BILKENT UNIVERSITY
ENGINEERING FACULTY
DEPARTMENT OF COMPUTER ENGINEERING

CS 299

SUMMER TRAINING REPORT

Ahmet Eren Gökalp

22302136

Performed at

AlpaTürk

11.08.2025 - 5.09.2025

Table of Contents

1 Introduction 3
2 Company Information 3
    2.1 About the company 3
    2.2 About your department 3
    2.3 About the hardware and software systems 3
    2.4 About your supervisor 4
3 Work Done 4
    3.1 Bug fixing 4
    3.2 Implementing new front-end feature 5
    3.3 Optimizing performance and testing 5
    3.4 API integration and documentation 6
4 Performance and Outcomes 7
    4.1 Solving Complex Engineering Problems 7
    4.2 Recognizing Ethical and Professional Responsibilities 7
    4.3 Making Informed Judgments 8
    4.4 Acquiring New Knowledge by Using Appropriate Learning Strategies 8
    4.5 Applying New Knowledge as Needed 9
    4.6 Awareness About Diversity, Equity, and Inclusion 9
5 Conclusions 9
References 11
Appendices 12

1 Introduction

I completed my summer training at AlpaTürk, a local organization that provides consultancy to local companies while also developing projects of their own. My motivation for choosing AlpaTürk was that the company specializes in full-stack web development and since it is a local organization it provided a collaborative environment where I could interact with experienced engineers.

During my training, I contributed to tasks across the development cycle [1], including debugging issues, implementing new features and optimizing performance. I made sure to deploy my academic knowledge to the fullest while improving my technical skills and understanding how professional engineer teams work on real projects. My contributions have had a direct impact on ongoing projects.

The rest of this report is organized as follows: Part 2 provides detailed information about AlpaTürk, the department I trained in, the technical environment, and my supervisor. Part 3 describes the work I completed during the training in detail. Part 4 discusses the outcomes of the training in terms of engineering performance, problem-solving, professional conduct, and continuous learning. Part 5 concludes the report with my reflections on my training. References and appendices are included at the end.

2 Company Information

2.1 About the company

AlpaTürk is a locally based software and consultancy organization. The company both develops its own projects and counsels partner companies. The company specializes in full-stack web application development, backend system design, and cloud-based deployment. Beyond the technical focus, AlpaTürk places importance on professional collaboration and mentorship.

2.2 About your department

I worked in the Software Development Department during my summer training. Its scope is cloud-based deployment, API integration, backend system design, and full-stack web application development. Project managers, front-end and back-end developers, and software engineers make up the team. I was able to take part in ongoing projects and help with both adding new features and debugging old code.

2.3 About the hardware and software systems

AlpaTürk uses standard workstations with multi-core processors and solid-state drives to manage resource-intensive operations. Developers use laptops like the MacBook [3], Dell XPS [2], or ThinkPad [4] models. Additionally the business uses cloud infrastructure for hosting and deployment. Testing devices like smartphones and tablets are used to verify responsiveness across platforms.

In terms of software the company mostly uses React on the front-end and Node.js and Express. Data is managed with PostgreSQL, MySQL or MongoDB

depending on the project. Collaboration is supported with GitHub. Docker is used for deployment and containerization. Design ideas are sketched with Figma. Testing and quality control use frameworks like Postman.

2.4 About your supervisor

My supervisor was Kürşat Önsay who is a computer engineer that graduated from Atatürk University Department of Computer Engineering in 2015.

Email: info@alpaturk.com.tr
Phone number: +90 532 546 20 10
Address: Atatürk Üniversitesi Teknokent Yerleşkesi A blok No: 105 Yakutiye/Erzurum

3 Work Done

Over the course of my four-week training at AlpaTürk, the work I completed can be categorized into four, each category being in different aspects of the software development process. These included bug fixing, where I identified and resolved issues in existing code; implementing new front-end features, which involved developing user-facing components and connecting them to backend services; optimizing performance and testing, focusing on improving efficiency and ensuring stability through structured validation; and API integration and documentation, where I worked on connecting external services to the application while also preparing internal documentation to support future development. The following subsections provide detailed explanations of each of these areas.

3.1 Bug fixing

I was assigned with looking into and fixing a persistent problem in one of AlpaTürk's web application projects toward the start of my training. User interface elements occasionally displayed either missing or duplicated information as a result of anomalies in API replies, which were identified as the cause of the issue. The team had identified this problem as having a negative effect on user experience, and in order to identify the root cause, a detailed examination of both frontend rendering and backend logic was necessary.

I started my job with a methodical debugging procedure. I examined how the backend services behaved in various scenarios using tools like Postman and console logs. I discovered throughout the process that the API endpoint in charge of obtaining user data was not appropriately accounting for situations in which optional fields were missing. The service generated incomplete responses that the frontend could not reliably understand, rather than returning a standardized null value. The primary cause of some fields being empty or duplicated on the client side was this mismatch.

I identified the root problem and then put in place a solution that dealt with the system's two levels. I changed the response-handling logic on the backend to ensure that all API outputs followed a standard structure, even if certain fields weren't present. I added further checks to the frontend to make sure that null or missing values would be treated properly rather than generating errors. After testing the solution several times in various user scenarios, I was able to confirm that the problem had been resolved.

I finished the assignment by recording the debugging procedure, the root cause that was found, and the steps taken to fix it in the team's internal issue-tracking system. Future developers who might run across similar issues might use this as a clear reference. In addition to increasing the application's correctness and reliability, fixing this bug gave me a better grasp of debugging techniques in a professional software setting.

3.2 Implementing new front-end feature

I was given the task of implementing a new front-end functionality for one of AlpaTürk's ongoing projects during the second week of my training. In order to allow users to modify personal information like their name, email address, and contact data directly through the interface, a user profile editing page has to be created. This feature, which enables regular updates to be performed without backend intervention, was seen to be important in improving usability.

Designing the page's layout in accordance with the project's current requirements was the first stage in the process. In order to maintain consistency and satisfy the unique requirements of this feature, I creatively modified the patterns and conventions that were already in use throughout the application rather than starting from scratch. I sent the team a draft of my design so they could assess it. I moved forward with the implementation step after getting approval.

Turning the design into responsive React components was the first step in the implementation process. I made sure the functionality operated on all devices while maintaining the application's overall design language by paying careful attention to positioning, typography, and responsiveness. After that, I connected the form to the backend API that managed the storing of user data. Error-handling features were introduced to give users unambiguous feedback in the event of invalid inputs or server issues, and input validation was set up to avoid wrong entries. I thoroughly tested the feature to make sure it was reliable by creating a variety of user scenarios, such as valid updates, incomplete submissions, and incorrect entries. To provide consumers with quick and understandable feedback, success and problem messages were also incorporated. I learned a lot about how to manage the transition from design approval to deployment, apply current project standards creatively, and match design with implementation through this process.

3.3 Optimizing performance and testing

I worked on increasing the effectiveness of a data-handling component that depended on a previously written sloppy sorting function [5] during the third week of my training. This approach was sufficient for small datasets, but as the volume of data expanded, it resulted in noticeable slowdowns that decreased the application's responsiveness. During that week, identifying and fixing this performance issue became my primary objective.

In order to solve the issue, I substituted a more effective quicksort [6] method for the current sorting logic. In order to achieve this, a conventional quicksort implementation has to be modified to fit the unique data structures of the project while maintaining smooth codebase integration. Faster execution times without losing accuracy or stability were the aim.

I tested the new algorithm's performance once it was implemented to see how well it worked. I created datasets of different sizes, executed the optimized and original versions, and recorded how long they took to execute. I also made graphs comparing the outcomes for various input sizes out of personal curiosity, using the format of performance analysis I had studied in my CS202 course (See Appendices). Although the company did not demand it, I decided to do this in order to gain a better understanding of how the algorithm would function in various scenarios. Although my quick sort implementation was similar in performance with the previous algorithm for small data sizes, it is evident that there is a large improvement for larger data inputs (see Appendix C).

I also performed accuracy testing by looking at edge cases including empty arrays, duplicate values, and inputs that had already been sorted in order to support the performance study. After I modified the code to account for all the edge cases, this verified that the optimization maintained accuracy while also improving speed. I improved my grasp of how theoretical ideas may be implemented in real-world scenarios while also helping to create a more dependable and scalable system by combining algorithmic design with careful testing.

3.4 API integration and documentation

I was tasked with integrating a third-party API into one of AlpaTürk's internal projects during the fourth week of my training. This job was to enable the application to retrieve and show external data in order to expand its capabilities. This necessitated managing errors, formatting results, and guaranteeing seamless interaction with the current system in addition to building a dependable connection with the API.

In order to understand the API's endpoints, authentication procedures, and limitations, I started by reading the documentation [7]. Following a successful application-service connection, I put the required backend functionality in place to retrieve and process the data. I linked the frontend components to the backend connection once it was stable so that users could see the data in an organized and understandable way. The application's ability to gracefully handle problems like invalid requests, network faults, or empty answers was given special consideration.

I created internal documentation to document the integration process along with the implementation after looking up how to actually write internal documentation [9]. The setup instructions, descriptions of the main API endpoints, and an explanation of the application's use of the data were all included in this documentation. In order to shorten onboarding times and avoid reoccurring issues, this was done to give upcoming devs and interns a clear point of reference.

Because it forced me to find a balance between technical accuracy, clarity, and maintainability, this project was really beneficial. When the API was successfully integrated, more functionality was added to the program, and the documentation helped the team share knowledge over time. I now have an improved understanding of the relationship between external services and practical projects, as well as the significance of effective communication in the field of professional software development.

4 Performance and Outcomes

The work I completed during my training helped me grow as an engineer. The experience provided chances to address real-world issues with wider implications, and consistently learn and use new information. In the subsections that follow, I'll go over the results of my training in the context of these standards, considering how the experience related to the knowledge and values I've been gaining at the university.

4.1 Solving Complex Engineering Problems

During my training, increasing the effectiveness of a data-sorting component was the most difficult technical task I faced. I replaced the less effective insertion sort method with a quicksort-based alternative, as detailed in Section 3.3. In addition to the actual coding, this assignment was complicated by the steps involved in determining the issue, considering potential solutions, and justifying my choice in relation to the project's goals. I had to go beyond just "making it work" and think about whether my modifications will actually enhance system performance in a long-term manner.

My understanding of algorithms served as the foundation for the technical aspect, but organizing the testing procedure was the real challenge. I had to choose which input sizes to test, how to measure performance, and how to meaningfully communicate my results. I was able to show scaling behavior and offer evidence of improvement by creating performance graphs with different data sizes. I learned from this phase that in engineering, problem-solving involves more than just coming up with a quicker solution; it also entails demonstrating that the solution functions in various scenarios.

I now have a clearer idea of what makes a "complex engineering problem" thanks to this experience. It forced me to simultaneously apply multiple elements of my academic background, including algorithm design, performance analysis, and scientific reasoning, while also developing my ability to explain why the solution was significant within the system's larger framework.

4.2 Recognizing Ethical and Professional Responsibilities

My understanding of how professional responsibilities and ethics influence day-to-day engineering work has improved during my training at AlpaTürk. I initially considered these responsibilities primarily in terms of obeying the law or avoiding errors, but I have now come to see that they are intrinsic in all phases of growth. For instance, I had to deal with private data like names and emails when developing the user profile function. This helped me realize how crucial data privacy is. I discovered that maintaining professional integrity includes even seemingly insignificant design decisions, such as verifying input fields or avoiding unintentional data exposure.

Collaboration practices were another place where I saw professional accountability. During code reviews, the team placed a strong emphasis on polite discussion, with the goal always being to improve the project rather than criticize specific people. This made it possible for criticism to be freely spoken without discouraging anyone. During my internship, I witnessed directly how maintaining this professionalism fosters collaboration and efficiency.

Additionally, I became more conscious of the long-term effects of taking short cuts. For example, even if it took longer in the short term, I saw that the organization

promoted careful error handling and documentation writing. In addition to being a technical choice, this was also a moral one because leaving behind ambiguous or unstable code would make things more difficult for developers in the future. This taught me that being a professional engineer means more than just completing jobs, it also means making sure that the solutions we make are dependable and maintainable for future generations.

4.3 Making Informed Judgments

One of the important lessons I gained during my training was understanding that technical decisions often extend beyond the immediate solution. For instance, I considered more than just performance when I chose to use a quicksort-based implementation in place of the usual sorting approach. I had to think about if the new solution would ultimately result in higher or lower development expenses, and whether it came with any additional hazards. Despite the seemingly minor optimization, I had to balance the technical advantages of this option against practical issues like readability, dependability, and the work future developers would need to do.

Additionally, I became more conscious of how technical decisions affect the economy and the environment. Less computational overhead results from faster and more effective code, which lowers resource usage and server costs when scaled. Even though a single function has little impact on its own, these optimizations add up to significant savings in large-scale systems. This helped me understand how even seemingly small engineering decisions can have a significant influence on expenses and therefore energy use.

When developing design choices, I took into account the significance of user experience and trust on a larger societal level. For example, when I was working on the profile editing feature, I discovered that providing users with clear feedback and appropriately addressing failures helps to increase system accessibility and trust. Developers might save time by using a sloppy approach, but users would become frustrated and confused. It also helped that my experience with such applications usually frustrates me to the point of not using the application anymore as a user. This helped me realize that making well-informed decisions in engineering takes more than just technical proficiency; it also includes knowing how choices affect how people use technology.

4.4 Acquiring New Knowledge by Using Appropriate Learning Strategies

I frequently came across technologies and procedures during my training that I had never used before, so I had to constantly look up and learn new information. My work on API integration serves as a prime illustration. This was the first time I had to link a real external service to a project. I used official documentation [7] and online tutorials to get ready, dividing the issue into smaller parts like rate restriction, error handling, and authentication. I created an organized learning strategy that created a mix between theory and practice by carrying out direct experiments in the codebase and comparing my findings with the documentation.

Learning through mentorship and observation was another strategy I used. I carefully observed how experienced developers handled issues during code reviews and team conversations. For example, observing how they handled exceptions,

wrote comments, and organized changes provided me with real-world knowledge that is rarely covered in our classes. I asked questions when I ran across problems, which helped me make sure I understood and improved the effectiveness of my conversations with mentors.

4.5 Applying New Knowledge as Needed

I had to immediately apply the knowledge I learned throughout my training to the tasks I was given. The work on API integration was an example of this. I put the backend logic to retrieve data into practice and linked it to the frontend after learning how the API's authentication and endpoints worked. I immediately implemented the error-handling strategies I had learned, making sure the application could handle unsuccessful requests without breaking. This demonstrated to me that freshly acquired knowledge only becomes valuable when it is effectively applied to practical problems.

Another example was when I was establishing a development environment and got to learn Docker. After reading the instructions [9] and testing configurations, I was able to put my limited practical experience with containerization to use by creating a working Dockerfile that made deployment easier. The idea that engineering learning is most successful when swiftly followed by application is confirmed by witnessing the containerized program execute consistently across different machines.

These experiences helped me to understand that putting new information to use is a never-ending cycle that involves determining what is lacking, learning it well, and then putting it into practice right away. In addition to helping the company with its tasks, this helped me gain confidence in my capacity to adjust to new tools and technologies, which would be crucial for my future career.

4.6 Awareness About Diversity, Equity, and Inclusion

I learned more about the value of diversity, equity, and inclusion in a professional engineering environment throughout my training at AlpaTürk. Although the company's small size, I saw that team members had a variety of professional and educational backgrounds. Because each person brought their own experiences and ideas to the table, this diversity of perspectives frequently affected how issues were treated and resolved. For me, it was an obvious illustration of how a team that values different viewpoints can produce better technical results.

The company's emphasis on equity in collaboration caught my attention as well. Junior developers and interns were encouraged to share their thoughts during conversations, and their opinions were given the same weight as those of more experienced engineers. Even though it wasn't part of the original job, the team appreciated my suggestion to use graphs to test performance gains. The need of providing everyone with an equal chance to speak and be heard was made stronger by this.

5 Conclusions

My summer training at AlpaTürk was an important step in enhancing the theoretical foundation I gained at Bilkent University with real-world software

engineering. I helped with integrating an external API, optimizing system performance, developing new front-end functionality, and fixing existing modules. I was able to apply and enhance my understanding of ideas I had studied in university classes on programming, data structures, and software engineering thanks to each of these assignments.

I improved my ability to evaluate complex systems and identify the root cause of technical issues by debugging and performance optimizing. I was able to address these problems methodically, test theories, and evaluate outcomes thanks to my programming and algorithmic training in Bilkent. I got firsthand experience with modern programming tools like React, Node.js, Docker, and AWS while working with APIs and developing new features. These tools help me apply what I've learned in the classroom to the workplace.

Beyond technical abilities, this experience helped my development in areas that are more difficult to develop in a classroom. Using version control systems, I learnt how to work with others in a team and how to carry out my responsibilities with professionalism. Additionally, I grew more conscious of the social and ethical elements of engineering, especially the significance of open cooperation, clear documentation, and data privacy.

In conclusion, I was able to transform my academic knowledge into professional expertise throughout my time at AlpaTürk. It demonstrated how engineering ideas are used in real-world situations, how important it is to keep learning new things, and how Bilkent's focus on teamwork, ethical awareness, and careful evaluation has a significant impact on business. I gained direction and confidence for my future as a computer engineer as a result of this encounter.

References

[1] “Software Development Process.” https://www.geeksforgeeks.org/software-engineering/software-development-process/ [Accessed: Oct. 01, 2025].

[2] “XPS 15 Laptop: Dell XPS Laptop Computers | Dell USA.” https://www.dell.com/en-us/shop/dell-laptops/xps-15-laptop/spd/xps-15-9530-laptop. [Accessed: Oct. 01, 2025].

[3] “MacBook Air - Technical Specifications.” https://www.apple.com/macbook-air/specs/. [Accessed: Oct. 01, 2025].

[4] “PSREF.” https://psref.lenovo.com/Product/ThinkPad/ThinkPad_T16g_Gen_3?tab=spec. [Accessed: Oct. 01, 2025].

[5] “Insertion Sort Algorithm.” https://www.geeksforgeeks.org/dsa/insertion-sort-algorithm/. [Accessed: Oct. 02, 2025].

[6] “Quick Sort.” https://www.geeksforgeeks.org/dsa/quick-sort-algorithm/. [Accessed: Oct. 02, 2025].

[7] “Google Maps Platform | Google Developers.” https://developers.google.com/maps/documentation. [Accessed: Oct. 02, 2025].

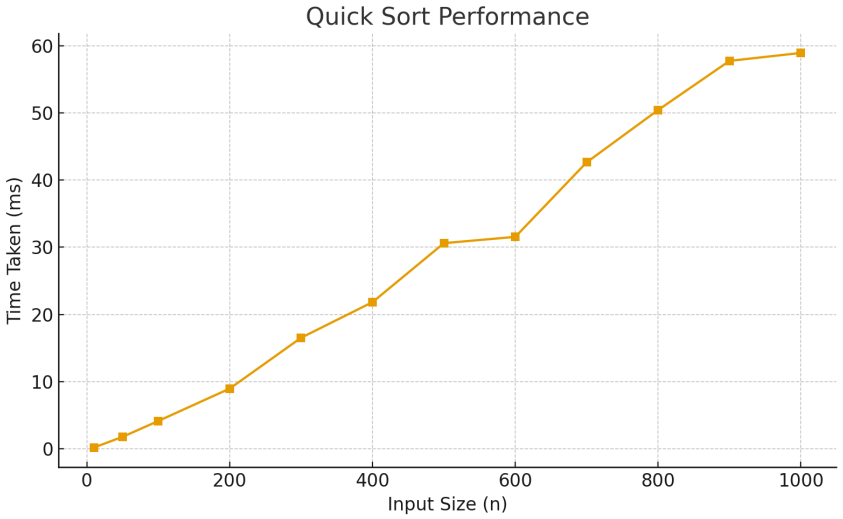
[8] “Internal Documentation: How to Create, Tips & Examples.” https://document360.com/blog/internal-documentation/. [Accessed: Oct. 02, 2025].

[9] “A Docker Tutorial for Beginners.” https://docker-curriculum.com/. [Accessed: Oct. 02, 2025].

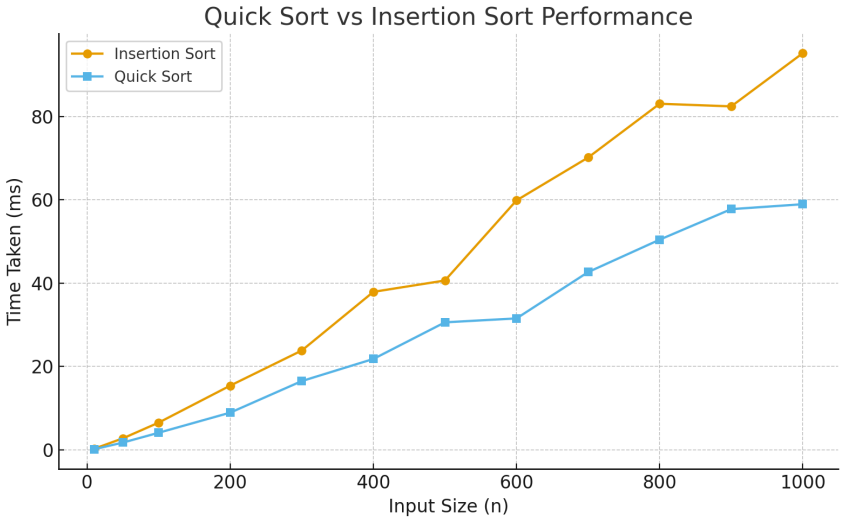
Appendices

Appendix A: Performance of insertion sort

Appendix B: Performance of quick sort



Appendix C: Performance comparison of quick sort and insertion sort



Self-Checklist for Your Report

Please check the items here before submitting your report. This signed checklist should be the final page of your report.

- ☑ Did you provide detailed information about the work you did?
- ☑ Is supervisor information included?
- ☑ Did you use the Report Template to prepare your report, so that it has a cover page, has all sections and subsections specified in the Table of Contents, and uses the required section names?
- ☑ Did you follow the style guidelines?
- ☑ Does your report look professionally written?
- ☑ Does your report include all necessary References, and proper citations to them in the body?
- ☑ Did you remove all explanations from the Report Template, which are marked with yellow color? Did you modify all text marked with green according to your case?

Signature: _____