Bilkent University

Department of Computer Engineering

Senior Design Project

Newspector: A Reliable Platform for News

Low-Level Design Report

Ahmet Ayrancıoğlu 21601206

Deniz Dalkılıç 21601896

Kaan Gönç 21602670

Supervisor: Varol Akman

Jury Members: Halil Altay Güvenir and Özcan Öztürk

Low-Level Report

February 17, 2020

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

1. Introduction

The news is one of the most important parts of everyone's life as it is a medium for them to learn about the developments in the world. The news directly affects people and society as a whole. No matter where you are in the world either traveling or working in the office you can get all the latest updates from the world. Without the circulation of the news, it is impossible for people to know what is happening around them. The latest news covers all the news from various fields such as sports, entertainment, fashion, charity events and much more. The world is changing every moment and we can keep an eye on the changes through the news. Therefore, the importance of the news in our daily lives cannot be measured in words. This important role of news in the world leads various news companies to be founded and thus creating a great rivalry between such sources on being the fastest and the most accurate one among all to deliver news to society. An important question that should be asked here is how much we should believe such news companies, and which one to follow if we want to access the latest news first.

When we consider these issues, the concept of accessing reliable breaking news comes out to be a real concern for many people. Anything may happen at any point in time, so it is very important for the media to stay alert to cover the incidents when it happens. The breaking news gets a lot of attention when it comes to the news portal. The breaking news may be related to any incident or the latest discovery around the world. An important point is that there are too many news sources on the web and only a limited number of them seem to be producing reliable news. We are planning to define worldwide known and trusted news companies' websites and twitter accounts as our sources of data in our project. And our interpretations about the companies will be based on the speed of publishing the breaking news and the reliability of the content of the news.

This report explains an overview of the low-level architecture and in-depth design of the Newspector system. Additionally, this report discusses the trade-offs of our design choices and the engineering standards that will be used. Documentation guidelines for the ease of understanding will also be provided. After documentation guidelines, information about the packages and interfaces of the Newspector's systems will be presented. Finally, a subsystem decomposition diagram along with the class diagrams explaining each subsystem and a detailed look into each of the Newspector's software components will conclude the report.

1.1 Object Design Trade-Offs

1.1.1 Functionality vs Usability

Newspector is a simple application that is mainly designed for daily use. Its target audience range is wide as any person who has a smart device is able to download and start using it any time they want. It offers great functionalities however these functionalities will have no value if the user is not able to understand and perform them easily. In order to maintain a highly usable system, we avoided adding too much functionality to our application so that the users will find it a lot easier to get used to the application features. We will provide simple functionalities such as “search news”, “follow the news” or “report source”. The reason for this is that we would not like to compromise the usability for increasing the functionality of the application. Although we will have a small number of functionalities in our application, we are planning to provide a tutorial mechanism to our users so that they will have a chance to learn the basics of the application without making any attempts to learn by themselves during the real experience.

1.1.2 Security vs Cost

In terms of security and cost, what we are planning to achieve with our application is to avoid a traditional example of a “trade-off”. As our application will only be storing the news preferences of users, we won’t be developing our own database and implementing specialized security algorithms but rather use Google services for performing the authentication and storing the user information. By this way, we will be saving a lot from the cost of implementation and security will not be a primary issue for our application

1.1.3 Portability vs Performance

Users will be able to use Newspector on a great variety of operating systems such as Android, IOS and Windows. It can be used as both a mobile and a web application. In order to achieve this in a small amount of time, we have used the Flutter framework which enables us to easily develop a cross-platform application. As we do not use a native language to develop the Newspector for each of the platforms, we are taking the risk of losing performance. However, right now the Flutter framework seems to operate well in any platform and does not seem to have any noticeable impacts on the user experience.

1.1.4 Compatibility vs Extensibility

In terms of extensibility, Newspector is planned as a project that can be extended indefinitely. Newspector should be developed with feature updates in mind and potential changes in our flow of information. Additionally, Newspector is aiming to be compatible with both IOS and Android and potentially Xiaomi's new operating system versions that will be released in the future. While supporting and adapting to future versions of these operating systems Newspector should also still be compatible with older versions of these systems.

1.1.5 Quality vs Time

The concept of delivering breaking news is significantly important. However, we also believe that the way we deliver this news to the users is as important as the news content. Our application is based on providing statistics on the reliability and speed of the news sources as well as providing the news content with real-time notifications. This is a serious action which can have varying impacts on both the people and the companies, therefore we are aware that our application should have a high quality of achieving what it promises. In order to achieve these, we are planning to spend a lot of time in every detail of the application features in order not to have a misleading impact on our users and sources.

1.2 Interface Documentation Guidelines

In this report all of the diagrams are coherent with the following guidelines:

- Class names are in the pascal-case format where the first character of a word is in uppercase and the rest of the characters are in lowercase. (e.g. ‘ClassName’)
- Variable names are in the camel-case format where the first character of a word is in uppercase, except the very first character, and the rest of the characters are in lowercase. (e.g. ‘varName’)
- Function names are similarly formatted to variable names except the function names end with a pair of parentheses (e.g. functionName()).
- The names of the variables and functions can have a prefix such as public, private, final, and etc.

A sample class in a class diagram is represented as follows:

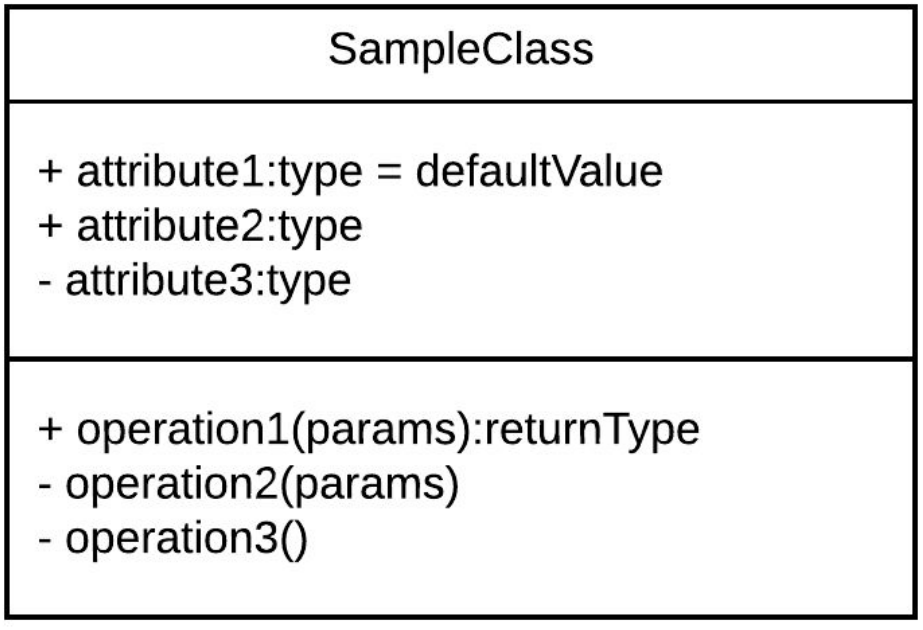


Figure 1 - Sample Class Diagram Piece 1

A sample class in class interfaces is represented as follows:

SampleClass	
Description of class	
Attributes	
-attribute1	Description of attribute1
Operations	
+operation1(param : int) : float	Description of operation1

Figure 2 - Sample Class Diagram Piece 2

1.3 Engineering Standards

- **UML**

  Unified Modeling Language (UML), is a standardized modeling language used by engineers that help system and software developers for specifying, visualizing, constructing, and documenting software systems. [1] The UML is useful and proven to be successful in the modeling of large and complex systems. The UML is a very important part of developing object-oriented software and the software development process. In this report, the UML standard is used for representing class interfaces, hardware-software components and decomposing systems to packages.

- **IEEE**

  IEEE provides various engineering standards for unified understanding amongst engineers. In this report, the references follow the IEEE citation standard.

1.4 Definitions, acronyms, and abbreviations

- **API:** Application Program Interface
- **JSON:** JavaScript Object Notation
- **UI:** User Interface
- **SQL:** Structured Query Language
- **NoSQL:** Not Only SQL
- **NLP:** Natural Language Processing
- **NLU:** Natural Language Understanding
- **ML:** Machine Learning

2. Packages

The low-level design of our project is split into 3 main subsystems: Fetcher, Analyzer, and View. These subsystems are connected to each other through a central database and they are truly modular and independent of each other. Only the View subsystem represents the client-side and it is the only subsystem that the user will have any interactions with. The Fetcher and the Analyzer subsystems are server-side and will work in the background in our servers. Fetcher subsystem is responsible for collecting and storing the raw news data for later analysis. The Analyzer subsystem is responsible for analyzing the raw news data and computing valuable insights into the news articles we have collected. Finally, the View subsystem is responsible for presenting the news and our analysis of the news to the user.

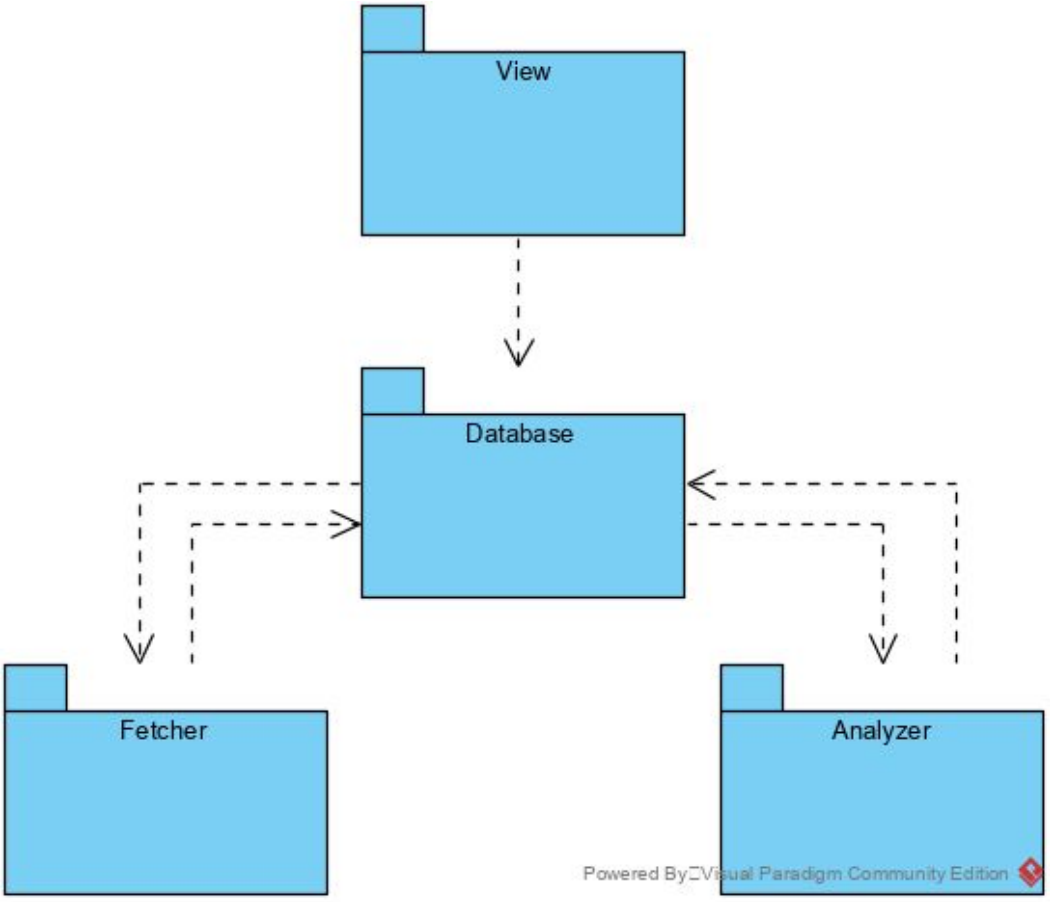


Figure 3 - Subsystem Decomposition

2.1 Fetcher Subsystem

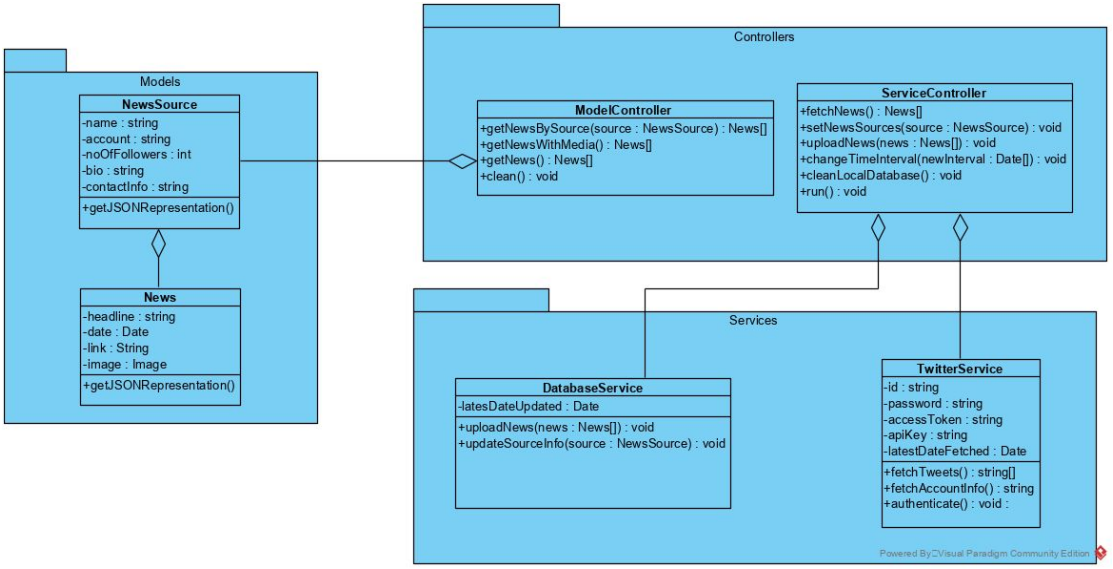


Figure 4 - Fetcher Class Diagram

Models Package

Models Package consists of models used to represent objects that will be gathered and stored in our database. Models do not contain any internal logic and their only function is to store information and change the format of the information.

Controllers Package

Controllers Package consists of controllers that control the flow of information. Controllers act as a bridge between services that are connected to outside systems and our models.

Services Package

Services Package consists of services that allow the system to fetch news information and store the gathered news in our own databases. They are connected to outside packages and APIs and provide the main functionality of this subsystem.

2.2 Analyzer Subsystem Packages

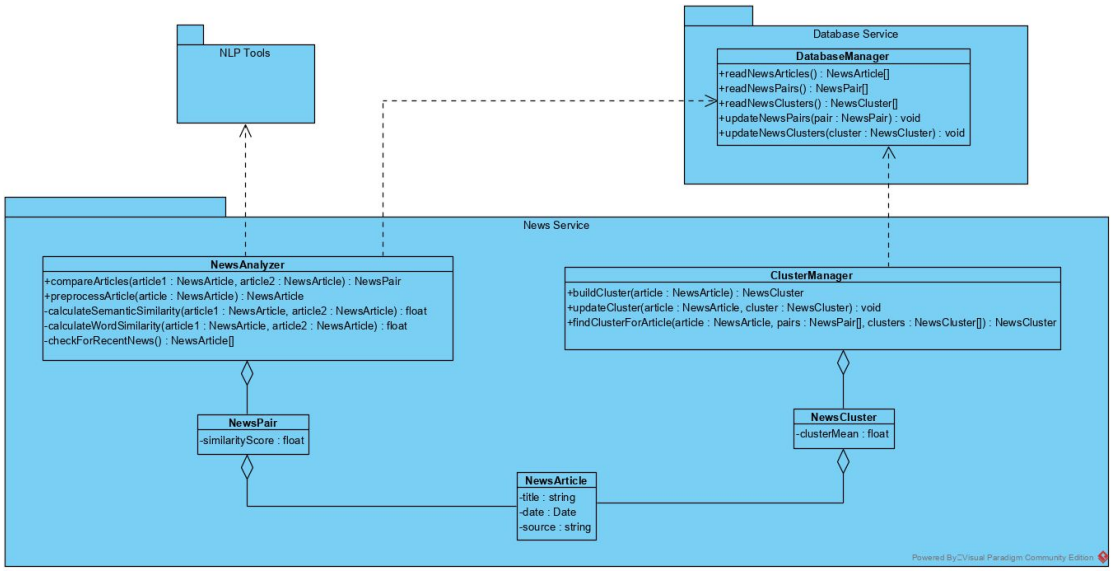


Figure 5 - Analyzer Class Diagram

NLP Tools Package

NLP Tools Package is a black box for the current version of the project. This package is going to be filled with some specific tools which we will use for NLU, sentiment analysis, and etc. parts of the project. Currently, there are some tools such as NLTK library [2], Doc2vec library [3], and semantic similarity calculators that we use but there isn't any certainty that we will keep using them in the future.

Database Service Package

This package is for the database related operations of the Analysis subsystem such as reading from, inserting to, or updating the database with respect to the requests of the classes of News Service package.

News Service Package

This package is for the news related operations of the Analysis subsystem such as analyzing and comparing news or building and updating clusters of news.

2.3 View Subsystem Packages

View Subsystem is responsible for showing the news articles, news sources and our analysis to the users in a concise and clear way.

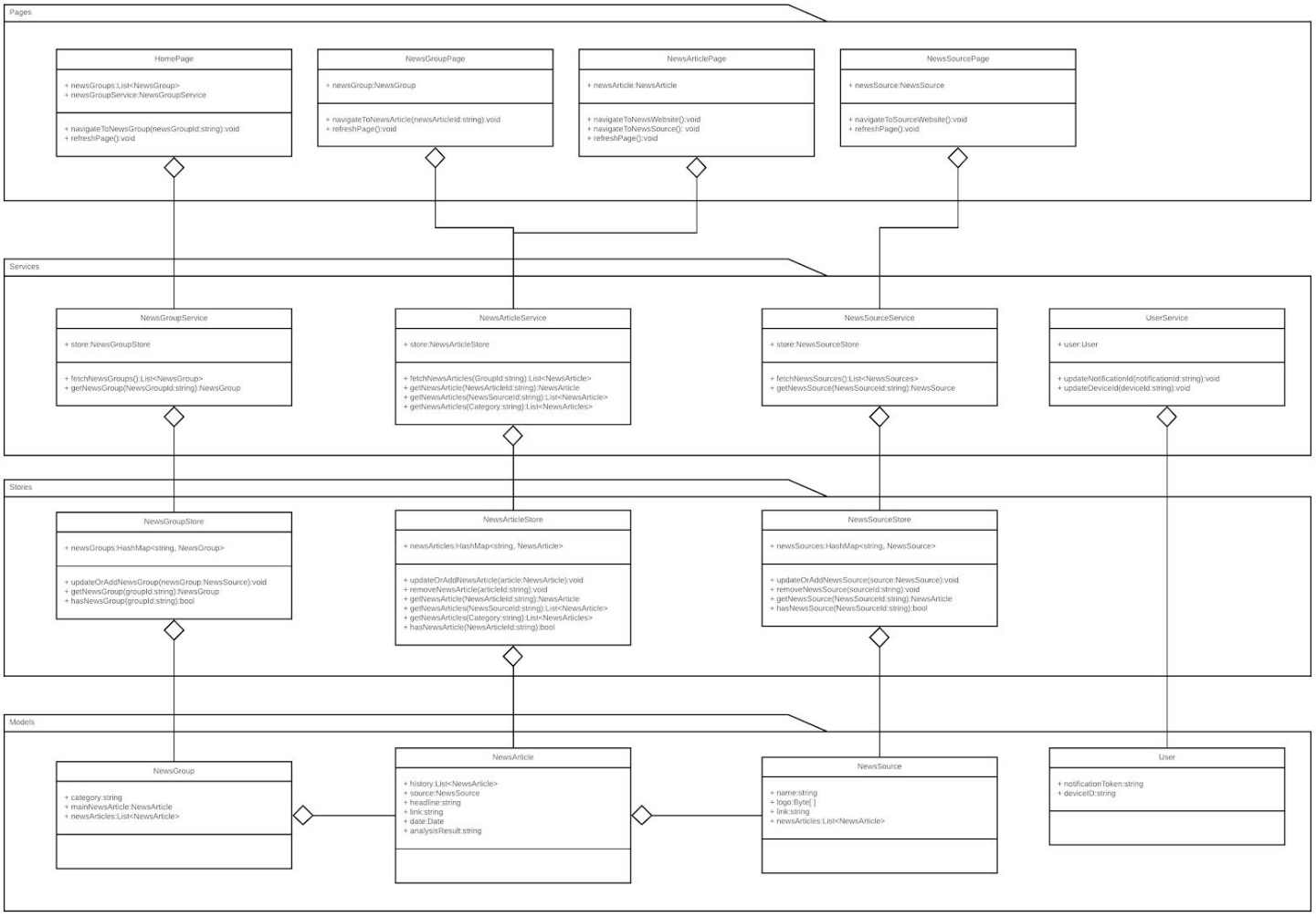


Figure 6 - View Class Diagram

(See Appendix A)

Page Package

The Pages package consists of page classes that are responsible for showing our content to users. These pages do not have complex logic inside them and are not responsible for modifying any of our data directly. These classes are a bridge between our view login and user interactions.

Services Package

The services package consists of services. These services provide functionality and access to application data to the pages package. Services are responsible for providing a clear interface for the pages package for accessing, modifying and updating the application data. For each unique object in our application, there is a service that handles functionality for that object. Services are also connected to stores, local storage, and our database and are responsible for synchronization of these 3 storage solutions.

Stores Package

The Stores package consists of stores. These stores store unique objects that our application uses. Stores are custom data structures that ensure every object is unique and can be accessed with a unique ID. In short, they are custom hashtables with extra functionality.

Model Package

The model package consists of object models. Object models are classes that represent the objects in our application. They do not contain any logic and only attributes that can be read and written by services. Models are stored in stores in our application.

3. Class Interfaces

3.1 Fetcher Subsystem Class Interfaces

Models Package

News	
Represents the raw news data gathered from outside sources.	
Attributes	
-headline : string	The headline of the news article
-date : Date	Publish date of the news article
-link : string	Publish link of the news article
-image : Image	The image posted with the news
Operations	
+getJSONRepresentation()	Returns the JSON representation of this object.

NewsSource	
Represents the news source information.	
Attributes	
-name : string	Name of the news source
-account : string	Account name of the news source
-noOfFollowers : int	Number of followers of the news source account
-bio : string	Biography of the new source account
-contactInfo: string	Contact information of the news source
Operations	
+getJSONRepresentation()	Returns the JSON representation of this object.

Services Package

DatabaseService	
Database Service is responsible for providing access to the database and modifying the contents of the database.	
Attributes	
-latestDateUpdated : Date	The latest date database was updated
Operations	
+uploadNews(news : News[]) : void	Uploads the recent news to the database.
+updateSourceInfo(source : NewsSource) : void	Updates the news source due to the given NewSource object.

TwitterService	
Twitter service is responsible for gathering news from twitter using the twitter API.	
Attributes	
-id : string	The unique ID of the Twitter developer account.
-password : string	The password of the Twitter developer account
-accessToken : string	Access token of the Twitter API
-apiKey : string	Key for the Twitter API
-latestDateFetched : Date	The latest date news was fetched
Operations	
+fetchTweets() : string[]	Fetches the tweets and returns them as a string list.
+fetchAccountInfo() : string	Fetches the account info for the news sources.
+authenticate() : void	Authenticates the Twitter API.

Controllers Package

ModelController	
The model controller is responsible for structuring the gathered news into an easy to use and store format.	
Operations	
+getNewsBySource(source : NewsSource) : News[]	Returns the news that belongs to the given source.
+getNewsWithMedia() : News[]	Returns the news with the media the include.
+getNews() : News []	Returns the news articles.
+clean() : void	Cleans the data that is temporarily stored.

ServiceController	
Service Controller is responsible for creating a bridge between our local storage in the subsystem and the remote storage.	
Operations	
+fetchNews() : News[]	Fetches the news by calling the helper functions such as fetchTweets() from TwitterService
+setNewsSources(source : NewsSource) : void	Initiates the NewsSource objects
+uploadNews(news : News[]) : void	Uploads the news by calling helper functions such as uploadNews() from DatabaseService.
+changeTimeInterval(newInterval : Date[]) : void	Changes the time interval of news fetching.
+cleanLocalDatabase() : void	Cleans the local database of this subsystem.
+run() : void	Runs this subsystem.

3.2 Analyzer Subsystem Class Interfaces

Database Service Package

DatabaseManager	
This class contains the database management functions. The classes of the News Service package call these functions in order to perform their roles.	
Operations	
+readNewsArticles() : NewsArticle []	Reads the news articles from the database.
+readNewsPairs() : NewsPair []	Reads the news pairs from the database.
+readNewsClusters() : NewsCluster []	Reads the news clusters from the database.
+updateNewsPairs(pair :NewsPair) : void	Updates the news pairs table with respect to the pairs parameter.
+updateNewsClusters(cluster : NewsCluster) : void	Updates the news clusters table with respect to the cluster parameter.

News Service Package

NewsArticle	
This class is an entity that represents the news article objects.	
Attributes	
-title : string	Title of the news article
-date : Date	Publish date of the news article
-source : string	The source which published the news article

NewsPair	
This class is an entity that is composed of a pair of news articles and the similarity score between them.	
Attributes	
-similarityScore : float	The similarity score that is calculated with respect to the results of sentiment and word similarities between two news articles.

NewsCluster	
This class is an entity that is composed of a bunch of news articles that have the same content and the mean value of the cluster.	
Attributes	
-clusterMean : float	The mean value of the cluster that is calculated with respect to the similarities between the news articles in the cluster.

NewsAnalyzer	
This class is responsible for the core computations of this project. This class is where the news articles are preprocessed and the similarities and relations between the news articles are calculated using NLP tools.	
Operations	
-calculateSemanticSimilarity(article1 : NewsArticle, article2 : NewsArticle) : float	Calculates the semantic similarity between two news articles using NLP tools.
-calculateWordSimilarity(article1 : NewsArticle, article2 : NewsArticle) : float	Calculates the word similarity between two news articles using NLP tools.
-checkForRecentNews() : NewsArticle []	This method is called continuously in order to check for recent news.
+compareArticles(article1 : NewsArticle, article2 : NewsArticle) : NewsPair	Compares two news articles by callingcalculateSemanticSimilarity and calculateWordSimilarity methods and creates a NewsPair object.
+preprocessArticle(article : NewsArticle) : NewsArticle	Preprocesses the given news article before the analysis process.

ClusterManager	
This class is responsible for grouping the news articles that have the same content and creating new clusters for unique news.	
Operations	
+buildCluster(article : NewsArticle) : NewsCluster	Creates a new cluster for the given news article.
+updateCluster(article : NewsArticle, cluster : NewsCluster) : void	Updates the given cluster with respect to the given news article.
+findClusterForArticle(article : NewsArticle, pairs : NewsPair[], clusters : NewsCluster[]) :NewsCluster	This method finds the cluster (if any) which the given news article should belong to by examining the news pairs.

3.3 View Subsystem Class Interfaces

Model Package

NewsGroup	
NewsGroup consists of multiple news articles that are about the same news topic.	
Attributes	
+category: string	The category of the newsGroup.
+mainNewsArticle:NewsArticle	The NewsArticle to represent the NewsGroup and the news topic.
+newsArticles:List<NewsArticle>	The NewsArticle belonging to this NewsGroup.

NewsArticle	
Represents a single news article and holds every information about the news article.	
Attributes	
+history: List<NewsArticle>	Holds the information about the history of this particular news article to keep track of the changes made to the article.
+source: NewsSource	The news source that published the article.
+headline: string	The headline of the news article.
+link: string	The link tot he original news article.
+date: Date	The date the news article was published.
+analysisResult:string	Analysis result of the news article.

NewsSource	
Represents a news source that is used to fetch news articles from.	
Attributes	
+name: string	The name of the news source.
+logo: Byte[ ]	The logo of the news source.
+link: string	The link to the website of the news source.
+newsArticles: List<NewsArticle>	The news articles published by this news source.

User	
Represents the user of the application.	
Attributes	
+notificationToken: string	Notification token for sending notifications.
+deviceId: string	Device Id to identify the device.

Stores Package

NewsGroupStore	
NewsGroupStore is a class that holds NewsGroup for easy access.	
Attributes	
+newsGroups: HashMap<string, NewsGroup>	HashMap for storing the newsgroups with their unique id.
Operations	
+ hasNewsGroup(groupId:string):bool	Checks if a newsGroup exists with the given Id.
+ updateOrAddNewsGroup(newsGroup:NewsSource):void	Checks if a newsGroup exists in the hashmap, update the group if a group exists, adds if it does not exist.
+ getNewsGroup(groupId:string):NewsGroup	Gets a newsgroup with the given Id.

NewsArticleStore	
NewsArticleStore is a class that holds NewsArticles for easy access.	
Attributes	
+newsArticles: HashMap<string, NewsArticle>	HashMap for storing the news articles with their unique id.
Operations	
+ hasNewsArticle(NewsArticleId:string):bool	Checks if a news article exists with the given Id.
+ getNewsArticles(Category:string):List<NewsArticles>	Gets news articles with the given category.
+ getNewsArticles(NewsSourceId:string):List<NewsArticle>	Gets news articles with the given news source.
+ getNewsArticle(NewsArticleId:string):NewsArticle	Gets a news article with the given Id.
+ removeNewsArticle(articleId:string):void	Removes the news article with the given Id.
+ updateOrAddNewsArticle(article:NewsArticle):void	Checks if a news article exists in the hashmap, update the article if a group exists, adds if it does not exist.

NewsSourceStore	
NewsSourceStore is a class that holds NewsSources for easy access.	
Attributes	
+newsSources: HashMap<string, NewsSource>	HashMap for storing the news sources with their unique id.
Operations	
+ hasNewsSource(NewsSourceId:string):bool	Checks if a news source exists with the given Id.
+ getNewsSource(NewsSourceId:string):NewsArticle	Gets a news source with the given Id.
+ removeNewsSource(sourceId:string):void	Removes the news source with the given Id.
+ updateOrAddNewsSource(source:NewsSource):void	Checks if a news source exists in the hashmap, update the source if a group exists, adds if it does not exist.

Services Package

NewsGroupService	
NewsGroupService is a class that provides functionality to fetch, remove, update newsgroups that is displayed in the application.	
Attributes	
+ store: NewsGroupStore	Store to hold newsgroups.
Operations	
+ fetchNewsGroups(): List<NewsGroup>	Returns the latest newsgroups that are available in the database.
+ getNewsGroup(NewsGroupId:string):NewsGroup	Returns the newsgroup with the given Id.

NewsArticleService	
NewsArticleService is a class that provides functionality to fetch, remove, update news articles that are displayed in the application.	
Attributes	
+ store: NewsArticleStore	Store to hold news articles.
Operations	
+ fetchNewsArticles(GroupId:string):List<NewsArticle>	Returns the latest news articles that belong to the given group.
+ getNewsArticle(NewsArticleId:string):NewsArticle	Returns the news article with the given Id.
+ getNewsArticles(NewsSourceId:string):List<NewsArticle>	Returns the news articles that are published by the given source.
+ getNewsArticles(Category:string):List<NewsArticles>	Returns the news articles that are in the given category.

NewsSourceService	
NewsSourceService is a class that provides functionality to fetch, remove, update news sources that are displayed in the application.	
Attributes	
+ store: NewsSourceStore	Store to hold news sources.
Operations	
+ fetchNewsSources(): List<NewsSources>	Returns the latest news sources that are available in the database.
+ getNewsSource(NewsSourceId:string):NewsSource	Returns the news source with the given Id.

UserService	
UserService is a class that provides functionality to update the preferences and the identification of the user.	
Attributes	
+ user: User	The user that is currently active.
Operations	
+ updateNotificationId(notificationId:string):void	Updates the notification Id of the user.
+ updateDeviceId(deviceId:string):void	Updates the device Id of the user.

Pages Package

HomePage	
Home Page is the main page the user can interact with the application. It shows the newsgroups in an organized way in a list.	
Attributes	
+ newsGroups:List<NewsGroup>	Latest newsgroups to be displayed.
+ newsGroupService: NewsGroupService	NewsGroup service to get the latest newsgroups from available.
Operations	
+ navigateToNewsGroup(newsGroupId:string):void	Navigates the user to the page of the clicked newsgroup.
+ refreshPage():void	Refreshes the home page to fetch the latest newsgroups.

NewsGroupPage	
Newsgroup page is the page where users can see every news article about the same topic with our analysis of the news articles.	
Attributes	
+ newsGroup: NewsGroup	Newsgroup to be displayed.
Operations	
+ navigateToNewsArticle(newsArticleId:string):void	Navigates the user to the page of the clicked news article.
+ refreshPage():void	Refreshes the newsgroup page to fetch the latest version of the page.

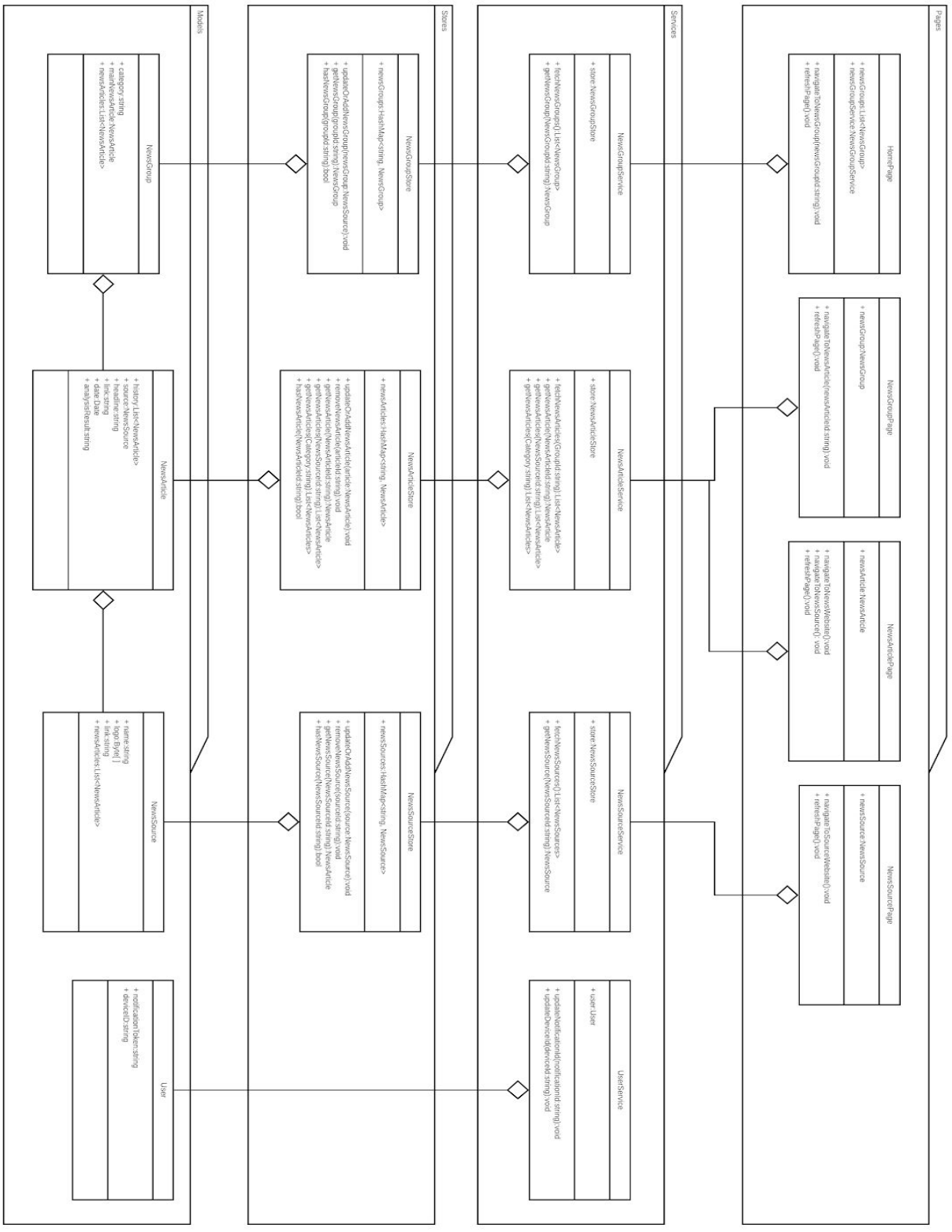
NewsArticlePage	
News Article Page is the page where the users can see a news article in detail.	
Attributes	
+ newsArticle:NewsArticle	News article to be displayed.
Operations	
+ navigateToNewsWebsite():void	Navigates the user to the website of the clicked news article.
+ navigateToNewsSource(): void	Navigates the user to the news source page of the clicked news article.
+ refreshPage():void	Refreshes the newsgroup page to fetch the latest version of the page.

NewsSourcePage	
The news source page is the page where users can see detailed information about a particular news source.	
Attributes	
+ newsSource: NewsSource	News source to be displayed.
Operations	
+ navigateToSourceWebsite():void	Navigates the user to the website of the news source.
+ refreshPage():void	Refreshes the news group page to fetch the latest version of the page.

4. References

1. What is Unified Modeling Language (UML)? [Online]. Available: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/. [Accessed: 17-Feb-2020].
2. "Natural Language Toolkit," *Natural Language Toolkit - NLTK 3.4.5 documentation*. [Online]. Available: https://www.nltk.org/. [Accessed: 17-Feb-2020].
3. "gensim: topic modeling for humans," *Radim: Machine learning consulting*. [Online]. Available: https://radimrehurek.com/gensim/models/doc2vec.html. [Accessed: 17-Feb-2020].

Appendix A


Pages
HomePage
+ newsGroups: List<NewsGroup>
+ newsGroupService: NewsGroupService
+ navigateToNewsGroup(newsGroupId:string):void
+ refreshPage():void
NewsGroupPage
+ newsGroup:NewsGroup
+ navigateToNewsArticle(newsArticleId:string):void
+ refreshPage():void
NewsArticlePage
+ newsArticle:NewsArticle
+ navigateToNewsWebsite():void
+ navigateToNewsSource():void
+ refreshPage():void
NewsSourcePage
+ newsSource:NewsSource
+ navigateToSourceWebsite():void
+ refreshPage():void
Services
NewsGroupService
+ store:NewsGroupStore
+ fetchNewsGroups():List<NewsGroup>
+ getNewsGroup(NewsGroupId:string):NewsGroup
NewsArticleService
+ store:NewsArticleStore
+ fetchNewsArticles(GroupId:string):List<NewsArticle>
+ getNewsArticle(NewsArticleId:string):NewsArticle
+ getNewsArticles(NewsSourceId:string):List<NewsArticle>
+ getNewsArticles(Category:string):List<NewsArticle>
NewsSourceService
+ store:NewsSourceStore
+ fetchNewsSources():List<NewsSources>
+ getNewsSource(NewsSourceId:string):NewsSource
UserService
+ user:User
+ updateNotificationId(notificationId:string):void
+ updateDeviceId(deviceId:string):void
Stores
NewsGroupStore
+ newsGroups:HashMap<string, NewsGroup>
+ updateOrAddNewsGroup(NewsGroup:NewsSource):void
+ getNewsGroup(groupId:string):NewsGroup
+ hasNewsGroup(groupId:string):bool
NewsArticleStore
+ newsArticles:HashMap<string, NewsArticle>
+ updateOrAddNewsArticle(article:NewsArticle):void
+ removeNewsArticle(articleId:string):void
+ getNewsArticle(NewsArticleId:string):NewsArticle
+ getNewsArticles(NewsSourceId:string):List<NewsArticle>
+ getNewsArticles(Category:string):List<NewsArticles>
+ hasNewsArticle(NewsArticleId:string):bool
NewsSourceStore
+ newsSources:HashMap<string, NewsSource>
+ updateOrAddNewsSource(source:NewsSource):void
+ removeNewsSource(sourceId:string):void
+ getNewsSource(NewsSourceId:string):NewsArticle
+ hasNewsSource(NewsSourceId:string):bool
Models
NewsGroup
+ category:string
+ mainNewsArticle:NewsArticle
+ newsArticles:List<NewsArticle>
NewsArticle
+ history:List<NewsArticle>
+ source:NewsSource
+ headline:string
+ link:string
+ date:Date
+ analysisResult:string
NewsSource
+ name:string
+ logo:Byte[ ]
+ link:string
+ newsArticles:List<NewsArticle>
User
+ notificationToken:string
+ deviceID:string