Bilkent University

Department of Computer Engineering

Senior Design Projects

TechRank

Low Level Design Report

Serhat Hakkı Akdağ, Alperen Ustaömer, Mehmet Oğuz Göçmen, İlhami Kayacan Kaya, Pelin Elbin Günay

Supervisor: Çiğdem Gündüz Demir
Jury Members: Mustafa Özdal and Selim Aksoy

February 18, 2019

This report is submitted to the Department of Computer Engineering of Bilkent University inpartial fulfillment of the requirements of the Senior Design Project course CS491/2.

Table of Contents

Low Level Design Report

TechRank

1 Introduction

Online shopping is one of the rapidly growing trends in the last decade. People think that doing shopping by using websites or shopping apps is easier and less time consuming than doing shopping in the stores. Because, they can search the products they want to buy by using search bar without losing time while finding the products in the store. Customers are surfing on e-commerce websites not only for buying the products online but also, even when people are buying things from stores, they are checking out reviews and comments of products online. In any case, checking out reviews is quite common nowadays, as it is hugely beneficial. However, people do not want to read hundreds of comments due to timing constraints. In addition, it is not always easy to understand general consensus on comments because there are inconsistencies among comments. Because of that, seeing the products' rates that are constituted by analyzing all comments is beneficial for the user. By observing rates about the products, customers can easily decide which product is worth buying.
When we consider the common usage of online shopping, we decided to develop a web application in order to help people to decide and find out which technological product is beneficial and necessary for them by analyzing user comments from trusted e-commerce websites and by rating and ranking them into some related categories. Our application will analyze all comments scattered on web, then it will rank the devices on the same category using criteria. TechRank will also decide on overall rating for the device and overall rating for the producer company of the device. In this report, we give an overview of the low-level design of our system. Design trade-offs, engineering standards and interface documentation guidelines are defined. After that, packages and interfaces are described. Finally, class diagram and components are presented.

1.1. Design Trade-Offs

1.1.1. Functionality vs. Usability

People prefer our system instead of searching product in eBay, Amazon etc. due to saving their time when they are getting general information about product. That's why, both usability and functionality of our system are important for us to be succeeded. Our user interface must be user friendly in order to provide users to spend their time effectively in our system. Unless user can use the system properly, user may prefer to use Amazon or eBay to search the product. On the other hand, our system has much functionality in order to make the system more preferable. Because of that our system must also have functionality. Our main design goal is to keep the balance between functionality and usability while providing maximum usability.

1.1.2. Memory usage vs. Performance

TechRank Web Application is requiring huge amount of memory to be able to store fetched data about products from commonly used web pages since countless different products are exist inside these pages. Therefore, to be able to use this huge memory, performance of the system should be reduced. However, performance is also one of our main purpose inside the system. That's why our prior design purpose should be finding a mid-way that system should be able to provide both of enough performance and memory usage.

1.1.3. Performance vs. Reliability

In our system, we have large dataset which includes huge number of comments that are comes from trusted e-commerce websites. This dataset will also grow day by day while we pull new comments. This means that we use huge amount of memory and it decreases the performance and the response time of the system. On the other hand, while we process the comments, the more comment we process, and the more reliable ranking and rating results we obtain. That's why, in this context, our prior design goal is reliability.

1.2. Interface Documentation Guidelines

Interface documentation is used in the documentation like following.

ClassName: Description of class
Attributes: AccessModifiers Type AttributeName
Methods: AccessModifiers ReturnType methodName(Parameters)

Figure 1 Interface Outlines of Packages

Class names are given in camel case format followed by description of the class.

Attributes are given in the format shown above. AccessModifiers are shown by characters '+', '-' and '#'. Symbols indicates the following:

'+' : Class is public

'-' : Class is private

'#': Class is protected

Methods are given in the format shown in table.

1.3. Engineering Standards

Our reports are written by considering IEEE report format which is an engineering standard. This provides us to make our reports easily understandable. We also used UML (Unified Modeling Language) which is also an engineering standard to visualize our system by forming the diagrams, use cases, scenarios and subsystem decomposition.

1.4. Definitions, Acronyms, and Abbreviations

API: Application Programming Interface.

UI: User Interface.

MVC: Model-view-controller which is an architectural pattern.

SQL: Structured Query Language which is a programming language of database.

NLP: Natural Language Processing

HTTP: Hypertext Transfer Protocol

ACID: It is a property set for transaction in relational database. Properties are atomicity, consistency, isolation, durability

HTML: Hypertext Markup Language

CSS: Cascading Style Sheets

JPA Repository interface: Java Persistence API repository interface which is found in Spring Framework package

DTO: Data Transfer Object

2. Packages

For TechRank, three-tier design pattern is chosen. The main working principle of the system of application is based on client-system. In this system a client side send a request by interacting with the interface, then server side responds this request according to desired data by request. The main reason why three tier architecture is selected since, it supports the workflow in the client-server system while separating server side from the client side by limiting direct interactions between end user and database. Briefly three tier design is the only design pattern that provide sufficient features to system of TechRank. As it can be understood from its name, design choice of TechRank separates and classifies all of the system into 3 main components. Which are:

- Presentation Tier
- Logic Tier
- Data Tier

Each of these tiers has their own qualities and missions inside the system composition. The presentation layer is responsible for providing Graphical User Interface to the clients and listening any action that client is performing on this interface. This tier is also responsible for making REST calls according to the user actions on the front-end and sending HTTP commands to the related endpoint of the server by means of using request-related controller and service modules that define the request endpoints. Logic tier is responsible for listening for any action on the server by the use of predetermined endpoints. Controllers and services found in this subsystem are responsible for handling the logic behind the REST calls (such as converting RequestDTO's to Entity classes, navigating requests to related controllers and services in order to communicate with the database). Data tier is responsible for communicating with database and retrieving data required for logic tier to process and return back to the presentation tier.

In this process, data tier uses repositories which communicate with databases using the request-related entities.

2.1. Presentation Tier

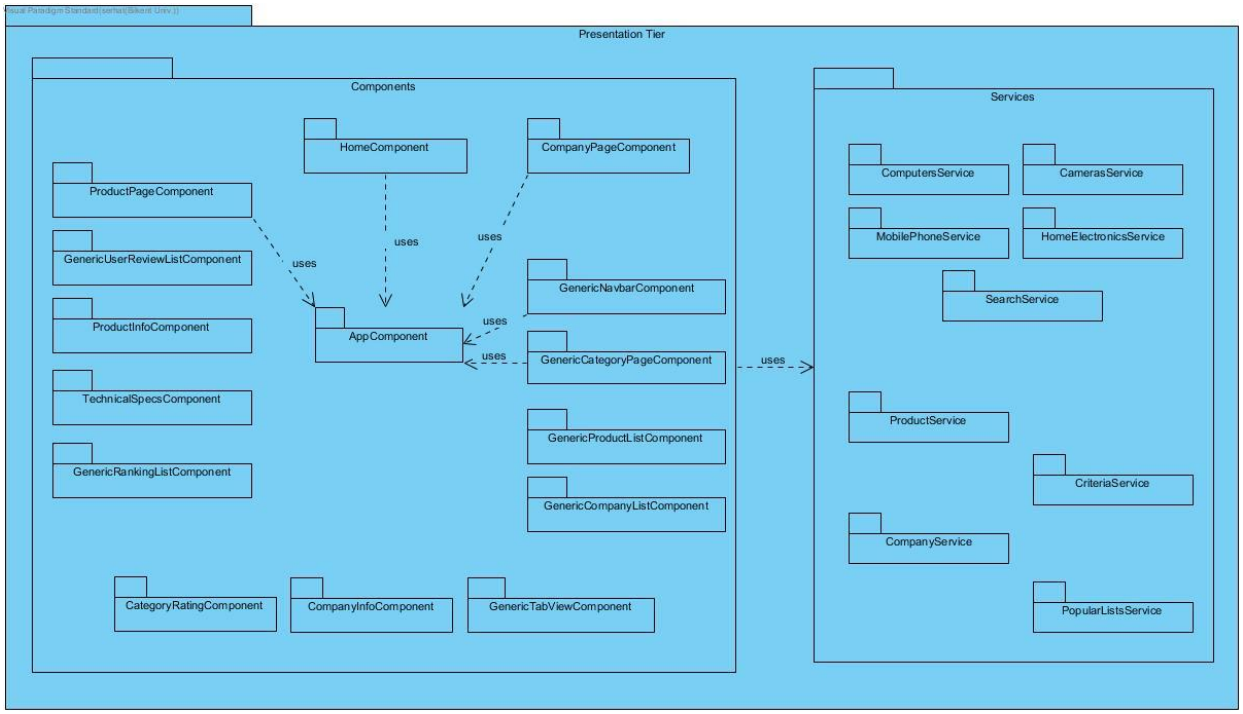

Figure 2 Presentation Tier Diagram

Note: Generic components have multiple inheritances with the other component. Because of heavy load on the diagram they are not shown on the diagram.

Presentation tier is responsible for managing interactions between the end-user and the user interface of the application
Logic tier is consisting of 2 different parts which are:

- Components
- Services

2.1.1. Components

This package inside presentation tier includes HTML and CSS files that will be presented to the client. It also creates a pattern for the design of front end.
AppComponent: AppComponent class is basically a design part of the TechRank web page which will be common for all of the pages inside the web application. It will contain web page header.
HomeComponent: HomeComponent is the class that is designed for UI design of the main (entering) page of the webpage. This component will contain popular ranking lists inside it which will be created by using other components.
ProductPageComponent: ProductPageComponent is responsible from design of the product pages. This component will contain all of the technicalSpecsComponent,

ProductInfoComponent, GenericUserReviewListComponent, GenericRankingListComponents

CompanyPageComponent: CompanyPageComponent is responsible from design of the company pages. This component will contain CompanyInfoComponent and CategoryRatingComponent.

Other components are:

- **GenericCategoryPageComponent**
- **GenericTabViewComponent**
- **GenericNavbarComponent**
- **GenericProductListComponent**
- **GenericCompanyListComponent**
- **GenericUserReviewListComponent**
- **GenericRankingListComponent**
- **ProductInfoComponent**
- **TechnicalSpecsComponent**
- **CompanyInfoComponent**
- **CategoryRatingComponent**

Note: detailed content of these components explained in third part of the report.

2.1.2. Services

ProductService: This module's functions are used by **ProductPageComponent** and all of other components that required product info form backend, in order to make REST calls to the server. It handles the product related endpoints and their communication with the server.

CompanyService: This module's functions are used by **CompanyPageComponent** in order to make REST calls to the server. It handles the company related endpoints and their communication with the server.

PopularListService: This module's functions are used by **HomeComponent** in order to make REST calls to the server. By using this class home component can reach all 4 popular list from backend. It handles the popular list related endpoints and their communication with the server.

CriteriaService: This module's functions are used by **most of the Components** in order to make REST calls to the servers. It handles the criterion related endpoints and their communication with the server.

ComputersService: This module basically gets company and product list that belongs to this category from backend.

Note: Other Category Services that similar to ComputersService are not described in detail since their content is same with ComputersService.

Note: Search bar related endpoints and their RESTful services are handled inside all the components and services according to the type of the query passed by the client to the search-bar.

2.2. Logic Tier

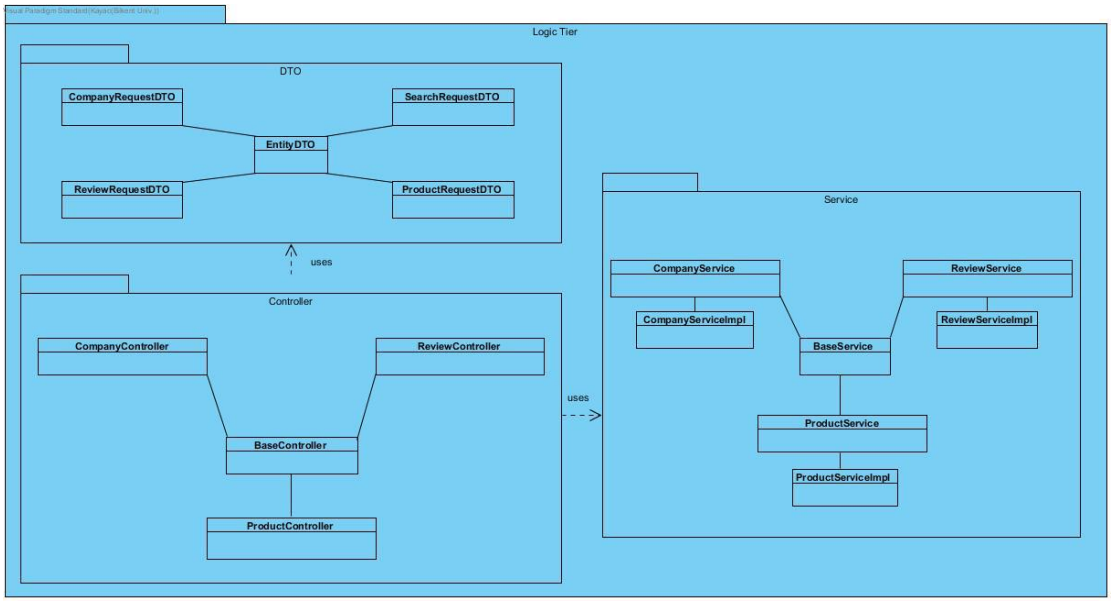

Figure 3 Logic Tier Diagram

Logic tier is consisting of 3 different components which are:

- DTO
- Controller
- Service

2.2.1. DTO

EntityDTO: Base parent class that is responsible for holding information common to all the DTOs. DTO classes are used in the process of communication with the frontend. The server does not respond to the requests directly with the entity classes but with DTO classes to either encapsulate the entity classes or limit the response messages from the server regarding security concerns. DTO classes reshape the entity classes and are then used by controllers while responding the request.

CompanyRequestDTO: This class is responsible for deciding what to return if a company-related request is detected.

ReviewRequestDTO: This class is responsible for deciding what to return if a review related request is detected.

SearchRequestDTO: This class is responsible for deciding what to return if search request is detected. It in most of the cases encapsulates list of objects to be used and returned back to client.

ProductRequestDTO: This class is responsible for deciding what to return if a produce related request is detected.

2.2.2. Controller

CompanyController: This class responsible for handling the REST request endpoints related to companies originating from the client. It uses CompanyService to get data (in Company type) and then convert it to CompanyDTO to respond to the request queried by the client.

ReviewController: This class responsible for handling the REST request endpoints related to user reviews (comments) originating from the client. It uses ReviewService to get data (in Review type) and then convert it to ReviewDTO to respond to the request queried by the client.

ProductController: This class responsible for handling the REST request endpoints related to products originating from the client. It uses ProductService to get data (in Product type) and then convert it to ProductDTO to respond to the request queried by the client.

2.2.3. Service

CompanyService: This interface is used by CompanyController which uses this class' methods for retrieving related information from the database.

CompanyServiceImpl: This class provides the implementation for CompanyService interface. This class uses CompanyRepository class to ask for the data in the database.

ReviewService: This interface is used by ReviewController which uses this class' methods for retrieving related information from the database.

ReviewServiceImpl: This class provides the implementation for ReviewService interface. This class uses ReviewRepository class to ask for the data in the database.

ProductService: This interface is used by ProductController which uses this class' methods for retrieving related information from the database.

ProductServiceImpl: This class provides the implementation for ProductService interface. This class uses ProductRepository class to ask for the data in the database.

2.3. Data Tier

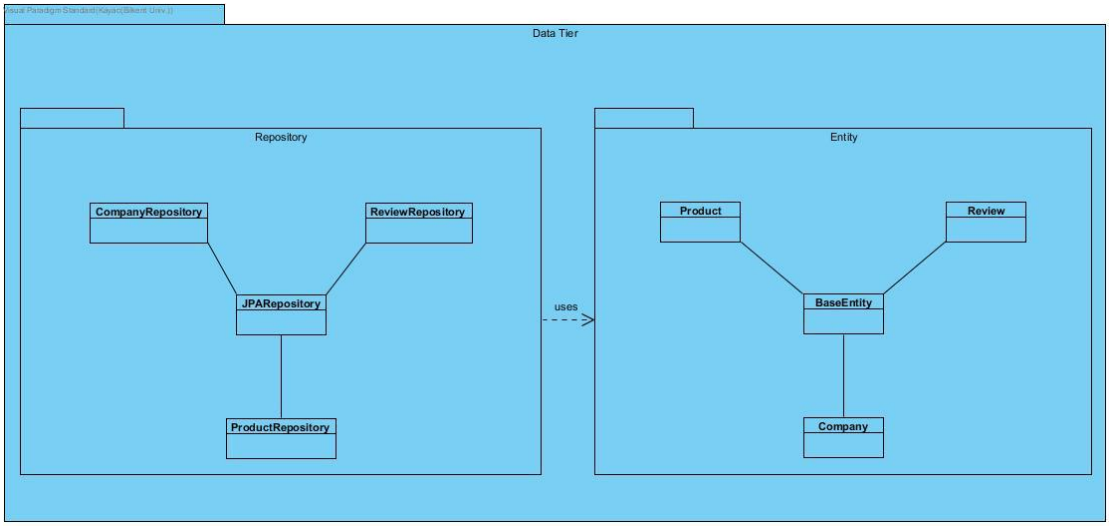

Figure 4 Data Tier Diagram

Data Tier consists of 2 different components which are:

- Repository
- Entity

2.3.1. Repository

CompanyRepository: This class is used by CompanyService, it returns back to it after retrieving data related to Company entity. This class extends JPARepository interface (found in Spring Framework package) which provides wide range of interactive methods to retrieve data from the database.

ReviewRepository: This class is used by ReviewService, it returns back to it after retrieving data related to Review entity. This class extends JPARepository interface (found in Spring Framework package) which provides wide range of interactive methods to retrieve data from the database.

ProductRepository: This class is used by ProductService, it returns back to it after retrieving data related to Product entity. This class extends JPARepository interface (found in Spring Framework package) which provides wide range of interactive methods to retrieve data from the database.

2.3.2. Entity

BaseEntity: Base parent class of entity objects which holds common information that all the entities share. Custom entity classes below which extends this class will be used by related repositories while retrieving information from database.

Product: This class represents the Product entity which is basically a representative of an object retrieved from Product table.

Review: This class represents the Review entity which is basically a representative of an object retrieved from Retrieve table.

Company: This class represents the Company entity which is basically a representative of an object retrieved from Company table.

Note: Repository interfaces, by means of using JPARepository interface, create a vast amount of methods related to the entity object that it is bound to so that complex queries can be handled.

3. Class Interfaces

3.1. Presentation Tier

3.1.1. Components

Class Name	AppComponent
Description	AppComponent class is basically a design part of the TechRank web page which will be common for all of the pages inside the web application. It will contains web page header.
Attributes	No attributes are included in AppComponent.
Methods	public void searchByKeyword()

Class Name	HomeComponent
Description	HomeComponent is the class that is designed for UI design of the main (entering) page of the webpage. This component will contains popular ranking lists inside it which will be created by using other components.
Attributes	popularList1 : List <Product> popularList2 : List <Product> popularList3 : List <Product> popularList4 : List <Product>
Methods	public void displayPopularList()

Class Name	GenericNavbarComponent
Description	GenericNavbarComponent is designed for the usage of a common navbar for all of the different pages. By this way same navbar could be added into all of different UI's without requiring new components for all page navbars. Same Navbar component will be called for required places.
Attributes	No attributes are included in GenericNavbarComponent.
Methods	public void showComputersCategory() public void showMobilePhoneCategory() public void showAudioCategory() public void showTV_ElectronicsCategory() public void showCamerasCategory()

Class Name	GenericCategoryPageComponent
Description	This Class is implemented for design of category page interface.
Attributes	No attributes are included in GenericCategoryPageComponent.
Methods	public void displayProducts() public void displayCompanies()

Class Name	GenericProductListComponent

Description	In Techrank webpage there are too much places that desired product list is required for the user interface. That's why, when a product list will be required this component will be called by desired data inside it.
Attributes	productList1 : List <Product>
Methods	public void showProductList () public void showProductWithKeyword(searchKeyword)

Class Name	GenericTabComponent
Description	This Component will be called all of the tab segments inside the webpages. By this way product, Category, Product Criteria, search pages can call this same component for creating tab segments.
Attributes	No attributes are included in GenericTabComponent.
Methods	public void displayProducts(String searchKeyword) public void displayCompanies(String searchKeyword) public void displayComments() public void displayTechnicalSpecs() public void displayCriteriaProductRanking() public void displayCriteriaComments()

Class Name	GenericCompanyListComponent
Description	GenericCompanyListComponent is implemented for the easier and more organized front-end design of user Interface parts that contains list of companies.
Attributes	No attributes are included in GenericCompanyListComponent.
Methods	public void showCompanyList () public void showCompaniesWithKeyword(searchKeyword)

Class Name	ProductPageComponent
Description	ProductPageComponent is responsible from design of the product pages. This component will contain all of the technicalSpecsComponent, ProductInfoComponent, GenericUserReviewListComponent, GenericRankingListComponents
Attributes	productId : int

	productName: String
Methods	public void showProductPage(int productId) public void showProductPage(String productName)

Class Name	CompanyPageComponent
Description	CompanyPageComponent is responsible from design of the company pages.
Attributes	productId : int productName: String
Methods	public void showCompanyPage(int companyId) public void showCompanyPage(String companyName)

Class Name	ProductInfoComponent
Description	This component will be responsible for design of product information inside the product page.
Attributes	productID: int productName : String
Methods	public void showProductInfo(productID)

Class Name	TechnicalSpecsComponent
Description	This class is responsible for listening and managing user interactions with the user interface that are related with product's specifications.
Attributes	productID : int productName: String
Methods	public void showTechSpecs()

Class Name	GenericUserReviewListComponent
Description	This class is responsible from containing reviews inside it and displaying these reviews to end users by user interface.

Attributes	- reviewID: int - reviewContent : String
Methods	public void showReviews() public void selectReview()

Class Name	GenericRankingListComponent
Description	Ranking Lists are very commonly used components inside TechRank web application since ranking is very essential for the logic of the application. This component will provide required ranking table for all of the request. Only its content will be changed according to request from end user.
Attributes	- currentProductList : Product [] - currentCompanyList : Company []
Methods	public void selectCategory()

3.1.2. Services

Class Name	ComputersService
Description	ComputersService is the class that send requests to backend to provide desired information about computers to User Interface.
Attributes	computerProductList : List <Product> computerCompanyList : List <Company>
Methods	public List <Product> getProductList() public List <Company> getCompanyList()

Note: *There will be 6 other class like Computer service inside our projects for all categories. Since their content is the same with the Computer service, they are not contained inside report.*

Class Name	ProductService
Description	This class handles the product related endpoints and their communication with the server. It will takes data from back end and send them to components.

Attributes	-currentProduct : Product
Methods	public Product getProduct(int id)

Class Name	CompanyService
Description	This class handles the company related endpoints and their communication with the server.It will takes data from back end and send them to components.
Attributes	-currentCompany : Company -companyId: int -companyName: String
Methods	public Criterion getCompany(int id)

Class Name	CriteriaService
Description	This class handles the criteria related endpoints and their communication with the server. It will takes data from back end and send them to components.
Attributes	currentCriterion : Criterion criterionId : int criterionName : String
Methods	public Criterion getCriterion() public String getCriterionName()

Class Name	PopularListService
Description	This class send request from front end to backend to be able to get most popular lists. Then it will send these datas to component part.
Attributes	popularList1: List<Product> popularList2: List<Product> popularList3: List<Product> popularList4: List<Product>
Methods	public List<Product> getPopularList1() public List<Product> getPopularList2() public List<Product> getPopularList3() public List<Product> getPopularList4()

3.2. Logic Tier

3.2.1. DTOs

Class Name	CompanyRequestDTO
Description	This class is responsible for deciding what to return if a company-related request is detected.
Attributes	- companyID: int - companyName: String - overallCompanyRating: float - overallCompanyRanking: int - products: ArrayList<ProductResponseDto>
Methods	public String getCompanyName public void setCompanyName(String companyName public float getOverallCompanyRating() public void setOverallCompanyRating(float overallCompanyRating) public int getOverallCompanyRanking() public void setOverallCompanyRanking(int overallCompanyRanking) public ArrayList<ProductResponseDTO> getProducts() public void setProducts(ArrayList<ProductResponseDTO> products) public int getCompanyID() public void setCompanyID(int companyID)

Class Name	ReviewRequestDTO
Description	This class is responsible for deciding what to return if a review related request is detected.
Attributes	- reviewID: int - commentDate: Date - commentSource: String - commentBody: String - commenterName: String
Methods	public Date getCommentDate() public void setCommentDate(Date commentDate) public String getCommentSource() public void setCommentSource(String commentSource) public String getCommentBody() public void setCommentBody(String commentBody) public String getCommenterName()

	public void setCommenterName(String commenterName) public int getReviewID() public void setReviewID(int reviewID)

Class Name	SearchRequestDTO
Description	This class is responsible for deciding what to return if search request is detected.
Attributes	- products: List<ProductResponseDTO> - companies: List<CompanyResponseDTO>
Methods	public List<ProductResponseDTO> getProducts() public void setProducts(List<ProductResponseDTO> products) public List<CompanyResponseDTO> getCompanies() public void setCompanies(List<CompanyResponseDTO> companies)

Class Name	ProductRequestDTO
Description	This class is responsible for deciding what to return if a produce related request is detected.
Attributes	- productID: int - productName: String - productImageURL: String - productTechSpecs: String - overallProductRating: float - overallProductRanking: int - company: CompanyResponseDTO - reviews: ArrayList<ReviewResponseDTO>
Methods	public String getProductName() public void setProductName(String productName) public String getProductImageUrl() public void setProductImageUrl(String productImageUrl) public String getProductTechSpecs() public void setProductTechSpecs(String productTechSpecs) public float getOverallProductRating() public void setOverallProductRating(float overallProductRating) public int getOverallProductRanking() public void setOverallProductRanking(int overallProductRanking) public int getProductID() public void setProductID(int productID) public ArrayList<ReviewResponseDTO> getReviews()

	public void setReviews(ArrayList<ReviewResponseDTO> reviews)

3.2.2. Controllers

Class Name	CompanyController
Description	This class is responsible for handling the REST request endpoints related to companies originating from the client.
Attributes	# companyService: CompanyService # productService: ProductService
Methods	No methods are included.

Class Name	ReviewController
Description	This class is responsible for handling the REST request endpoints related to user reviews (comments) originating from the client.
Attributes	# reviewService: ReviewService
Methods	No methods are included.

Class Name	ProductController
Description	This class is responsible for handling the REST request endpoints related to products originating from the client.
Attributes	# productService: ProductService # companyService: CompanyService # reviewService: ReviewService
Methods	No methods are included.

3.2.3. Services

Class Name	CompanyService
Description	This class is used by CompanyController which uses this class' methods for retrieving related information from the database.

Attributes	# companyRepository CompanyRepository
Methods	No methods are included.

Class Name	ReviewService
Description	This class is used by ReviewController which uses this class' methods for retrieving related information from the database.
Attributes	# reviewRepository: ReviewRepository
Methods	No methods are included.

Class Name	ProductService
Description	This class provides the implementation for ProductService interface.
Attributes	# productRepository: ProductRepository
Methods	No methods are included.

3.3. Data Tier

3.3.1. Repositories

Class Name	CompanyRepository
Description	This class is used by CompanyService, it returns back to it after retrieving data related to Company entity.
Attributes	There are not attributes included.
Methods	public List<Company> findAll() public Company findById(Integer companyId) public Company save(Company company) public void deleteById(Integer companyId)

Class Name	ReviewRepository

Description	This class is used by ReviewService, it returns back to it after retrieving data related to Review entity.
Attributes	There are not attributes included.
Methods	public List<Review> findAll() public Review findById(Integer reviewId) public Review save(Review review) public void deleteById(Integer reviewId)

Class Name	ProductRepository
Description	This class is used by ProductService, it returns back to it after retrieving data related to Product entity.
Attributes	There are not attributes included.
Methods	public List<Product> findAll() public Product findById(Integer reviewId) public Product save(Product product) public void deleteById(Integer productId)

3.3.2. Entities

Class Name	Product
Description	This class represents the Product entity which is basically a representative of an object retrieved from Product table.
Attributes	- productID: int - productName: String - productTechSpecs: String - productRating: float - productRanking: int - company: Company
Methods	public int getProductID() public void setProductID(int productID) public String getProductName() public void setProductName(String productName) public String getProductTechSpecs() public void setProductTechSpecs(String productTechSpecs) public float getProductRating() public void setProductRating(float productRating)

	public int getProductRanking() public void setProductRanking(int productRanking) public Company getCompany() public void setCompany(Company company)

Class Name	Review
Description	This class represents the Review entity which is basically a representative of an object retrieved from Retrieve table.
Attributes	- reviewID: int - commentDate: Date - commentSource: String - commentBody: String - commenterName: String - product: Product
Methods	public int getReviewID() public void setReviewID(int reviewID) public Date getCommentDate() public void setCommentDate(Date commentDate) public String getCommentSource() public String getCommentBody() public void setCommentBody(String commentBody) public String getCommenterName() public void setCommenterName(String commenterName) public Product getProduct() public void setProduct(Product product)

Class Name	Company
Description	This class represents the Company entity which is basically a representative of an object retrieved from Company table.
Attributes	- companyID: int - companyName: String - companyRating: float - companyRanking: int
Methods	public int getCompanyID() public void setCompanyID(int companyID) public String getCompanyName() public void setCompanyName(String companyName) public List<Product> getProductList()

	public void setProductList(List<Product> productList) public float getCompanyRating() public void setCompanyRating(int companyRating) public int getCompanyRanking() public void setCompanyRanking(int companyRanking)

4. Glossary

GUI: Graphical User Interface is a form of user interface that allows users to interact with electronic devices through visual indicators such as buttons and menus, instead of typing command labels or text navigation. [1]

Three Tier Architecture: Client-server architecture that the functional process logic, data access, computer data storage and user interface are kept as independent modules. [2]

Client: Computer which is capable of obtaining information and applications from a server.

Server: Computer which manages access to a service in the network.

Angular: It is a platform that enables us to build applications in the web. [3]

Gitlab: It is a single application for the whole software development lifecycle. [4]

5. References

[1]"GUI (Graphical User Interface) Definition", *Techterms.com*, 2019. [Online]. Available: https://techterms.com/definition/gui. [Accessed: 15- Feb- 2019].

[2]"What is Three-Tier Architecture? - Definition from Techopedia", *Techopedia.com*, 2019. [Online]. Available: https://www.techopedia.com/definition/24649/three-tier-architecture. [Accessed: 11- Feb- 2019].

[3]"Angular Docs", *Angular.io*, 2019. [Online]. Available: https://angular.io/docs. [Accessed: 18- Jan- 2019].

[4]"The first single application for the entire DevOps lifecycle - GitLab", *GitLab*, 2019. [Online]. Available: https://about.gitlab.com/. [Accessed: 09- Feb- 2019].