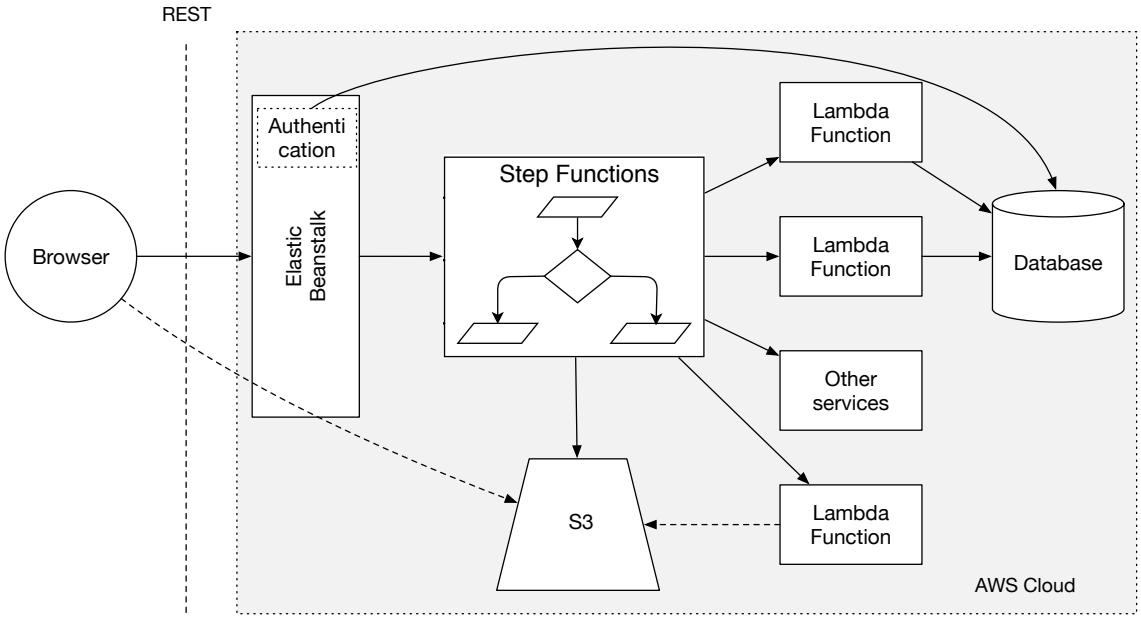

Figure 1 - Container Diagram of the Project

The frontend applications will access the backend to perform, at least, the following operations:

- Log in/Log out using a password.
- Set an appointment by selecting specialty, doctor, and schedule a date.
- Pay.
- Log into the clinic using facial recognition.
- Get indication of time, room, and other information relevant for the appointment.

The frontend applications interact with the backend application using a REST API supported by a Django project. Access to the backend goes through **Elastic Beanstalk**. Students may use standard **Django** (as opposed to the Django REST framework) in Elastic Beanstalk. This layer should be minimalistic and is essentially a gateway to other services. Importantly, this is not a Django assignment! Students should not write all the logic in the Django project. Also, students must not use the Django **authentication and authorization** libraries. They should implement their own solution using JSON Web Tokens. This enforcement serves tutorial purposes alone. In a real scenario, students should resort to standard libraries. The Django technological stack is not mandatory; students may use other technologies, but, in this case, they should discuss their options with the professor.

Pólo II · Pinhal de Marrocos · 3030-290 Coimbra · Portugal · Tel.: +351 239 290 000 · info@dei.uc.pt · http://www.uc.pt/fctuc/dei

Workflows written in **Step Functions** should be a central part of the backend implementation. Workflows have the great advantage of being easier to understand than Python code. This makes the interaction between engineers and managers simpler. The workflow may need to interact with different services, including **Lambda** functions and databases, like **Dynamo**.

Users start by logging in to the service using a standard username/password mechanism. Then, the site provides a number of dropdown lists where users select what they want. Once the user does this selection, a (step functions) workflow starts on the server with the purpose of ensuring that the user pays within a limited amount of time (to release the reservation otherwise) and to wait for an appointment or to discard the appointment and terminate the workflow after the set date. Students do not need to build a database of doctors and times available; they can create data on the fly and assume that all reservations occur on a single resource, e.g., divided in hours. For example, if a patient sets an appointment for June 5th at 16h, then that slot should not be available anymore. Hence, students should discuss and carefully think about the details of how to ensure that reservations are done and released properly. This application should also let users pay their appointments. While a button for that purpose is enough, the implementation of this operation should be idempotent. Finally, users should be able to see their appointments state (waiting for payment, scheduled, or finished).

Students should do a second frontend application where they list appointments and their state. A button to set the appointment as finished should be available. This second website does not require a login. Note that sending an SMS, an email, or some other notification is not necessary.

Utilization of Technologies

Boto3
Boto3[1] is a Python Software Development Kit (SDK) to operate AWS resources. The Django project and the Lambda functions will make extensive use of Boto3. Boto3 includes APIs to control a large spectrum of AWS services, e.g., databases, or Step Functions.

Storage
Students should minimize the use of the database that is configured by default in a Django project, SQLite (preferably not use it at all), and use RDS and other AWS services, like SimpleDB or DynamoDB. Usage and configuration of **more than a single type of**

[1] https://boto3.amazonaws.com/v1/documentation/api/latest/index.html.

Pólo II · Pinhal de Marrocos · 3030-290 Coimbra · Portugal · Tel.: +351 239 290 000 · info@dei.uc.pt · http://www.uc.pt/fctuc/dei

databases is encouraged, e.g., to keep authentication data for the log in and to keep information of the state of an appointment.

Storage of HTML and JavaScript
Even though a good solution to store HTML and JavaScript would be AWS S3, this option will raise several challenges related to Cross-origin resource sharing (CORS). Hence, students may prefer to serve these contents directly from their Django project.

Rekognition
According to AWS[2], "Rekognition offers Computer Vision capabilities, to extract information and insights from images and videos". To set Rekognition ready to identify persons in photographs, students need to perform a preliminary step of preparing a collection of faces and the creation of an external index to match the faces that Rekognition identifies with external real users. To improve results students may use more than one picture per person.

Additional Challenge

A challenge for students is to define their interface using Open API/Swagger. In addition to providing a clear definition of the REST interface, this technology enables the creation of fancy testing interfaces, server skeletons, and other valuable features.

Evaluation

Regarding the split of work
Students are required to showcase their acquired skills in both the design and implementation components of the project. It is essential to ensure that all students engage in both aspects of the project, rather than dividing tasks in a way that some students are solely responsible for design while others handle implementation. This approach will considerably penalize the entire group. This aspect is assessed during project development as well as in the final defense of the project.

[2] https://aws.amazon.com/rekognition/?nc1=h_ls.

Pólo II · Pinhal de Marrocos · 3030-290 Coimbra · Portugal · Tel.: +351 239 290 000 · info@dei.uc.pt · http://www.uc.pt/fctuc/dei