BILKENT UNIVERSITY

ENGINEERING FACULTY

DEPARTMENT OF COMPUTER ENGINEERING

CS 399

SUMMER TRAINING REPORT

Deniz Yüksel

21600880

Performed at

Sibertek

16.07.2019 – 15.08.2019

Table of Contents

1 Introduction

In the summer of 2019, I had an internship at Sibertek for 20 workdays, between 16.07.2019 and 15.08.2019. Founded in 1999, Sibertek has its office in Ankara Cyberpark. The main reason I applied for Sibertek was because of the significance of secure web applications they produce to other businesses in the finance sector. My supervisor was Abdussamet Ceylan and he assigned me to build a web project on managing suggested proposals. While building the application, I learned about many useful tools that will be mentioned throughout this report.

2 Company Information

2.1 About the company

Sibertek is a company founded in 30.12.1999. The company mostly deals with systems, database, network and web applications specifically for clients. It is a company that has 100 per cent domestic capital. The clients are both from private sector and from government establishments.

2.2 About your department

I was not employed in a specific department but I had my supervisor in the web development department. I was assigned my own project that my supervisor oversaw and we had daily meetings. In essence, I wanted to work on a company project however I was told the projects are too large and setting up a company project on my computer would be heavy. They also told me I would have to spend a lot of time just for understanding the company projects, and they are also confidential.

2.3 About the hardware and software systems

In the company, the employees use Windows 10 operating systems with sufficiently enough requirements. Mainly, the programmers use IntelliJ for IDE, Java and NodeJS for web development [1].

2.4 About your supervisor

My supervisor was Abdussamet Ceylan. He studided Computer Engineering in Kırıkkale University. Then he continued his masters study in Computer Science in Gazi University. He has certifications on web programming, entrepreneurship, and introduction to cybersecurity by CISCO [2]. He has been working in Sibertek since July 2018.

In the company, Mr. Ceylan is currently a back-end web programmer specialized in NodeJS and Java language.

3 Work Done

3.1 A Brief Information About the Project

When I started my internship in the company, I was told the projects were immensely big and too complicated for me to understand. Also they were confidential. Therefore, my supervisor offered me a rather simple project for me that would cover a wide range of web technologies. Briefly, my project was to build a project proposal system that allowed users to publish their project ideas. Then, those ideas would be listed on the main board as card views. The idea format may or may not have a file attached. After the ideas are listed, a user with a project manager role can “lock” an idea, which will disable other project managers or standard users locking it. A locked icon will indicate that the idea is not available for other project managers therefore only one manager can work on the project. If a project manager decides not to work on the project anymore, one can release the lock of the project to make it available to other managers. A project manager and a card therefore has one-to-many relationship. There is also an admin user who is able to delete and unlock the idea cards. Below is a screenshot from the main board of the project.

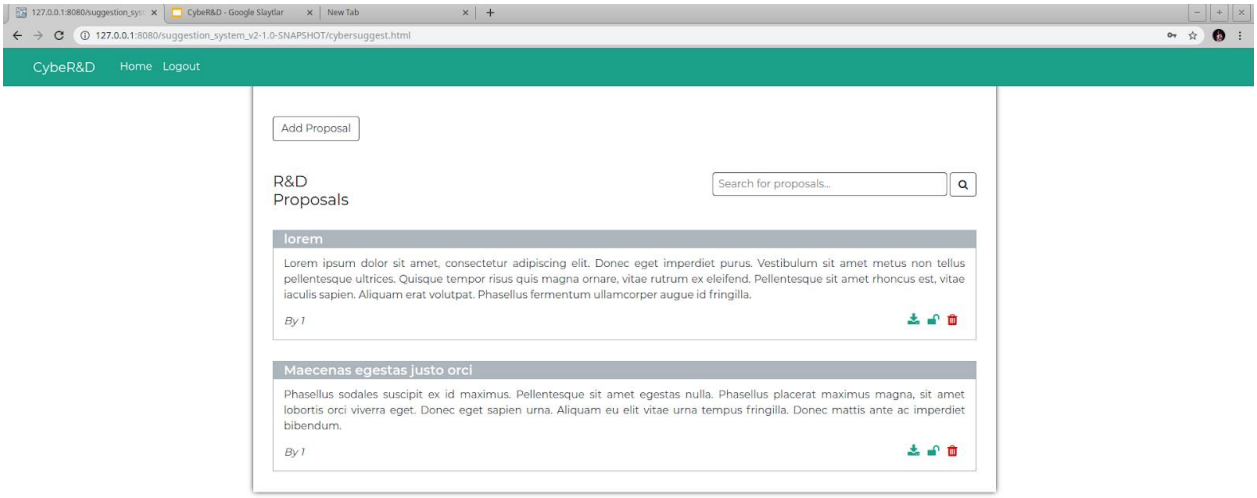

Figure 1

3.2 Setting Up Maven & WildFly Application Server

I already had Xubuntu and NetBeans IDE installed in my machine [3, 4]. I continued to work with my setup which I also managed my dependencies with Maven [5]. The first step I needed to perform was to setup an application server so that I could use localhost to run my web application. I used WildFly Application Server by JBoss [6]. By reading documentations of JBoss and using JBoss Command Line Interface, I managed to integrate WildFly with NetBeans.

3.3 Running a PostgreSQL Instance with Docker

After integrating the WildFly server, my supervisor gave me a task for installing and learning how to use Docker [7]. After reading documentations and learning about docker images and containers, I downloaded the official instance of PostreSQL from Docker hub [8]. I also downloaded a proper JDBC driver for PostreSQL. Then, I added

the JNDI driver name on WildFly console in order to be able to use Java Persistence API (JPA) by Hibernate [9]. I also downloaded PgAdmin III which is a tool to write manual SQL statements to check my tables' consistencies [10]. An example view from PgAdmin III is included as Figure 2 below.

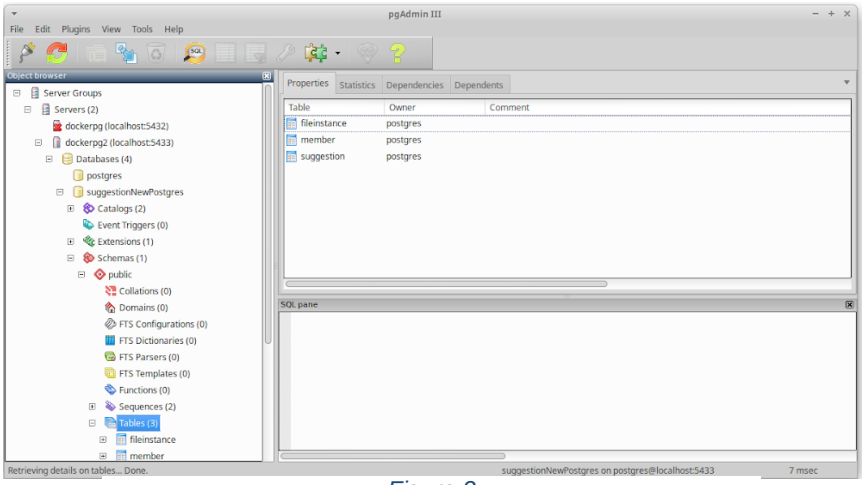

Figure 2

3.4 Learning Java Persistence API (JPA)

My project needed a database connection as I mentioned earlier. However, I did not use any SQL statements while coding. Instead, I used Hibernate's implementation of JPA which had Java methods and annotations that achieved SQL's aims like inserting members to tables and creating tables with appropriate entities and properties. Hibernate creates an xml file that specifies the table creation strategies and connection to WildFly server. An example of a created empty table after running the project is shown in Figure 3. The code required to create the table is the Suggestion class, which is included in appendices section.

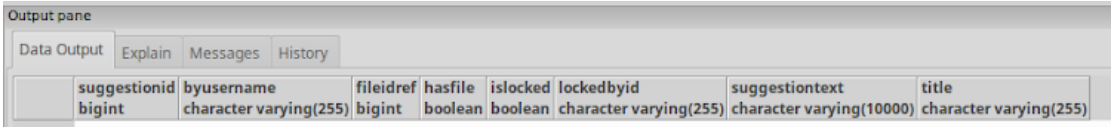

Figure 3

3.5 Building the Project Using Java Servlets & Java Server Pages (JSP)

After I setup the server and the database, my supervisor employed me with the new task of starting the project itself. As I mentioned earlier, I used methods and annotations of JPA in my classes. My first aim was to make a multi-page web application which were consist of several Java Server Pages and [11]. JSP's are html pages that are enhanced with inserting Java code in them. The front-end of the website was implemented with JSP.

For implementing the back-end, I used Java Servlets [12]. Each Java Servlet class handles one type of request. Java Servlets are rather primitive and implementing different servlets for different purposes can end up with many servlet classes. After I implemented the website using Java Servlets and JSP, my supervisor encouraged me to implement the project with a more different and rather advanced way. I started reading about REST API.

3.6 Re-building the Project Using RESTEasy and SwaggerUI

After I started reading, I included the dependency of JaxRs by RESTEasy [13]. I also found a JaxRs activator code which was required in order to use the Rest services. Switching to a different design did not require many changes by means of algorithm. I copied and pasted most of my previous codes by only doing required changes in the syntaxes. Briefly, I moved all my Servlet codes to one service file. However, after adapting the architecture, designing a RESTful single page web application was more practical in a sense that I was able to test my services from a UI console, named Swagger UI [14]. The service code included login, signup, adding, filtering, deleting and locking suggestions. Figure 3 is a screenshot from the Swagger UI console. Figure 4 shows the comparison between the implementations in sections 3.5 and 3.6.

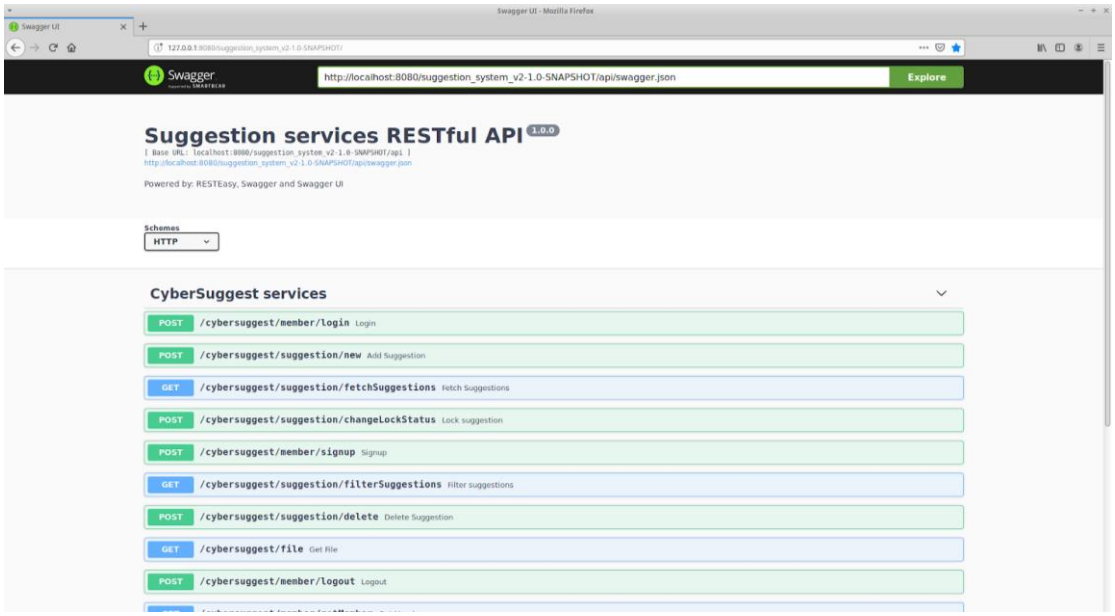

Figure 3

Implementation in Section 3.5	Implementation in Section 3.6
Multi-page application, static (HTML)	Single-page application, dynamic (JS)
Easier to grasp the context	Harder to grasp the context (JSON, JS)
HTTP requests and responses are done directly with Request Dispatcher	HTTP requests and responses are done indirectly with XML HTTP Request
Stateful (user information is stored in the session)	Candidate for Stateless (session map is a variable in service code)

Figure 4

3.7 Designing the Front-end with Bootstrap

The website I built finally needed a proper user interface. Hence, I chose Bootstrap 4 while designing the front-end [15]. I had learned Bootstrap while implementing the front-end page for the term project of CS 353 course. This is why I chose Bootstrap.

3.8 Running an EC2 Machine on Amazon Web Services (AWS)

The web project practically became real after I launched it on Amazon Web Services (AWS) [16]. While doing that, I subscribed for a free membership on AWS and I used docker to create and manage my application. I faced some struggles because of my company's security issues. Therefore, I had to test some features when I arrived to dorm after work.

4 Performance and Outcomes

4.1 Applying Knowledge and Skills Learned at Bilkent

As a Bilkenter, I was in conscience that Bilkent emphasizes several important points. For example, Bilkent emphasizes the importance of brainstorming and planning before coding the project. Therefore, me and my supervisor Mr. Ceylan had daily meetins where we discussed the daily progress and tasks. We planned the project structure together and I was able to find help when necessary. I did not draw any UML diagrams because the project was small and did not needed any communication between other people. However, I inspired from other developers in the company about how they use known solutions to specific problems.

The knowledge I learned in CS 353 course helped me directly. I used my SQL skills while testing the validity of information in my tables. I also used my mother language which is Java while developing the back-end of the project.

4.2 Solving Engineering Problems

There were constantly problems while I was developing the web application in both implementations. This was because it was my first web application. The hardest problem I faced was parsing JSON objects. In essence, parsing JSON objects is trivial with libraries like GSON in my entity classes [17]. However,

4.3 Team Work

Unfortunately, I was not placed in any team therefore I did not experience team work. However, I tried to observe other developers and the communication between them. Also, I realized that communication mistakes between businesses can lead to huge mistakes and re-implementation of some features for projects.

4.4 Multi-Disciplinary Work

In my project, as I mentioned earlier I worked alone. However, as a social person, I had dialogues with the other software engineers about their projects and how crucially important communication is. There was an incident in which I asked a question to a senior software engineer about my relational database model because my supervisor was busy. I tried to ask the questions in my mind to variety of people in the company to compare their viewpoints. Thus, I also managed to meet employees that are not computer engineers. The multi-disciplinary work was a concept that I did not apply, but instead observed.

4.5 Professional and Ethical Issues

As I mentioned earlier, I could not work on a company project therefore I did not learn about professional and ethical issues directly. I was given my own project and I would deliver it whenever I wanted to. In the end, I finished my project and it was successful, so at least I obeyed my principles of work ethics.

4.6 Impact of Engineering Solutions

Honestly, I have to admit that I did not encounter any engineering problems in my project. In my opinion, an engineering problem does not cover the errors that occur while parsing JSON objects. Another common error I faced was when I tried to deploy the .war file I generated for WildFly. Although I tried to observe engineering problems and solutions around, I did not face any serious problem except runtime errors. Some of those runtime errors took me a long time to figure out and some of them just disappeared after a clean build.

4.7 Locating Sources and Self-Learning

My supervisor assigned me coding tasks every day until I finished the project. Also as I advanced further my supervisor gave me tasks about reading documentations e.g. for Docker. I also setup the WildFly application server myself, and I found documentations of it as well. In this internship, I noticed that one does not simply learn

by doing the same things with another person on a video. Instead, I figured out that reading documentations was a stronger but a tougher way to learn.

4.8 Knowledge about Contemporary Issues

I learned that although Java is not the most popular language in web development, there are many enterprises that use Java and it is used by companies in the finance sector.

4.9 Using New Tools and Technologies

I had not built any website on my own before. My first experience was with Java Servlets and JSP. Although they seemed to be a bit archaic, I felt much stronger after I rebuilt the website with a REST implementation and Swagger UI. At the end, I had a bit of an idea on developing a web application with Java.

Amongst the tools I learned, there was also Docker. Docker helped me on two major issues that were running the database (PostgreSQL) and deploying the application on AWS.

5 Conclusions

The most important thing I learned in this company was building a website with a Java on back-end. While doing that, I learned to setup application servers like WildFly. I learned how to create Docker containers and manage them. I learned how to use PostgreSQL with PgAdmin III and most importantly I learned to put all these blocks together to come up with a working web application. Although I did not manage to work on company projects, I was mentored with good care. Overall, this internship covered a lot information on web development as a whole.

References

[1] JetBrains. (2019). *IntelliJ IDEA: The Java IDE for Professional Developers by JetBrains*. [online] Available at: https://www.jetbrains.com/idea/ [Accessed 7 Oct. 2019].

[2] Cisco. (2019). *Cisco – Türkiye*. [online] Available at: https://www.cisco.com/c/tr_tr/index.html [Accessed 7 Oct. 2019].

[3] community, T. (2019). *Xubuntu*. [online] Xubuntu.org. Available at: https://xubuntu.org/ [Accessed 28 Sep. 2019].

[4] Netbeans.org. (2019). *Welcome to NetBeans*. [online] Available at: https://netbeans.org/ [Accessed 28 Sep. 2019].

[5] Porter, B., Zyl, J. and Lamy, O. (2019). *Maven – Welcome to Apache Maven*. [online] Maven.apache.org. Available at: http://maven.apache.org/ [Accessed 28 Sep. 2019].

[6] Wildfly.org. (2019). *WildFly Homepage · WildFly*. [online] Available at: https://wildfly.org/ [Accessed 7 Oct. 2019].

[7] Hub.docker.com. (2019). *Docker Hub*. [online] Available at: https://hub.docker.com/ [Accessed 7 Oct. 2019].

[8] Hub.docker.com. (2019). *Docker Hub*. [online] Available at: https://hub.docker.com/_/postgres [Accessed 7 Oct. 2019].

[9] Docs.oracle.com. (2019). *37 Introduction to the Java Persistence API (Release 7)*. [online] Available at: https://docs.oracle.com/javaee/7/tutorial/persistence-intro.htm [Accessed 7 Oct. 2019].

[10] Page, D. (2019). *Download*. [online] Pgadmin.org. Available at: https://www.pgadmin.org/download/ [Accessed 7 Oct. 2019].

[11] Tutorialspoint.com. (2019). *JSP Tutorial - Tutorialspoint*. [online] Available at: https://www.tutorialspoint.com/jsp/ [Accessed 7 Oct. 2019].

[12] www.javatpoint.com. (2019). *Learn Servlet Tutorial - javatpoint*. [online] Available at: https://www.javatpoint.com/servlet-tutorial [Accessed 7 Oct. 2019].

[13] Resteasy.github.io. (2019). *RESTEasy - JBoss Community*. [online] Available at: https://resteasy.github.io/ [Accessed 7 Oct. 2019].

[14] Swagger.io. (2019). *Swagger UI | API Development Tools | Swagger*. [online] Available at: https://swagger.io/tools/swagger-ui/ [Accessed 7 Oct. 2019].

[15] Mark Otto, a. (2019). *Bootstrap*. [online] Getbootstrap.com. Available at: https://getbootstrap.com [Accessed 7 Oct. 2019].

[16] Amazon Web Services, Inc. (2019). *Amazon Web Services (AWS) - Cloud Computing Services*. [online] Available at: https://aws.amazon.com/ [Accessed 7 Oct. 2019].

[17] GitHub. (2019). *google/gson*. [online] Available at: https://github.com/google/gson [Accessed 7 Oct. 2019].

Appendices

An example of an entity object with JPA annotations:

```
package
tr.net.deniz;

import com.google.gson.Gson;
import com.google.gson.JsonObject;
import javax.persistence.*;

/**
 *
 * @author yukseldeniz
 */
@Entity
public class Member {

    @Id
    private String userName;
    private String email;
    private String password;
    private String name;
    private String surname;
    private String userType;

    public Member() {
    }

    //JSON CONSTRUCTOR
    public Member(String json) {
        Gson gson = new Gson();
        JsonObject jObj = gson.fromJson(json, JsonObject.class);
        this.userName = jObj.get("userName").getAsString();
        this.password = jObj.get("password").getAsString();

        if (jObj.get("email") != null) {
```

```
            this.email = jObj.get("email").getAsString();
        }


        if (jObj.get("name") != null) {
            this.name = jObj.get("name").getAsString();
        }


        if (jObj.get("surname") != null) {
            this.surname = jObj.get("surname").getAsString();
        }


        if (jObj.get("userType") != null) {
            this.userType = jObj.get("userType").getAsString();
        }


    }


    public Member(String userName, String email, String
password, String name, String surname, String userType) {
        this.userName = userName;
        this.email = email;
        this.password = password;
        this.name = name;
        this.surname = surname;
        this.userType = userType;


    }


    public String getName() {
        return name;
    }


    public void setName(String name) {
        this.name = name;
    }


    public String getSurname() {
```

```
        return surname;
    }

    public void setSurname(String surname) {
        this.surname = surname;
    }

    public String getUserName() {
        return userName;
    }

    public void setUserName(String userName) {
        this.userName = userName;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public String getUserType() {
        return userType;
    }

    public void setUserType(String userType) {
        this.userType = userType;
```

```
            }
        }
```

The signup service, as a service example:

```
//
Signup
service
        @POST
        @Produces(MediaType.TEXT_PLAIN)
        @Consumes(MediaType.APPLICATION_FORM_URLENCODED)
        //@Produces(MediaType.MULTIPART_FORM_DATA)
        //@Consumes(MediaType.MULTIPART_FORM_DATA)


        @ApiOperation("Signup")
        @Path("/member/signup")
        public boolean SignUp(
                @FormParam("memberObj") Member member) {


            String userName = member.getUserName();
            String email = member.getEmail();
            String password = member.getPassword();
            String name = member.getName();
            String surname = member.getSurname();
            //String userType = member.getUserType();


            if (("".equals(userName) || userName == null) ||
("".equals(password) || password == null) || ("".equals(email) ||
email == null)
                    || ("".equals(surname) || surname == null) || (name ==
null || "".equals(name))) {
                return false;
            } else {


                try {
                    //Member member = new Member(userName, email,
password, name, surname, userType);
                    Member compared = em.find(Member.class, userName);
                    transaction.begin();
```

```
                if (compared == null) {


                    try {
                        member.setUserType("standart");
                        em.persist(member);
                        transaction.commit();


                        return true;
                    } catch (RollbackException ex) {

Logger.getLogger(SuggestionService.class.getName()).log(Level.SEVERE,
null, ex);
                    } catch (HeuristicMixedException ex) {

Logger.getLogger(SuggestionService.class.getName()).log(Level.SEVERE,
null, ex);
                    } catch (HeuristicRollbackException ex) {

Logger.getLogger(SuggestionService.class.getName()).log(Level.SEVERE,
null, ex);
                    } catch (SecurityException ex) {

Logger.getLogger(SuggestionService.class.getName()).log(Level.SEVERE,
null, ex);
                    } catch (IllegalStateException ex) {

Logger.getLogger(SuggestionService.class.getName()).log(Level.SEVERE,
null, ex);
                    } catch (SystemException ex) {

Logger.getLogger(SuggestionService.class.getName()).log(Level.SEVERE,
null, ex);
                    }
                } else {
                    String memUserName = member.getUserName();
                    String comparedUserName = compared.getUserName();
                    if (!(memUserName.equals(comparedUserName))) {


                        try {
                            em.persist(member);
                            transaction.commit();
```

```
                        return true;
                    } catch (RollbackException ex) {

Logger.getLogger(SuggestionService.class.getName()).log(Level.SEVERE,
null, ex);
                    } catch (HeuristicMixedException ex) {

Logger.getLogger(SuggestionService.class.getName()).log(Level.SEVERE,
null, ex);
                    } catch (HeuristicRollbackException ex) {

Logger.getLogger(SuggestionService.class.getName()).log(Level.SEVERE,
null, ex);
                    } catch (SecurityException ex) {

Logger.getLogger(SuggestionService.class.getName()).log(Level.SEVERE,
null, ex);
                    } catch (IllegalStateException ex) {

Logger.getLogger(SuggestionService.class.getName()).log(Level.SEVERE,
null, ex);
                    } catch (SystemException ex) {

Logger.getLogger(SuggestionService.class.getName()).log(Level.SEVERE,
null, ex);
                    }
                } else {
                    return false;
                }
            }
        } catch (javax.transaction.NotSupportedException ex) {

Logger.getLogger(SuggestionService.class.getName()).log(Level.SEVERE,
null, ex);
        } catch (SystemException ex) {

Logger.getLogger(SuggestionService.class.getName()).log(Level.SEVERE,
null, ex);
        }
    }
    return false;
}
```

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, the 8 major sections and 13 subsections specified in the Table of Contents, and uses the required section names?
- ☐ Did you follow the style guidelines?
- ☐ Does you 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: _____