THE AMERICAN UNIVERSITY IN CAIRO

الجامعة الأمريكية بالقاهرة

Software Engineering

Initial Milestone

Group 3

Submitted To: Dr. Sherif Aly

Submitted By:

Abdalla Elragal - abdalla.elragal@aucegypt.edu - 900171502

Tayam badawy - tayambadawy@aucegypt.edu - 900183911

Ramy Badras - ramybadras@aucegypt.edu - 900194248

Salma Laban - salmalaban@aucegypt.edu - 900191369

Sherif Wessa - sherifwessa@aucegypt.edu - 900203171

Ali Salem - alisalem@aucegypt.edu - 900183226

Table of Contents

Scrum Methodology

What is the Scrum methodology?

Scrum is an incremental Agile software development methodology that is adopted by companies that want fast delivery and development of software products. The Scrum methodology is faster than the plan-based methods (Castillo-Barrera et al., 2018). Each iteration or increment is called a sprint, and it usually lasts for 2 to 4 weeks. The aim of each sprint is to build the most crucial features first and produce a workable version of the product if possible (Peek, 2022). The functional requirements of the software product are written as user stories that are written from the users' perspective. The complexity of each user story should be estimated beforehand to ensure that an adequate workload is assigned to each sprint (Castillo-Barrera et al., 2018). Any time before starting a sprint or after completing a sprint, the stakeholders can edit or add new features to the requirements, which makes this method flexible to handle the frequent changes in customers' needs (Peek, 2022).

How is the Scrum methodology conducted?

Scrum workflow involves three main components: processes, roles and artifacts. First of all, Scrum Roles involve continuous collaboration of three roles: a scrum master (SM), product owner (PO), and the scrum team. Second, Scrum Processes include the kickoff, the sprint, the sprint planning meeting, the sprint review meeting, and the daily Scrum. And lastly, the Scrum Artifacts consist of product and sprints backlog, and graphics (Carneiro, 2018).

For the Scrum roles, the Scrum master (SM), or a system analyst, is mainly responsible for introducing the Scrum method to the whole team, assuring that the scrum practices and approaches are being followed in a valid way, as well as eliminating any impediments or obstacles that pop up during the execution process. The product owner, which includes a project manager and a business analyst, is responsible for determining the requirements and the specification of the software project they are working on. In addition, PO creates the product backlog which is all the initial requirements for the project, as well as the sequence of deliverables to be made by the team. In other words, the product owner is the one who gives the project team a clear vision of what they are going to develop. Finally, the Scrum team, consisting of programmers and testers, work on each item in the product backlog that has been gathered by the product owner (Permana, 2015) (Adi, 2015).

Scrum processes require a kickoff meeting, where a basic, non-detail product backlog is created, as well as the main objectives of the project. Afterwards, a sprint planning meeting is conducted that includes the team, the PO and the SM, where they start creating a detailed backlog. Another Scrum process is the Daily Scrum which are check-in meetings that are conducted daily for no longer than 15 minutes to check what has already been done and what needs to be done next. Lastly, A sprint review meeting is done at the end of each sprint to discuss functionality and inspect what has been developed so far (Permana, 2015).

In the Scrum artifacts phase, the product owner ranks the project requirements in the product backlog according to their priority and maintains it. Also, the team meets for sprint planning,

where they specify the effort needed to complete each requirement in the product backlog (Carneiro, 2018).

Advantages and disadvantages

One of the main features in agile scrum development is the frequent meetings between the different teams working on the project or else known as “daily scrum”, which from what the name implies, those meetings could be held on a daily basis. The idea of having frequent meetings comes with its own set of advantages and disadvantages. Starting with the advantages, having a daily meeting where the different teams get to share updates on progress among themselves and voice their concerns frequently can help enhance everyone’s vision on the end product and everyone is more involved in the process, making them have an overview of the project building process and not just specifically their part only. Those meetings can also help make sure that everyone is on track and understand the requirements clearly as they are constantly reminded of them. Finally, through sharing updates on progress frequently, the teams should be motivated as progress is visible and clear. However, those frequent meetings may become highly time consuming in some cases, such as unresolved conflict between different teams’ work. There is also the possibility that some team members may not be willing to attend those meetings as they have nothing to share and believe they do not need to listen to other teams share their updates on their progress. The success of those meetings highly relies on the different teams’ ability to communicate with one another, therefore high morale must be kept among everyone involved in the process. In the case of a sudden change in the team members, the new member is required to be caught up to date with the process in order to be able to continue the work. If implemented correctly the agile scrum methodology can prove to be useful for the client

to have a clear vision on the progress and the efficiency of the team, however, can prove costly if implemented poorly from the stated reasons above.

Possible Client

- Food vendors
- Auc students
- Auc faculty
- Auc staff

Overview of the System

The major goal of this application is to cut down on the amount of time that faculty, staff, and students spend waiting for their orders and walking to the campus food stores. As individuals might arrange or place orders in advance, this would make it simpler for the retailers to be more efficient and give themselves more time to deliver the products or pick it up from the stations located all around campus. This would encourage faculty and employees to be more effective and productive during their free time.

Customers

- Choose their preferred food outlet from a list of restaurants and cafeterias in the university, access their menus, and choose their desired order.
- Add their orders to the shopping cart.
- Pay for orders with their AUC pay accounts or credit cards.
- Choose how they want to pick up their orders, either through passing by the restaurant or the nearest pick up point.
- Track their orders.

- Give a rating and review their orders.
- Access their order history.

Food vendors

- View and receive orders.
- Confirm orders.
- Update menus (add/delete items, edit prices).
- View transactions.

Product Backlog

Functional Requirements

1. **Sign up**

   The user must sign up with the AUC email to ensure that the user belongs to the AUC community (staff, faculty, students) and a password containing 8 characters with 1 uppercase letter and 1 special character.

2. **Logging in**

   The user uses the email and password that was used when signing up.

3. **Verify password**

4. **Navigate menu**

   A menu used to make the application more user friendly that the user can use to maneuver easily through the application.

5. **Select item (Add to Cart)**

   The user must select 1 item at a time and add it to the cart.

6. **Customize item**

   After selecting an item the user has the ability to customize what they ordered (if applicable) for example a sandwich without olives.

7. **Remove item (Remove from Cart)**

   An option if the user wants to remove an item from the cart the user must select 1 item at a time or delete the whole cart.

8. **Selecting location**

   After the user has confirmed his order, he should choose whether to pick up the order from the food vendor or the designated pick-up stations at each building. If the user chooses the pick-up stations, he has to also select the building.

9. **Pay for order (Credit or Debit Card / Auc Pay)**

   The next step is for the user to pay for the order. The user can do so either by paying by credit or debit card or Auc pay. The user must input the credit/debit card number, name, expiration date and CVV or the user can scan the card and he won't need to fill in this information. However, if the user chooses to pay through Auc pay, he will be directed to AUC's website to pay.

10. **Payment Confirmation**

    After the amount has been paid a message appears on the screen: payment of: X amount has been successfully made.

11. **Order confirmation**

    An email will be sent to the user's AUC email containing the invoice of the order made which includes the order number, the amount paid and the items that have been ordered.

12. **Cancel order**

    The user can cancel an order if the order has not yet been made by the food vendor.

13. **Track the order**

    The user has the option to track the order and know whether the order is still in the food vendor or is on its way.

14. **Pick Up Confirmation**

    After the food has been picked up an email will be sent to the vendor and the customer confirming that the order has been picked up by the customer.

15. **Review Order**

    The user has the option to review the order by giving it a rating on a scale from 1 to 5

16. **Support**.

    If the user faces any problems the user can chat with an employee to help him through the application.

17. **Orders history**

    The user can view the orders history up to 10 orders. The order history contains which food vendor the user chose, the items, and the amount paid.

18. **Edit menu**

    The vendors would be able to add new items or remove old ones as well as update the items according to inventory (stock).

The system's objective is to make the customer's ordering procedure as simple as possible, therefore the capability offered through the ordering system is limited to that which is necessary to carry out the desired task.

Nonfunctional Requirements

1. **Reliability**

- The application should handle any number of users in order not to crash.
- The application should notify the vendors once an order has been placed by a customer

2. **Usability**

- UI/UX of the system should be simple and user-friendly to ensure the easiness of the purchasing process.
- The design of the user interface would be consistent with the AUC theme.

3. **Responsiveness / Speed**

- The system should always respond quickly to users' needs, and it should also be flexible to allow any modifications.

4. **Maintainability**

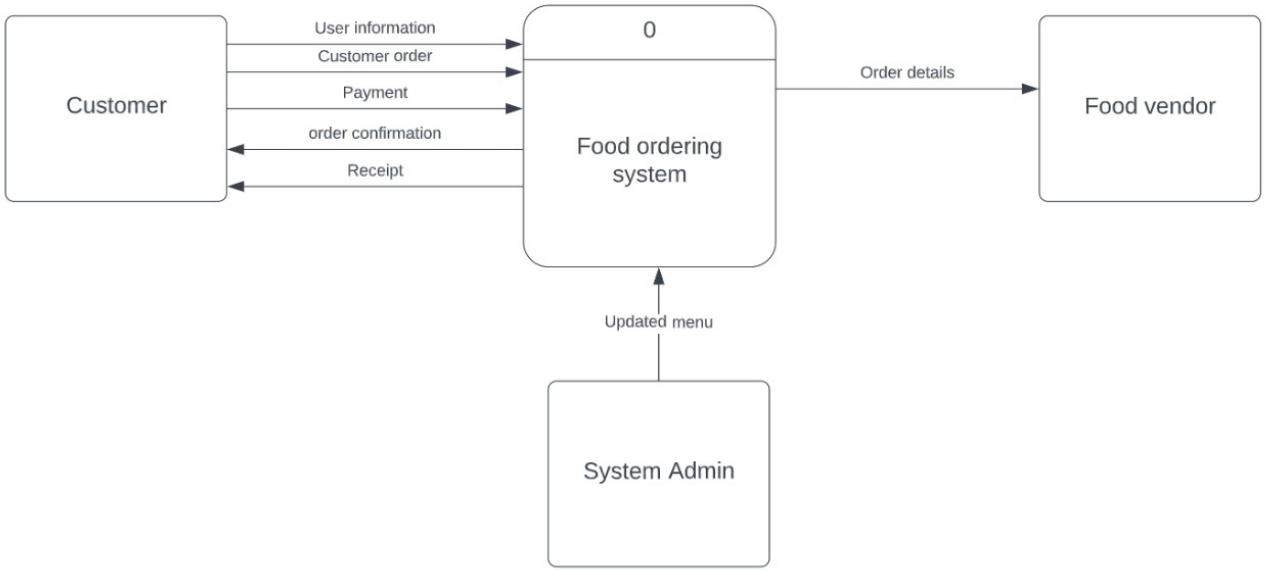
- The system should be maintainable to ensure its effectiveness through continuous updates, restoration, and backup in case of any failures.

5. **Security**

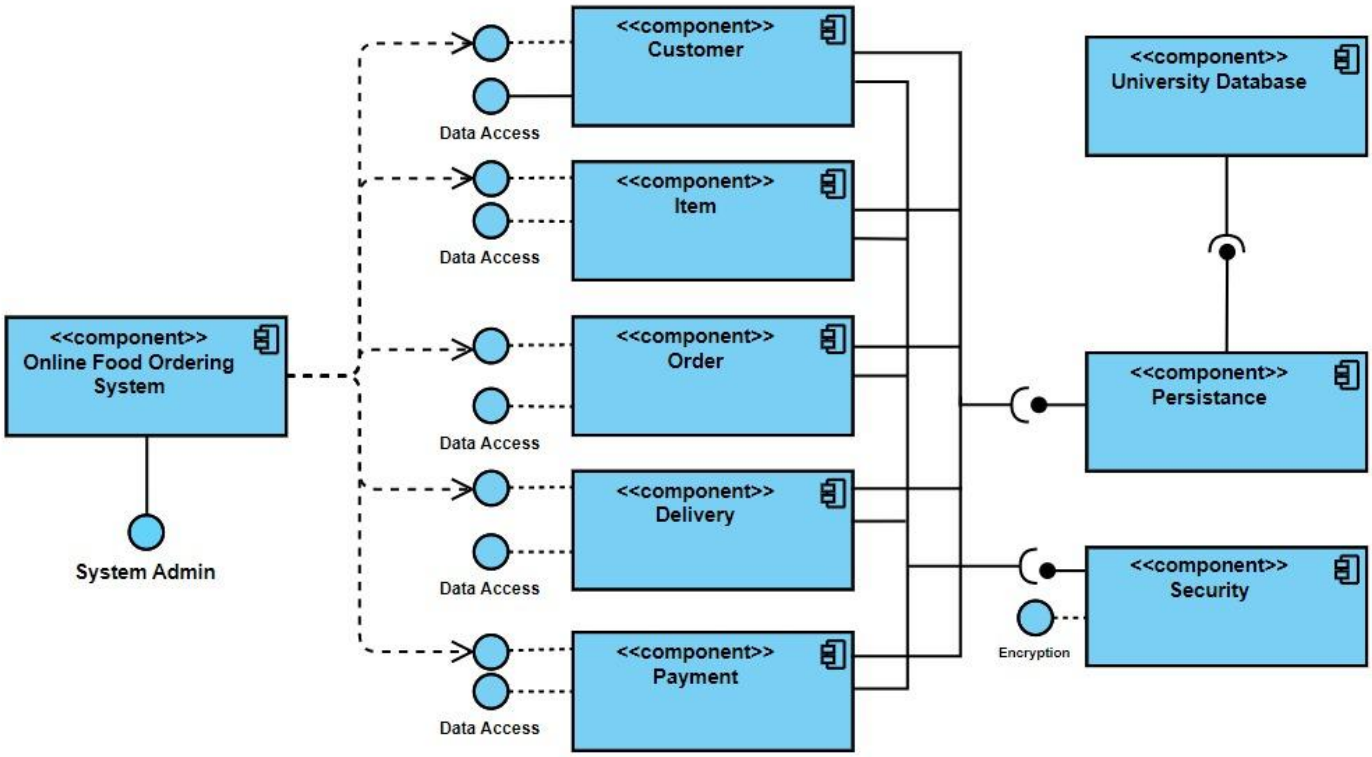
- Database firewalls can be used to protect against security attacks
- Customers' information and payment data should be encrypted to maintain a securable system

High-level Architecture of the System

Context Diagram



UML Component Diagram



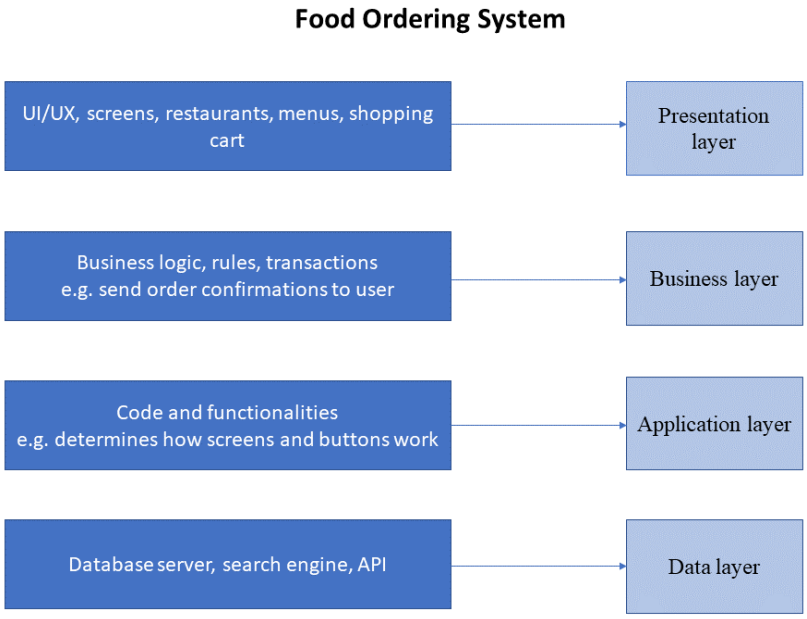
Software Architecture Model

For system architecture, we considered two main perspectives: logical architecture and physical architecture.

Logical Architecture

In order to represent the overall structure of our software architecture, organize its components in the most efficient way, and show how each component interacts with the other components of the program, we decided on two software architecture patterns that would best fit the purpose of our program: Layered pattern and Microservices pattern.

In the layered architecture pattern, program components are organized in horizontal layers (one above the other). Layers are independent from each other, yet they are interconnected. The pattern consists of four main layers: Presentation layer, Business layer, Application layer, and Data layer. Layered pattern is ideal for our program because we can simply edit and modify any layer in the architecture without affecting other layers as they are independent from each other. Also, objects are easily modified; since all objects are located in the same place, we can easily find the object we want to change. In addition, its scale matches the scale of our program and it is easy to implement (Richards, 2015).

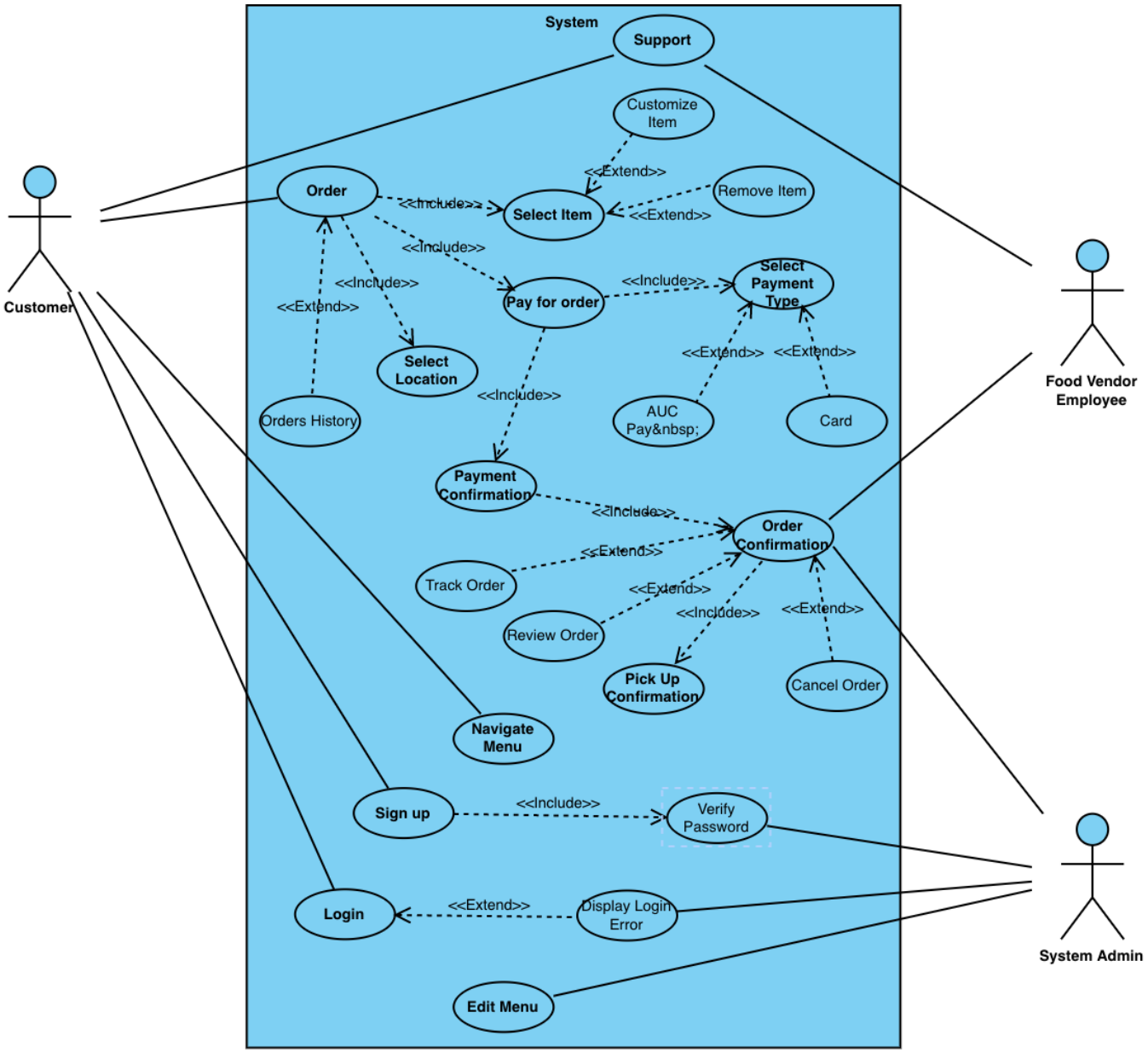


Physical Architecture

The physical architecture is the arrangement of physical objects and interfaces to satisfy the logical requirements. Therefore, our system will require some physical elements such as:

- Servers: to store system database and users' data
- Hardwares: For using the application as well as entering and updating data
- Internet/wifi connection and routers: for communication between user and system.
- Navigation system: to identify the location of the user / food station to deliver food.

Set of Use Cases



Identified Risks

- A surge in the number of users during assembly hour might overwhelm the servers.
- Vendors may not be willing to spare the extra resources to adopt the new system.
- Vendors may lack time of delivery because of their location.
- Lack of application awareness
- Some of the customers and vendors may not be able to use the system as it's a new method for them.

Conclusion

All in all, we believe that our application would be very beneficial. The main goal of this software application is to decrease the amount of time that faculty, staff, and students spend waiting for their orders and walking to the campus' food vendors. Both students and faculty will benefit from being able to spend their free time more efficiently and productively once they begin ordering meals through the app and receiving their food and drinks whenever and wherever they want. We also believe that the stated functional requirements are easy to implement. By using the Scrum methodology, we will be able to build the software quickly and smoothly, and we will also be able to update the requirements at any time if needed.

References

Adi, P. (2015). Scrum method implementation in a software development project management. *International Journal of Advanced Computer Science and Applications*, *6*(9). https://doi.org/10.14569/ijacsa.2015.060927

Castillo-Barrera, F. E., Amador-Garcia, M., Perez-Gonzalez, H. G., Martinez-Perez, F. E., & Torres-Reyes, F. J. (2018). Adapting bloom's taxonomy for an agile classification of the complexity of the user stories in Scrum. *2018 6th International Conference in Software Engineering Research and Innovation (CONISOFT)*. https://doi.org/10.1109/conisoft.2018.8645899

Cho, Juyun Joey, "*An Exploratory Study on Issues and Challenges of Agile Software Development with Scrum*" (2010). All Graduate Theses and Dissertations. 599. https://digitalcommons.usu.edu/etd/599

L. B. Carneiro, A. C. C. L. M. Silva and L. H. Alencar, "Scrum Agile Project Management Methodology Application for Workflow Management: A Case Study," 2018 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), 2018, pp. 938-942, doi: 10.1109/IEEM.2018.8607356. https://ieeexplore.ieee.org/document/8607356

Peek, S. (2022, June 29). *What is agile scrum methodology?* Business News Daily. Retrieved October 8, 2022, from https://www.businessnewsdaily.com/4987-what-is-agile-scrum-methodology.html

Permana, P. A. G. (2015). Scrum method implementation in a software development project management. International Journal of Advanced Computer Science and Applications, 6(9), 198-204. https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.695.8055&rep=rep1&type=pdf

Richards, M. (2015). Software architecture patterns (Vol. 4, p. 1005). 1005 Gravenstein Highway North, Sebastopol, CA 95472: O'Reilly Media, Incorporated. http://jasonmaggart.net/wp-content/uploads/2022/08/Software-Architecture-Patterns.pdf