



CELAL BAYAR UNIVERSITY ENGINEERING FACULTY

COMPUTER ENGINEERING

SUMMER PRACTICE

Internship Report

Information of the Intern

Name and Surname : BERKE ALPASLAN
Student Number : 210316016
Company Title : KOD YAZILIM PROJE HİZ.
TUR. TİC. A.Ş.
Internship Period : 01/07/2024 – 29/07/2024



Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
<div><h2>TABLE OF CONTENTS</h2><div><div><div>CHAPTER 1 INTRODUCTION</div><div>1.1 Personal and Academic Background</div><div>1.2 Internship Overview</div><div>1.3 Purpose of the Internship</div><div>1.4 Project Description</div><div>1.5 Objectives and Scope of the Report</div></div><div>1</div></div><div><div>CHAPTER 2 INFORMATION ABOUT THE WORKSPACE</div><div>2.1 Business Activities and Organizational Structure</div><div>2.2 Usage of Computers within the Organizations</div><div>2.3 Pre-Software Analysis and Design Processes</div><div>2.4 Familiarization and Implementation of Software</div><div>2.5 Sample Project Implementation</div><div>2.5.1 Project Overview</div><div>2.5.2 Development Tools</div></div><div>3</div></div>	

Internship Authority:

Signature:

EK 11- Günlük Rapor Sayfası

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
<div><div>2.5.3 Architectural Approach for Backend6</div><div>2.5.3.1 Onion Architecture6</div><div>2.5.3.1.1 Domain Layer7</div><div>2.5.3.1.2 Application Layer7</div><div>2.5.3.1.3 Persistence Layer9</div><div>2.5.3.1.4 Infrastructure Layer9</div><div>2.5.3.1.5 Presentation Layer10</div><div>2.5.3.2 CQRS (Command Query Response Segregation)11</div><div>2.5.3.3 Mediator Pattern12</div><div>2.5.3.4 Marker Pattern13</div><div>2.5.3.5 Generic Repository Design Pattern14</div><div>2.6 Investigation of Computer Networks and Their Hardware and Structural Features15</div><div>2.7 Developed Software Codes, Sample Data, and Results16</div><div>2.7.1 Backend Source Code Examples16</div><div>2.7.1.1 Domain Layer Codes16</div><div>2.7.1.2 Application Layer Codes17</div><div>2.7.1.3 Infrastructure Layer Codes18</div><div>2.7.1.4 Persistence Layer Codes19</div><div>2.7.1.5 Presentation Layer Codes20</div><div>2.7.1.5 Database Sample Data and Tables21</div><div>2.7.2 Frontend Source Code Examples23</div></div>	
Internship Authority:	Signature:

EK 11- Günlük Rapor Sayfası

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
<div>CHAPTER 3 INTERNSHIP DIARY JOURNAL32</div> <div>3.1 Day 1 (01/07/2024)32</div> <div>3.2 Day 2 (02/07/2024)32</div> <div>3.3 Day 3 (03/07/2024)33</div> <div>3.4 Day 4 (04/07/2024)34</div> <div>3.5 Day 5 (05/07/2024)35</div> <div>3.6 Day 6 (08/07/2024)36</div> <div>3.7 Day 7 (09/07/2024)37</div> <div>3.8 Day 8 (10/07/2024)37</div> <div>3.9 Day 9 (11/07/2024)38</div> <div>3.10 Day 10 (12/07/2024)39</div>	
Internship Authority:	Signature:


EK 11- Günlük Rapor Sayfası

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
<div>3.11 Day 11 (16/07/2024) 39</div> <div>3.12 Day 12 (17/07/2024) 40</div> <div>3.13 Day 13 (18/07/2024) 40</div> <div>3.14 Day 14 (19/07/2024) 41</div> <div>3.15 Day 15 (22/07/2024) 41</div> <div>3.16 Day 16 (23/07/2024) 43</div> <div>3.17 Day 17 (24/07/2024) 43</div> <div>3.18 Day 18 (25/07/2024) 44</div> <div>3.19 Day 19 (26/07/2024) 44</div> <div>3.20 Day 20 (29/07/2024) 44</div> <div>CONCLUSION 45</div> <div>REFERENCES 46</div>	
Internship Authority:	Signature:


Name & Surname: Berke Alpaslan																															
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.																															
Start Date: 01/07/2024	End Date: 29/07/2024																														
<div><h2>TABLE OF FIGURES</h2><table><tr><td>Figure 1 .NET Core Logo</td><td>2</td></tr><tr><td>Figure 2 Kod Yazılım Logo</td><td>4</td></tr><tr><td>Figure 3 Onion Architecture Schema</td><td>8</td></tr><tr><td>Figure 4 Onion Architecture on Internship Project</td><td>10</td></tr><tr><td>Figure 5 The layers referenced by the presentation layer in the project's Onion architecture</td><td>10</td></tr><tr><td>Figure 6 CQRS Pattern Schema</td><td>11</td></tr><tr><td>Figure 7 Mediator Pattern Schema</td><td>12</td></tr><tr><td>Figure 8 Mediator Pattern for GetAllCustomer in the Project</td><td>12</td></tr><tr><td>Figure 9 IReadRepository in the Project</td><td>13</td></tr><tr><td>Figure 10 Reservation Entity Class in the Project</td><td>16</td></tr><tr><td>Figure 11 ITokenHandler Interface in the Project</td><td>17</td></tr><tr><td>Figure 12 IUserService Interface in the Project</td><td>17</td></tr><tr><td>Figure 13 Handler Class of Mediator Pattern for CreateCustomer Operation in the Project</td><td>17</td></tr><tr><td>Figure 14 TokenHandler Class for Create AccessToken in the Project</td><td>18</td></tr><tr><td>Figure 15 ServiceRegistration Class of Persistence Layer Services in the Project</td><td>19</td></tr></table></div>		Figure 1 .NET Core Logo	2	Figure 2 Kod Yazılım Logo	4	Figure 3 Onion Architecture Schema	8	Figure 4 Onion Architecture on Internship Project	10	Figure 5 The layers referenced by the presentation layer in the project's Onion architecture	10	Figure 6 CQRS Pattern Schema	11	Figure 7 Mediator Pattern Schema	12	Figure 8 Mediator Pattern for GetAllCustomer in the Project	12	Figure 9 IReadRepository in the Project	13	Figure 10 Reservation Entity Class in the Project	16	Figure 11 ITokenHandler Interface in the Project	17	Figure 12 IUserService Interface in the Project	17	Figure 13 Handler Class of Mediator Pattern for CreateCustomer Operation in the Project	17	Figure 14 TokenHandler Class for Create AccessToken in the Project	18	Figure 15 ServiceRegistration Class of Persistence Layer Services in the Project	19
Figure 1 .NET Core Logo	2																														
Figure 2 Kod Yazılım Logo	4																														
Figure 3 Onion Architecture Schema	8																														
Figure 4 Onion Architecture on Internship Project	10																														
Figure 5 The layers referenced by the presentation layer in the project's Onion architecture	10																														
Figure 6 CQRS Pattern Schema	11																														
Figure 7 Mediator Pattern Schema	12																														
Figure 8 Mediator Pattern for GetAllCustomer in the Project	12																														
Figure 9 IReadRepository in the Project	13																														
Figure 10 Reservation Entity Class in the Project	16																														
Figure 11 ITokenHandler Interface in the Project	17																														
Figure 12 IUserService Interface in the Project	17																														
Figure 13 Handler Class of Mediator Pattern for CreateCustomer Operation in the Project	17																														
Figure 14 TokenHandler Class for Create AccessToken in the Project	18																														
Figure 15 ServiceRegistration Class of Persistence Layer Services in the Project	19																														
Internship Authority:	Signature:																														

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
<p>Figure 31 When You Click on the Hotel You Want on the Home Page, the Page that Appears 29</p> <p>Figure 32 Payment Page After Choosing How Many Days You Booked For in the Project 29</p> <p>Figure 33 Comment Section for a Hotel in the Project 30</p> <p>Figure 34 Notification alert after posting a comment and the listed comments in the Project 30</p> <p>Figure 35 The Starting Section of the Page Where Hotels are Listed in the Project 31</p> <p>Figure 36 The Display of a Listed Hotel on the Hotel Listing Page and Information About its Star Rating 31</p> <p>Figure 37 Sample PostgreSQL Table Creation Code Provided by Our Internship Supervisor for the Project 33</p> <p>Figure 38 Docker Container of our Project's PostgreSQL Database 34</p> <p>Figure 39 The Postman App to API Test 34</p> <p>Figure 40 An excerpt from the Migration class automatically generated using EntityFramework for my entities 35</p> <p>Figure 41 Interceptor for BaseEntity Class 36</p> <p>Figure 42 CreateAccessToken Class for AccessToken Creation After Sign In 37</p> <p>Figure 43 The code that allows the Client URL of our project to connect with the API under CORS policies 38</p> <p>Figure 44 Validation for CreateUser Operation in the Project 38</p> <p>Figure 45 HttpClientService of My Frontend Learning Project 42</p>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
<div><h2>CHAPTER 1</h2><h3>INTRODUCTION</h3><h4>1.1 Personal and Academic Background</h4><p>My name is Berke Alpaslan, and I am currently a second-year student in the Computer Engineering program at Manisa Celal Bayar University. As part of my academic curriculum, I am required to complete an internship to gain hands-on experience in the field of software engineering.</p><h4>1.2 Internship Overview</h4><p>This internship was conducted at “Kod Yazılım” specifically in the “Software Development” department, from 01/07/2024 to 29/07/2024. The organization is renowned for its innovative solutions in the hospitality sector, and it provided an excellent environment to apply and enhance my technical skills.</p><h4>1.3 Purpose of the Internship</h4><p>The main purpose of this internship was to bridge the gap between theoretical knowledge and practical application, particularly in designing and developing software systems. This aligns with my career goal to become a proficient software engineer, specializing in system architectures and web development.</p></div>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
<p>1.4 Project Description</p> <p>During my internship, I was tasked with developing a hotel reservation system using .NET Core, ASP.NET, Angular, and PostgreSQL. My primary focus was on the backend architecture, employing the Onion Architecture model to ensure scalability and maintainability. This project not only allowed me to deepen my understanding of these technologies but also helped me to gain practical experience in applying complex architectural patterns like CQRS and Mediator.</p> <p>1.5 Objectives and Scope of the Report</p> <p>This report aims to detail my activities throughout the internship, describe the projects I worked on, and discuss the challenges I faced and the solutions I implemented. It will also reflect on the skills I developed and the knowledge I gained, which are vital for my future career in software development.</p> <div></div> <p>Figure 1 .NET Core Logo</p>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan			
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.			
Start Date: 01/07/2024	End Date: 29/07/2024		
<div><h2>CHAPTER 2</h2><h3>INFORMATION ABOUT THE WORKSPACE</h3></div> <div><h4>2.1 Business Activities and Organizational Structure</h4><p>“Kod Yazılım” was established with the goal of developing software that forms the infrastructure for a digital business approach, tailored to the needs of accommodation facilities and tourism agencies in the tourism sector. Since 2003, the company has been providing services in the field of sectoral project and software production. Leveraging its expertise and a customer satisfaction-oriented work principle, “Kod Yazılım” offers its accumulated knowledge to the tourism industry through hotel management, agency, and accounting programs.</p><p>The Computer Center, where I was placed, is central within the company’s structural organization, ensuring the sustainability and security of the technological infrastructure.</p></div> <div><h4>2.2 Usage of Computers within the Organizations</h4><p>Computers at “Kod Yazılım” are utilized across various departments, including software development, system management, customer support services, and administrative operations. Furthermore, the training and support department, which educates customers on how to use the produced software, plays a role just as critical as the software development department itself.</p></div> <table><tr><td>Internship Authority:</td><td>Signature:</td></tr></table>		Internship Authority:	Signature:
Internship Authority:	Signature:		

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
<p>2.3 Pre-Software Analysis and Design Processes</p> <p>Before the commencement of software projects, the Project Management Department conducts the analysis and design processes. These include needs analysis, system design, and user experience design. During my internship, I actively participated in these processes, gaining firsthand experience in the initial stages of project planning and execution.</p> <p>2.4 Familiarization and Implementation of Software</p> <p>During my internship, I developed a hotel reservation system utilizing .NET Core, ASP.NET, Angular, and PostgreSQL technologies. Throughout this process, I became intimately familiar with these platforms and tailored the software to meet real-world needs. My work primarily focused on backend aspects such as database design and API development, but it also encompassed integration of the user interface. Additionally, I had the opportunity to establish the infrastructure necessary for a scalable and secure system architecture that integrates seamlessly with these technologies. The practical application of each technology significantly enhanced my technical skills and deepened my understanding of software development processes.</p> <div></div> <p>Figure 2 Kod Yazılım Logo</p>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
 2.5 Sample Project Implementation <p>My main project involved developing a backend system for a hotel reservation platform using Onion Architecture to ensure scalability and maintainability. This project was aligned with the organization's time frames and allowed me to apply complex architectural patterns like CQRS and Mediator.</p> 2.5.1 Project Overview <p>The main objective was to develop a hotel reservation platform to manage user bookings, monitor room availability, and process payments securely and efficiently. I played a role in developing the backend system for this project.</p> <p>The project utilized a combination of .NET Core and ASP.NET for the backend development, Angular for the frontend interface, and PostgreSQL for the database management. This technology stack was chosen for its robustness, scalability, and widespread industry support.</p> 2.5.2 Development Tools <p>The main objective was to develop a hotel reservation platform to manage user bookings, monitor room availability, and process payments securely and efficiently. I played a role in developing the backend system for this project.</p> <p>The project utilized a combination of .NET Core and ASP.NET for the backend development, Angular for the frontend interface, and PostgreSQL for the database management. This technology stack was chosen for its robustness, scalability, and widespread industry support.</p>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
 2.5.3 Architectural Approach for Backend <p>While working on the backend part of our project, we planned not to use separate APIs for fetching data from different databases or for operations, but instead, we used a single database and handled all operations, including database CRUD (Create, Read, Update, Delete), through a single API due to our basic level. Given that we used a monolithic structure, instead of architectures created for multi-API communications like Microservices, we employed other software architectures in the backend that still reduced dependencies among components. These architectures are as follows:</p> 2.5.3.1 Onion Architecture <p>Onion Architecture is a layered architectural pattern that aims to create a maintainable and testable software application by dividing it into different layers, each with a specific responsibility and level of abstraction. This architecture is designed with various layers expanding from the inside out to facilitate the maintenance and development of software. Essentially, the core functionality of the application is located at the innermost layer, and each layer is designed to be independent of the layers outside of it. In Onion Architecture, the fundamental principle is that each layer depends on the more central layers. The layers from the inside out are the Domain, Application, Infrastructure, Persistence, and Presentation layers.</p> <p>Onion Architecture promotes Loosely Coupling by ensuring that application layers only depend on the inner layers, eliminating Tightly Coupling. It supports the Separation of Concerns principle by organizing the project into responsibility-based layers, making maintenance easier. Additionally, it enhances testability, as unit tests can be independently developed for each layer. Unlike traditional models, Onion Architecture places the data layer (Persistence Layer) as the outermost layer, facilitating development regardless of data origin.</p>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
<p>2.5.3.1.1 Domain Layer</p> <p>It is the central layer of the architecture. The Domain and database entities for the entire application are created in this layer.</p> <ul style="list-style-type: none">• Entities• Value Object• Enumeration• Exceptions <p>2.5.3.1.2 Application Layer</p> <p>This layer serves as an abstraction layer between the Domain layer and the application's business/service layer. All interfaces, whether repositories or services, are defined here. The goal is to demonstrate a loosely coupled approach to data access. It references the Domain layer. All objects that are relevant to the entire application, along with the necessary interfaces, are defined in this layer.</p> <ul style="list-style-type: none">• Custom Exception• Response Object• Request Parameters Object• DTO Objects• ViewModels Objects• Interfaces• Mapping• Validators	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan

Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.

Start Date: 01/07/2024

End Date: 29/07/2024

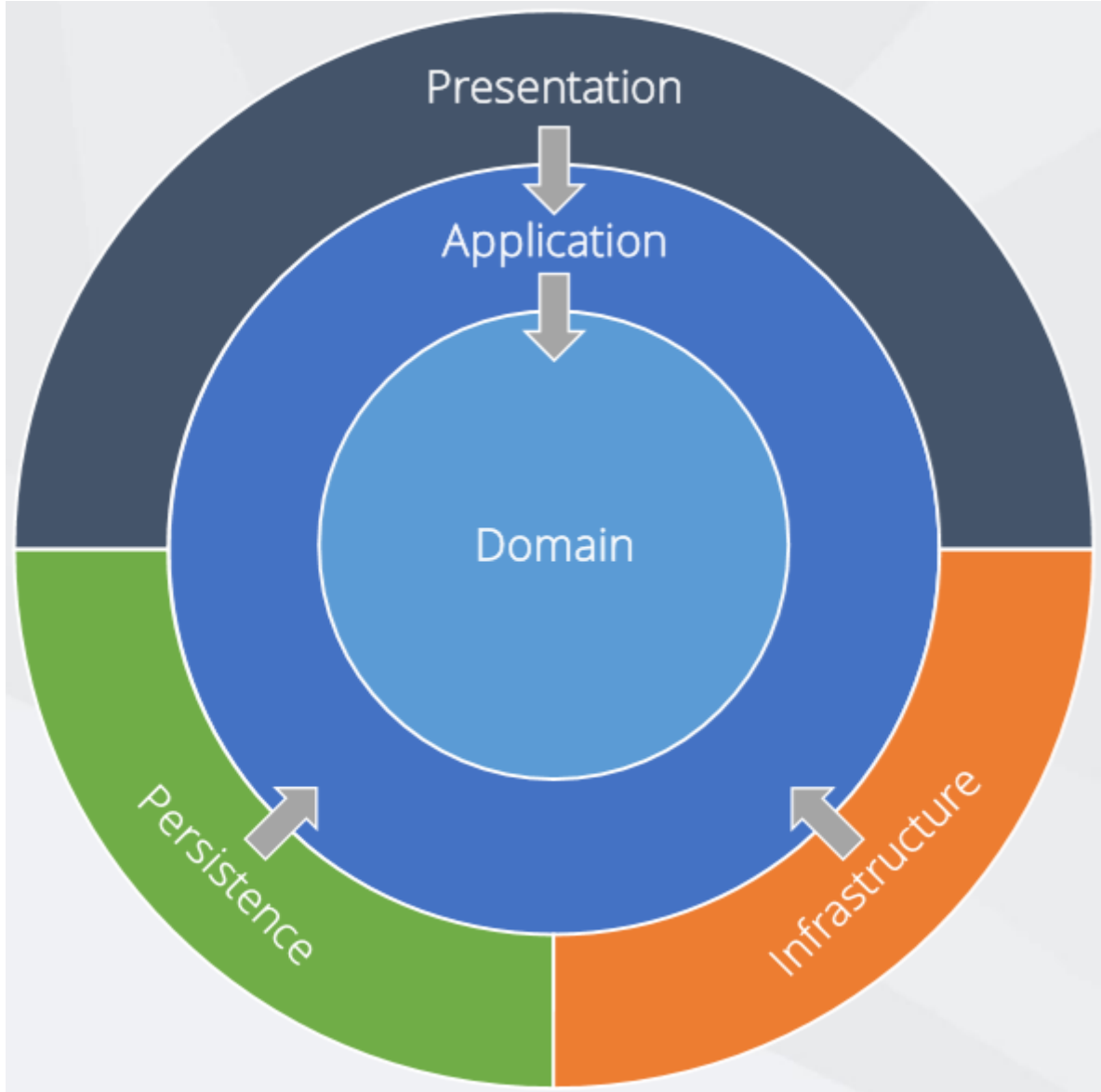


Figure 3 Onion Architecture Schema

Internship Authority:

Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
<p>2.5.3.1.3 Persistence Layer</p> <p>DbContext, migration, and database configuration tasks are performed in this layer. Additionally, interfaces from the Application layer are implemented here. As it is the outermost layer, no other layer will have dependencies on this layer.</p> <ul style="list-style-type: none">• DbContext• Migrations• Configurations• Interface Implementation <p>2.5.3.1.4 Infrastructure Layer</p> <p>In fact, the Persistence layer is typically integrated with this layer. External structures to be added to the system are usually incorporated in this layer. Consequently, as this is also the outermost layer, it should not have dependencies from any other layer.</p> <ul style="list-style-type: none">• Email/SMS• Notification• Payment	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 0107/2024	End Date: 29/07/2024

2.5.3.1.5 Presentation Layer

It is the layer where the user interacts with the application.

- Console App
- Web App
- MVC
- Web API

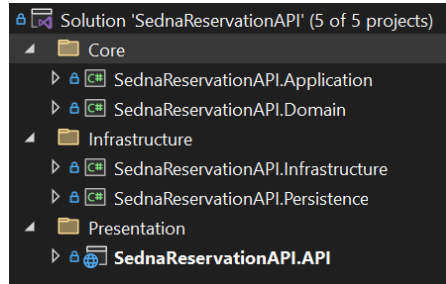


Figure 4 Onion Architecture on Internship Project

	Name	Path
<input checked="" type="checkbox"/>	SednaReservationAPI.Application	C:\Users\berke\Downl...
<input checked="" type="checkbox"/>	SednaReservationAPI.Domain	C:\Users\berke\Downl...
<input checked="" type="checkbox"/>	SednaReservationAPI.Infrastructure	C:\Users\berke\Downl...
<input checked="" type="checkbox"/>	SednaReservationAPI.Persistence	C:\Users\berke\Downl...

Figure 5 The layers referenced by the presentation layer in the project's Onion architecture

This image shows the menu in an IDE where the layers that the Presentation layer should reference according to the Onion architecture are selected. Normally, in Onion architecture, the outer layers depend on the inner layers. However, there is an exception for the Presentation layer, which may also reference the Infrastructure and Persistence layers located in the same outer layer when necessary. In my project, I utilized this exception to manage dependencies effectively.

Internship Authority:	Signature:
-----------------------	------------

Name & Surname: Berke Alpaslan

Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.

Start Date: 01/07/2024

End Date: 29/07/2024

2.5.3.2 CQRS (Command Query Response Segregation)

The command query responsibility segregation (CQRS) pattern separates the data mutation, or the command part of a system, from the query part. You can use the CQRS pattern to separate updates and queries if they have different requirements for throughput, latency, or consistency.

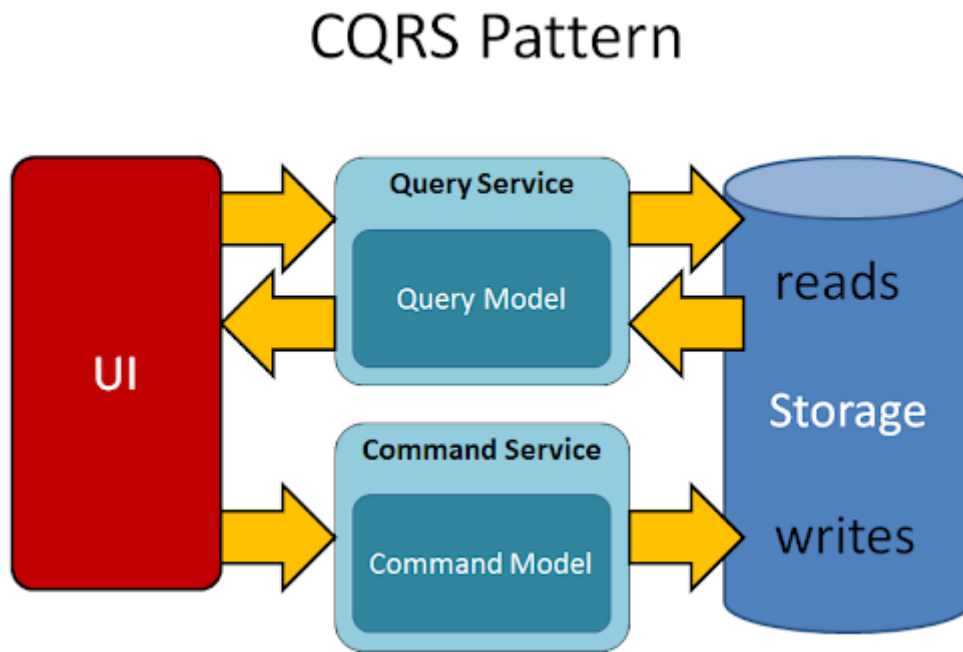


Figure 6 CQRS Pattern Schema

Internship Authority:

Signature:

Name & Surname: Berke Alpaslan

Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.

Start Date: 01/07/2024

End Date: 29/07/2024

2.5.3.3 Mediator Pattern

The Mediator pattern is a design pattern that allows you to manage a complex network of interactions between various objects within a single mediator object. This pattern centralizes communication and control between objects in the system, simplifying relationships and interactions across the system by reducing the direct connections that objects have with each other. The communication process typically involves three main components:

- **Request:** Requests sent by client objects to the mediator. These requests are used to trigger a function or operation within the system.
- **Handler:** Components within the mediator that address these requests and execute the appropriate actions. There can be one or several handlers designated for each type of request.
- **Response:** The outcomes returned by handlers to the client objects after completing the operations. This can include the results of an operation or the necessary data.

Thus, the Mediator pattern minimizes the direct interactions among objects within the system, promoting a cleaner and more manageable architectural structure.

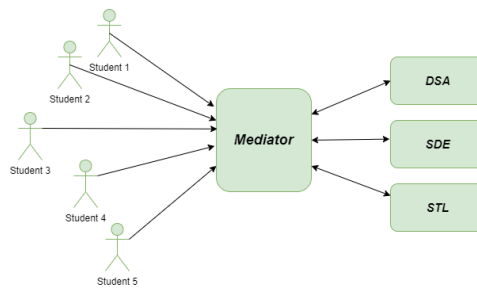


Figure 7 Mediator Pattern Schema

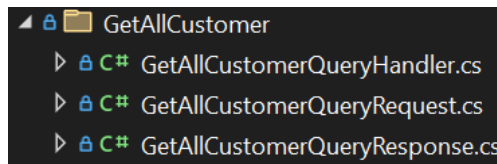


Figure 8 Mediator Pattern for GetAllCustomer in the Project

Internship Authority:

Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024

2.5.3.4 Marker Pattern

The Marker Pattern plays a simple yet effective role among software design patterns. This pattern is used to mark or identify specific objects, typically without carrying any methods or data. Also known as marker interfaces, this pattern is used to apply specific operations to certain objects or to enforce certain behaviors on them. These interfaces act as a type of label to identify objects that possess particular characteristics or behaviors.

Advantages:

- **Simplicity:** The Marker Pattern allows you to easily identify specific objects without complicating the structure of the application.
- **Flexibility:** It is easy to dynamically add or remove markers from objects, which enhances the modularity and flexibility of the system.

Disadvantages:

- **Limited Scope:** It does not add functionality, only serving to tag objects. This may not be sufficient in some cases.
- **Type Safety Issues:** In languages like Java, the use of marker interfaces can sometimes lead to type safety problems because these interfaces do not contain any methods, which can cause runtime errors.

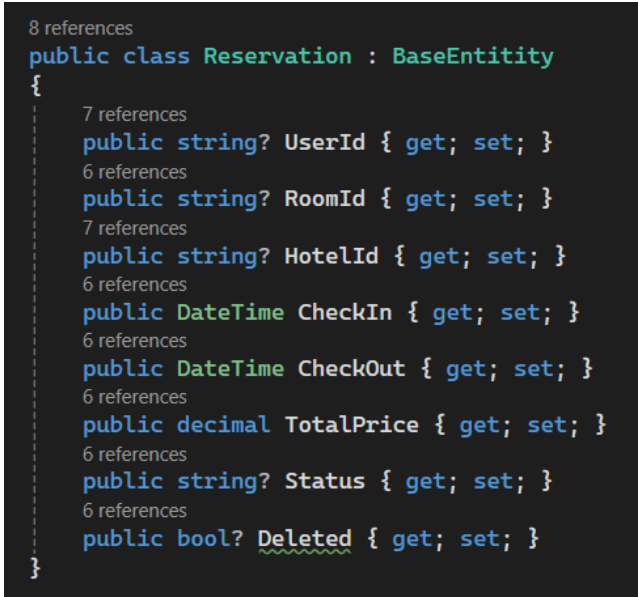
```
8 references
public interface IReadRepository<T> : IRepository<T> where T : BaseEntity
{
    8 references
    IQueryable<T> GetAll(bool tracking = true);
    1 reference
    IQueryable<T> GetWhere(Expression<Func<T, bool>> method, bool tracking = true);
    1 reference
    Task<T> GetSingleAsync(Expression<Func<T, bool>> method, bool tracking = true);
    15 references
    Task<T> GetByIdAsync(string id, bool tracking = true);
}
```

Figure 9 IReadRepository in the Project

Internship Authority:	Signature:
-----------------------	------------

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
<p>2.5.3.5 Generic Repository Design Pattern</p> <p>The Generic Repository Design Pattern is a structural approach in software design that abstracts data access mechanisms for different types of entities in an application. This pattern aims to reduce redundancy by centralizing common CRUD (Create, Read, Update, Delete) operations into a single, generic repository rather than having separate data access logic scattered across the application.</p> <p>Key Features:</p> <ul style="list-style-type: none">• Abstraction: It abstracts the details of the data access layer from the business logic, promoting a separation of concerns. This helps in isolating the business logic from how data is persisted in the database.• Reusability: By using a generic repository, developers can reuse the same data access logic across different types of entities, reducing code duplication and fostering cleaner code.• Maintainability: Changes to the data access logic need to be made in one place, improving maintainability. This also simplifies upgrades or changes to the database access technology. <p>Advantages:</p> <ul style="list-style-type: none">• Decoupling: The pattern helps decouple the application from specific data storage mechanisms, making it easier to switch out the underlying database without affecting the rest of the application.• Testing: Simplifies testing by allowing mock implementations of the repository in unit tests, ensuring that business logic can be tested separately from data access. <p>Disadvantages:</p> <ul style="list-style-type: none">• Over-Abstraction: Sometimes, the abstraction can be excessive, complicating scenarios where specialized behavior is needed for certain types of entities.• Complex Queries: Handling complex queries can be challenging, as the generic nature might not accommodate all the specific requirements without additional customization. <p>This design pattern is particularly useful in large applications with numerous entity types, where it can significantly streamline data handling strategies.</p>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
<div>2.6 Investigation of Computer Networks and Their Hardware and Structural Features</div> <p>The company's computer network is fortified with high-security firewalls and a range of network devices including routers, switches, and servers to ensure robust security and efficient data flow. The IT Department actively manages and analyzes this complex network infrastructure, conducting regular reviews to assess the performance and security of each device.</p> <ul style="list-style-type: none">Network Security Protocols: Advanced security protocols are employed to protect data integrity and privacy. This includes the use of encryption, intrusion detection systems, and secure VPN services for remote access.Performance Optimization: The network is designed to optimize performance through the use of high-capacity backbone connections, quality of service (QoS) configurations, and traffic prioritization techniques, ensuring that critical applications receive the bandwidth they require.Disaster Recovery and Business Continuity: The network architecture is structured to support disaster recovery plans. This includes redundant hardware and backup systems that can quickly restore functionality in the event of a system failure.Scalability and Flexibility: As the company grows, the network infrastructure is regularly updated and scaled to meet increasing demands. This scalability ensures that the network can support more users, more data, and more complex applications without degradation in performance.Compliance and Standards: The network is maintained in compliance with relevant industry standards and regulations, which helps to protect the company against legal and security risks. Regular audits are conducted to ensure ongoing compliance with these standards. <p>The IT Department's proactive approach in managing the network not only maintains its efficiency and security but also aligns with the company's strategic goals, supporting overall productivity and operational excellence.</p>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
<div>2.7 Developed Software Codes, Sample Data, and Results</div> <p>Throughout the internship, I documented the software codes and associated test scenarios in detail. Using sample data sets for tests, we evaluated the software’s performance and reliability, ensuring it met the project requirements and standards.</p> <div>2.7.1 Backend Source Code Examples</div> <div>2.7.1.1 Domain Layer Codes</div> <div></div> <p>Figure 10 Reservation Entity Class in the Project</p>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024

2.7.1.2 Application Layer Codes

```
4 references
public interface ITokenHandler
{
    3 references
    DTOs.Token CreateAccessToken(int minute);
    2 references
    string CreateRefreshToken();
}
```

Figure 11 ITokenHandler Interface in the Project

```
14 references
public interface IUserService
{
    2 references
    Task<CreateUserResponse> CreateAsync(CreateUser user);
    3 references
    Task UpdateRefreshToken(string refreshToken, AppUser user, DateTime accessTokenDate, int addOnAccessToken);
    2 references
    Task<List<ListUser>> GetAllUserAsync();
    3 references
    Task<AppUser> GetByIdUser(string id);
    2 references
    Task<AppUser> UpdateUser(UpdateUser user);
}
```

Figure 12 IUserService Interface in the Project

```
1 reference
public class CreateCustomerCommandHandler : IRequestHandler<CreateCustomerCommandRequest, CreateCustomerCommandResponse>
{
    readonly ICustomerWriteRepository _customerWriteRepository;

    0 references
    public CreateCustomerCommandHandler(ICustomerWriteRepository customerWriteRepository)
    {
        _customerWriteRepository = customerWriteRepository;
    }

    0 references
    public async Task<CreateCustomerCommandResponse> Handle(CreateCustomerCommandRequest request, CancellationToken cancellationToken)
    {
        await _customerWriteRepository.AddAsync(new()
        {
            Name = request.Name,
            Email = request.Email,
            Password = request.Password,
            Phone = request.Phone,
            Age = request.Age,
            Gender = request.Gender
        });

        await _customerWriteRepository.SaveAsync();
        return new();
    }
}
```

Figure 13 Handler Class of Mediator Pattern for CreateCustomer Operation in the Project

Internship Authority:	Signature:
-----------------------	------------

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024

2.7.1.3 Infrastructure Layer Codes

```
2 references
public class TokenHandler : ITokenHandler
{
    readonly IConfiguration _configuration;

    0 references
    public TokenHandler(IConfiguration configuration)
    {
        _configuration = configuration;
    }

    3 references
    public Application.DTOs.Token CreateAccessToken(int minute)
    {
        Application.DTOs.Token token = new();

        // Symmetry of SecurityKey
        SymmetricSecurityKey securityKey = new(Encoding.UTF8.GetBytes(_configuration["Token:SecurityKey"]));

        // Encryption of SymmetricSecurityKey
        SigningCredentials signingCredentials = new(securityKey, SecurityAlgorithms.HmacSha256);

        // Settings for Create Token
        token.Expiration = DateTime.UtcNow.AddMinutes(minute);
        JwtSecurityToken securityToken = new(
            audience: _configuration["Token:Audience"],
            issuer: _configuration["Token:Issuer"],
            expires: token.Expiration,
            notBefore: DateTime.UtcNow,
            signingCredentials: signingCredentials
        );

        // Token Creator
        JwtSecurityTokenHandler tokenHandler = new();
        token.AccessToken = tokenHandler.WriteToken(securityToken);

        // Refreash Token Creator
        token.RefreshToken = CreateRefreshToken();
        return token;
    }

    2 references
    public string CreateRefreshToken()
    {
        byte[] number = new byte[32];
        using RandomNumberGenerator random = RandomNumberGenerator.Create();
        random.GetBytes(number);
        return Convert.ToBase64String(number);
    }
}
```

Figure 14 TokenHandler Class for Create AccessToken in the Project

This TokenHandler class generates a user-specific token that facilitates authentication for a certain period when the user logs into the site.

Internship Authority:	Signature:
-----------------------	------------

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024

2.7.1.4 Persistence Layer Codes

```
0 references
public static class ServiceRegistration
{
    1 reference
    public static void AddPersistenceServices(this IServiceCollection services, IConfiguration configuration)
    {
        services.AddDbContext<SednaReservationAPIDbContext>(options => options.UseNpgsql(configuration.GetConnectionString("PostgreSQL")));
        services.AddIdentity<AppUser, AppRole>(options =>
        {
            options.Password.RequiredLength = 5;
            options.Password.RequireNonAlphanumeric = false;
            options.Password.RequireDigit = false;
            options.Password.RequireLowercase = false;
            options.Password.RequireUppercase = false;
        })
        .AddEntityFrameworkStores<SednaReservationAPIDbContext>();

        services.AddScoped<ICustomerReadRepository, CustomerReadRepository>();
        services.AddScoped<ICustomerWriteRepository, CustomerWriteRepository>();
        services.AddScoped<IHotelReadRepository, HotelReadRepository>();
        services.AddScoped<IHotelWriteRepository, HotelWriteRepository>();
        services.AddScoped<IPaymentReadRepository, PaymentReadRepository>();
        services.AddScoped<IPaymentWriteRepository, PaymentWriteRepository>();
        services.AddScoped<IReservationReadRepository, ReservationReadRepository>();
        services.AddScoped<IReservationWriteRepository, ReservationWriteRepository>();
        services.AddScoped<IReviewReadRepository, ReviewReadRepository>();
        services.AddScoped<IReviewWriteRepository, ReviewWriteRepository>();
        services.AddScoped<IRoomReadRepository, RoomReadRepository>();
        services.AddScoped<IRoomWriteRepository, RoomWriteRepository>();
        services.AddScoped<IRoomTypeReadRepository, RoomTypeReadRepository>();
        services.AddScoped<IRoomTypeWriteRepository, RoomTypeWriteRepository>();

        services.AddScoped<IUserService, UserService>();
        services.AddScoped<IAuthService, AuthService>();
    }
}
```

Figure 15 ServiceRegistration Class of Persistence Layer Services in the Project

The code snippet demonstrates how to use Dependency Injection (DI) to register services with the Inversion of Control (IoC) container in an ASP.NET Core application. The AddPersistenceServices method is an extension method for IServiceCollection that configures and registers various services required by the application.

- **AddDbContext:** Registers the SednaReservationAPIDbContext with the IoC container, configuring it to use a PostgreSQL database connection string from “appsettings.json”.
- **AddIdentity:** Configures and registers identity services using the AppUser and AppRole classes for user authentication and authorization. It also sets password requirements.
- **AddEntityFrameworkStores:** Integrates Identity with Entity Framework, allowing the application to use EF for storing identity-related data.
- **AddScoped:** Registers several repository and service interfaces with their corresponding implementations in the IoC container. The scoped lifetime ensures that each HTTP request gets a new instance of the service.

Internship Authority:	Signature:
-----------------------	------------

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024

2.7.1.5 Presentation Layer Codes

```
[Route("api/[controller]")]
[ApiController]
1 reference
public class CustomersController : ControllerBase
{
    private readonly ICustomerReadRepository _customerReadRepository;
    private readonly ICustomerWriteRepository _customerWriteRepository;
    private readonly IMediator _mediator;

    0 references
    public CustomersController(ICustomerReadRepository customerReadRepository, ICustomerWriteRepository customerWriteRepository, IMediator mediator)
    {
        _customerReadRepository = customerReadRepository;
        _customerWriteRepository = customerWriteRepository;
        _mediator = mediator;
    }

    [HttpGet]
    0 references
    public async Task<IActionResult> Get([FromQuery] GetAllCustomerQueryRequest getAllCustomerQueryRequest)
    {
        List<GetAllCustomerQueryResponse> response = await _mediator.Send(getAllCustomerQueryRequest);
        return Ok(response);
    }

    [HttpGet("{id}")]
    0 references
    public async Task<IActionResult> GetById([FromRoute] GetByIdCustomerQueryRequest getByIdCustomerQueryRequest)
    {
        GetByIdCustomerQueryResponse response = await _mediator.Send(getByIdCustomerQueryRequest);
        return Ok(response);
    }

    [HttpPost]
    0 references
    public async Task<IActionResult> addRoomType(CreateCustomerCommandRequest createCustomerCommandRequest)
    {
        CreateCustomerCommandResponse response = await _mediator.Send(createCustomerCommandRequest);
        return Ok(response);
    }

    [HttpDelete("{id}")]
    0 references
    public async Task<IActionResult> deleteRoomType([FromRoute] DeleteCustomerCommandRequest deleteCustomerCommandRequest)
    {
        DeleteCustomerCommandResponse response = await _mediator.Send(deleteCustomerCommandRequest);
        return Ok(response);
    }

    [HttpPut]
    0 references
    public async Task<IActionResult> updateRoom([FromBody] UpdateCustomerCommandRequest updateCustomerCommandRequest)
    {
        UpdateCustomerCommandResponse response = await _mediator.Send(updateCustomerCommandRequest);
        return Ok(response);
    }
}
```

Figure 16 CustomerController Class for Operate Customer CRUD Operations at Presentation Layer in the Project

This code example from the Presentation layer demonstrates the use of MediatR and CQRS patterns. The CustomersController uses dependency injection to include repositories and the MediatR mediator, which forwards requests to appropriate handlers in the Application layer. This design keeps the controller clean and focused on handling HTTP requests, avoiding code clutter and maintaining separation of concerns.

Internship Authority:	Signature:
-----------------------	------------

EK 11- Günlük Rapor Sayfası

Name & Surname: Berke Alpaslan

Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.

Start Date: 01/07/2024

End Date: 29/07/2024

2.7.1.5 Database Sample Data and Tables

Columns	Column Name	#	Data type	Identity	Collation	Not Null	Default	Comment
	Id	1	text		default	[v]		
	Name	2	text		default	[]		
	Age	3	text		default	[v]		
	Gender	4	bool			[]		
	RefreshToken	5	text		default	[]		
	RefreshTokenExpirationDate	6	timestampz			[]		
	isAdmin	7	bool			[]		
	UserName	8	varchar(256)		default	[]		
	NormalizedUserName	9	varchar(256)		default	[]		
	Email	10	varchar(256)		default	[]		
	NormalizedEmail	11	varchar(256)		default	[]		
	EmailConfirmed	12	bool			[v]		
	PasswordHash	13	text		default	[]		
	SecurityStamp	14	text		default	[]		
	ConcurrencyStamp	15	text		default	[]		
	PhoneNumber	16	text		default	[]		
	PhoneNumberConfirmed	17	bool			[v]		
	TwoFactorEnabled	18	bool			[v]		
	LockoutEnd	19	timestampz			[]		
	LockoutEnabled	20	bool			[v]		
	AccessFailedCount	21	int4			[v]		

Figure 17 PostgreSQL Table of “AspNetUsers” Entity which is Created with IdentityFramework in the Project

Grid	Id	Name	Age	Gender	RefreshToken	RefreshTokenExpirationDate	isAdmin	UserName	NormalizedUserName
1	5a516827-2e8a-46f2-8661-d8c23536c386	Berke Alpaslan	22	[]	lrz+lrAhXPbX96lyxWs6nuGIU6lvGpwf9s2y	2024-07-31 20:07:54.824 +0300	[]	BaraCuda	BARACUDA

Figure 18 Sample "AspNetUsers" Data for Project Testing

Internship Authority:

Signature:

EK 11- Günlük Rapor Sayfası

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024













 Id	1	uuid		[v]
 userId	2	text	default	[]
 Name	3	text	default	[]
 Address	4	text	default	[]
 Phone	5	text	default	[]
 Email	6	text	default	[]
 Description	7	text	default	[]
 StarRating	8	float4		[v]
 Star	9	int4		[v]
 ImageUrl	10	text	default	[]
 standartRoomPrice	11	text	default	[]
 CreatedDate	12	timestampz		[]
 UpdatedDate	13	timestampz		[]
 DeletedDate	14	timestampz		[]
 Deleted	15	bool		[v]

Figure 19 PostgreSQL Table of “Hotels” Entity in the Project

[illegible]

Figure 20 Sample "Hotel" Data for Project Testing

<p>Internship Authority:</p>	<p>Signature:</p>
------------------------------	-------------------

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
<div>2.7.2 Frontend Source Code Examples</div> <div><pre>getHotelById(id:string): Observable<HotelModal> { return this.http.get<HotelModal>(this.url+"/Hotels/"+id) } addHotel(hotel: HotelModal): Observable<any> { return this.http.post<HotelModal>(this.url+"/Hotels", hotel) } deleteHotel(id:string): Observable<any> { return this.http.delete(this.url+"/Hotels/"+id) } updateHotel(hotel: HotelModal): Observable<any> { return this.http.put(this.url+"/Hotels/", hotel) } getMyHotels(userId:string): Observable<HotelModal[]> { return this.http.get<HotelModal[]>(this.url+"/Hotels/myHotels/"+ userId) }</pre></div>	
<p>Figure 21 HTTPClient Codes to Communicate Project API for "Hotel" CRUD Operations</p> <p>This code snippet belongs to the "HotelService" TypeScript file in the frontend, which is used to interact with the API. The this.url represents the base URL of our API, which remains constant.</p>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
<div><pre>import { HttpClient } from '@angular/common/http'; import { Injectable } from '@angular/core'; import { roomModel } from '../model/room'; import { Observable } from 'rxjs/internal/Observable'; @Injectable({ providedIn: 'root' }) export class RoomService { url:string ="https://localhost:7171/api"; constructor(private http: HttpClient) { } get(): Observable<roomModel[]> { return this.http.get<roomModel[]>(this.url+"/Rooms") } getById(id:string): Observable<roomModel> { return this.http.get<roomModel>(this.url+"/Rooms/"+id) } add(hotel: roomModel): Observable<any> { return this.http.post<roomModel>(this.url+"/Rooms", hotel) } delete(id:string): Observable<any> { return this.http.delete(this.url+"/Rooms/"+id) } update(hotel: roomModel): Observable<any> { return this.http.put(this.url+"/Rooms/", hotel) } getRoomByHotelId(id:String):Observable<roomModel[]>{ return this.http.get<roomModel[]>(this.url+"/Rooms/hotel/"+id) } }</pre></div>	
<p>Figure 22 HTTPClient Codes to Communicate Project API for "Room" CRUD Operations</p> <p>This is another service similar to HotelService, named RoomService.</p>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan

Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.

Start Date: 01/07/2024

End Date: 29/07/2024

```
src > app > hotels > hotel > reservation > reservation.component.html > h2.text-center
Go to component
1 <h2 mat-dialog-title class="text-center">Rezervasyon </h2>
2 <mat-dialog-content class="mat-typography">
3   <form [formGroup]="reservationForm" class="example-form">
4
5
6     <input type="hidden" formControlName="userId" placeholder="User Id">
7     <input type="hidden" formControlName="totalPrice" [value]="totalPrice">
8     <input type="hidden" formControlName="hotelId" placeholder="Hotel Id">
9     <input type="hidden" formControlName="roomId" placeholder="roomId">
10
11     <!-- DateIimepicker -->
12     <mat-form-field>
13       <input formControlName="checkIn" matInput [matDatepickerFilter]="checkInFilter" [matDatepicker]="checkInPicker" placeholder="Check-in
14       Date" (dateChange)="onCheckInDateChange($event)">
15       <mat-datepicker-toggle matSuffix [for]="checkInPicker"></mat-datepicker-toggle>
16       <mat-datepicker #checkInPicker></mat-datepicker>
17       <mat-error *ngIf="reservationForm.get('checkIn')?.hasError('required')">
18         Check-in Tarihi gereklidir
19       </mat-error>
20     </mat-form-field>
21     <mat-form-field>
22       <input formControlName="checkOut" matInput [matDatepickerFilter]="checkOutFilter" [matDatepicker]="checkOutPicker"
23       placeholder="Check-out Date" [min]="checkInDate" (dateChange)="onCheckOutDateChange($event)">
24       <mat-datepicker-toggle matSuffix [for]="checkOutPicker"></mat-datepicker-toggle>
25       <mat-datepicker #checkOutPicker></mat-datepicker>
26       <mat-error *ngIf="reservationForm.get('checkOut')?.hasError('required')">
27         Check-out Tarihi gereklidir
28       </mat-error>
29     </mat-form-field>
30
31     <input type="hidden" formControlName="status" value="PNC" placeholder="status">
32
33     <div class="total-price">
34       <label><b>Toplam fiyat:</b> {{totalPrice}} ₺</label>
35     </div>
36   </form>
37 </mat-dialog-content>
38 <mat-dialog-actions align="end">
39   <button mat-button mat-dialog-close>Cancel</button>
40   <button (click)="addReservation()" mat-button [disabled]="!reservationForm.valid">Rezervasyon Yap</button>
41 </mat-dialog-actions>
```

Figure 23 HTML Code of "Reservation" Page

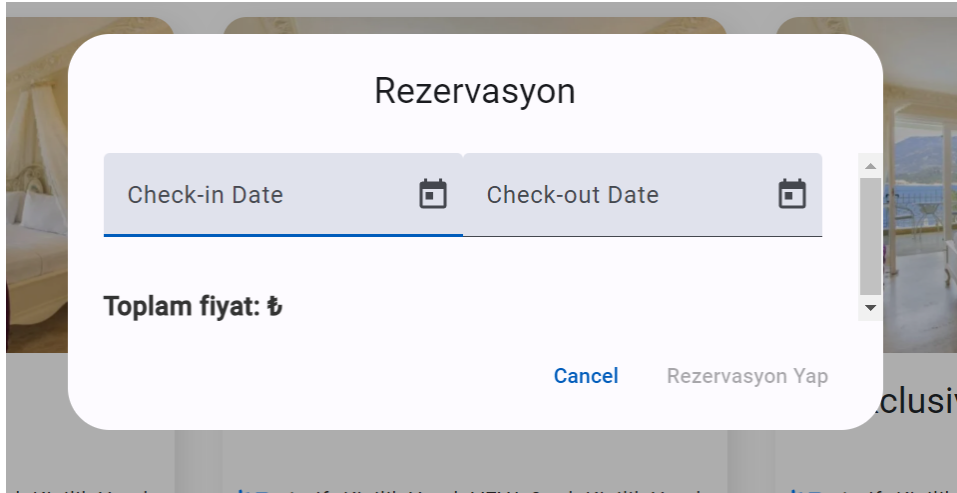


Figure 24 Reservation Page of the Project

These images contain the HTML code for the reservation page and how it appears visually.

Internship Authority:

Signature:

EK 11- Günlük Rapor Sayfası

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024

```
<section class="vh-100">
  <div class="container py-5">
    <div class="row d-flex justify-content-center align-items-center h-100">
      <div class="col-12 col-md-8 col-lg-6 col-xl-5">
        <div class="card text-white" style="border-radius: 1rem; background-color: #434853;">
          <div class="card-body p-5 text-center">
            <form (ngSubmit)="login()" >
              <div class="mt-md-4 pb-5">
                <h2 class="fw-bold mb-2 text-uppercase">Giriş Yap</h2>
                <div data-mdb-input-init class="form-group form-outline form-white mb-4">
                  <label for="username" class="form-label">Kullanıcı Adı/Email</label>
                  <input [(ngModel)]="email" type="text" name="email" class="form-control form-control-lg" placeholder="Kullanıcı Adı/Email"/>
                </div>
                <div data-mdb-input-init class="form-group form-outline form-white mb-4">
                  <label for="password" class="form-label">Şifre</label>
                  <input type="password" name="pass" placeholder="Şifre" [(ngModel)]="password" required class="form-control form-control-lg"/>
                </div>
                <p class="small mb-3 pb-lg-2"><a class="text-white-50" routerLink="/password-reset">Şifreni mi unuttun?</a></p>
                <button data-mdb-button-init data-mdb-ripple-init class="btn btn-outline-light btn-lg px-5" type="submit">Giriş Yap</button>
              </div>
            </form>
            <div>
              <p class="mb-0">Hesabın yok mu? <a routerLink="/register" class="text-white-50 fw-bold">Kayıt Ol!</a></p>
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>
</section>
```

Figure 25 HTML Code of Login Page

Figure 26 Login Page of the Project

Here are the login page layout and its HTML code.

Internship Authority:	Signature:
-----------------------	------------

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024

```
export class SignupComponent {
  public errorMessage: string = ""
  public signUpForm: FormGroup
  constructor(
    private userService: UserService,
    private fb: FormBuilder,
    private _snackBar: MatSnackBar,
    private router: Router,
  ) {
    this.signUpForm = this.fb.group({
      name: ['', [ValidationService.nameValidator()]],
      userName: ['', [ValidationService.usernameValidator()]],
      email: ['', [ValidationService.emailValidator()]],
      age: ['', [ValidationService.ageValidator()]],
      phone: ['', [ValidationService.phoneValidator()]],
      gender: [false],
      password: ['', [ValidationService.passwordValidator()]],
      confirmPassword: ['', [ValidationService.confirmPasswordValidator()]],
    }), {
      validators: (group: AbstractControl): ValidationErrors | null => {
        let pass = group.get('password').value;
        let confirmPass = group.get('confirmPassword').value;
        var a = pass == confirmPass && confirmPass == pass ? null : { notSame: true };
        return a;
      }
    })
  }

  submitRegister() {
    if (this.signUpForm.valid) {
      var user = this.signUpForm.value
      user.phone = "0" + user.phone;
      user.age = user.age + ""

      this.userService.createUser(this.signUpForm.value).subscribe((resp) => {
        if (!!resp) {
          this._snackBar.open('Başarıyla kayıtlı oldun', '', { duration: 4000 });
          this.router.navigate(["login"]);
        }
      }, (error) => {
        console.error('Signup failed:', error);
        this.errorMessage = 'Kayıt olurken bir hata oluştu. Hata Mesajı: ' + error.message;
      })
    }
  }
}
```

Figure 27 HTML Code of SignUp Page

KAYIT OL

İsim Soyisim*

Kullanıcı Adı *

İsim Soyisim

Kullanıcı Adı

Email *

Telefon Numarası *

Örn: örn@gmail.co

Örn: 535353535

Yaş *

Cinsiyet *

Yaş

Erkek

Yeni Şifre *

Yeni Şifre Tekrarı *

Yeni Şifre

Yeni Şifre Tekrarı

Kayıt Ol

Zaten hesabın var mı? [Giriş Yap](#)

Figure 28 Signup Page of the Project

Internship Authority:	Signature:
-----------------------	------------

EK 11- Günlük Rapor Sayfası

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024



Figure 29 "Hotel Admin" Page for Managemet in the Project

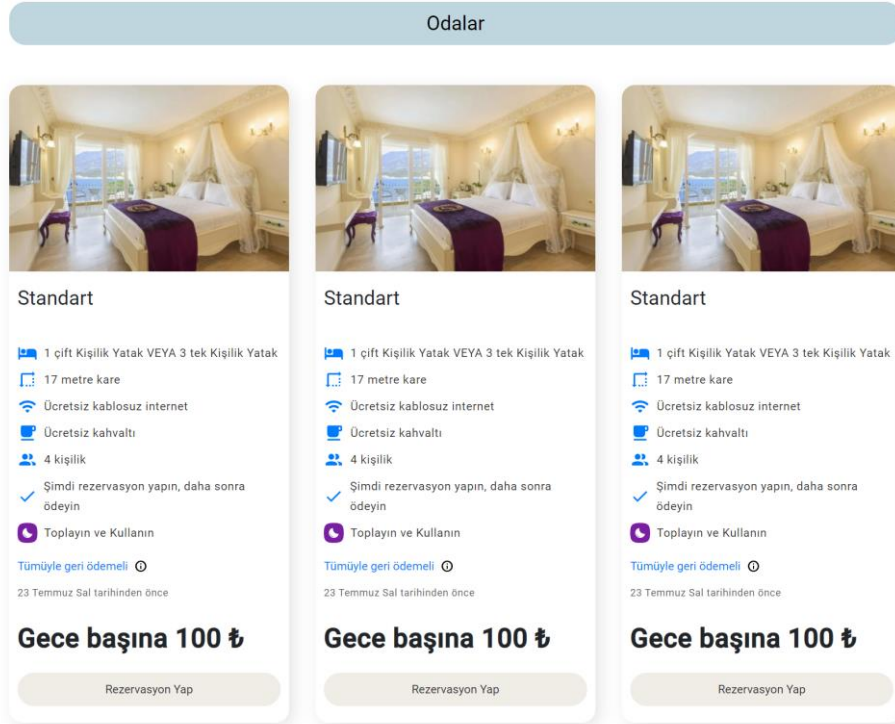


Figure 30 The Page where the Rooms of a Hotel are Listed in the Project

Internship Authority:	Signature:
-----------------------	------------

EK 11- Günlük Rapor Sayfası

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024

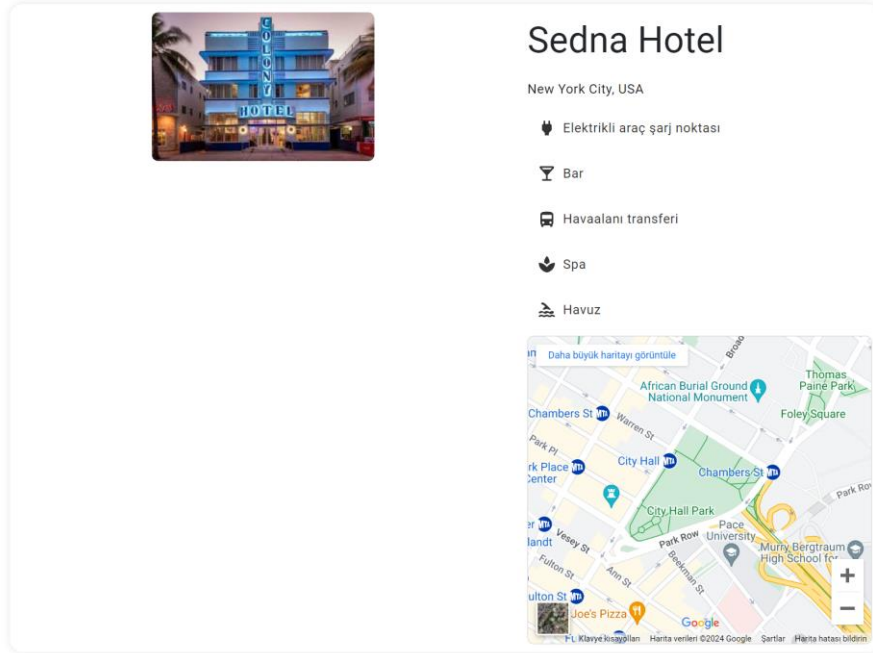


Figure 31 When You Click on the Hotel You Want on the Home Page, the Page that Appears

Ödeme Sayfası

Oda Bilgileri

Sedna Hotel

All-Exclusive

Giriş Tarihi: Jul 31, 2024

Çıkış Tarihi: Aug 3, 2024

Toplam Fiyat: 1500 ₺

Ödeme Bilgileri

Kart Üzerindeki İsim

Kart Numarası

Son Kullanma Tarihi Ay

Son Kullanma Tarihi Yıl

CVV

Ödemeyi Tamamla

Figure 32 Payment Page After Choosing How Many Days You Booked For in the Project

Internship Authority:	Signature:
-----------------------	------------

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024

Yorum Yap

Değerlendirme

5 Yıldız

Yorum

Good Service!

Submit

Daha Önce Yorum Yapılmamış.

Figure 33 Comment Section for a Hotel in the Project

Yorumlar

alpaslanberke@gmail.com ★★★★★

Aug 1, 2024
Good Service!

alpaslanberke@gmail.com ★★★★★

Aug 1, 2024
Good!

Başarıyla yorum yapıldı.

Figure 34 Notification alert after posting a comment and the listed comments in the Project

Internship Authority:

Signature:

EK 11- Günlük Rapor Sayfası

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024

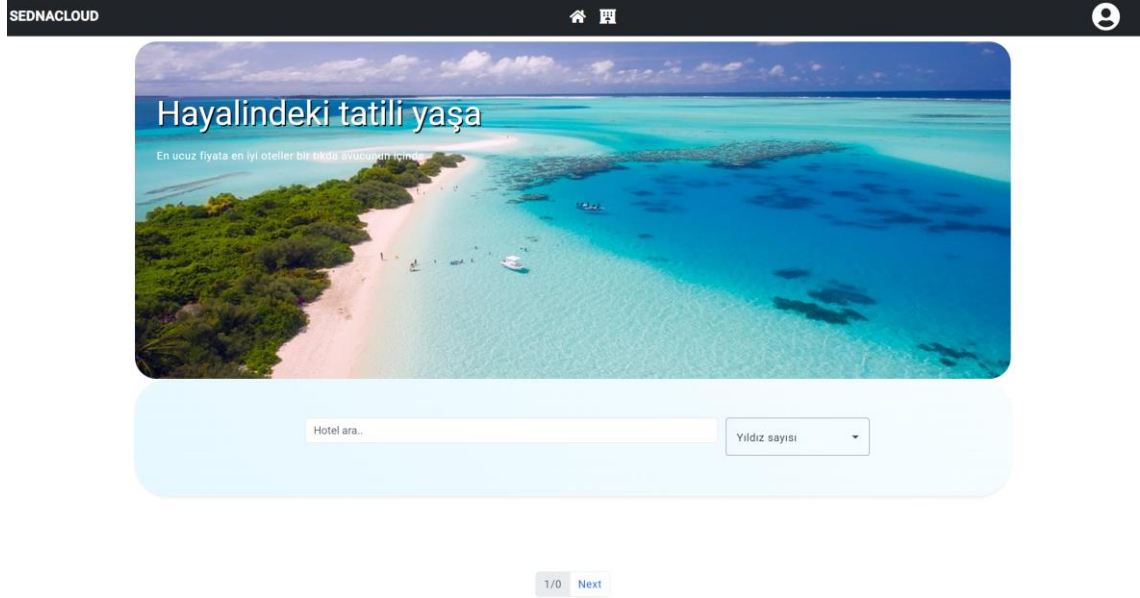


Figure 35 The Starting Section of the Page Where Hotels are Listed in the Project

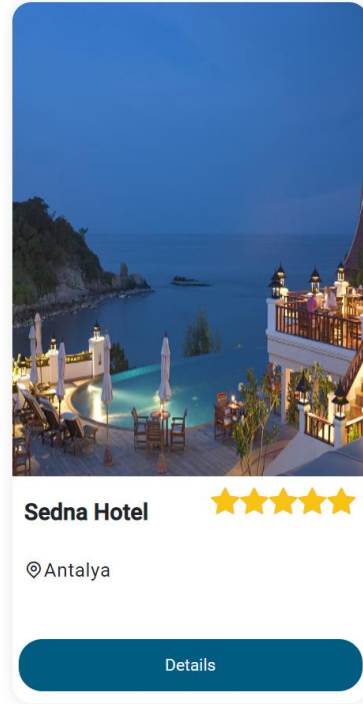
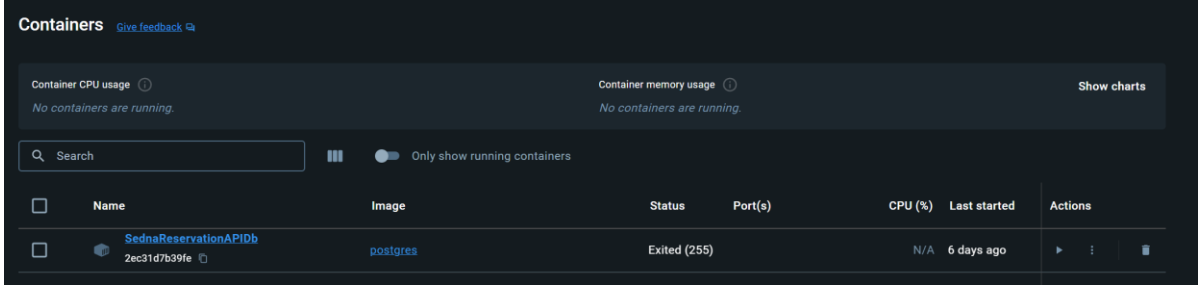
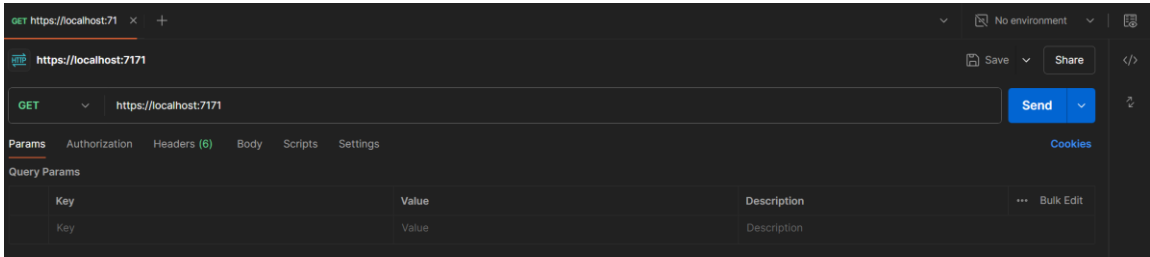


Figure 36 The Display of a Listed Hotel on the Hotel Listing Page and Information About its Star Rating

Internship Authority:	Signature:
-----------------------	------------

Name & Surname: Berke Alpaslan			
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.			
Start Date: 01/07/2024	End Date: 29/07/2024		
<div><h2>CHAPTER 3</h2><h3>INTERNSHIP DIARY JOURNAL</h3></div> <div><h4>3.1 Day 1 (01/07/2024)</h4><p>On my first day of the internship, I was introduced to the company by the Human Resources Manager, Ms. Gülhan Koyak. Afterwards, I received information about the technologies and programming languages used by the company from our intern supervisor. Upon learning that they use .NET on the backend, Angular on the frontend, and PostgreSQL for the database, I decided to focus on the backend. Thus, I spent the rest of the day watching training videos on ASP.NET and C# to complete my first day.</p><h4>3.2 Day 2 (02/07/2024)</h4><p>Since the company uses "Angular" for the frontend, I watched training videos on "HTML, CSS, JavaScript" to establish a foundation before learning Angular. Additionally, today I gained access to the company's own training videos and took the opportunity to review them.</p><p>Towards the end of the day, it was mentioned that our group of five interns would be expected to write a project using the software languages and technologies utilized by the company. The topic, details, and task distribution of the project will be provided in the coming days.</p></div> <table><tr><td>Internship Authority:</td><td>Signature:</td></tr></table>		Internship Authority:	Signature:
Internship Authority:	Signature:		

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
 3.3 Day 3 (03/07/2024) <p>We were tasked with creating a "Hotel Reservation System" web application using ASP.NET, Angular, PostgreSQL, and other languages and technologies used by the company. The data and data types we will use for this project were shared with us in the form of PostgreSQL example table creation scripts. Consequently, I watched ASP.NET training videos and installed PostgreSQL, Visual Studio, and other necessary applications on my computer, focusing on this project. I also watched new training videos related to the languages we will be using in this project (C#, TypeScript, HTML, CSS).</p> <p>Towards the end of the day, I set up the project skeleton for the backend in ASP.NET, in accordance with the "ONION ARCHITECTURE."</p> <p>Database Relational Table Structure</p> <pre>CREATE TABLE Users (id SERIAL PRIMARY KEY, name VARCHAR(100) NOT NULL, email VARCHAR(100) UNIQUE NOT NULL, password VARCHAR(255) NOT NULL, phone VARCHAR(20) NOT NULL, age INT CHECK (age >= 0), gender VARCHAR(10) CHECK (gender IN ('male', 'female', 'other')), created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP, updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP, deleted BOOLEAN DEFAULT FALSE);</pre> <p>Figure 37 Sample PostgreSQL Table Creation Code Provided by Our Internship Supervisor for the Project</p>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
<div><h3>3.4 Day 4 (04/07/2024)</h3><p>Today, in our project based on the "ONION ARCHITECTURE," I wrote the necessary foundational code in line with the architecture and then attempted to establish a PostgreSQL connection. For this purpose, I worked on a separate test project independent from the main project. If I can fully master it by tomorrow, I will try to implement the PostgreSQL connection for our main project.</p><p>To ensure the database remains continuously active, I installed Docker on my computer. I also installed "Postman" to test API communication in the later stages of our project.</p><p>I started the project by setting up software architectures with ASP.NET, and since I am more interested in the backend, I decided to take part in the backend development aspect of the project.</p><div></div><p>Figure 38 Docker Container of our Project's PostgreSQL Database</p><div></div><p>Figure 39 The Postman App to API Test</p></div>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024

3.5 Day 5 (05/07/2024)

I was able to establish the PostgreSQL connection for our project. Using the "EntityFrameworkCore" framework within ".NET," I executed my code to first create a "Migration," and then access the database I opened in a "Container" on Docker. This allowed me to generate tables suitable for each entity in my project within this local database using the created "Migration."

To perform database CRUD operations, I decided to use the "Generic Repository Design Pattern" and watched videos related to it.

Since we are conducting this project as intern students, we are using the "Git" version control system to take checkpoints of our progress and save them to our jointly established GitHub repository. To avoid any confusion, each of us created a branch under our own name and records our work on our respective branches.

```
1 reference
public partial class mig_1 : Migration
{
    /// <inheritdoc />
    [Obsolete]
    protected override void Up(MigrationBuilder migrationBuilder)
    {
        migrationBuilder.CreateTable(
            name: "AspNetRoles",
            columns: table => new
            {
                Id = table.Column<string>(type: "text", nullable: false),
                Name = table.Column<string>(type: "character varying(256)", maxLength: 256, nullable: true),
                NormalizedName = table.Column<string>(type: "character varying(256)", maxLength: 256, nullable: true),
                ConcurrencyStamp = table.Column<string>(type: "text", nullable: true)
            },
            constraints: table =>
            {
                table.PrimaryKey("PK_AspNetRoles", x => x.Id);
            });
    }

    migrationBuilder.CreateTable(
        name: "AspNetUsers",
        columns: table => new
        {
            Id = table.Column<string>(type: "text", nullable: false),
            Name = table.Column<string>(type: "text", nullable: true),
            Age = table.Column<string>(type: "text", nullable: false),
            Gender = table.Column<bool>(type: "boolean", nullable: true),
            RefreshToken = table.Column<string>(type: "text", nullable: true),
            RefreshTokenExpirationDate = table.Column<DateTime>(type: "timestamp with time zone", nullable: true),
            IsAdmin = table.Column<bool>(type: "boolean", nullable: true),
            Username = table.Column<string>(type: "character varying(256)", maxLength: 256, nullable: true),
            NormalizedUsername = table.Column<string>(type: "character varying(256)", maxLength: 256, nullable: true),
            Email = table.Column<string>(type: "character varying(256)", maxLength: 256, nullable: true),
            NormalizedEmail = table.Column<string>(type: "character varying(256)", maxLength: 256, nullable: true),
            EmailConfirmed = table.Column<bool>(type: "boolean", nullable: false),
            PasswordHash = table.Column<string>(type: "text", nullable: true),
            SecurityStamp = table.Column<string>(type: "text", nullable: true),
            ConcurrencyStamp = table.Column<string>(type: "text", nullable: true),
            PhoneNumber = table.Column<string>(type: "text", nullable: true),
            PhoneNumberConfirmed = table.Column<bool>(type: "boolean", nullable: false),
            TwoFactorEnabled = table.Column<bool>(type: "boolean", nullable: false),
            LockoutEnd = table.Column<DateTimeOffset>(type: "timestamp with time zone", nullable: true),
            LockoutEnabled = table.Column<bool>(type: "boolean", nullable: false),
            AccessFailedCount = table.Column<int>(type: "integer", nullable: false)
        },
        constraints: table =>
        {
            table.PrimaryKey("PK_AspNetUsers", x => x.Id);
        });
    }
}
```

Figure 40 An excerpt from the Migration class automatically generated using EntityFramework for my entities

Internship Authority:

Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
 3.6 Day 6 (08/07/2024) <p>In our project, to facilitate database querying and manipulation tasks, I implemented the "Generic Repository Design Pattern" and "Marker Pattern," and wrote the necessary generic interfaces and concrete classes at “Application Layer” of Onion architecture. I also added read and write repository concretes for each table.</p> <p>I wrote an interceptor for the "BaseEntity" class, which contains data required in every entity type. This interceptor automatically adds, updates, or deletes common properties such as creation date, update date, and deletion date in the database according to the action performed. This automation prevents the need for manually entering these details each time.,</p>  <pre>2 references public override async Task<int> SaveChangesAsync(CancellationToken cancellationToken = default) { var datas = ChangeTracker.Entries<BaseEntity>(); foreach (var data in datas) { _ = data.State switch { EntityState.Added => data.Entity.CreatedDate = DateTime.UtcNow, EntityState.Modified => data.Entity.UpdatedDate = DateTime.UtcNow, EntityState.Deleted => data.Entity.DeletedDate = DateTime.UtcNow, EntityState.Unchanged => data.Entity.UpdatedDate = DateTime.UtcNow, }; } return await base.SaveChangesAsync(cancellationToken); }</pre>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
<p>3.7 Day 7 (09/07/2024)</p> <p>In our project's "Presentation" layer, I decided to adopt a new software pattern to write "Controller" classes in a more organized and easily adaptable manner for future changes. As we aim to develop a forward-looking project, we will implement the "CQRS Design Pattern," facilitated by the "Mediator" pattern, to ensure easy adaptation to future modifications. Today, I established the necessary infrastructure for the CQRS Design pattern. Moving forward, as our project progresses, we will use this CQRS pattern when writing methods for CRUD operations.</p> <p>Additionally, towards the end of the day, I conducted research on the ASP.NET Identity Framework as we intend to securely store our entities in the database using this mechanism.</p> <p>3.8 Day 8 (10/07/2024)</p> <p>I laid the groundwork for using the Identity Framework in our project to securely store data in the database in an encrypted form. I gathered information about the necessity of using "Tokens" for authentication processes and wrote basic-level code to generate Access Tokens using "JWT" in our project. If I am successful in producing the Access Token, the next step will involve implementing the production of a "Refresh Token" that will generate a new Access Token as the validity of the current one expires.</p> <p>Figure 42 CreateAccessToken Class for AccessToken Creation After Sign In</p> <pre>3 references public Application.DTOS.Token CreateAccessToken(int minute) { Application.DTOS.Token token = new(); // Symmetry of SecurityKey SymmetricSecurityKey securityKey = new(Encoding.UTF8.GetBytes(_configuration["Token:SecurityKey"])); // Encryption of SymmetricSecurityKey SigningCredentials signingCredentials = new(securityKey, SecurityAlgorithms.HmacSha256); // Settings for Create Token token.Expiration = DateTime.UtcNow.AddMinutes(minute); JwtSecurityToken securityToken = new(audience: _configuration["Token:Audience"], issuer: _configuration["Token:Issuer"], expires: token.Expiration, notBefore: DateTime.UtcNow, signingCredentials: signingCredentials); // Token Creator JwtSecurityTokenHandler tokenHandler = new(); token.AccessToken = tokenHandler.WriteToken(securityToken); }</pre>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
<div>3.9 Day 9 (11/07/2024)</div> <p>Before adding information from the client to the database, I implemented "Fluent Validator" in the backend to check if the necessary conditions are met for each table. Additionally, since our project will have a client operating in a browser, I watched necessary videos to learn about CORS security policy and the Same Origin Policy. This allowed me to write the required code in the backend according to CORS policies, enabling requests from Angular's client side on "localhost:4200" to our API.</p> <p>Figure 43 The code that allows the Client URL of our project to connect with the API under CORS policies</p> <pre>builder.Services.AddCors(options => options.AddDefaultPolicy(policy => policy.WithOrigins("https://localhost:4200", "http://localhost:4200").AllowAnyHeader().AllowAnyMethod()));</pre> <p>Figure 44 Validation for CreateUser Operation in the Project</p> <pre>0 references public CreateUserValidator() { RuleFor(u => u.Name) .NotEmpty() .NotNull() .WithMessage("Please do not leave username blank!") .Length(2, 30) .WithMessage("Please enter name with length between 2-30!"); RuleFor(u => u.Phone) .NotEmpty() .NotNull() .WithMessage("Please do not leave phone number blank!") .Length(10, 11) .WithMessage("Please enter valid phone number!"); RuleFor(u => u.Password) .NotEmpty() .NotNull() .WithMessage("Please do not leave password blank!") .Must(BeValidPassword) .WithMessage("Please enter valid password!"); RuleFor(u => u.Age) .NotEmpty() .NotNull() .WithMessage("Please do not leave age blank!") .GreaterThan(0) .WithMessage("Your age can't be less than 0!") .Must(BeValidAge) .WithMessage("Please enter valid age!"); RuleFor(u => u.Email) .NotEmpty() .NotNull() .WithMessage("Please do not leave mail blank!") .Must(BeValidMail) .WithMessage("Please enter valid mail!"); RuleFor(u => u.Gender) .NotEmpty() .NotNull() .WithMessage("Please do not leave gender blank!") .Must(BeValidGender) .WithMessage("Please enter valid gender!"); }</pre>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
 3.10 Day 10 (12/07/2024) <p>In line with the previously established CQRS and Mediator architectures, I wrote the necessary handler functions for all database operations (create, delete, update, get) for each table.</p> <p>I made a very basic start on generating a refresh token. After laying the foundation for the refresh token and postponing detailed processes for later, I installed Angular on my computer, as we had sufficiently developed the backend of our project. I began learning the fundamentals of Angular and its rules. My fellow interns also started to focus more on the frontend.</p> <p>From now on, as the backend code we write will be developed in conjunction with the frontend, the backend is almost complete, with just a few parts missing, allowing us to start working on the frontend.</p> 3.11 Day 11 (16/07/2024) <p>I have initiated a basic implementation of the RefreshToken structure using JWTBearer and have it operational, although there are still a few shortcomings. Currently, I can generate refresh tokens. Additionally, the "Connection String" used to load the migrations we wrote to the database was previously hardcoded within necessary functions, which contradicted the SOLID principles. Although I attempted to address this issue early in the project, the ConfigurationManager class was initially unable to read the "appsettings.json" file. This issue also emerged when we needed to extract necessary parameters for token usage in adherence to SOLID principles. I revisited this problem and resolved it. Now, our project can retrieve the required parameters from the "appsettings.json" file in the backend.</p>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
 3.12 Day 12 (17/07/2024) <p>As backend interns, we recently updated the validations I had initially implemented using Fluent Validation. Subsequently, my colleagues working on the frontend requested a few additions and corrections from me in the backend. Additionally, we encountered some errors in certain parts of the backend. Throughout the day, besides updating validations, I primarily focused on these additions, modifications, and corrections in the backend. With these adjustments, we have now made significant progress on the frontend as well, and the development of a functional website with several working features has begun.</p> 3.13 Day 13 (18/07/2024) <p>Although there are still tasks to be completed in the backend, I have managed to develop it to a level sufficient for advancing the frontend. Thus, desiring to engage with the frontend and learn more, I watched videos about Angular today and began building the frontend of our project from scratch. Since my involvement in the frontend is for learning purposes, the parts I work on will not be used in the actual project.</p> <p>I also set up the infrastructure for our website's "Admin" and "UI" sections and established a "Multiple Layout" system. Planning to use "Material" for the Admin part and "Bootstrap" for the UI, I installed the necessary software on my computer and created interfaces using a simple UI and Admin structure.</p>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
 3.14 Day 14 (19/07/2024) <p>Before starting the connection processes with the backend in the Angular section, I downloaded the "jQuery", "AlertifyJS", and "Toastr" modules for visual presentation to my project and wrote the necessary import statements and codes for notification cards. Having worked significantly on the backend, I also implemented the required backend changes and corrections requested by my colleagues working on the main project's frontend. Additionally, I made a basic start on creating my own HTTPClientService in my Angular project.</p> 3.15 Day 15 (22/07/2024) <p>Yesterday, I laid the foundation for my own HTTPClientService, and today, I completed it by creating a fully functional “Custom HttpClientService” that communicates effectively with the API. Subsequently, in the frontend of my test project, I developed a component intended for posting hotel listings on the admin page. I also wrote another component to display these published hotel listings on the admin page. Through the component designed for creating hotels, I connected to the API using my custom HttpClient and successfully facilitated the transfer of the created hotels to the database. On the backend, I revised and integrated the filtering code written by my fellow intern for filtering hotel listings on the UI side into our main backend project.</p>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
	
<p>Figure 45 HttpClientService of My Frontend Learning Project</p>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
 3.16 Day 16 (23/07/2024) <p>Having previously set up the hotel creation in the frontend to connect with the API and save to the database, I had only written a component for listing purposes. Today, when I tried to execute the listing, I received an <code>HttpResponseError</code>. Despite the error, the API connection was established, and I was receiving responses. I spent most of today resolving this issue. I discovered that the error was due to the "CreateHotel" controller, which we had written using the Mediator architecture. It was expecting data in the request, but I had not provided any data from the frontend, leading to the error. Once I made the necessary adjustments in the backend, I resolved the issue.</p> <p>I am undertaking my frontend work purely to gain insights and learn about Angular and its associated languages (TypeScript, HTML, SCSS). Since I am not involved in the frontend part of our main project, it is unlikely that this work will be included in our project.</p> 3.17 Day 17 (24/07/2024) <p>I am part of the backend team for our project. My colleagues working on the frontend have nearly completed the project at a basic level. Once the backend was finished, the frontend development phase began. From the start of the frontend development, I have maintained constant communication, providing support by making necessary backend corrections and additions as required by the frontend team. However, there have been some very minor changes made to the backend without my knowledge during the frontend development. Today, I obtained the updated backend code from my colleague and made the necessary small corrections. As a significant change, I removed the "Validation" codes in the backend to make it easier to test our project in the future. Additionally, we have started integrating the work done by our colleagues on the frontend. Soon, we will have unified "API" and "CLI" projects.</p>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
<p>3.18 Day 18 (25/07/2024)</p> <p>Today, I assisted my colleagues working on the frontend by brainstorming ideas on how to implement our desired features into the project. We collaborated at the computer to integrate these features into our project.</p> <p>We are no longer making significant changes to the backend, except for very minor adjustments. While it may not be fully stable, we now have a functional ASP.NET backend project that effectively handles our tasks and communicates seamlessly with our PostgreSQL database.</p> <p>3.19 Day 19 (26/07/2024)</p> <p>Today, based on the knowledge I gained while writing my own frontend test project, I joined my colleagues working on the frontend. Together, we brainstormed and wrote the necessary code to add final features, make corrections, and resolve any errors we encountered. While we can't call it fully stable, we now have a functional frontend project that connects seamlessly with our API. I spent the entire day working with my team on these tasks.</p> <p>3.20 Day 20 (29/07/2024)</p> <p>Today, we ran our completed backend (API) and frontend (CLI) projects simultaneously to test the functionalities of our project one last time. After completing our tests, we presented our project to the internship supervisor. The supervisor provided feedback on what we could add or improve in future stages of the project.</p> <p>With a project completed at a basic level, I have now finished my 20-day internship period.</p>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
<div><h2>CONCLUSION</h2><p>Throughout this internship and the given project, I gained substantial knowledge and valuable experience in various areas, especially in backend development, full stack development, project planning and management, and teamwork. During the backend development process, I gained in-depth knowledge in database operations, API creation and management using ASP.NET and EntityFramework. I also learned extensively about software architectures and practically applied design principles by implementing architectures such as CQRS, Mediator, Generic Repository, and Marker.</p><p>On the frontend, I learned about modern web technologies by using Angular and understood the importance of visual presentation through working with various UI libraries. By integrating libraries such as JQuery, AlertifyJS, and Toastr, I enhanced user experience with notification systems. In terms of project planning and management, I gained experience in task allocation, time management, and efficient use of resources. I improved my teamwork skills, realizing the critical role of communication and collaboration within a team in achieving project success.</p><p>The experiences I gained during this internship will greatly contribute to my professional career and help me work more effectively and efficiently in software development processes. This internship period was invaluable as it provided an opportunity to put what I have learned into practice and apply it in real-world projects.</p></div>	
Internship Authority:	Signature:

Name & Surname: Berke Alpaslan	
Company Title: KOD YAZILIM PROJE HİZ. TUR. TİC. A.Ş.	
Start Date: 01/07/2024	End Date: 29/07/2024
<div><h2>REFERENCES</h2><p>Kod Software. (n.d.). About Us. Kod. Retrieved from https://www.kod.com.tr/Home/About</p><p>Yıldız, G. (2021, March 6). Nedir Bu Onion Architecture? Tam Teferruatlı İnceleyelim. Retrieved from https://www.gencayyildiz.com/blog/nedir-bu-onion-architecture-tam-teferruatli-inceleyelim/</p><p>Yıldız, G. (2021, March 8). CQRS Pattern Nedir? MediatR Kütüphanesi ile Nasıl Uygulanır? Retrieved from https://www.gencayyildiz.com/blog/cqrs-pattern-nedir-mediatr-kutuphanesi-ile-nasil-uygulanir/</p><p>Alpaslan, B. (n.d.). SednaReservationAPI. GitHub repository. Retrieved from https://github.com/BerkeAlpaslan/SednaReservationAPI</p><p>Alpaslan, B. (n.d.). SednaReservationCLI. GitHub repository. Retrieved from https://github.com/BerkeAlpaslan/SednaReservationCLI</p><p>Ulu, G. (n.d.). Sednacloud. GitHub repository. Retrieved from https://github.com/gozdeulu07/Sednacloud</p><p>Ulu, G. (n.d.). Sednacloud-Client. GitHub repository. Retrieved from https://github.com/gozdeulu07/Sednacloud-Client</p></div>	
Internship Authority:	Signature: