COMP 200

Summer Practice Report

Ömer Bilbil
042101089

01.07.2025 – 31.07.2025, 21 workdays
Submitted: *29.09.2025*

MEF University

Computer Engineering Program

Executive Summary

During my internship within the DevOps team at the Turkish Technology firm, I gained a lot of new information and also enhanced my technical capabilities significantly. Working on this project gave me the avenue to use contemporary tools and technologies that streamline software development as well as operations and save time. Throughout this internship, I acquired hands-on experience with real practices and learned how companies efficiently employ programs and automate their business processes in order to maintain them extremely reliable and operational.

Docker was one of the main technologies that I learned throughout this internship. I used Docker to run applications inside containers. With this, I realized how Docker enables applications to run in a deterministic manner across environments, skipping most of the compatibility issues. I also studied Kubernetes, which taught me how to scale and maintain containerized applications in an efficient manner. This taught me about orchestrating complex systems and how organizations maintain stability by adding more workload.

Also, I gained handy experience deploying CI/CD (Continuous Integration and Continuous Deployment) pipelines using Jenkins. Through this, I witnessed first-hand how automation accelerates software release and minimizes errors in the deployment process. Also, I gained experience with OpenShift, an enterprise Kubernetes platform, which further added to my skill set for deploying and maintaining applications on live production environments. Developing with OpenShift gave me practical knowledge of how enterprises handle live deployments without sacrificing security, reliability, and scalability.

Apart from mastering all these basic DevOps tools, I also learned through a project involving artificial intelligence. Through the project, I created an AI-based solution that calculates the most appropriate configuration for systems automatically. Through this, my solution was intended to minimize manual effort, eliminate much of the intervention of human factors, and enhance overall efficiency in operation. This experience further cemented my understanding of how AI can be integrated into DevOps operations to promote innovation and automation in system maintenance.

In addition to gaining experience with several DevOps tools and technologies, this internship greatly improved my problem-solving abilities, adaptability, and understanding of DevOps culture. I realized how collaboration, continuous improvement, and automation are the pillars of modern software development and operations. Overall, this internship was extremely beneficial to my studies and professional life. The experience and theoretical understanding I have obtained will certainly contribute to my future professional growth, research work, and technological contributions.

Contents

1. Company and Sector

Figure 1.1
Turkish Tecnology's logo
Address: Halkalı Merkez Mah. Basın Ekspres Cad. No:3 Küçükçekmece / İstanbul

1.1 Overview of the Company and Sector

Turkish Airlines Technology and Informatics Inc. is a technology company that belongs 100% to Turkish Airlines. The company was founded in 2021 to support the digital transformation of the airline industry. Its main goal is to create new technology solutions that make air travel easier, safer, and more efficient. The company works in many areas such as software development, IT infrastructure, data analytics, artificial intelligence, machine learning, cybersecurity, and financial technology solutions.

One of its important projects is TK Wallet, which helps passengers make faster payments and receive quick refunds. It also develops biometric systems that allow passengers to pass through airport checkpoints without contact, making travel faster and safer. Another big project is the TROYA Passenger Service System, which is designed to improve all parts of the passenger journey according to international standards.

The company uses advanced tools like Red Hat OpenShift AI to build and run artificial intelligence models. With Artificial intelligence, it can save time, improve flight planning, and reduce costs. For example, thanks to Artificial intelligence and optimization systems, the company has saved around 10 million dollars in fuel and operations.

Innovation is a very important part of Turkish Airlines Technology and Informatics Inc. The company supports new ideas from both employees and passengers. Through platforms like IDEAPORT (for employees) and IDEALIST (for passengers), many creative solutions are collected and turned into real projects. It also works with startups through its terminal program, helping new businesses grow while benefiting from their innovative ideas.

In conclusion, Turkish Technology is not only a technology company for Turkish Airlines but also a leader in aviation technology. It continues to work on new digital solutions, improve the travel experience, and increase efficiency. With its focus on artificial intelligence,

innovation, and collaboration, the company is expected to play an important role in the future of global aviation. (Turkish Airlines Technology and Informatics Inc., 2024)

1.2 Organization of the Company

Turkish Technology has a large and well-structured organization to service its large portfolio of technological endeavors. The company has about 3500 engineers who are specialized in a number of technical areas such as software development, artificial intelligence, data analytics, and cybersecurity. The engineers are distributed in 165 departments with specific tasks and projects being given to them.

The departments are under some directorates, which schedule the work of various teams and facilitate smooth operations of projects. Over them exist group directorates, which take care of the larger strategic goals of the company and distribute resources among various directorates. At the top-level management, the company has a General Manager, who beholds all of the company's operation and growth. He reports to the Chairman, who provides overall guidance and ensures that the operations of the company tie in with the long-term strategy of Turkish Airlines.

This organization in several levels helps Turkish Technology manage complicated technology projects and promote innovation effectively. With a team of engineers and separate specialized departments, Turkish Technology can handle different areas at once and easily respond to new arising issues in the aviation technology sector.

TEAM:

Murat Yazar: DevOps Solutions Manager
Muhammed Metin Uluyardımcı: Expert DevOps Engineer
Muhammed Talha Şahin: Senior DevOps Engineer
Cemal İhsan Sofuoğlu: Senior DevOps Engineer
Ahmet Said Oyanık: Associate DevOps Engineer
Rengin Keten: Associate DevOps Engineer
Selçuk Kınalı: Junior DevOps Engineer

a. Production/Service System

Turkish Technology is a technology company focused on designing and deploying digital solutions for the aviation industry. The company adopts an Agile development approach that enables fast response to customer needs. Regular feedback from aircraft maintenance experts, operations managers, and IT specialists is integrated into the development process to guarantee continuous improvement.

At its core service infrastructure lies advanced software and data analysis infrastructure aimed at enhancing maintenance processes, simplifying component tracking, and enhancing overall operating efficiency.

Three stages constitute the company's service system:

Data Collection and Analysis:

Data are gathered from aircraft maintenance processes, flight operational history, and component usage reports. These datasets are analyzed for computing trends in system performance as well as maintenance requirements.

Algorithm and Software Development:
Based on the data that was processed, software modules are developed for managing optimal maintenance planning. Artificial intelligence and machine learning-based algorithms are utilized to predict failures and ease maintenance procedures.

Personalized Solution Delivery:
The systems developed are installed in the operations of the airlines. User-friendly interfaces enable the maintenance teams to monitor processes in real time, and tailored reporting modules and alert systems provide tailored solutions for each client.

By this process, Turkish Technology reduces the cost of operation and minimizes downtime in flight, thereby making airline operations more efficient and dependable.

b. Professional and Ethical Responsibilities of Engineers

In Turkish Technology, DevOps engineering teams are responsible for designing, deploying, and maintaining software systems supporting aviation operations to the highest levels of reliability, security, and efficiency. Their core professional tasks are:

The provision of reliable and stable deployment pipeline and minimizing downtime on software upgrade and new feature release.

Ensuring system security and regulatory compliance with aviation rules, cybersecurity best practices, and international standards.

Harmonization of CI/CD processes for accelerated delivery cycles without compromising the quality or safety of software.

Principles of Ethical Behavior applied by the engineering teams include:

Transparency, in disclosing deployment processes, system updates, and potential operating impacts to all stakeholders.

Inclusivity, through engineering deployment processes and tool sets that facilitate diverse teams within the organization, regardless of technical proficiency.

Responsibility for operation and customer data management, whereby all data is transmitted, stored, and processed securely and in accordance.

With such practices, Turkish Technology's DevOps teams have an efficient, secure, and collaborative DevOps environment, which enables the delivery of quality technological solutions that live up to the stringent operating requirements of the aviation sector.

c. Future of the Sector / Contemporary Issues

I conducted an interview with my supervisor from the Turkish Technology office. He has been in aviation software for over a decade. He described that Turkish Technology

specializes in digital solutions for aviation. The company builds systems for aircraft maintenance, flight operations and data analysis. There are approximately 500 employees today and most of them are engineers and IT experts.

He clarified to me that the aviation sector is a bit different from other sectors like e-commerce or social media. Safety and reliability come first in aviation. Any update in software must be tested dozens of times before it comes to real airplanes. It retards the work compared to other sectors, but it must be done. Conversely, an online store can launch a new version of their site several times a day with minimal risk.

My supervisor also spoke about the future issues. One of the significant issues is having to manage and process large data from airplane sensors. Planes produce enormous amounts of data on every flight, and the data must be stored and analysed in a timely fashion. Another issue is that there is a shortage of engineers who know both aviation regulations and modern software practices.

Looking ahead, he believes that emerging technologies will reshape the industry over the next five to ten years. Artificial intelligence and big data analytics will forecast maintenance problems before they happen, which can be cost-effective and improve safety. Internet of Things will allow for around-the-clock monitoring of aircraft components. He also thinks 3D printing can be used for some spare parts to reduce waiting time.

He compared aviation to the automotive industry. Cars already use many smart systems and can update software over the air. Aviation will go down the same road but more slowly because of strict regulations.

I gained from this interview an understanding that the future of aviation technology is very bright. Turkish Technology will need to adopt AI, IoT and other new technologies, but always with careful consideration to safety and quality. For myself as a student, it shows how DevOps principles can be applied even in a life-critical field like aviation, but with special care compared to faster-moving industries.

2. Summer Practice Description

I learned a lot about DevOps and modern software development during my summer internship at Turkish Technology. My primary computer engineering tasks were to study continuous integration and continuous delivery (CI/CD) principles and how to build a pipeline. I observed how the company creates automated workflows to build, test, and deploy applications.

I learned the company's own applications and observed how a pipeline is designed and executed. This explained to me the sense of linking various steps like code building, testing, and deploying to production without manual intervention.

I also learned how to apply critical tools in the DevOps world: Docker, Kubernetes, and OpenShift. Docker is used to build containers such that applications deploy the same across all environments. Kubernetes and OpenShift are used to orchestrate these containers, scale them, and run them reliably.

This work is directly related with my area of specialty in Computer Engineering. During university I was instructed basic programming and software engineering, yet through work in the company I gained to know how these technologies are used in real projects. I learned about how DevOps operations reduce the development time and make it more secure.

The task was interesting and demanding. It was exciting to see how large teams collaborate and how automation removes human mistakes. At first, it was difficult to keep up with all the new tools and commands, but my supervisor and coworkers helped me and I learned step by step.

By this practice, I gained technical knowledge on CI/CD and container technologies and improved my problem-solving ability. I also understood the importance of collaboration and communication in a working environment. Throughout this summer practice, I gained confidence and motivation to study DevOps more in my career life.

- **Activity Analysis**

One activity that was important to me was to review the code that is generated by other teams prior to being transmitted to the test environment. Our goal was to make sure that the code does not contain any concealed configuration data or hard-coded credentials. Usually, when the developers push their code to Bitbucket, a trigger fires off a process that is used to deploy the code into the test environment. Before that process, our DevOps team reviews the code for security issues. With some help from another intern and working with the team, I helped put in place a pipeline that would automatically identify possible hard-coded credentials or configuration information. We used a tool called TK-GPT to aid in the detection and to re-scan the code before proceeding with it.

We evaluated the performance of this pipeline by checking the number of risky files it could detect and how fast it completed scanning. For example, we tracked the number of repositories scanned within a day and how many problems were flagged. High detection rate and minimal false alarms were essential for excellent performance.

When we were verifying the data, we noticed some problems. Sometimes the pipeline reported false positives where the code contained words that looked like passwords but were not credentials. Scanning extremely big repositories may also take more time than expected. These issues show that we should improve the detection rules and maybe adjust the scanning procedure.
This project closely relates to my Computer Engineering degree as it encapsulates programming, security, and automation. It was difficult and rewarding to create a system that will secure sensitive information before it can be sent to the testing environment. I learned how DevOps practices can directly make software more secure and of better quality in a professional setting through this project.

a. Impact

Engineering solutions are able to reach the world in many ways and even a minimal change in a company can have far-reaching implications. At my internship, I helped to create a team that produced an automated software security scan before code was allowed into the test environment. This is an example of how engineering affects global, economic, environmental and societal aspects all at once. Globally, modern software is utilized by nations and industries like aviation, banking and healthcare. Better security in one company safeguards data that may be shared around the world and can encourage similar action in other companies. From a business point of view, leakage prevention of data and service failures cost companies billions of dollars, legal penalties and reputations. An automated check system also reduces the level of manual checks, lowering the operational cost and enabling engineers to focus on something else. From an environmental point of view, although the activity is computer software, automation leads to fewer paper reports, fewer unnecessary meetings and improved use of servers, saving power and carbon footprint of IT activities. To society at large, protection of individual information and online service security promotes public confidence in technology; in areas like aviation, where dependability and security are of most importance, secure software procedures also protect people's lives and personal data. This experience illustrates that engineering activity in security and DevOps is not only a technical endeavor but also a social responsibility, and that even a focused software pipeline can have global, economic, environmental and societal implications.

b. Team Work

The DevOps group was in daily meeting for the daily sprint meeting where every person described what he did and what was coming next in short. These short meetings kept everything organized and allowed problems to be recognized early.

One such special event was a retrospective session. Members of the team met physically and openly shared good and bad points of the last sprint. Everybody had a chance to share good stories, little complaints or improvement suggestions. This open talk created trust and enabled the team to prepare more effectively for the upcoming cycle.

In the security-check pipeline project, the working team consisted of a senior DevOps engineer, another intern, and myself. The senior engineer was responsible for technical direction, provided best practices, and brushed aside complex obstacles. The other intern and I worked on writing and testing scripts, peer reviewing each other's work and learning new tools.

The team was properly organized and the duties were defined. Planning tasks was left to the senior engineer, there was friendly and open communication, and all the members contributed to decision-making. As a result, work was finished on time and the pipeline produced the expected results.

This project illustrated how good open communication, honest feedback and clearly specified responsibilities can make an engineering project effective and enjoyable, and how effective team culture fosters learning and achievement.

c. Life Long Learning

Professional engineering practice requires a schedule of continuous learning and willingness to embrace new skills. During the internship, I clearly understood that learning at university is just a starting point and that there is always a need to look for new information and learn new technologies.

For successful professional practice, it is vital to acquire additional technical competencies like a thorough understanding of DevOps culture, CI/CD practices and container technologies like Docker, Kubernetes and OpenShift. These tools and practices keep changing rapidly, so the ability to learn them continuously is required. Additionally, good soft skills such as teamwork, communication, problem solving and time management are needed since engineering work is almost always done in teams. Good attitude, curiosity and willingness to ask questions are also applicable attitudes for a professional engineer.

To be able to analyze the activity and complete the project, more information were needed beyond the content that I was exposed to in my classes. As an example, in order to create the pipeline that scans for hardcoded credentials I had to learn about how a CI/CD pipeline operates in general, how you can take advantage of trigger mechanisms in Bitbucket, and how security checks can be automated. I also had to learn about best practices in software security and how DevOps teams integrate these checks before code is sent to a test environment.

I managed to gather this relevant information through various means. One, I interviewed my senior engineers and team members; what they said enlightened me on company practice and real usage. Two, I studied the official manuals of Docker, Kubernetes and OpenShift, which gave me decent technical information and examples. I also went through online blogs and viewed short training clips on CI/CD pipelines and security scanning. Finally, I looked through the company's internal reports and previous work to see how similar problems were handled.

This experience was a lesson to me that lifelong learning is not merely taking classes or reading books. It is an active process of questioning, observing, learning and doing. By combining university expertise with competencies gained in the workplace and by being open-minded to autonomous learning, I am more prepared for a professional engineering career and for the future improvements in technology that will take place.

3. Conclusions

My summer internship with Turkish Airlines Technology and Informatics Inc. provided a full picture of how modern DevOps practices are being adopted in a high-reliability, safety-oriented industry such as aviation. I observed the way the company integrates innovation with strict regulatory and security compliance to offer reliable technology solutions to airlines. Through exposure to tools like Docker, Kubernetes, and OpenShift, I enhanced my technical skills along with learning how CI/CD pipelines simplify development and maintain quality.

Participating in the security-oriented exercise to detect hard-coded credentials opened my eyes to how DevOps is directly involved in protecting sensitive data and ensuring compliance. It also depicted the importance of finding balance between automation and careful human inspection to keep false positives low and continually improve processes.

Overall, this experience not only better prepared me as a technical professional but also taught me the value of teamwork, communication, and ethical responsibility in technology projects on a large scale. I gained more confidence with the use of engineering principles in solving real-world issues and had a clearer sense of how I can contribute to the advancement of aviation technology using secure, efficient, and innovative DevOps practices.

References

Turkish Airlines Technology and Informatics Inc. (2024) *About us*. Available at: https://www.turkishairlines.com (Accessed: 29 September 2025).

Daily Activity Tables

Day 1 – Meeting the DevOps Team

Today was my first day working in the company, and I was both excited and anxious. When I arrived in the morning, I was met by my mentor in the lobby and brought to meet the DevOps team. The office was nicely lit and ventilated, with many desks and big monitors showing dashboards and terminal windows with plenty of text. I didn't know much of what there was on the screens at this stage, but it looked interesting and somehow mysterious.

I interviewed the DevOps group individually. They were extremely friendly and told me what they did. One was doing cloud infrastructure, one automation scripts, and one Kubernetes clusters. They also mentioned that they work very closely with software developers on a regular basis and sometimes work with security teams.

My mentor introduced me to what I will be working on during my internship briefly in the afternoon. The plan is to learn the basics of DevOps first, take courses on Docker and Kubernetes, next learn CI/CD concepts, and finally integrate an AI-based project to scan repository sites for sensitive information. I was relieved because this is exactly what I was anticipating and looking forward to - hands-on learning.

At night, I had a small coffee break with my team. They told me that communication within DevOps is equally important as technical skills. This made me think that I have to pay attention not just to tools but also to how people work. I left the workplace motivated for the next day.

Date	Supervisor's Name	Signature
01.07.2025	Muhammed Metin Uluyardımcı	

Day 2 – Introduction to DevOps

Today was my first ever daily stand-up meeting. The session was short but effective. Everyone in the team spoke about what they had done the day before, what they are going to do today, and if they had any problems. I discovered that this style of meeting is part of the Agile approach. It makes the team aware of everything and solves problems in good time.

After meeting, my mentor gave me an introduction document to DevOps. I spent most of the morning reading and took notes. I learned that DevOps is not a tool or technology—it is a culture and way of thinking. It is the unification of "Development" and "Operations" to deploy software faster, more consistently, and with fewer bugs.

I also learned a few of the essential DevOps practices: continuous integration, continuous delivery, infrastructure as code, automated testing, and monitoring. These concepts sounded unfamiliar to me, but I realized their significance. For instance, if a company can test and deploy their applications automatically, they can release updates much quicker compared to a team that does everything manually.

During the afternoon session, my mentor showed me some real-life scenarios. On his monitor, I saw a pipeline automatically outputting code once a developer had committed it to GitHub. It was like magic—no deployment and compilation by hand, only automation. I started thinking about how such pipelines could make a team's life easier.

By the end of the day, I was convinced. Yesterday I had heard of the word "DevOps"; today I know that it is a collection of culture, practices, and tools that help to deliver software in a better and faster way.

Date	Supervisor's Name	Signature
02.07.2025	Muhammed Metin Uluyardımcı	

Day 3 – Deeper into DevOps Concepts

My mentor held a one-on-one learning session with me today. We sat in a meeting table, and he explained how DevOps changes the way businesses function. In the older models, developers and operations people are separated. Developers write code and "throw it over the wall" to operations, who then deploy it. This typically creates problems because the two groups don't always understand the difficulties of the other. DevOps gets around this by having them work together from the start.

He also spoke about how automation is necessary. Without automation, teams spend too much time doing repetitive tasks like copying files, running tests manually, or restarting servers. Automation tools make these tasks happen by themselves, typically in a matter of seconds.

Later in the afternoon, I watched a series of brief training videos on DevOps basics. The videos covered subjects that included version control systems (Git), containerization, cloud platforms, and monitoring tools. While it was too much to take in at once, I tried to take notes and focus on the major ideas.

Before leaving the office, I helped my mentor with a small task: checking a log file to see if a script ran successfully. It was a small contribution, but it made me feel like I was already part of the flow.

Date	Supervisor's Name	Signature
03.07.2025	Muhammed Metin Uluyardımcı	

Day 4 – Discovering Containers

Today I took my first deep dive into Docker. I had previously simply known that DevOps engineers required it but wasn't exactly sure why. My mentor described containers as small, portable boxes with all an application needs to run—its code, libraries, and even configurations. They can be moved anywhere and operate the same way no matter the computer or operating system unlike traditional installations.

I had Docker running on my laptop in the morning. It was as if I opened a window to another universe. My mentor showed me how containers are so much lighter in comparison to virtual machines and can be started in seconds. I was surprised that one laptop could have several different environments running at the same time without the laptop slowing down too much.

By the end of the day, I was already thinking about the idea of "images," which are basically blueprints for containers. They can be downloaded from online repositories or created manually. It made me realize that with Docker, you don't spend hours installing tools—you simply use an image that's already ready to rock. At the end of the day, I was left wondering how these containers communicate and interact with one another.

Date	Supervisor's Name	Signature
04.07.2025	Muhammed Metin Uluyardımcı	

Day 5 – Seeing the Bigger Picture of Docker

This morning, I learned more about where containers are involved in the broader DevOps universe. Previously, the developers would always say to us, "It works on my machine," and that was never a guarantee that it would work somewhere else. With containers, that is practically eliminated because everything the app requires is encapsulated inside.

My mentor defined Docker images as snapshots of an application—ready to be "brought to life" at any given time. Running, they are containers that are isolated from each other and from the underlying operating system. This is important because it shields applications from messing with other applications.

We also talked about volumes, which enable containers to hold and exchange information even if they have been closed. This was a new concept to me, and it caused me to consider how actual applications, such as databases, must secure their data.

By evening, I understood Docker is not a tool—Docker is a philosophy of thinking of software as being a self-contained, portable package. It's faster to deploy and easier to manage.

Date	Supervisor's Name	Signature
07.07.2025	Muhammed Metin Uluyardımcı	

Day 6 – Building My First Custom Environment

Today was creativity day. I learned how to create my own "recipe" for a container so that it starts off exactly the way I require. My mentor told me that this is one of the most helpful abilities in DevOps: creating environments that are consistent and reproducible for everyone on the team.

We discussed how operating systems handle containers. Containers, in contrast to virtual machines, share the kernel of the host system and are thus lighter in weight. This enables me to run many containers on one machine without squandering too many resources.

The idea of building my own image was interesting. It was similar to designing a small workspace, filling it with what I need, and shutting the door so nothing else could enter. Having viewed it up and running, it was sort of magical—like creating a mini computer inside my computer.

By the end of the day, it was simple to visualize how containers revolutionize the manner in which teams deploy and build software. They render it faster, cleaner, and much more dependable.

Date	Supervisor's Name	Signature
08.07.2025	Muhammed Metin Uluyardımcı	

Day 7 – Orchestrating with Docker Compose

Today I discovered that sometimes one container is not enough. Real-world applications are composed of many pieces a web server, a database, maybe even a cache system. My mentor introduced me to a tool that can create and manage all of these containers in one command, Docker Compose.

In the morning, I saw how this utility can describe an entire multi-container configuration in one file. It was like copying down a recipe, but instead of one dish, you prepare a full meal made up of different courses. With a single effortless command, all the containers can be spun up and networked with each other as if they were all in the same kitchen.

We talked about why this is beneficial for development and testing. Instead of booting up each piece manually, you can boot up the entire setup in seconds. By the afternoon, I was grateful to see how Docker Compose increases efficiency and eliminates mistakes, especially when working with multiple people. It was my initial exposure to "orchestration," the idea of managing lots of moving pieces simultaneously.

Date	Supervisor's Name	Signature
09.07.2025	Muhammed Metin Uluyardımcı	

Day 8 – First Impressions of Kubernetes

Today was just the start of something much bigger—Kubernetes. My mentor called it "the conductor of the container orchestra." Whereas Docker runs and controls individual containers, Kubernetes manages multiple of them across multiple machines so that all goes well.

I discovered clusters in the morning, those being groups of machines (or nodes) that can be operated together. Inside these clusters, Kubernetes runs "pods," which might be said to be small groups of containers. It decides where a pod should live, restarts them when they fail, and even duplicates them if the app needs to serve more users.

At first, it was intimidating. The diagrams seemed complicated, and there were so many new acronyms—namespaces, deployments, services. But I was reminded by my mentor that Kubernetes is like learning a new city. At first, you know the main streets, but eventually, you start learning the smaller streets.

At the end of the day, I came to the conclusion that Kubernetes is not so much about executing containers as it is about keeping them healthy, networked, and ready to scale as necessary.

Date	Supervisor's Name	Signature
10.07.2025	Muhammed Metin Uluyardımcı	

Day 9

Today I was able to witness firsthand how Kubernetes makes applications dependable. My mentor explained to me that pods are the most basic Kubernetes has, each housing a single container or multiple ones. When a container in a pod crashes, Kubernetes does not panic—it simply starts a new one.

We discussed why this matters so much in the real world. If a site crashes, users can leave and never come back. Kubernetes prevents this from occurring by checking on every pod like an overbearing parent. It can even replace pods with new ones without shutting down the whole system, so users won't even notice there was a disruption.

During the afternoon session, I saw a demonstration of Kubernetes deploying pods to several machines in the cluster. This makes sure that if any machine goes down, the application remains operational on others. It was nearly a safety net for containers, and they would never be shut down.

By the end of the day, I was more at ease with the underpinnings, though I still have a lot to learn. Kubernetes felt like a complex but powerful friend I was only just getting to know.

Date	Supervisor's Name	Signature
11.07.2025	Muhammed Metin Uluyardımcı	

Day 10 – Services and Networking in Kubernetes

My mentor told me early this morning that we would be covering how containers in Kubernetes can find and talk to each other. He explained that Kubernetes pods can be moved from one node to another, and their IP addresses could change as well. This complicates the ability to access a pod directly. That is why Kubernetes has something known as a "Service." A Service is like a constant point of contact. Even though the pod changes, the Service stays the same, so communication never gets interrupted.

I learned different Service types. "ClusterIP" is used for communication within the cluster, "NodePort" publishes the application outside the cluster through a port on the node, and "LoadBalancer" auto-provisions an external IP so people can access the application through the internet. It was similar to learning how to offer addresses to houses inside a city so people know where to go.

My mentor performed an example in the afternoon. He used two pods with a small web application. Afterward, he deleted one of the pods on purpose. I was thinking that the application would crash, but it didn't. The Service forwarded the traffic to the second pod without any issues. This showed me how Kubernetes was designed to be reliable.

At the end of the day, I felt more confident. Networking in Kubernetes was still convoluted, but I started to understand the "why" of it. It's not just plugging containers together—it's about making applications endure no matter what happens in the background.

Date	Supervisor's Name	Signature
16.07.2025	Muhammed Metin Uluyardımcı	

Day 11 – Deployments and Scaling in Kubernetes

Today was more about applications in a smarter way. My mentor demonstrated Kubernetes Deployments for me. A Deployment is like a manager that maintains the proper number of running pods and that they should be running with the proper version of the application.

In the morning, we rolled out a sample web application and formed a Deployment. It originally had two pods. We later scaled it to five pods using a single command. I was capable of seeing the extra pods being spun up in mere seconds. My mentor explained that this is called "horizontal scaling." It is used when users are greater and the application needs more resources to handle the traffic.

We also learned to roll out updates. Instead of restarting all the pods at the same time, Kubernetes replaces them one by one. This is an "update rolling" and guarantees that the application is never unavailable. Late in the afternoon, we practiced it by changing the background color of the web application. I committed the new code, and the Deployment rolled out the update to each pod until they all showed the new color—not a moment of downtime.

This day showed me that Kubernetes is more than just a tool to run containers, but it's also a way to run them in an optimal fashion, with the minimum amount of impact to users.

Date	Supervisor's Name	Signature
17.07.2025	Muhammed Metin Uluyardımcı	

Day 12 – Introduction to CI/CD

I started learning about CI/CD—Continuous Integration and Continuous Deployment today. I was informed by my mentor that it is one of the major principles in DevOps because it automates software delivery.

He explained Continuous Integration this morning. I.e., the developers push their code continuously to a shared repository, and upon each push, an automated build runs that compiles the code and executes tests. If the build or tests fail, the developers fix it immediately before the problem spirals out of control.

In the afternoon, we had a glimpse of Continuous Deployment. It is where the code that has been tested gets deployed to production or staging automatically without any human interaction. My mentor showed me a sample pipeline in GitLab. It had steps like "build," "test," and "deploy." When a developer pushed the code, the pipeline created a Docker image, tested it, and deployed to Kubernetes. Everything was done automatically without any human being interacting with the server.

I was amazed at how much time and effort CI/CD saves. Instead of doing everything manually and risking mistakes, the system ensures that everything is done in the correct sequence, every time.

Date	Supervisor's Name	Signature
18.07.2025	Muhammed Metin Uluyardımcı	

Day 13 – Building My First CI Pipeline

Today I created my first CI pipeline on GitLab. At first, it was extremely simple—it contained only a single job that printed "Hello World." But it made me happy because I was able to see the job get started, run, and finish successfully.

My mentor explained to me that pipelines are made of "stages" and "jobs." Stages run one after the other, and jobs within a stage can run in parallel. If one job fails, the pipeline stops. This ensures that rotten code never gets to the next step.

During the afternoon, we added a build phase that created a Docker image and pushed it into a local registry. It was sweet to behold my very own image in the registry. I could picture in my mind that one day I would push real application images here and deploy it into Kubernetes.

Ultimately, I realized that CI/CD is like having a factory assembly line. Once you set it up, it will naturally provide you with consistent and quality outputs.

Date	Supervisor's Name	Signature
21.07.2025	Muhammed Metin Uluyardımcı	

Day 14 – Automated Testing in CI/CD

This morning, I learned how to integrate automated tests into my pipeline. My mentor said without tests, automation is risky—you could be rolling out broken code and not even know it.

We started with simple unit tests for a small Python program. I added a "test" stage in the pipeline. The pipeline failed before deploying if the test failed. This safeguard is needed in actual projects since it prevents faults from reaching the user.

We intentionally introduced a bug in the code during the afternoon to see what would occur. The pipeline ran, the test failed, and deployment was prevented. It was pleasant to know that this system would protect the production from bad updates.

Date	Supervisor's Name	Signature
22.07.2025	Muhammed Metin Uluyardımcı	

Day 15 – Continuous Deployment to Kubernetes

It was one of the most fascinating days of my internship. We merged the CI/CD pipeline with Kubernetes. Now, whenever I commit code, it automatically gets built, tested, and deployed to the cluster.

We used kubectl commands in the deploy stage of the pipeline to deploy the new Docker image to the Kubernetes Deployment. I committed a change to the app's HTML, and within a few minutes, I could see the new version running in the browser.

Later in the afternoon, I attempted to make small adjustments to see how fast and stable the system was. Everything went smoothly. That was when I truly understood the power of DevOps automation.

Date	Supervisor's Name	Signature
23.07.2025	Muhammed Metin Uluyardımcı	

Day 16 – Starting the AI Config Project

I began the final and most unique part of my internship today—integrating AI into DevOps for security. My mentor explained how developers sometimes accidentally put passwords, API keys, or tokens into code. This is "hardcoding credentials," and it's problematic since if the code is exposed, attackers can steal this data.

We were to use an AI model to search repositories and detect sensitive information in a click. Morning was spent detailing how the model operates and what amount of data it needs. Afternoon was spent helping to prepare a list of examples: some with real-looking but fake credentials, and others with clean code. These would be used to train and test the AI.
I began the final and most unique part of my internship today—integrating AI into DevOps for security. My mentor explained how developers sometimes accidentally put passwords, API keys, or tokens into code. This is "hardcoding credentials," and it's problematic since if the code is exposed, attackers can steal this data.

We were to use an AI model to search repositories and detect sensitive information in a click. Morning was spent detailing how the model operates and what amount of data it needs. Afternoon was spent helping to prepare a list of examples: some with real-looking but fake credentials, and others with clean code. These would be used to train and test the AI.

Date	Supervisor's Name	Signature
24.07.2025	Muhammed Metin Uluyardımcı	

Day 17 – Data Collection for the AI Project

Today I spent most of the time on collecting and preparing code examples for the AI model. We needed samples of different types of credentials: passwords, API tokens, and database connection strings.

I also learned of "false positives," when the AI thinks there is a secret but it is not one. For example, the word "password" used in a remark might be interpreted as an actual password. We had to introduce such examples so that the AI could be taught to make no mistakes.

I had a much better understanding of how important data is in AI projects by evening. Even the best model would not work without good data.

Date	Supervisor's Name	Signature
25.07.2025	Muhammed Metin Uluyardımcı	

Day 18 – Testing the AI Model

We experimented with a pre-trained secret-detecting AI today. My mentor ran it on a few test repositories, and it identified the spoof passwords we had placed there for testing.

There were some false positives, however. That led to some debate about tuning the model and making it more precise. In the afternoon, I helped make more diverse examples to train on so that the AI would learn better.

It was interesting to see how AI may help in DevOps by incorporating security testing into the development process.

Date	Supervisor's Name	Signature
28.07.2025	Muhammed Metin Uluyardımcı	

Day 19 – Integrating AI into CI/CD

Today morning, I along with my mentor initiated the most important step of the AI security project—embedding the AI scanning process inside the CI/CD pipeline. The goal was simple: every time a developer pushes new code to the repository, the pipeline would scan for hardcoded credentials automatically before deploying the code.

We started by adding a new stage in the GitLab pipeline configuration. We named it "security-scan." In this stage, the AI tool would run and scan all the repository files for sensitive information like passwords, API keys, and tokens. My mentor informed me that this is a critical step because sometimes developers might commit mistakes and accidentally place secrets in their code. With this ongoing check, these mistakes can be caught right away.

Late morning, our first test. I pushed some test code with a dummy API key baked inside it. The pipeline rolled along just as usual: it built the Docker image, ran the automated tests, and deployed to the temporary environment. But just as the deployment was about to finish, the AI scan detected the dummy key and halted the pipeline in its tracks. There was a warning message that appeared in pipeline logs indicating: "Possible hardcoded credential found in app/config.js – Please review and remove."

It was amazing to watch it execute live. It validated the system to perform as intended. During the afternoon, we executed the test with good code to verify the pipeline would execute smoothly without errors if secrets were not found. It processed without a problem.

By day's end, I realized just how powerful this integration was. We had combined DevOps, security, and AI into a single automated pipeline. It was no longer simply about moving applications out the door fast—it was about having them be secure from day one as well. This was my first time seeing such a powerful example of "DevSecOps" in action.

Date	Supervisor's Name	Signature
29.07.2025	Muhammed Metin Uluyardımcı	

Day 20 – Final Testing and Documentation

Today was all about making sure our system worked perfectly and could be maintained in the future. My mentor told me that in professional environments, testing and documentation are just as important as building the system itself.

In the morning, we did a full end-to-end test of the pipeline. We followed the entire process:

1. **Push new code** to the repository.
2. **Pipeline starts** automatically.
3. **Build stage** creates a new Docker image.
4. **Test stage** runs automated unit tests.
5. **Deploy stage** updates the application on Kubernetes.
6. **Security-scan stage** uses the AI tool to check for hardcoded credentials.

We tried multiple scenarios—code with fake credentials, code with no credentials, and code with tricky patterns that might confuse the AI. The system responded correctly every time. If the AI found something, it stopped the pipeline and sent a clear warning. If not, it completed the deployment without any issues.

After confirming that everything was working, I started writing documentation. My mentor said this would help other team members understand how the pipeline works and how to improve it in the future. I wrote about:

- The purpose of each stage in the pipeline.
- How the AI scanning tool works.
- How to train the AI with new data.
- Common problems and how to fix them.

It took me most of the afternoon to complete the documentation, but it was worth it. I knew that when my internship ended, the project would still be useful for the team.

By the end of the day, I felt proud because we had created a complete, secure, and well-documented system that could be used in real production environments.

Date	Supervisor's Name	Signature
30.07.2025	Muhammed Metin Uluyardımcı	

Day 21 – Last Day and Reflection

Today was my final day at Turkish Airlines' DevOps team. I was sad and at the same time happy. Happy due to the fact that I had learned a lot of things in these three weeks, and sad because I would miss working with such a talented and kind-hearted team.

I prepared for my final presentation in the morning. I wanted to explain thoroughly everything that I learned and the effort that I put in. My slides were:

Week 1: Introduction to DevOps, learning about Docker, and creating custom container images.

Week 2: Learning Kubernetes, creating deployments, scaling apps, and creating CI/CD pipelines.

Week 3: The AI security project—data collection, testing the AI model, and implementing it into the pipeline.

As it was presentation time, I was anxious, but the team and my mentor just smiled and assured me. I took them through the process of how the pipeline worked from start to finish, and I showed them a live scan of the AI scanning for a fake credential. The team was impressed and praised me on it being a great addition to their security process.

My mentor offered me feedback after the presentation. He said that I had actually made a significant input to the department and that my greatest assets were my curiosity and willingness to learn. I felt overwhelmingly proud after hearing that.

In the night, I took a while to bid farewell to each member of the team. We talked about funny moments in the office, moments I was mistaken for Kubernetes commands, and the small victories—like when my very first pipeline executed successfully.

As I left the office for the last time, I was surprised at how much I had picked up in three weeks. Initially, I knew very little about DevOps. Now, I had hands-on experience with Docker, Kubernetes, CI/CD, and even using AI to enhance a security process. I was not just leaving with technical skills, but also an awareness of how collaboration, communication, and documentation are important in the real world.

I was grateful for the opportunity, and I promised to keep learning and improving day by day.

Date	Supervisor's Name	Signature
31.07.2025	Muhammed Metin Uluyardımcı	

Internship Documents
CHECKLIST
COMP 200 Engineering Practice
MEF University
2025-2026

For the completeness of your internship report submission, you are held responsible to check the submission of the following items. The marked final checklist should be included to your submitted internship report as the last page.

- ✓ Internship Application and Acceptance Form signed/stamped?
- ✓ Internship Report
- ✓ Student Evaluation Survey
- ✓ Internship Evaluation Form enclosed/sealed
- ✓ Bottom of Internship Report pages signed?
- ✓ Additional material (such as source codes) attached to the submitted report
- ✓ Internship Report bound and plastic-covered?
- ✓ Internship Report and all accompanying material submitted in an envelope.
- ✓ Hereby, I accept liability for the accuracy and integrity of the submitted contents.

Student Name: Ömer BİLBİL

Date: 29.08.2025

Signature