



UiT The Arctic University of Norway

Faculty of Science and Technology

Healthy Transportation Choices with IoT and Smart Nudging

A research to explore how can people's reliance on private transportation be reduced with the help of digital intervention

Fazal Mehmood

INF-3990 Master's thesis in Computer Science

June 2021

Declaration

I, Fazal Mehmood, hereby declare that this thesis in its entirety has been composed by myself and has not been submitted, in whole or part for any previous degree or professional qualification. However, I have been part of the Open Distributed Systems (ODS) research group, working in collaboration with Anders Andersen and Randi Karlsen as supervisors. The structure of the thesis is composed by referring to work previously done in the same arena specifically by Cosmin Radu Crciun and Jemea Lady Limunga. Their related research work also gave me insight and direction into the work that was previously done in this field. Any other form of information or inspiration gotten from other peoples work has been well referenced.

Abstract

Modern technology has provided people with ease of living but at the same time has given birth to the problems of equally modern nature. For instance, high reliance on private transportation has resulted in unintended consequences such as high level of air pollution and congestion in urban cities. Another main disadvantage that is often overlooked is related to the rise of several noncommunicable diseases that are caused due to excessive dependence on cars and lack of physical activity. This thesis is entirely dedicated to encounter serious hazards of lack of physical activity by choosing unhealthy transportation choices. The interaction between people and the computers has become ubiquitous over the span of years. People interact in digital environment for a number of reasons. From checking weather conditions to running multinational trading businesses, computer driven digital automation has taken over what has always remained a manual handiwork. Cognizant of the potency of computer driven services and its authority, we propose applying nudge theory to encourage users to choose healthy options when it comes to any type of mobility. The first step involves researching about collecting, storing and performing analysis on data from different resources and then suggesting different techniques to manipulate it in order to perform an effective nudge.

Table of Contents

Declaration	i
Abstract	ii
Table of Contents	v
List of Tables	vii
List of Figures	ix
Abbreviations	x
1 Introduction	1
1.1 Motivation	1
1.2 Challenges	1
1.3 Goals	2
1.4 Problem Statement	2
1.5 Approach	2
1.6 Results	2
1.7 Contribution	2
1.8 Limitations	3
1.9 Outline	3
2 Background	5
2.1 Health and Transportation	5
2.2 Nudging	5
2.3 Digital Nudging	6
2.4 Persuasive Design	7
2.5 Big data	9
2.5.1 Characteristics of Big Data	11
2.5.2 Classification of Big Data	12
2.5.3 Management of Big Data	13
2.6 Related Work	19
2.6.1 Green Transportation choices with IoT and Smart Nudging (Andersen et al., 2018)	19
2.6.2 Shop with your DNA (Vaughan, 2019)	19
3 Architecture and Design	21
3.1 Main Approach	21
3.2 Architecture	22
3.2.1 Client	22
3.2.2 Web APIs	22

3.2.3	Data Management Service	23
3.2.4	Nudge Service	23
3.2.5	External Sources	23
3.2.6	Persistent Storage	23
3.2.7	Temporary Storage	24
3.3	Database Structure	24
3.3.1	Test Users	24
3.3.2	Test Queries	25
3.3.3	Authentication	26
3.3.4	Nudge data	27
3.3.5	Nudge Verdict	28
3.3.6	User History	29
3.3.7	Data Model and cardinality	30
4	Implementation	31
4.1	Client Application	31
4.1.1	React JS framework	31
4.1.2	Client Application Content	31
4.1.3	Client Application Styling	31
4.1.4	Client Application logic	32
4.2	Back-End	32
4.2.1	Node JS	32
4.2.2	Express	32
4.3	Database	32
4.3.1	Mysql	32
4.3.2	Database Visualization Tool	33
4.3.3	Virtualization	33
4.3.4	web-scraping	33
4.4	Client Application Overview	33
4.4.1	Authentication	33
4.4.2	Query From	33
4.4.3	Results Component	33
4.5	Client Application Implementation	34
4.6	Back-end Implementation	34
4.7	Datbase	35
4.8	External Web Apis	36
4.8.1	Weather Data	36
4.8.2	Google Directions Api	36
4.8.3	Transit Mode Data	36
5	Evaluation	39
5.1	Internal Evaluation	39
5.1.1	Mapbox API	39
5.1.2	Beautiful Soup	40
5.1.3	Puppeteer	40
5.1.4	Data Collection	41
5.1.5	Data processing	41
5.2	External Evaluation	41
5.2.1	Accepted Nudges	41
5.2.2	Rejected Nudges	41

6	Discussion	43
6.1	Test Users	43
6.2	GDPR	43
6.3	consent forms	43
6.4	Legality of Web scraping	44
6.5	Involuntary solicitation	44
6.6	Other Insights	44
7	FutureWork	45
8	Conclusion	47
	Bibliography	47

List of Tables

2.1	Selection of Nudge principles, Description and Examples (Markus Weinmann, 2016) (Richard H. Thaler, April, 2010)	8
-----	---	---

List of Figures

2.1	Digital Nudge Life Cycle (C. Schneider and vom Brocke, Jul 2018).	7
2.2	Intersection Model (Mimmi Castmo, June, 2018) (Richard H. Thaler, April, 2010) (Oinas-Kukkonen and Harjumaa, 2009).	8
2.3	Data Volume Vs. Computational Speed (C.L. Philip Chen, January ,2014).	10
2.4	50% of 560 Companies believe that they can benefit from Big Data (C.L. Philip Chen, January ,2014).	10
2.5	Survey on Companies that tries to Benefit from Big Data Analytics (Russom, 2011) . . .	11
2.6	Characteristics of Big Data (Ibrahim Abaker Targio Hashem, July, 2014)	11
2.7	Classification of Big Data (Ibrahim Abaker Targio Hashem, July, 2014) & (Makridis, August, 2018)	12
2.8	Using commodity computing for Big data (Ibrahim Abaker Targio Hashema, January 2015)	14
2.9	The Olap Process (Rouse, 2018)	18
3.1	Shows an overview of the architecture, the data components and their relationship. . . .	21
3.2	Figure attempts to depict the basic architecture of the application in a graphical form where different components and their corresponding relations can be observed. The arrows represent different events and the flow of data.	23
3.3	Detailed Architecture of the Nudge App.	24
3.4	Test User	25
3.5	Users queries	26
3.6	Authentication	26
3.7	Nudge Data	27
3.8	Nudge Verdict	28
3.9	Users History	29
3.10	Data Tables and their Relations	30
4.1	Bus schedul snippet (TromsKortet, 2020)	37
5.1	MapBox api	40

Abbreviations

DALY	=	Disability Adjusted-Life Years
WHO	=	World Health Organization
ACID	=	Atomicity, Consistency, Isolation, Durability
API	=	Application Programming Interface
BASE	=	Basically Available, Soft state, Eventually consistent
CAP	=	consistency, availability, partition tolerance
CSS	=	Cascading Style Sheets
HTTP	=	Hypertext Transfer Protocol
OLAP	=	On-Line Analytical Processing
RDMS	=	Relational Database Management System
SPA	=	Single Page Application
SQL	=	Sequential Query Language
ES	=	Ecma Script
XML	=	eXtensible Markup Language
IDE	=	Integrated Development Environment
JSON	=	JavaScript Object Notation
NoSQL	=	Not only SQL
GDPR	=	General Data Protection Regulation

Introduction

1.1 Motivation

Physical activity and exercise have been considered as one of the most important daily activities to maintain a healthy lifestyle. During the old times, people would find themselves engaged in extensive labour and going long distances with either by foot or with the help of animals. It would sometimes even involve carrying heavy goods on their backs or heads. The entire day would be consumed with physical commotion as that was the only available option.

As the human mind has always been busy tailoring new ways to attain comfort and efficiency, as a result newer innovations and discoveries were made in all aspects of life including transportation. From using horses to attaching carts to them and than creating engines, the life cycle of different means of travel has taken several twists and turns. Despite it made mobility more convenient and far less time consuming, the drawbacks it has brought to the society are also immense and can not be overlooked.

Lack of physical activity has been known to cause several chronic noncommunicable diseases such as heart diseases, mental illness, obesity and high blood pressure. According to a report by WHO (World Health Organization), there are nearly 3.2 million deaths and 69.3 million DALYs (Disability Adjusted life Years) that are caused due to lack of physical activity (Shanthi Mendis, 2014) each year.

Based on the above mentioned facts, we would like to dedicate this project to finding ways to promote physical activity by encouraging users to choose healthy transportation choices.

1.2 Challenges

The biggest challenge to overcome is how to effectively nudge people when it comes to deciding what type of transport to take from one place to another. For that, we need to collect large amount of heterogeneous data from different resources. It would involve several factors to take into account such as Users data, bus schedules, weather conditions, user's preferences and distance/time of journeys and a number of other factors that can be incorporated to devise a strategy in order to suggest any means of travel that would involve some sort of physical activity. We also need to filter the raw data and combine and transform usable data after conducting analysis on it.

Furthermore, we aim to dwell into understanding the psychological aspects of users and what factors dictate their behavior towards certain recommendations.

1.3 Goals

The ultimate goal is to investigate how a persuasive system can be designed that effectively inspire it's users to adapt to transportation which as a result, enables them to be physically more active. For that we intent to acquire and process data for nudge purpose.

We also build a limited prototype implementation that is exposed to multiple volunteers and than collect their feedback and determine what can be improved. We look into different sources and explore several different technologies and techniques to investigate the best approach to perform smart nudges.

1.4 Problem Statement

Physical inactivity due to high use of private transportation is one of the leading causes for several diseases. For decades, cars have remained one of the most convenient and commonly used source of transportation. To motivate people to reduce their reliance on private transportation, a digital nudge is to be employed that can change overall behavior of the user.

This can not be done unless a user is given strong incentive to change his/her behavior. Our task is to find ways a user can be influenced in this regard. For that, we need to collect data from heterogeneous resources and based on the data we need to propose a nudge that can steer users towards a desired choice.

1.5 Approach

Collecting the data regarding different modes of transportation is the basic element that we require since it is pivotal that the nudge will be performed based on the data we will have on our hand. For that, we need not only real time dynamic data that is susceptible to changes but also relevant data that matches user's requirements.

Since different modes of transportation follow strict schedules of their own besides the schedule or plan the user may have made, therefore we need real time data that provides information on different types of data when it comes to travelling by means of different modes. For example, if the user is interested in taking bus, than the schedule of the transit and how many interchanges user may have to make in order to reach it's destination is important.

1.6 Results

Our research led us to reveal that the users who do not own a private car are relatively easier to nudge than those who owns one. Furthermore, we observed that users would prefer to accept a nudge that encouraged them to take a walk when the nudge was based on shorter distances. We also discovered that whether is an important component that plays a crucial role in dictating users behavior. Finally, a generic nudge presented to all users would invoke varying and most often undesirable responses from different users therefore the nudge must be tailored for the specific users. We also perform analysis of different technologies that can be useful in creating an actual app.

1.7 Contribution

We have contributed in this project by introducing an approach where users personal information coupled with different factors concerned with a specific journey is used to create nudges. We also participated in researching on what type of data would be needed to devise a nudge for our users. We accessed data from different external resources and our efforts also revolve around a small scale experimentation based on a limited prototype implementation of the project that collects, stores, analyse and presents relative data to the test users. Furthermore, we observe test users feedback to our nudges and try to understand

what are the main elements that govern their behaviour towards nudges. We also explore different factors that lead a user towards rejecting a nudge.

1.8 Limitations

The real time data in our case is mainly the data that is relevant to transit and weather. For transit data, we have to use web-crawlers/web-scrapers to fetch real time data and give it's access to the user. Most websites these days are heavily java-script rendered and in order to perform scraping on such web pages can be time inefficient.

We used react js and Node js to build a prototype implementation. We believe the real application should also work on smart phones and should therefore be built using frameworks such as react native etc. We did not incorporate data regarding Skiing tracks and their availability due to lack of time therefore our implementation does not provide any insight on how successful or unsuccessful such nudges would be.

An application that tracks and monitors the travel history of users must implement a strong security and privacy mechanism to ensure the that the sensitive data is protected. We did not investigate that in too much depth, however, we did suggest some solutions that are discussed in discussion part.

1.9 Outline

This project comprises of 7 chapters. The introduction of this project has been dedicated an entire chapter termed also as Introduction where we throw light on our motivation to investigate the problem in question, different challenges, and our goals. In the background chapter, we reveal the history of Digital Nudging and relevant techniques, the innovation of modern means of transportation and their consequences. Furthermore, we encompasses on the concept of Big data and how it is handled through different processing tools. We also discuss how this data can be effectively stored. Finally, we include few examples of related work that has been done in this area. Chapter 3 (Architecture and Design) details the architecture and design of our solution. Chapter 4 (Implementation) uncovers different tools and technologies that can be used to develop the smart nudging app. In chapter 5, we perform a critical analysis and evaluation of our research and provide reasoning for different outcomes. The discussion chapter describes different techniques that can be incorporated in order to further improve the design. We also discuss the outcomes of our research and the drawback of our assumptions. It also takes into account the different important factors that were not extensively investigated in the project. Chapter 6 (Future work) briefly suggests any future work that can be done and finally we conclude our work in chapter 7 (Conclusion).

Background

2.1 Health and Transportation

Cars have remained one of the most convenient and commonly used means of travel for over a century. Undoubtedly, the invention and innovation of cars spanning over several decades, may have changed the public perspective on distance and long tiresome journeys, however the advancement also altered the overall layout of urban cities and than gradually disseminated this change globally. Newer Cities were built, older ones transformed and than connected to each other through bridges and in order to accommodate smooth flow of traffic, roads and highways were built around those cities. Thus the invention of modern means of transportation not only changed the way these cities looked like but also how they would eventually sound.

Despite cars proved to be a comfortable and reliable way of travelling, they also created some serious unforeseen consequences to the overall health of people. Apart from air pollution, cars are also responsible for limiting the physical activity of their consumers. A 12 year study comprising of 300,000 people suggests that lack of physical activity is more harmful and resulting in a higher rate of fatalities than obesity (Gallagher, 2015). A number of other diseases such as High Blood Pressure, Hypertension and cardiovascular diseases are related to lack of physical activity. Furthermore, the advent of cars also impacted other healthy means of travel such as walking and cycling. According to a survey conducted by Gallup, 83% of U.S citizens frequently drives a car (Brenn, 2018).

2.2 Nudging

Human decision making is not without it's flaws. There are various social and psychological factors that influence a user's choice while making a decision (Richard H. Thaler, 2013). These psychological effects can either consciously or unconsciously lead a user to make predictable mistakes that can further result in making poor choices (Tobias Mirsch, 2017a). The choices that do not have any immediate implications are easily taken and for that users often rely on heuristics. It is a debatable subject whether there are any decisions which do not have a long term impact no matter how trivial they appear. Of course, given the timing, duration, circumstance and the type of a decision can make a difference to the life of user and the society.

Owing to this fact, the concept of choice architecture was introduced. The term was first coined by Thaler and Sunstein (R.H. Thaler, 2008) which suggests that the many ways a choice can be presented to a user in fact determines what choice would be ultimately made. Furthermore, it describes the choice architect as same as the design of a building where the overall structure of the building such as placement of doorways, hallways and allocation of bathrooms etc, guides it's inhabitants(Weber, 2012) .

There are several examples of choice architecture in a physical sphere. However, A famous example for that is the change in design of a cafeteria to steer students towards a healthier meal without restrict-

ing them from choosing unhealthy foods available on the menu. This is achieved by putting healthy food options at eye level, thus making it more convenient to reach when compared to unhealthy options (R.H. Thaler, 2008). Hence creating a choice environment and then making changes to it in order to change the preferences of users and encouraging them to make a desirable decision is called Nudging.

The concept of nudging is no longer a theoretical concept. After the inception of the idea, Nudging is not limited to the books of Behavioral economics. In fact governments in the US, UK, Germany and many more have implemented departments of behavioral economics (Team, 2015-2016). The fact that decisions are becoming more and more influenced by digital environment, the nudges too can now be performed in a digital manner as well.

2.3 Digital Nudging

The practice of making decisions while in front of a screen has become a common practice. In fact, a growing number of decisions these days are governed by some sort of digital intervention. For example, the choice of clothing on a particular day can rely on the digitally provided weather forecast for that specific day.

Furthermore, picking the right flight or investing in a financial firm might entirely rely on the electronic content specially catered for bringing the client to a mutually beneficial arrangement. Hence from making trivial choices to significantly important decisions, digital content and the way it is presented has spontaneously become the driving force in dictating consumer's preferences. Such an approach where a user's choices can be affected through UI design elements in digital environments is referred to as **Digital Nudging** (Tobias Mirsch, 2017b). These UI design elements can consist of images, text, questions or audio/video.

A number of worldwide issues such as global warming, non communicable diseases and air pollution etc require awareness among common masses. For that, it is crucial that real time guidance and support is provided to the public. Owing to this fact, a behavioral change is what we require which can be accomplished by providing effective nudges. There are number of factors that need to be carefully considered In order to present a successful nudge to the user.

1. **Non-Restricting:** It is important to remember that when nudging a user, the freedom of the user to make their own decisions should not be compromised. The nudge should be merely suggestive and should not in any way restrict user's liberty.
2. **Practicality:** Another important rule to follow when providing a nudge is to make sure that the nudge being performed is a possible one. For example, nudging a user to take an impractical mean of transportation under special circumstances (e.g. storm, fog, etc.) might be a useless attempt to make a nudge. Similarly, the nudge must be tailored for different conditions and should be implemented in a practical manner.
3. **Clarity:** The ultimate objective to nudge a user should be thought and considered beforehand. It needs to be clear what is the purpose behind nudging a user. For that, the goal should be clearly defined and the design of the nudge system should take into account all the factors that revolve around it.
4. **Transparency:** The nudges must be transparent and all the possible options should be given to the user without clouding any that are not desired.

There are several steps that are involved when performing a nudge. **Fig. 2.1** shows the different components that govern the life cycle of a digital nudge.

The first step involves the basic design and the definition of the goal. The design of the system should consider the elements that needs to be incorporated so that a nudge can be performed. The detailed

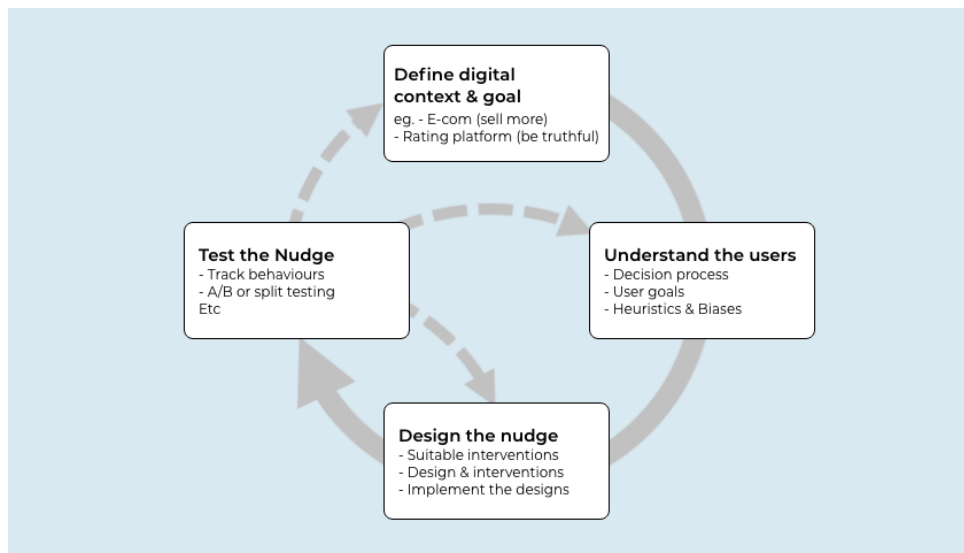


Figure 2.1: Digital Nudge Life Cycle (C. Schneider and vom Brocke, Jul 2018).

Nudge principles are illustrated in **Table. 2.1**.

Secondly, it is also important to understand the users. Studies have shown that the efficiency of nudge is highly dependant on who the nudge is being sent to. The nudge should be customised based on the culture, gender, age group and social values etc. To summarise, the target audience should be well studied and their goals should be premeditated.

Step 3 involves the practical design of the nudge that comprises of clearly defined goals, understanding of users, nudging elements to change a behavior. All these attributes should also be meticulously implemented while designing the nudge.

The last step in the nudge life cycle involves testing the nudge for it's effectiveness. Testing the nudge would give feedback regarding the degree of effectiveness of the nudge. This step is very important since it helps the nudge architects to determine whether the users accepted the nudge or not. By tracking behaviour, it could be established which nudges were successful and which were not. The designer can than retract to the second and third step of the Digital Nudge life cycle.

2.4 Persuasive Design

Persuasive design is another prominent methodology that resonates with some of the design elements and principles of nudging. Like digital Nudging, persuasive design is also a design strategy that is used to steer people towards making certain decisions (Mimmi Castmo, June, 2018).

Both Digital nudging and Persuasive Design relies on different psychological and social theories and is used in different digital choice environments where decisions are made such as e-commerce and organizational management (Mimmi Castmo, June, 2018) (Markus Weinmann, 2016).

Despite Digital nudging and persuasive design serve same fundamental objective that is to trigger some sort of behavioral change, both these strategies can not be considered synonymous to each other. The main difference that separates Digital Nudging from Persuasive design is the design strategy of both approaches(Mimmi Castmo, June, 2018). Persuasive design employs a behavioral oriented strategy and hence is more attitude oriented (Katarina Segersthl, 2017) while Digital nudging deals with decision making and is used to navigate users towards set goals or behaviour. **Figure. 2.2** based on (Richard H. Thaler, April, 2010) & (Oinas-Kukkonen and Harjumaa, 2009) illustrates an intersection model be-

Nudge Principles	Description	Example
INCENTIVE	Making incentives more salient to increase their effectiveness	Telephones that are programmed to display the running costs of phone calls
Understanding Mapping	Mapping information that is difficult to evaluate to familiar evaluation schemes	Mapping megapixels to maximum printable size when advertising a digital camera instead of pointing to megapixels
Defaults	Preselecting options by setting default options	Automatic renewal of subscriptions
Giving Feedback	Providing users with feedback when they are doing well and when they are making mistakes	Electronic road signs with smiling or sad faces depending on the drivers speed
Expecting Error	Expecting users to make errors and being as forgiving as possible	Requiring people at an ATM to retrieve the card before they receive their money in order to help them avoid forgetting the card.
Structure Complex Choices	Listing all the attributes of all the alternatives and letting people make trade-offs when necessary	Online product configuration systems that make choices simpler by guiding users through the purchase process

Table 2.1: Selection of Nudge principles, Description and Examples (Markus Weinmann, 2016) (Richard H. Thaler, April, 2010)

tween the two strategies where the similarities and differences can be observed.

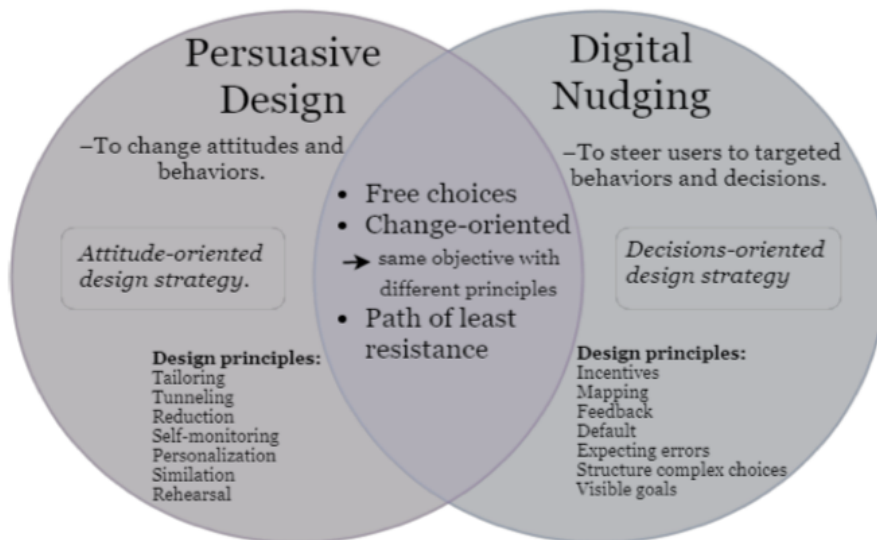


Figure 2.2: Intersection Model (Mimmi Castmo, June, 2018) (Richard H. Thaler, April, 2010) (Oinas-Kukkonen and Harjumaa, 2009).

The design principles of Persuasive design is different than that of Digital Nudging as illustrated in **Table 2.1**. However, both strategies share similarities when it comes to allowing freedom of choice, influencing behaviour change and making desirable behavior the path of least resistance.

The detailed account of Persuasive Design and its principles are beyond the scope of this project. Following are a minor description of each design principle of Persuasive Design.

1. **Tailoring:** Tailoring is done when information is customised for a specific user since it is more effective to use. It is important to remember the interest, needs and other factors so that the user can be motivated towards a certain behaviour (Oinas-Kukkonen and Harjumaa, 2009).
2. **Tunneling:** Refers to the guided process of making user get close to the target behavior through experience.
3. **Reduction:** In Reduction, complex activity is broken into smaller steps and those steps that are not relevant and unnecessary are removed (PI Kraft, June 2008). The users almost always prefer the path of least resistance, therefore it is important to offer a way to accomplish a task by using the least amount of effort.
4. **Self Monitoring:** It is important that the users are aware of their activity so that they can make changes to their behaviour (Mimmi Castmo, June, 2018). It can work as a reminder that can inform user about the direction they are going in (Oinas-Kukkonen and Harjumaa, 2009).
5. **Personalization:** Refers to the technique of catering the content according to a specific user. The more the content is personalised, the more it holds the ability to persuade a user (Oinas-Kukkonen and Harjumaa, 2009).
6. **Simulation:** is to allow user to observe the relationship between the cause and the effect with consideration of user's actions and behaviours (Oinas-Kukkonen and Harjumaa, 2009).
7. **Rehearsal:** Rehearsal allows users to adjust their attitudes and behavior through rehearsing and practice of an action (Oinas-Kukkonen and Harjumaa, 2009).. For example, a flying simulator can enable pilots to rehearse flying in different weather conditions (Oinas-Kukkonen and Harjumaa, 2009).

2.5 Big data

Big data is a term used for massive amount of data that comes with its own set of challenges to deal with. These data are generated from different sources such as online transactions, videos, audios, images, click streams, logs, posts, search queries, health records, social networking interactions, science data, sensors and mobile phones and their applications (Paul C. Zikopoulos, 2012).

The management of raw data has become a challenge as the the rate at which information is gener- ated exceed Moor’s law, (C.L. Philip Chen, January ,2014).

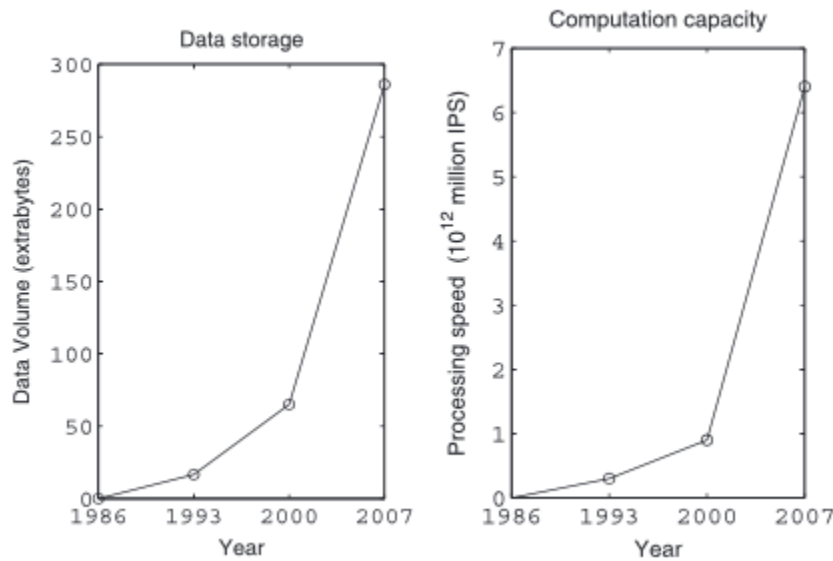


Figure 2.3: Data Volume Vs. Computational Speed (C.L. Philip Chen, January ,2014).

Unlike the conventional form of data, Big data relies on special tools and methods to be handled and can not be processed or analyzed using traditional tools (Paul C. Zikopoulos, 2012). The generation of new data on daily basis is not only immense but also very difficult to process. According to (James Manyika, May, 2011), 30 billion pieces of content are shared on Facebook every month and the scale is expected to increase in future. Moreover, 72 hours of videos are uploaded to YouTube every minute. Google on a daily basis processes about 24 Petabytes (24,000 Terabytes) (Thomas H. Davenport and Bean, Fall ,2012). Many companies have access to such wealth of data but find it very difficult to draw value out of it since the data is scattered, disorganised and unstructured (Paul C. Zikopoulos, 2012). Therefore, in an era where anything can be stored and data is produced on a massive and rapid scale, the dire need for organising, managing, and getting value out of huge amount of raw data is unprecedented.

Although the phenomenon of big data and it’s accession is seen as a breach of privacy by the general public, the role it can play to devise not only private commerce for better but also national economies can not be overlooked. According to a research (James Manyika, May, 2011), data can play a significant role in benefiting world economy by enhancing productivity as well as producing economic surplus and other opportunities for the consumers. Hence ”Using big data yields to better predictions and using better predictions yields to better decisions (Andrew McAfee, October, 2012)”

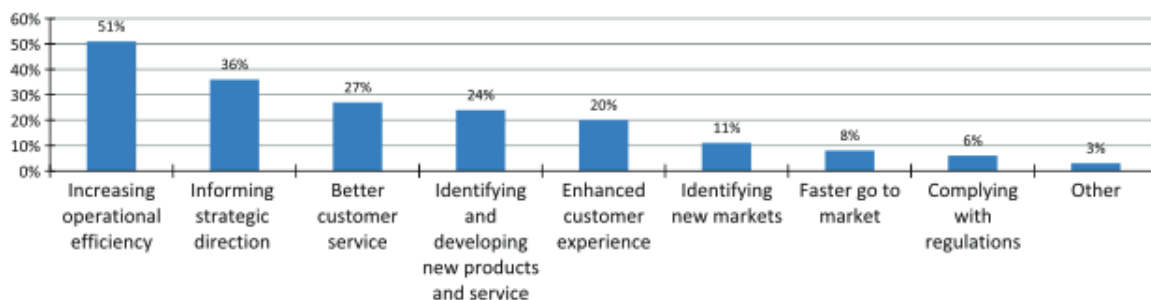


Figure 2.4: 50% of 560 Companies believe that they can benefit from Big Data (C.L. Philip Chen, January ,2014).

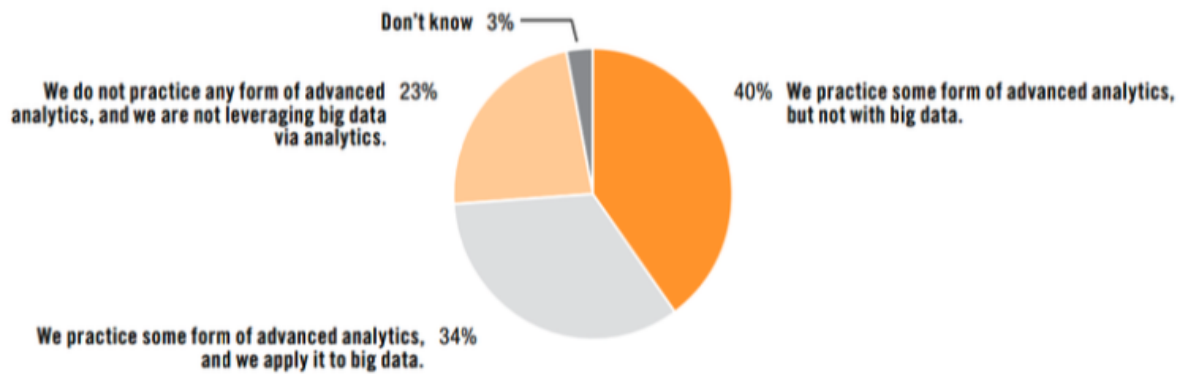


Figure 2.5: Survey on Companies that tries to Benefit from Big Data Analytics (Russom, 2011)

2.5.1 Characteristics of Big Data

Big data can be defined by the following characteristics.

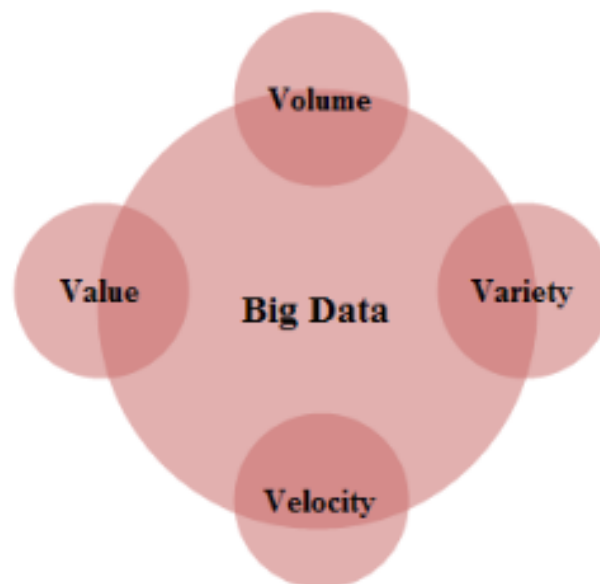


Figure 2.6: Characteristics of Big Data (Ibrahim Abaker Targio Hashem, July, 2014)

1. **Volume:** The term volume refers to the size of data set. It is one of the immediate challenges to overcome. The ability to gain from by processing huge chunks of data is the biggest motivation of data scientists (Ishwarappa, 2015).
2. **Variety:** Refers to the different categories the data can be classified into. Big data is usually scattered, heterogeneous, and very arduous to analyze.
3. **Velocity:** Refers to the increasing speed at which the data is produced and can be stored, processed and analyzed by relational databases (Ishwarappa, 2015).
4. **Value:** This is the ultimate goal to achieve when concerned with big data. The purpose of dealing with big data is motivated by extracting value out of it. The value of the data must not exceed the computational cost it requires for it to be valuable.

Characteristics of Big data can be further extended to a few more Vs based on the requirement. Following are some of the other attributes of Big Data that are worth mentioning.

5. **Veracity:** This characteristic correspond to the degree of the correctness of data. When dealing with big data, one may encounter undesired data that needs to be filtered out in order to acquire accuracy of the data.
6. **Volatility:** Volatility concerns with the degree at which the data modifies. As explained above, big data generates at a very high speed and is also susceptible to high degree of variation. Therefore it is important to take into consideration when to utilize this data before it is destroyed.
7. **Validity:** Validity covers how valid the data is. Big data is often context oriented and is valid with respect to the requirement. Data might be useful for one particular application and not so much for the other. In order for data to become valid, special tools are used to clean and organize it. According to Forbes, data scientists spend 60% of their time cleaning the data so that it can become valid (Press, March, 2016).

2.5.2 Classification of Big Data

Big data can be classified based on the source, content format, relevant data stores, Data staging Techniques, and Data processing as shown in **Figure. 2.7**. Big data can be classified based on the overflow

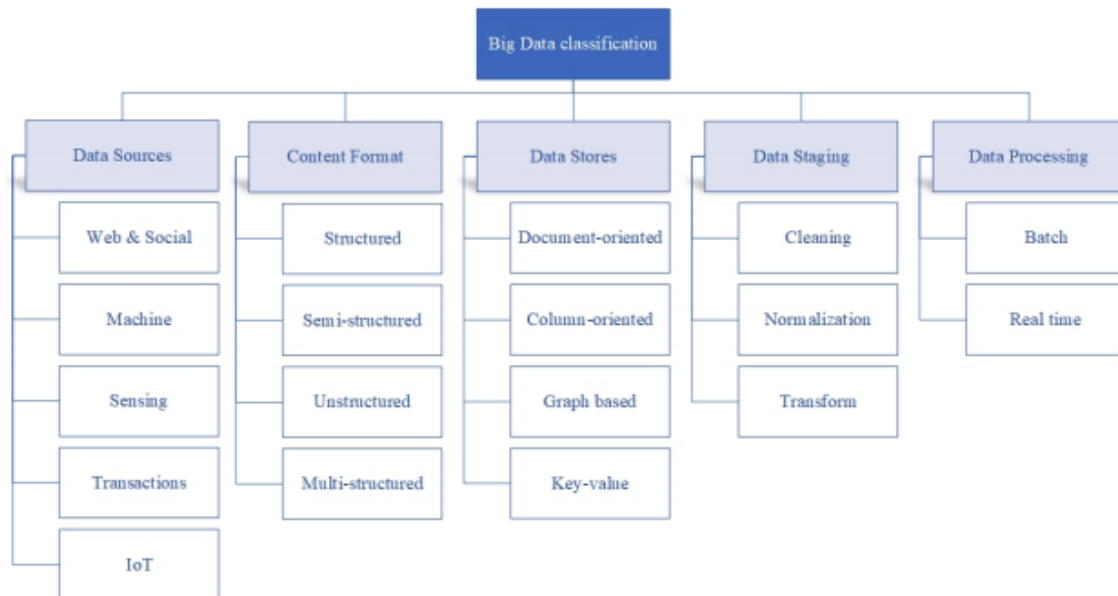


Figure 2.7: Classification of Big Data (Ibrahim Abaker Targio Hashem, July, 2014) & (Makridis, August, 2018)

of information from different sources such as data generated by sensors, Machine, transactions, or medical devices and IoT. This data can be further classified based on it's content format. Data can either be entirely structured or unstructured. An example of structured data is traditional Relational Database Management System (RDBMS). It can also be identified as semi-structured which refers to different entities within data lacking any pre-defined structures (Makridis, August, 2018). Multi-structured is referred to as a blend of all other data format (i.e. Structured, Unstructured, and Semi-Structured). The classification of big data also depends on the data store that is used in order to accumulate this data. There are several options available based on the type, quantity and operation of data that needs to be stored. Data stores can further have several categories such as Key-value stores, Document stores, Extensible record stores, and Relational Databases etc. (Cattell, December, 2010). The data goes through a staging process also called pre-processing before it can be finally processed. Staging technique involves cleaning the data, normalizing it and then transforming it as shown in Figure. 2.7. Undesired chunks of data are discarded, redundancy removed and the rest is formalized as to give a proper structure and format. Finally processing is performed on the homogenized data of batch type or stream type.

2.5.3 Management of Big Data

Data management of Big Data deals with organizing and processing of data. It also deals with technologies that should be used to store, analyze and process data. When dealing with Big Data, it is important to consider various challenges beforehand such as accessing, storing, manipulating, and representing data.

Following are some of the techniques and technologies that can be utilized while managing big data.

Relational Databases

Relational databases have remained the primary source of storing data for more than 3 decades. Most financial and business firms used to highly rely on relational data and thus needed a relational database for storing that data as well. RDBMS is still highly in demand and is used for storing relational data. The data is stored in various tables where columns define the data while the data itself resides in rows. These rows and columns reflects relations between data. The relational structure of data makes it easier to run queries on different tables at the same time. Sql (Structured Query Language) is used in to manipulate data stored in these tables.

RDBMS support ACID transaction properties (Sir, July, 2016) that guarantee several elements of a transaction. A short description of each ACID characteristic is as follows.

1. **Atomicity:** A transaction is considered atomic if the update is either propagated in it's entirety or is not executed at all.
2. **Consistency:** A transaction must be consistent. If not than it should abort.
3. **Isolation:** Parallel transaction should not make any changes to the system once the transaction has completed. The system should remain in the same as it was before the transaction.
4. **Durability:** The changes made during a transaction should be permanent and should not disappear if the system sustains a failure.

As the user requirement, hardware organization and the data itself changed, the need for more robust means of storing and processing data were sought out. Relational databases are also known to be used in instances where the preservation of persistent data is required which is not the case with Big data. The weaker support for scalability makes it very difficult for relational databases when handling Big data which is prone to rapid growth and susceptible to change.

The content format of big data is also a challenge since relational databases are not designed to cope with Big data Variety and it is very difficult to handle especially when it comes to unstructured data (Wael M.S. Yafooz, December 2013).

Cloud Computing

Due to the shortcomings of Relational databases, the need for an infrastructure capable of highly scalable and available means of data storage and processing facility was sort out. Cloud computing provides this infrastructure that allows a user to utilize a number of configurable resources such as accessing, storing and processing data. The term "cloud" is used to refer to the fact that geography of the computer systems no longer holds any prominence since the component of a software system need not to remain at a single location and might reside in multiple unseen computer systems that might as well be scattered over different locations (Hayes, 2008). The resources offered by cloud computing can be increased or reduced as per the requirement of a user and therefore Cloud computing is also often referred to as *on-demand computing* (Michael Armbrust, 2010).

As mentioned earlier, dealing with big data can be a very time consuming task that requires an infrastructure that can deal with the ever-changing and rapidly growing data. Cloud computing comes in handy

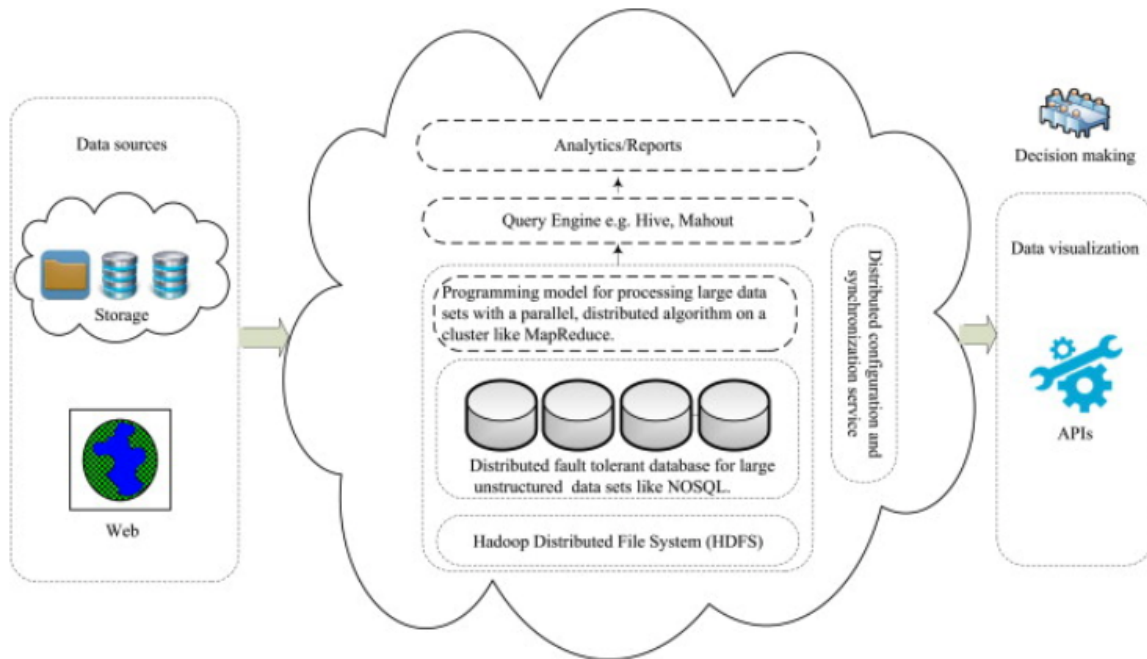


Figure 2.8: Using commodity computing for Big data (Ibrahim Abaker Targio Hashema, January 2015)

when dealing with such huge amount of data. Big data allows users to process distributed queries across multiple data-sets using commodity computing and return resultant sets efficiently (Ibrahim Abaker Targio Hashema, January 2015).

The data itself can be partitioned across multiple servers and then replicated further in order to enhance availability in the event of a server failure. Unlike RDBMS which supports ACID properties, systems that follow a model that allows users to partition and replicate data across the network are formulated by CAP-theorem. Introduced by Eric Brewer, Cap-theorem argues that there is a fundamental trade-off between consistency, availability, and network partition tolerance (Brewer, February 2012). Consistency roughly refers to a shared common and concurrent state between all the replicas of a single data item while availability simply ascertains that a response is received for a given request. In other words consistency is the property that each response from any server returns the right response to any request made to it while availability pertains to that each request eventually harvest a response from the server (Seth Gilbert, 2012). The system would have to prioritize availability over consistency in the event of a network partitioning since inaccessibility to the data entirely can hinder a task from completing. There are several techniques that are used to ensure consistency in distributed environments (Mustaque Ahmad, September, 1999) such as eventual, explicit or causal consistency (Valter Balegas, April, 2015) (Burckhardt, 2014). Nonetheless cloud storage offers one of the best solutions when it comes to dealing with big data, the need for handling this huge amount of unstructured data has led to the rise of No-sql technologies (Bhagal and Choksi, 2015).

No-Sql

One of the main reasons No-sql technologies grew rapidly popular amongst developers is the need for avoiding unwanted complexity. There are use cases where the ACID properties especially consistency that is inherently supported by RDBMS are not necessary (Strauch et al., 2011). No-sql databases can co-exist with relational sql databases. However, they address the following 3 main differences than that of RDBMS.

1. Nosql databases are non-relational and support unstructured data unlike RDBMS. They do not have a rigid schema but have either flexible or no schema at all.
2. Another property of Nosql databases is that they are very effective when it comes to scaling out.

The operation can be spread out to a number of server nodes. Hence it is easy to horizontally scale out No-sql databases as compared to RDBMS which were not designed for horizontal scaling.

3. Nosql databases prioritize availability by compromising consistency. They do not guarantee data integrity as effectively as that of RDBMS. Therefore they do not support ACID properties but rather facilitate BASE properties.

The term ACID reflects a paradigm of one database versus many users where the transaction occurs in an exclusive manner while allowing the possibility to change a value. Contrary to that, BASE properties involve around a scenario where data is scattered and the synchronization of data is infeasible (Chandra, 2015). Base properties are briefly described as follows.

1. **Basically available:**
Refers to a constant state of availability by implementing a high degree of replication.
2. **Soft state:**
Consistency is not guaranteed. The system might not share a same state therefore the user's application should be responsible for safeguarding consistency (Chandra, 2015).
3. **Eventually consistent:**
Updates are propagated to each replica over a course of period to ensure consistency. Therefore at a given time, the system may or may not be entirely consistent.

The main difference between RDBMS and Nosql technologies is that of the data model. Most Nosql databases fall in to four of the categories. These categories are as follows.

1. **Key-Value stores:**

Key value stores are used for storing key value pairs. They can be entirely schema free and relies on a hash table where a unique key points to a certain attribute or value corresponded to that specific key only (Bhogal and Choksi, 2015). Although key-value stores have a simpler structure, the schema free design allow them to be more efficient when it comes to querying distributed data. Key value stores can be further classified into In-memory key-value stores and persistent key value stores. In-memory key-value stores such as Redis (Redis, 2020) and Memcached (Memcached, 2020) allow storage of data in memory while persistent key value stores like BerkeleyDb and Voldemort keep data on disk (Grolinger et al., 2011).

2. **Column based Stores:**

Most relational databases management systems store data in the form of row. However, column based databases allow the data to be stored in different column families where a single unique row key, also known as primary key corresponds to these column families. Each row has a set of column families, and different rows can have different column families. These column families then further acts as key to identify different values stored in each column (Grolinger et al., 2011). This column based data storing technique makes it convenient to manipulate values in a column without touching any other columns (Mehra et al., 2015). Column based datastores are considered to be highly scalable and are well suited for workload such as On-line Analytical processing (OLAP) (Bhogal and Choksi, 2015) (Bonnet et al., 2011).

3. **Document Databases:**

Document databases have gained popularity due to their support for storing JSON (Javascript object notation). Document based data stores also allow users to work with data immediately without the need to define a schema upfront (Chasseur et al., 2013). Similar to key value stores, document stores uses unique keys to locate documents. However, document stores also allow indexing based on the content of documents which differentiates them from key value stores (Grolinger et al., 2011). They are schema free thus allowing to store unstructured data more conveniently.

4. Graph Databases:

As the name suggests, graph databases are based on a graph data model. The idea originated from graph theory. A graph database stores data in form of a graph where nodes represent objects and edges act as the relationship between the objects. Specific key value pairs are used to identify those relationship and properties linked with different nodes. Graph databases are widely used in cases where the relationship between data is more important than the data itself (Bhogal and Choksi, 2015) for example, social networking, generating recommendations, and in conducting forensic investigation (Moniruzzaman and Hossain, 2013).

Data Extraction tools

Big data analysis has emerged as an indispensable mean of discovering hidden insight and trends that occur frequently in large data sets. These data sets are too massive and complex for analysis without the intervention of computational tools (Jaseena and David, 2014). Technologies such as Spark, Hadoop and Mapreduce allow drawing meaningful knowledge from unstructured and complex massive amount of data.

Data Analytics

Before computers becoming a common household item, the storage of information remained a manual task where data relating to a particular case would reside in a hard form inside a register. For example, the data about a certain retail shop would give insight on what has been bought and by whom. However with the advent of computer systems and consequently online shopping, it became possible to not only hold the information what has been purchased but also what has been looked at. This gave an immense possibility to discover different trends in the behaviour of a particular customers/users or a genre of users and how these users can be influenced for further mutually inclusive gains (McAfee et al., 2012). The outburst of this information and it's potential to maximize the business gains paved it's path to developing techniques where such hidden potentials of massive amount of data can be revealed. Such techniques are termed as Data analytics. The techniques involve predictive analytics, data mining, statistical analysis, and complex SQL. The list further goes on to data visualization, artificial intelligence, natural language processing, and database capabilities that support analytics such as MapReduce, in-database analytics, in-memory databases, columnar data stores (Russom et al., 2011).

There are different types of analytics based on the technologies and architectures used for big data analytics. The major types of data analytics are briefly described as follows.

1. Predictive analysis:

In predictive analysis, the data is looked for any hidden insights that can allow to forecast any future trends. This method is mainly used in marketing spheres in order to understand customers preferences. Technologies and methods such as regression analysis, neural networks, and machine learning are used to preform predictive analysis (Watson, 2014).

2. Descriptive analysis:

Descriptive analytics deals with the past data events as in to make sense out of what has been happening. In other words, descriptive analytics allow to dig deep into data so as to find out "what has occurred" (Watson, 2014).

3. Diagnostic analysis:

Diagnostic analytics allows to perform a clinical examination of the data so that the root cause of different trends and behaviours can be determined within the data. This is to say, Diagnostic analytics answers the question, "why did it happen" (Youssra and Sara, 2018).

4. Prescriptive analysis:

Prescriptive analytics is more of a solution find tactic. While descriptive analytics deals with the past events concerning data, and predictive analytics, on the other hand is concerned with predicting future trends, Prescriptive analysis, however looks for the best possible solution in the light of results drawn from other types of analytics (Youssra and Sara, 2018).

OLAP

On-line analytical processing has become an essential component when it comes to dealing with decision support in online environments. OLAP is usually performed on data stored inside a data warehouse.

A data warehouse contains most granular, subject-oriented, time varying, non volatile and primitive data (Inmon, 2005). Since data warehouses often contain data from multiple operational databases, over time, the data stored inside a data warehouse can be orders of magnitude larger than operational databases and hence require specialized complex queries, scans, joins, and aggregates (Chaudhuri and Dayal, 1997).

Furthermore, to perform olap in order to achieve complex analysis and facilitate data visualization, the data inside a data warehouse is organized in a multidimensional database also referred to as a multi-dimensional cube where data is modeled based on different dimensions of interests. For example data regarding a sale firm might include different dimensions of interests such as sale item, cost, sales district and product etc (Chaudhuri and Dayal, 1997).

There are 5 different types of operations that can be performed the above-mentioned multidimensional data models. These OLAP operations are described briefly as follows on the next page:

- **Rollup:** Also referred to as drill up summarizes the data along a particular dimension.
- **Drill-down:** On the other hand, Drill down allows an analyst to narrow down on different aspects of data along a dimension.
- **Slice:** With slicing, a specific excerpt from the entire data can be observed. For example in case of a sale dimension, with slice, sales in a particular time frame can be analyzed.
- **Dice:** With this technique, data from multiple dimension can be analyzed in a combine fashion.
- **Pivot:** A new glimpse into data can be attained by simply rotating the different axes of the multi-dimensional cube.

The OLAP process

How data is prepared for online analytical processing (OLAP)

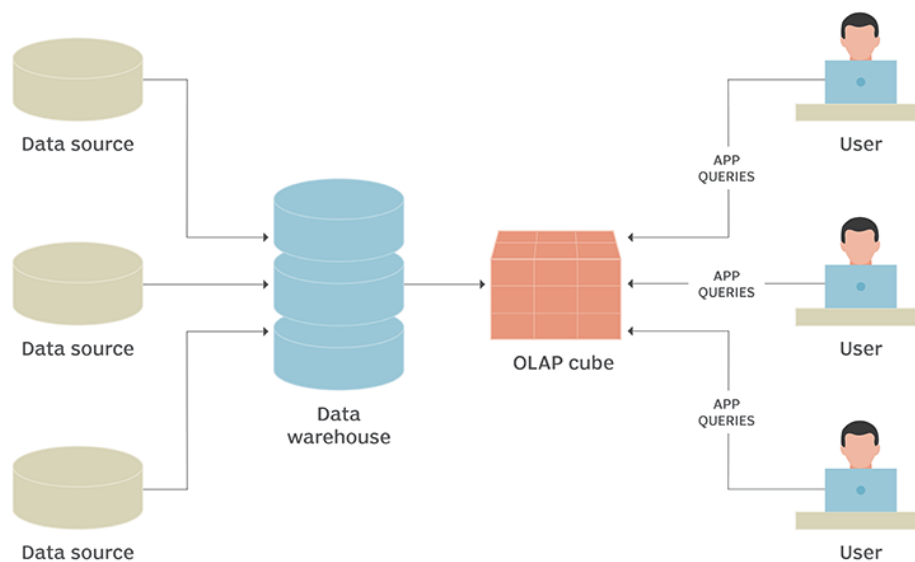


Figure 2.9: The Olap Process (Rouse, 2018)

OLAP operations can be combined with different data analytics technique such as Descriptive or predictive analysis etc. Data analytics technique are already described above.

Approximate Query Processing

The performance of OLAP is very important when it comes to making online decision. However with ever increasing large data sets, utilizing OLAP can become inefficient and costly. With the surge in the volume of data being produced, ironically the data analysis tools themselves have become the bottlenecks when dealing with data analytics and related activities. The software and hardware limitations also hinder a fast response to the query and consequently resulting in delayed decision making.

The traditional query processing require including all the relevant subject tuples and discarding irrelevant ones in order to make a best decision. In other words, precision is essentially favoured at the cost of efficiency, productivity and creativity (Miller, 1968). However, this arrangement would no longer be required lest the incomplete or partial inclusion of the relevant tuples can lead to an estimation of the decision that is as same as the traditional query processing where precise values are preferred. The later technique is termed as Approximate query processing.

In an AQP, only a small fraction of the relevant tuples is processed in order to provide fast, approximate answers (Mozafari, 2017). Existing Approximate query processing can be divided into following two categories:

- Online generation: The samples are selected online and then run against OLAP queries.
- Offline generation: The samples are pre-fetched offline and than run against OLAP queries.

APQ particularly gives better performance against aggregate functions such as SUM, AVG, COUNT, MAX and MIN etc (Li and Li, 2018).

2.6 Related Work

This section is dedicated to throw light on the practical and theoretical work done in the field of digital nudging and the factors surrounding it. Moreover, Publications that incited inspiration to work in this arena are also acknowledged.

2.6.1 Green Transportation choices with IoT and Smart Nudging (Andersen et al., 2018)

This paper brings about the different aspects of digital nudging. It specifically targets nudging for transportation that is environment friendly. It throws light on the concept and factors that revolve around smart nudging as well as make a comprehensive note on how an effective nudge can be performed by elaborating on all the steps involved. It also expands on different techniques and technologies. Furthermore, the paper describes different challenges that can be encountered and give suggestions on how to overcome them. The article is eloquently written and well referenced. This very article was used for aspiration to take this project as an academic endeavor.

2.6.2 Shop with your DNA (Vaughan, 2019)

A newly devised approach to allow customers to shop dietary items that are best for them based on their DNA. The DNA is analysed on the spot for different customers after testing for genes associated with caffeine metabolism and a predisposition for hypertension, high cholesterol and type 2 diabetes. A personal profile is created. The profile allows customers to shop different items by matching the barcodes and then comparing it to different parameters in their profile. A nudge is proposed to users every time they scan an item with either a red or green flash reflecting weather the item should be bought or avoided, respectively.

Architecture and Design

This chapter deals with the approach that is devised to solve the problem in question. We elaborate on general architecture, design and the management of the app. We will also throw light on the different data management techniques and the technologies that can allow us to implement our design.

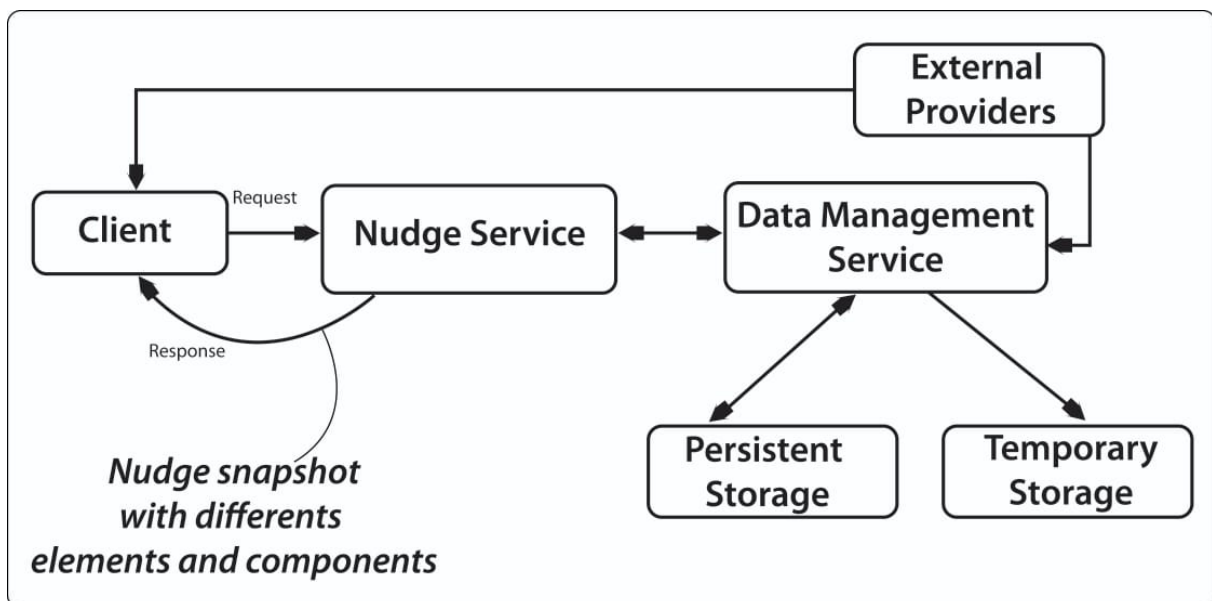


Figure 3.1: Shows an overview of the architecture, the data components and their relationship.

3.1 Main Approach

The nudging app that we are developing is not a conventional app that mainly deals with the concerns over its functionality. In other words, conventional apps are considered functional as long as there is a response to a request in a reasonable amount of time to a decent number of users. The nudging app however would not only rely on its correct and quick functionality but also the human factor of the users since the app aims towards triggering a behavioral change when it comes to the users preferences towards certain means of transportation. Thus the main challenge remains not only how to access, and manipulate data but how the processed data should be presented to the user that in the end convinces users to choose a healthier transport. The solution can be best summarized in a theoretical fashion by introducing the rule of 3 *rights*. The *right* type of data that must be presented in a *right* way at a *right* time. This simple rule further complicates our requirement as the right time, data and its presentation might differ from user to user. Therefore understanding the user and their preferences will also be an important part of our design.

The specifically tailored or adaptive nudges can only be designed if there is a certain pattern to follow. For that, some knowledge of user for which the nudge is being catered for is also necessary as personalized nudges can only be created if there is some insight available on the type of users we are dealing with. For that, different user profiles need to be administered in the system which will expand as the user continues to utilize the app.

A thorough analysis of the nudge data pertaining to specific users needs to be done in order to provide users with a constant feedback. This feedback would include any variation in their physical activity. The same information would be exploited while performing explicit dynamic nudges.

The design can be further divided into 3 components i.e UI, Nudge service and the data Management service. This is done due to each component's key role in the implementation of the APP. The front end would contain all the UI elements that users are limited to interact with. The UI needs to have not only ease of use prerequisites that most traditional apps are comprised of but also has to have a digital embodiment of the choice architecture that was previously discussed in section 2.2.

The nudge service decides what nudge to be pushed based on the users profile and data fetched from other sources. The nudge service takes several factors into account and follows a mathematical model before creating a nudge. Nudges are divided into two categories as well. The implicit and the explicit nudges. The implicit nudges would contain feedback data that comprises of different elements of users utility of the app over a course of period. These nudges follow a blending of the principles of persuasive design and digital nudging. The implicit nudges however, will be done through the design of UI elements of different data components alone and the very fashion they are displayed in. These nudges can be further divided into dynamic and static nudges. We will discuss these two types of nudges in detail later.

The 3rd component would be the data management service that would deal with the data manipulation and its storage. This component is the backbone of the entire project since it provides the foundation for the entire project to be build upon. We will dedicate spacial attention to this component while discussing the entire project.

3.2 Architecture

This section goes into detailed account of different services and the communication that is done between them. We also expand on some of the new concepts introduced in the previous section.

3.2.1 Client

The client side represents the front end UI component of the application that a user would interact with. The client app would run on both web as well as mobile. The app would be responsible for interacting with back-end technologies and governing different triggers. The client app can also interact independently with external sources of data for fetching specific data. The client app serves as the face of the entire project and this is where the nudges will be presented to the users in both explicit and implicit manner. The front end UI is used to serve the nudges that would be eventually created.

3.2.2 Web APIs

The web API layer would ensure passing the client requests to different services. It also ensures creating data in the right format and then sending it to the right recipients.

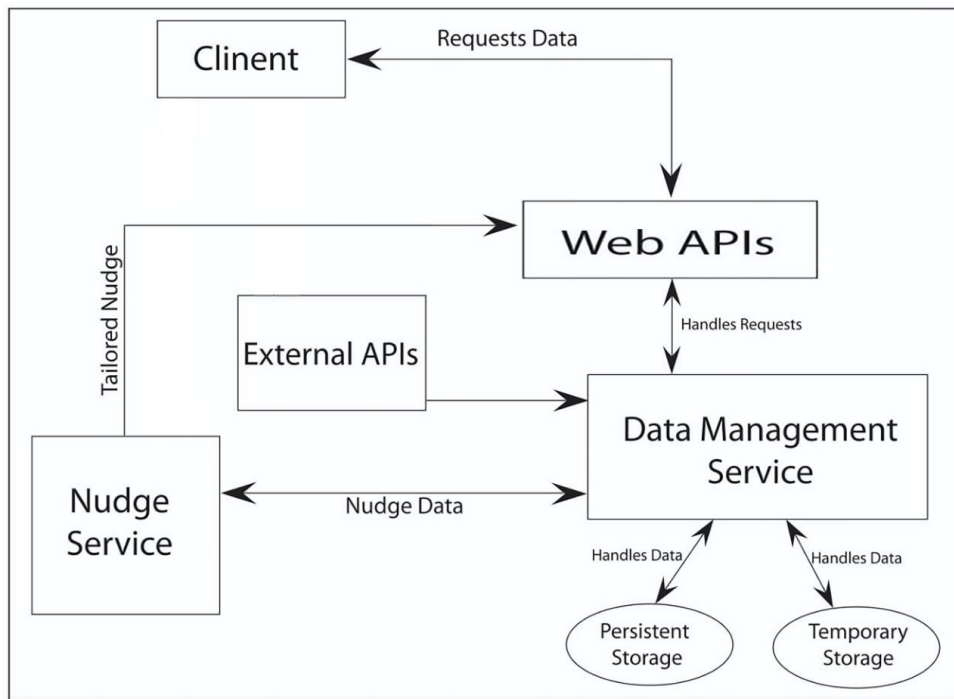


Figure 3.2: Figure attempts to depict the basic architecture of the application in a graphical form where different components and their corresponding relations can be observed. The arrows represent different events and the flow of data.

3.2.3 Data Management Service

As mentioned earlier, this service would be responsible for accession, preservation, conversion, and serving the data. The service would rely on different sources to store data such as persistent, temporary and data from external sources. The service also stores data based on the utility of users. However, the personal data of the users is preferred to be stored at the client side.

All other services would rely on Data management service for the accession of data regardless of where the data is located.

3.2.4 Nudge Service

This service is responsible for performing the actual nudge. Once a request has been received from the client through web API and the data service has processed the request, the nudge service then receives the data and then based on different parameters decides in which fashion to present data to the user. The ultimate goal is to present data to win users favor of choosing a transportation choice of our making. This is how an implicit nudge can be performed. The explicit nudge will highlight the drawbacks of certain transports and emphasis on advantages of the others. It would also allow a user to keep track of their past choices and their impact.

3.2.5 External Sources

The 3rd party data resources which provide different types of data would be used for accession of data. The external providers are used for weather forecast, traffic information, travel direction, travel distance and bus schedules. Long-term uniform data is stored and served from the data store in order to improve efficiency.

3.2.6 Persistent Storage

Relational database will serve to store persistent data. The database will store long term data queried from external data sources. Database also contains information about users, nudges and other historic

data.

3.2.7 Temporary Storage

This storage would serve the purpose of enhancing the efficiency of the App. Instead of making a remote trip to fetch data, the request can be served from the memory if the cached data suffices the requirement.

A detailed architecture of all components is illustrated by the following figure.

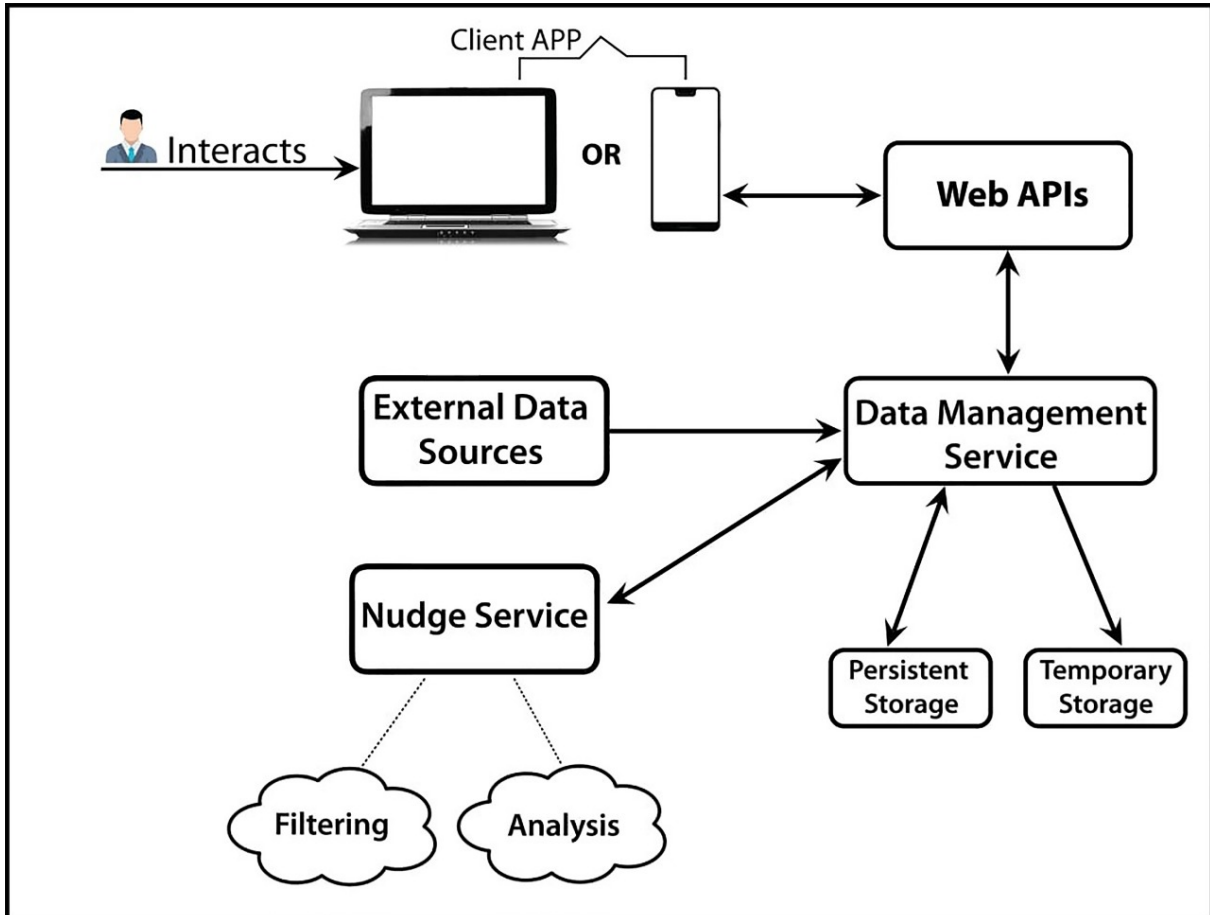


Figure 3.3: Detailed Architecture of the Nudge App.

3.3 Database Structure

This architecture is created as a basis for research and to investigate our set targets for the sake of this project. In a real scenario, the architecture would differ from our assumptions.

An instance of Mysql database would be used for storing data. There are six different tables that chosen to store data. The tables would contain information about users, their preferences and queries, nudge data and nudge verdict. A separate table would store data related to different observable patterns in the data present in the other documents. This document can also be used to perform explicit nudges.

The following subsections describe the database architecture of each data module.

3.3.1 Test Users

This table stores data about the users accounts. The pseudoFirst and pseudoLast columns entries represent the pseudo credentials that would be assigned to identify users from one another. This is a test









COLUMNS	DATA TYPE	NOT NULL
 id	INT	YES
 pseudoFirst	VARCHAR(256)	YES
 pseudoLast	VARCHAR(45)	YES
 createdAt	TIMESTAMP	NO
 modifiedOn	DATETIME	YES
 weight	FLOAT	YES
 age	TINYINT	YES
 height	FLOAT	YES

Figure 3.4: Test User

approach to hide the identity of the users. The actual information would maintain the entries for user credentials that can be safeguarded with the use of external authentication protocol. The table would also store information about the other properties associated with each user such as their weight, age and height.

The design of this table in a real world application should make sure that there are none security loopholes unexplored and the data is protected against any type of security vulnerability such as cross-site request forgery etc.

3.3.2 Test Queries

This is a special table that stores data about the user queries and what nudge has been proposed. The table takes an Id attribute as primary key. Initial and final fields store data regarding the journey user wishes to make. The preferred_mode is a null-able field. However, the field is nonetheless populated with a default selected value in the form unless user changes it.

The field testuser is used as a foreign key in order to create a relation with Test User table. Proposed_mode attribute stores data that the application intends for to be accepted.

Finally, the date_time field is used to store data regarding the time and date of journey. The time, date and the Proposed transport are required fields to gather data on demand based on the nudge algorithm that later runs a utility service that performs different operation on the fetched raw data and converts it into a presentable form against user's properties before it is finally rendered.








COLUMNS	DATA TYPE	NOT NULL
 id	INT	YES
 intial	VARCHAR(256)	YES
 final	VARCHAR(256)	YES
 date_time	DATETIME	YES
 preferred_mode	TINYINT	NO
 proposed_mode	INT	YES
 testuser	INT	YES

Figure 3.5: Users queries

3.3.3 Authentication

We suggest allowing users different methods for authentication purposes. We believe the one of the best and secure ways to employ authentication would be via use of electronic id's (i.e. (Signicat, 2021)). Testuser attribute would be used as a foreign key in authentication table.

For a test user in development environment, we can rely on legacy methods such as basic authentication via username and password.







COLUMNS	DATA TYPE	NOT NULL
 id	INT	YES
 testuser	INT	YES
 uuid	VARCHAR(255)	YES
 method	VARCHAR(255)	NO
 created_at	TIMESTAMP	NO
 updated_at	TIMESTAMP	NO

Figure 3.6: Authentication

3.3.4 Nudge data

This table would store Nudge data that is used to represent to user. The nudge data comprises of Queries ID, weather type, distance, duration, cloud coverage, sky coverage, real feel. This data is corresponding to the specific queries user has made.

COLUMNS	DATA TYPE	NOT NULL
 <u>id</u>	INT	YES
 testquery	INT	YES
 testuser	INT	YES
 weather	TINYINT	NO
 walking_dist	INT	NO
 bicycling_dist	INT	NO
 transit_dist	INT	NO
 car_dist	INT	NO
 car_time	TIME	NO
 transit_time	TIME	NO
 walking_time	TIME	NO
 bicycling_time	TIME	NO

Figure 3.7: Nudge Data

The purpose of this table is to store the information about the data we would receive when the user makes a request for a specific journey. The table takes Id as a primary key and 'testuser' and 'testquery' attributes as foreign keys to establish a relation with Test Users and Test Queries table, respectively. Regardless of what preferred mode user specifies while requesting a result, we process all relevant possibilities and therefore also need to store it in the database. The properties such as the distance of different modes of travel, and the time it takes a user to make a journey using any of the given modes would also be stored in this table. The table would help to understand how a nudge was tailored and it can allow developers to improve their implementation.

3.3.5 Nudge Verdict

Nudge verdict is a table that is used to save data that is processed after the utility operations have deduced results. For example, a journey that may require a user to walk up-to 2 miles can than be processed










COLUMNS	DATA TYPE	NOT NULL
 id	INT	YES
 testquery	INT	YES
 testuser	INT	YES
 preferredmode	TINYINT	YES
 distance_travelled	INT	NO
 timeconsumed	INT	NO
 caloriesburned	INT	NO
 estimatedsteps	INT	NO
 timesaved	INT	NO

Figure 3.8: Nudge Verdict

for how many estimated calories could be burned while accepting the nudge etc. The table only saves data regarding the nudge that has been accepted. We use 'preferredmode' field to store the nudge and it's properties are stored in the rest of the fields. The nudge is proposed on the basis of high ranking of the nudge score. These scores are determined by data processing services. Furthermore, scores are co-dependant on each other. For example, a score for weather data can be assigned based on the cloud coverage, temperature, wind speed or all of them for instance. A clear sky with little chance of change during the time users wishes to travel would automatically be assigned a higher nudge score compared to a day that is not ideal for a walk.

Similarly the distances between the two destinations are a crucial piece of information that can be used to determine what type of transport should be proposed. Interestingly enough, a better traffic situation

would get a lower score since this can prevent user from taking a healthier transport. Hence all the factors that can be exploited for a particular journey at a given time are rated on this scoring system and then the final nudge is made after a verdict has reached. The bias here is to recommend a healthier option as the first priority, however, the information about the other means of transportation would also be presented.

This table is created for analysis reasons. The purpose of this table is to store the different parameters that were used in the algorithm that would eventually generate the nudge. Storing these parameters in the persistent storage can also give basis for further review of the algorithm and any more improvements that are needed.

3.3.6 User History

This table would serve the purpose of performing explicit nudges and gross analysis. The table would contain relational data from other tables. It contains a log of most frequent queries performed by the user and a log of date/time when those queries were made. It would also take into account the preferred transportation based on a given set of nudge data in the past against the query table

The table takes 'testuser' as a foreign key. The rest of the fields are updated every time users log in and interact with the application.

The last_active field would store information about the time of user's most recent activity. The transaction field updates every time a user makes a query. Furthermore, has_accepted and has_rejected attributes store a count of how many nudges users has accepted and how many were repudiated.

Finally, most_initial_searched and most_final_searched fields would store records of the most frequent queries user has made. The table can also prove to be useful if the application follows a subscription model and would help to understand converging patterns in the data that would help to understand the user.









COLUMNS	DATA TYPE	NOT NULL
 id	INT	YES
 testuser	INT	YES
 last_active	DATETIME	NO
 transactions	INT	NO
 has_accepted	INT	NO
 has_rejected	INT	NO
 most_initial_searched	VARCHAR(255)	NO
 most_final_searched	VARCHAR(255)	NO

Figure 3.9: Users History

3.3.7 Data Model and cardinality

The diagram given below represents all the tables, their relations and cardinality. Our research resulted in the design of these tables. Every table holds a unique key to identify each record separately. The same keys are used as foreign keys to construct relations between different tables.

Test Users table has a foreign key in each table so that data stored in those tables can be traced back to the individual user. Collectively, the tables serve the purpose of storing queries, nudge data and user's preferences.

The real world application would also contain mandatory tables such as storing registration tokens as well as for storing different logs for events and user actions.

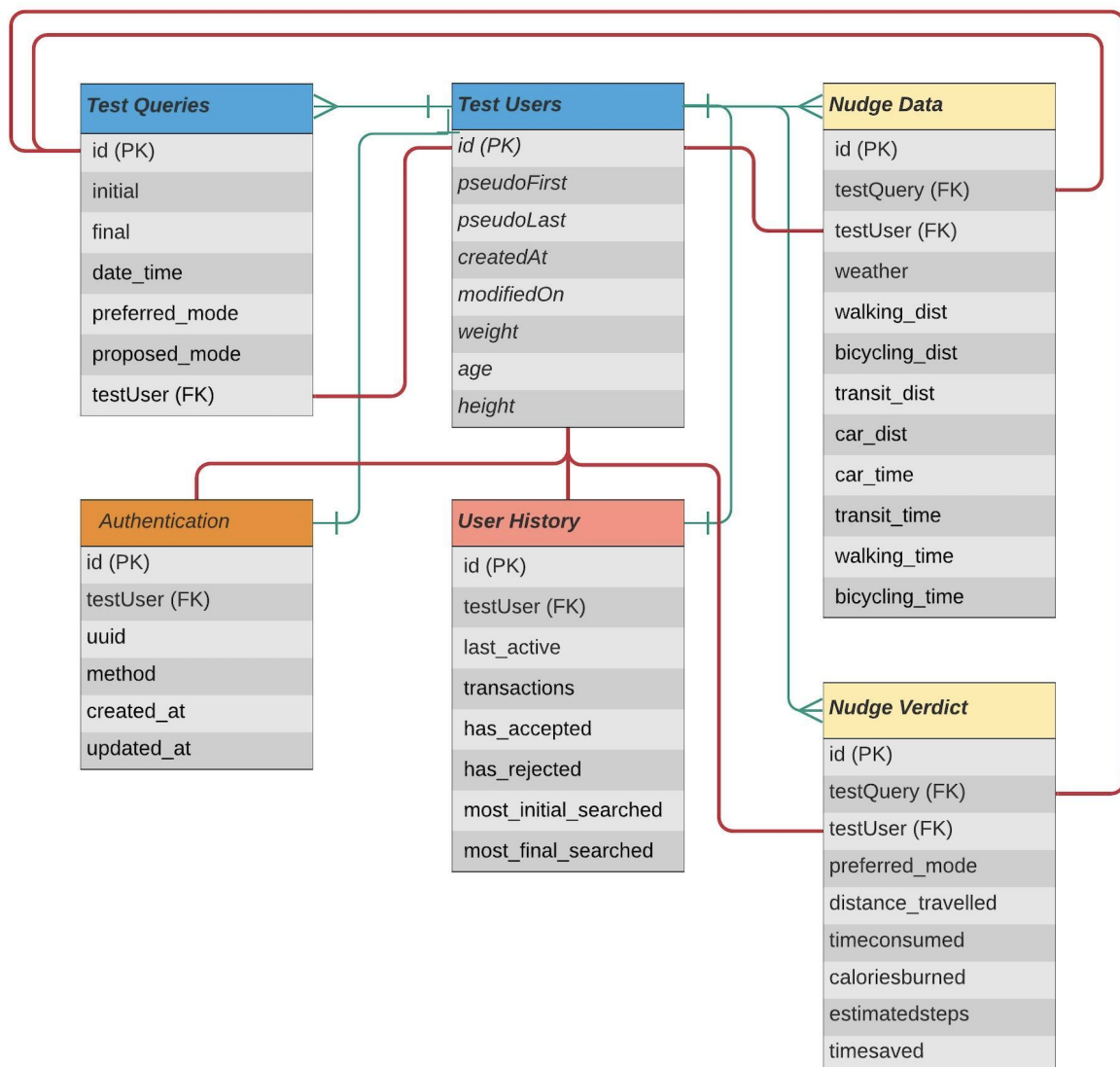


Figure 3.10: Data Tables and their Relations

Implementation

The tools and technologies described in this section relates to the implementation proposed specifically to perform experiments and different exploration to determine users behaviors to different nudges. We propose our solution by interacting with several of the technologies described below and go deeper into highlighting different advantages of relying on these technologies. Our implementations do not pertain to an ideal implementation of the actual app but rather a blending of different tests and experience gained from interacting with different tools. Nevertheless, it helps to understand how an actual app can take advantage of the insight we provide with our prototype proposed design. We will discuss different technologies that can be used for research/actual implementation purpose when dealing with collecting, storing, processing and representing data. The shortcomings of the prototype proposed design and different routes that were taken during the execution of different tests are discussed later in 'Evaluation' and "Discussion" chapter.

4.1 Client Application

Following tools and technologies would be used for developing the Client Application.

4.1.1 React JS framework

We selected React js as our Client Application framework. React JS is the most popular Javascript framework that is used for developing state of the art single page Client Application applications (Aggarwal, 2018).

Modern application involves frequent DOM manipulation so that the dynamic content of the websites can be rendered to the users. This is due to different user and server events. React however, introduces a new concept of a virtual DOM. A virtual DOM is a fast and actual in-memory representation of the DOM which allows us to translate any changes to the actual DOM (Fedosejev, 2015). This prevents an entire page refresh and consequently provides a faster interactive environment where users can manipulate the content of the page.

4.1.2 Client Application Content

The content of a web page would be handled by html (hyper-text-markup-language). We included html5 in our implementation. However, react is not limited to which version of html we want to work with unless particularly specified. This version of html is supported by all latest versions of browsers.

4.1.3 Client Application Styling

We relied on CSS for styling the content. Our design focuses on presenting the data in a chronological fashion where the desired content is placed above all other. The CSS explored in this project is of basic

nature. However, the actual implementation would require special attention dedicated towards styling of different UI components written in a markup language for better nudge purposes.

4.1.4 Client Application logic

React allows us to use an even powered up version of vanilla Javascript called JSX (Javascript XML). JSX is a syntax extension for Javascript (reactjs.org, 2020). React introduces the idea of components which contains markup and logic. This has become possible by the use of JSX. Any valid Javascript expression can be rendered inside a curly braces block with its corresponding markup. This serves our purpose even better since JSX allow us conditional rendering of logic as well as markup with the corresponding styling. JSX is compiled by Javascript preprocessors to Javascript (ES6). Javascript ES6 was introduced to allow developers to write quality code without dealing with a lot of abstractions that could turn out to be a lengthy process and thus more time consuming (Prusty, 2015).

4.2 Back-End

We explored Two different frameworks for designing the back-end. We initially attempted to select django framework for back-end with a No-sql database. Later we switched to working with more conventional back-end framework written in javascript called Express that facilitates Client Application implemented in React in a more efficient way. Following tools and technologies were used for the implementation of the Back-end.

4.2.1 Node JS

Node Js is a serverside Javascript environment that allows execution of javascript code. Asynchronous execution makes node highly scalable for data intensive real time applications. The javascript was previously only capable of execution within the browser itself. However, node is written in C++ by translating Chrome JavaScript V8 engine. The non blocking single threaded nature of Node can serve multiple clients at the same. With the Client Application in React and back-end in Node, the entire application can be implemented using a single programming language(Tilkov and Vinoski, 2010).

4.2.2 Express

Express is a lightweight Node JS framework. Express allows us to simplify Node JS functionality by offering its own middleware and routing. The middleware stack enables developer to handle one monolithic request into smaller handler function which makes it easier to handle. It also reduces the complexity of Node's HTTP server's complexity (Hahn, 2016).

4.3 Database

There were two databases that we choose to look into. A relational database and a document based no-sql database. Relational database was for quering users data as their profile information, nudge history and travel history. The no-sql database on the other hand was specifically explored for querying data for bus timings.

4.3.1 Mysql

Mysql is a relational database that comes in handy when quering related data stored in different tables that comprises of rows and columns. It allows us to perform different join operation on data that is stored in different tables and can be viewed in graphical form (Győrödi et al., 2015). We use knex (Knex js, 2020) query builder for our querying our Mysql database. Knex is primarily made for Node.js and therefore provides great flexibility and support for callbacks and asynchronous promises.

4.3.2 Database Visualization Tool

Mysql Workbench version 8.0 was used for visualization of data stored in our database.

4.3.3 Virtualization

Docker is a software development technology that offers an easy way to develop containerized apps. It offers a level of virtualization so that apps contained inside a docker container can be easily deployed and executed regardless of what machine they run on.

4.3.4 web-scraping

Web-scraping is one of the most convenient ways to acquire useful data from different web pages. It comes in handy while working with websites that do not provide their own APIs for transfer of data. This technique essentially turns the entire internet into one's very own personal database. Cognizant of the potency of web-scraping techniques, we too implemented web scraping. We explored different web-scraping tools such as Puppeteer, nightwatch, cheerio and beautiful soup. In the end we narrowed down our choice to Puppeteer since it works well with node.js.

4.4 Client Application Overview

The front-end design allows user a simple user interface to communicate with the back-end and access corresponding data. The Client Application is a single page application with 3 components. Although a sign-up component was created but kept dormant after we felt no need for it as the test users are provided with their own credentials.

4.4.1 Authentication

User Authentication can either be done with a legacy method or via some electronic ID. The app would contain the historical data that comprises of users travels. Since the data is of sensitive nature, the user must be authenticated with some of the best resources available. Signicat is one of the most reliable means of authentication in Europe. It allows users of different software products to be authenticated with a number of authentication methods. A legacy authentication model with username and password can be followed after the user has authenticated themselves with their electronic ID.

4.4.2 Query From

A simple UI component where different data parameters related to travel plans are collected. These parameters mainly involve starting and ending point of his journey. Another parameter is user's desired mode of transportation. The third and final user input parameter is time and date of travel. This is an optional field that contains a placeholder of current date and time. It can be left unfilled but the user would be notified that leaving this field empty will render results according to the current date and time.

4.4.3 Results Component

The results component is an inbuilt component that renders after user has submitted their queries. This component contains all the data that is fetched from different resources such as external APIs and scraped data from web. The component is not just for viewing purposes but is also made interactive so that the user can provide feedback. This is the most vital part of our design when it concerns Client Application since we only attempt to perform a nudge here but also store user response for further analysis.

4.5 Client Application Implementation

The Client Application design involves following steps. The very First step deals with a simple form submission with users query credentials. The queries credentials are processed in the second step, user queries are stores and the result of different calculations is loaded to the individual user. The results loaded to the user are also stored in the database. The third and final step involves storing the users feedback data. The feedback is fetched after user had given their opinion in the form of performing different operations on the Client Application. This opinion can be a simple acceptance of any of the choices that were presented to the user. Data corresponding to that specific choice is then stored as the actual nudge data. This data can be later accessed by the user to follow his previous track record. The data rendered after the user has submitted his queries are done so by following the principles of digital nudging specifically choice architecture where the recommended options are presented above the rest.

Despite users are allowed to choose a mode of transportation, the data processing service processes all possible transport options between the two endpoint of the journey user wishes to make. A bias is created in the database with specific locations that users take on regular basis. This bias is created for fetching data to user queries. The data involves transit routes, car, bicycle, and walking which is pre-fetched and stored in database from external sources. However, the implementation does not limit the user from giving any unknown values. The unknown values in a query are dealt by fetching data directly from the external web apis like google directions, distance matrix and open weathermap.

Javascript moment library allow us to manipulate time and date and perform various different operations on it. Time is a factor that is used and exploited both for answering users queries and generating data that can turn out to be useful when nudging users. The Client Application communicates with the backend through HTTP requests. The query data is first cleaned at the Client Application by using utility services that shape data for further processing in the back-end before it is sent out. This may involve adding instruction tags as parameters for what operation to perform on data at the back-end such as whether the data should be fetched from database or requested from an external web api. The conditional routing would come in handy with the use of data filtering service at the Client Application. The same services perform different operations on data that is retrieved from the database. For example adding proper date and time format, calculating the time it takes for a normal walk and converting it for a user that would rather run or brisk walk. This is done through predefined parameters. We also implemented an algorithm for determining how many calories a user will burn based on a simple formula that works on BMI(Body-Mass Index) and age . The step increase count with choosing a transport that may require a physical exertion is included in our Client Application design by relying on a separate utility service that would manually provide information regarding the steps taken during a period of time

To summarise, The design of client application is done by exploring React framework's conditional rendering of html content. Furthermore, The design principles of choice architecture are observed by presenting our favorable choice as the very first choice in the list of options in our design. The user is presented with recommended choice after giving surreptitious advantage to a specific favored choice by rendering it first with css and html specifically tailored for it. The choices are rendered after creating a ranking of them. This is also done using a client side service. The choices are than presented in chronological order based on their rank score and along with the corresponding markup.

4.6 Back-end Implementation

One single file is designated as the gateway for all the routes made from the front end. The corresponding functionality is then triggered using express middle-ware. The gateway file contains the necessary imports for different processes that would be required for execution. An instance of knex builds all the queries. Asynchronous promises are used to fetch independent data against knex queries. The raw queries can also be implemented where deemed necessary.

Express framework is used to reduce the complexity of Node JS and increase the flexibility of different http routes as discussed earlier. When concerning users, only two http requests are required for the entire design to fulfill its purpose. Get requests made from the client to get data against a specific get request and put requests are for storing the data. The authorized admin users can be allowed to perform all type of http requests.

The routes are made conditionally based on client application design after evaluating different factors in user queries. To elaborate, a user interested in traveling in mean time would get data that is fetched from either the repository or the external web sources. For example, in the event of a user wishing to travel to a specific location at a time that is within 1 hour in the future, any routes that would request data from the database would be bypassed and instead data will be fetched from either in-memory key value store or the external web apis.

Express middleware allow us to implement our own operations stored in a stack to access and manipulate different request parameters between the time-span a request is received and a response is sent. A simple reference to a next operation is made if the intended data does not reside inside the cached memory if available.

The transit options would allow to explore different timings of the available busses only if they are arriving soon. These timings sometimes can allow a window of opportunity to perform a nudge in case a bus is late from its schedule. This extra time is added to the time it would normally take a user to reach their destination. The journey that involve making two transits is also a vital element that we have focused in our design.

The estimated time consumed by merely walking to a destination or taking a car or bus might result in either same amount of time or even less. This is due to the waiting and transit time. The journeys involving car can be affected due to high volume of traffic. This can make users realize that walking is probably the best option. In order to make it even more appealing, we suggest making an estimate of how many calories a user will burn while choosing either to walk or bicycle. These options are not available for the users that choose to travel via a bus or car. This would left void deliberately. The lack of available information can potentially psychologically push users to avoid taking that mode altogether. However, its proved effect could not be established and requires further investigation.

The calorie estimation can also be done by using a dynamic front-end utility service that takes users physical attributes that were saved in his profile and the travel credentials retrieved from data returned from database. The service calculates calorie burn against a specific distance and the data is then presented to the user.

4.7 Database

Mysql database was deemed the best and most convenient option for implementing pre-requisites for our research. The database stores persistent data that facilitates different components of our logic. User profile elements are stored inside the databse beforehand. The data is stored manually regarding different locations that the user would most frequently travel to. These are the only pre-populated schemas that we prefer. The rest of schemas are populated once the user starts interacting with the client application. These include all the queries that user would make and the results generated once the queries have been processed.

Mysql's rigid schema allows us to perform closely knitted joins performed on multiple relational tables. This way we can relate different factors from multiple schema, apply our logic and deduce results. Furthermore, knex query builder makes it easier for to perform complex asynchronous querying with the help of javascript promises. This means the data management service can finish a task without waiting for the other operations to finish.

4.8 External Web Apis

Several different external sources are used for gathering data. Some of the sources are utilized through the Apis provided by these sources while others are done so by scraping the required data directly from the website. Following are the external apis that we use. We also throw light on how the corresponding data is processed.

4.8.1 Weather Data

Open weather map api (openweathermap, 2020) for accessing weather related data. The api provides fast, free and accurate weather information upto 7 days. The data is provided in both JSON and XML formats. It particularly benefits us as we use javascript for the entire project, we can directly use the returned JSON as a data object in the implementation.

The data can be stored in the persistent storage for several days to reduce the number of direct requests sent out to the database. The redundant stored information that is no longer valid can be expunged from the database by using Nodes-cron job which works as arbitrary functions to do a job depending on the specified time and date. It can also be set to trigger periodically. The same task can be achieved by relying on an in memory key value store such as Redis. Node allow us to store data into Redis database with an expiration date. This will benefit us in two ways. First the time it takes to receive a response to any request made to an external source can be marginally reduced. Furthermore, Since weather data is susceptible to change, the reliance on an in-memory key value store can deprecate our need for storing the data into a persistent storage entirely. Hence we do not need to worry about implementing node scheduled jobs.

The data upon reception from the api is analysed by the data management service. The elements such as, wind speed, cloud coverage and temperatures are compared and a ranking is established. This is important to establish whether this data supports the nudge we are trying to perform or not. This ranking decides how to render markup related to weather information to the user.

4.8.2 Google Directions Api

Google's direction api allow us for finding routes between locations for different modes of transportation. The api also provides information regarding the distance and the time concerning a particular journey. Furthermore, the api responds to http requests coupled with an api key and the defined parameters such as origin, destination and mode. Here origin serves the initial or current location of the user, destination relates to the location user desires to travel to and the mode being the type of transportation user intends to use.

The default mode is set to driving, however the keys for all means of transportation available for a particular journey are included in the JSON object sent from the api. This data is stored in the database after the users have performed certain operations while interacting with the client application.

4.8.3 Transit Mode Data

We found during our research that the transit information provided by the google directions api for Tromso was either incomplete, inaccurate or unavailable at all. In order to find more accurate and real-time bus schedules, we performed an experiment by referring to Troms Kortet website.

The sturcture of the website was carefully observed. A single url was discovered that remains constant throughout the re-rendering process of the search filters when interacted with. The parameters can be observed in the url while the website is processing the request made from the browser. This allowed us to perform dynamic web-scraping for each request with users parameters formatted within the url from

a client application.

The x-paths of different html tags were followed. The data was then carefully analysed and any factors that can benefit us in devising a nudge were identified. The website uses two different tags for presenting normal and delayed timings for busses. This problem was overcome by targeting all the relevant tags. The buses that were on the right schedule were missing the data (delayed time of arrival) we were looking for and therefore hinted at that the busses were on time. The total duration of journey would not include the delayed times therefore it was not possible to scrap data directly from the tag that contained it. The tags where the waiting time between making a transit to a different bus was also considered a valuable data fragment that can be used to include while creating a nudge. The following snippet helps to understand the challenge that lay overhead.

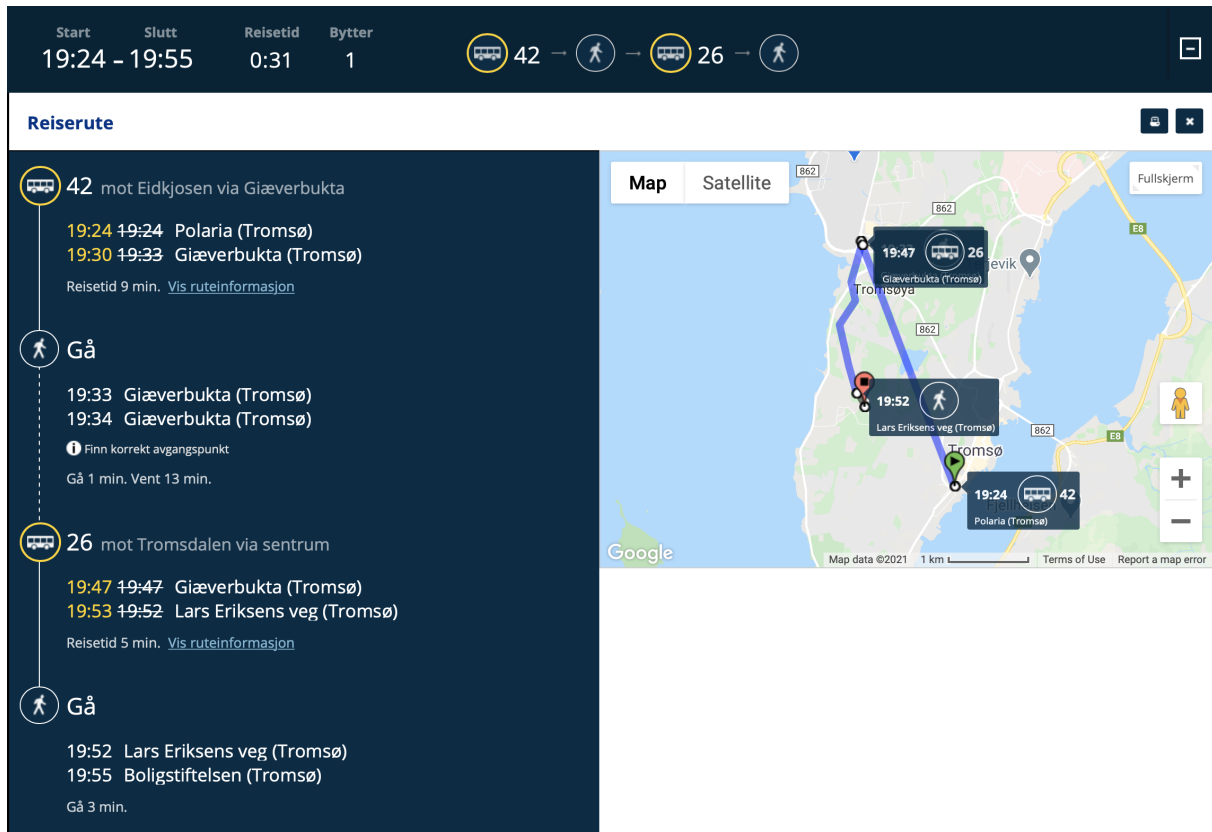


Figure 4.1: Bus schedul snippet (TromsKortet, 2020)

Nodes puppeteer library is used to perform this job. It remains a challenge to perform web scraping from the front-end since browser's CORS origin policy does not allow such requests due to browsers spectre security vulnerability. Performing web-scraping in production environment would be crucial task that needs to be performed very carefully leaving little to none margin of error.

Evaluation

In this section, we describe few investigative experiments performed either to find suitable technologies that can come in handy when implementing the project in a practical scenario or to observe the users behaviour and feedback . We also discuss some of the technologies we experimented with and gives grounds for why they were not included in the proposed implementation. The evaluation is done in parts by 6 volunteers who used the prototype implementation for 3 weeks. 4 of these volunteers owned a private car, a bicycle, and actively used transit as well. Two volunteers would only use a bus for transportation. It is important to mention that the prototype performed limited operations such as fetching data from external sources, calculation time and distance and scraping it directly from website. There were little efforts made into avoiding unforeseeable user generated bugs such as Unicode Transformation Formatting for Norwegian alphabets etc. The user feedback is recorded merely in terms of feedback gathered after interviewing them and the results deduced is an outcome of their experience and opinion. Nevertheless, the experiment and resulting feedback helped us to evaluate our design to a decent degree.

5.1 Internal Evaluation

In this section we will do an evaluation of the prototype implementation of our project and describe how effective our approaches were.

5.1.1 Mapbox API

We experimented with Mapbox api to add more interactivity. A simple dummy form was created that upon submission would trigger several states to update in react. The purpose of this experiment was to explore how much control a react implementation would have over the dynamic map rendered through the API.

It provides information regarding direction, navigation, traffic, time it takes for a journey using a specific form of transportation and different maps that can be controlled by the direction GLI. Moreover, users can zoom in and out to take a closer look at a location or explore a route. The easy drag and move functionality allows to navigate to any place in the world.

The implementation added interactivity to the page, however, mapbox require not only a strict paradigm to follow when manipulating the map that can create conflict when working with react. This is due to the different asynchronous operations that takes place when working with react. These problems can be solved by applying different solutions such as updating states of different coordinates in a synchronous way by relying on callback functions in react. This however can result in compromising the performance. Furthermore, mapbox directions api does not include any built in functionality to utilize the directions api. Our research was limited to the area surrounding Tromso city, therefore Restricting the map to certain coordinates would have also required extra effort. Mapbox does provide asynchronous execution but that would also require extra measures that was beyond the scope of this project. Nonetheless, the api

can be used for the sake of rendering elegant static maps that highlight a specific route once the user has submitted the form.

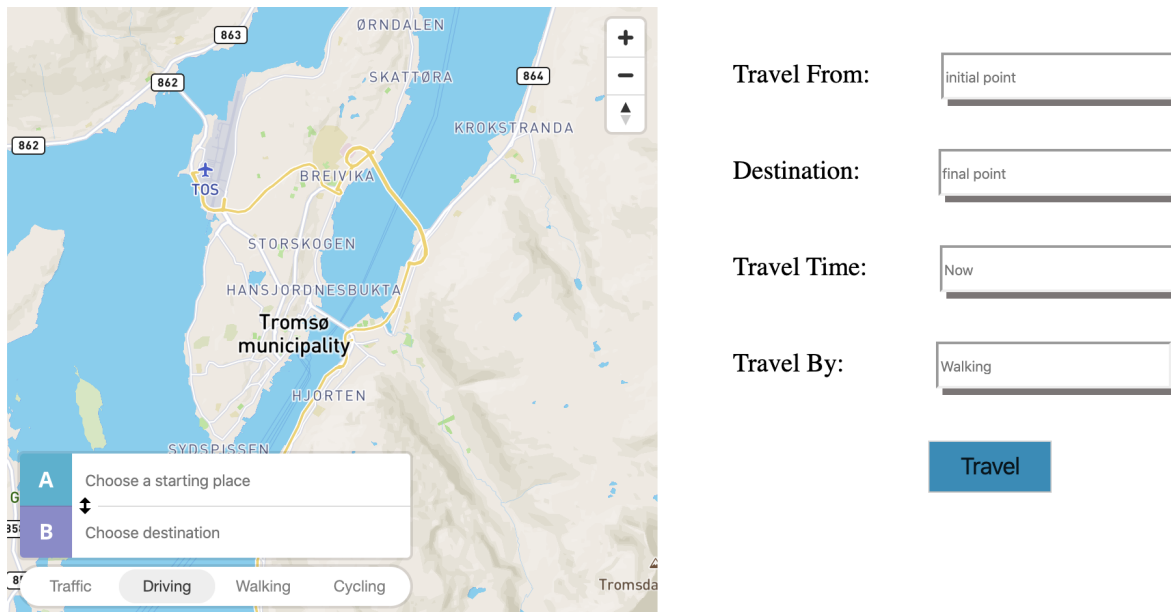


Figure 5.1: MapBox api

5.1.2 Beautiful Soup

We also experimented to propose an implementation that resembles client applications that gets served from two different back-ends. Therefore an attempt to initially write a back-end in django was made that would specifically perform on demand web-scraping for us. Django is a web framework written in python which has rich libraries for implementing different modules.

We worked with Python's requests, and beautiful-soup libraries to scrap data. The challenge to scrap data from websites that are heavily JavaScript rendered was overcome with the use of selenium and a google chrome web driver. Selenium is mainly used for testing purposes, however the html we were trying to access would not render unless there were a few clicks performed on the website.

We used selenium coupled with the web driver to achieve this task. This turned out to be a time inefficient operation with the job taking several seconds to finish. Moreover, we needed to implement extra functionality to convert our web driver to a headless web browser. Due to these drawbacks, we concluded to abandon this approach and decided to exclude it from the implementation.

5.1.3 Puppeteer

We preferred to work with puppeteer for its compatibility with Node. The other advantages puppeteer offers is the builtin headless chromium web-browser and the ability to parse javascript and html.

Extracting elements based on the x-path is a drawback since it can change at any time. This is why performing web-scraping with a hundred percent precision at every instance remains a difficult task to accomplish. Web-scraping is a resource intensive task that takes a toll on network, CPU and memory. An application running in production mode would compromise its reliability (in term of availability) at some point if the reliance on web-scraping for accessing deep rooted data that is hidden behind several links and events, is too high. In our implementation however, web-scraping turned out to be a useful technique that provided us with the data we needed for the research.

5.1.4 Data Collection

Data collection was done by sending out requests to all external API's and triggering web-crawlers. This was done so that all corresponding data can be displayed to the user. The only draw-back with this implementation was that it would take longer time to make a fetch cycle. Furthermore, the data would be stored in the database in 3 successive steps. First the data fetched was filtered out of any metadata and stored in the database. The same data was then analytically analysed with a data processing service. This new refined form was then displayed to the user and later stored again in the database. The final step involved storing data regarding users response in whether accepting or rejecting the nudge. This could have been improved by setting the data into constant objects and sending all the records at a single go to the database once the user had submitted the response. Another shortcoming was lack of parallel processing of different jobs.

5.1.5 Data processing

The data processing service resided mainly in the front-end. We used simple algorithms for adding different attributes to the data collected from the internal and external sources. This data was analysed with different parameters that would determine the rank of each data-element i.e. distance, time, mode, traffic and whether etc. The traffic information was only available for departure times either in the current time or some time soon in the future. Our algorithm would resolve a calorie estimation and highlight time of journey.

The processing of data does not in any way change the values of the data but rather determines how to present these values with additional information that would nudge users.

5.2 External Evaluation

External evaluation is based on the feedback from the test users. This was the most important discovery in the entire project that helped us understand what could be improved upon.

5.2.1 Accepted Nudges

Distance was the common factor among the accepted nudges that encouraged users to take a walk. The users feedback asserts that the bus timings and calorie information played a role in persuading them. Most of these journeys were searched for in real time where the arrival of bus and travel duration would have taken more time than to take a walk. Furthermore, almost all users regardless of what type of transportation they owned, exhibited a similar behavior when it came to the nudges where the aim was to take a walk.

Interestingly, the nudges where the bus was chosen pertained to a longer distance where walking was simply not an option. The users who did not own a car simply relied on the bus and discarded the recommended choice while the car owners persisted to take the car. This was an interesting key finding of the thesis which will be discussed in the next section.

The implementation lacked a constant determinant of a good weather during the winter therefore the weather data was mostly presented to the user based on the temperature and whether it was rainy or not. However, all the users admitted that weather information would be a dominant driving force in dictating their decision.

5.2.2 Rejected Nudges

One of the key findings of this thesis revealed that volunteers who owned a car were hard to persuade to either walk or take a bus in special circumstances. We believe, that presenting them bus information with emphasize on duration, transit, and delays backfired on our intentions. This emphasize would

possibly work only on those users who intends to travel to a fairly shorter distance and do not have any other options but to walk or to take a bus, however, users who owns a car would simply revert back to using the car. This was something entirely opposite of what we were trying to achieve. We believe the implementation failed to nudge users specifically the ones who owned a car to prefer a bus over a car where the travel distance was longer.

Discussion

The outcome of our research allowed us to see several drawbacks in the assumptions we had before starting our investigation into the problem in question. Bus information had remained a main focus in our implementation since the beginning of the project. However, surprisingly enough, it was revealed to us that presenting bus schedules and their timetables in a discouraging manner would actually result in a nudge that would make a user drive their private car if he/she owned one. In a scenario where none owned a car, the nudge can potentially result in achieving the intended outcome.

The bias for performing a nudge was also prevalent and despite users were given the freedom to choose, the ultimate choice did not translate into what we were intending. The main reason for that appears to be too much focus on performing a nudge where it was almost impractical to persuade user to take a desired option. We believe the implementation should have taken into account the impossibility of performing a nudge under certain circumstances.

The data analysis on the stored data regarding users journey and nudges could not be processed therefore any claims made are merely based on theory. The proposed implementation also does not encompass the context-aware functionality where the historical data is utilized against performing any nudges in the future as mentioned earlier.

6.1 Test Users

The training of the test users was an important aspect that could have been improved by putting more efforts. Our evaluation of the test implementation is based upon the response received from the test users. Furthermore, a relatively larger set of test users with a large data set would have allowed us to verify our assumptions in an even precise manner. Nevertheless, the feedback points out the drawbacks and highlights areas where more work is needed to be done.

6.2 GDPR

Applications that contains sensitive data regarding users personal information and their mobility can potentially compromise the GDP (General Data Protection) regulations and compliance if the data is not well protected. The smart nudge project would also fall in the category of such applications where users privacy and the protection of sensitive data is of paramount importance. An entire thesis can be dedicated to discuss this subject however we would like to discuss a few solutions to protect users privacy.

6.3 consent forms

An application such as ours should have the functionality of allowing users to download, sign and upload consent forms in order to process and store their personal information. This can either be

achieved by validating users and their consent forms via use of electronic ids or multi-factors authentication. Furthermore, Cron-jobs should be implemented to perform hard deletes on users log data and old records from the database.

6.4 Legality of Web scraping

The legality or in the least sense the morality behind performing web scraping has remained one of the profound questions that many developers have to face. Big companies often perform web scraping for their own personal use and gains. However, the extensive scraping of a web-resource with the intent of using it for personal reasons takes away the control from the original owner and therefore can fall into the category of immoral (if not illegal) access of a resource. We relied on web-scraping for the sake of experimentation, however, an actually application must take into account the morality and legality concerned with performing online bots.

6.5 Involuntary solicitation

The users were urged to report their feedback without getting involuntarily solicited to accept any nudges. The awareness of participation in a study can unconsciously lead users to accept nudges which they would not have accepted had they not been aware of the study they were involved in. We believe the users provided us with valid reliable information that can be used in the future for devising effective nudges with a an improved implementation.

6.6 Other Insights

The implementation should have the capability to determine what type of transport a user owned. This is very crucial since the system must not send a user a nudge that recommends them to take a bicycle to work when the user do not even own it in the first place. One way to achieve that would be to determine the availability or ownership of a particular transport based on the user's history of travel and means of transportation. The information can also be gathered by simply asking the user to volunteer this information. However, our research did not dwell too much into that aspect of the project.

Future Work

The solution and design we proposed for a smart nudging app is derived based on our work that revolved around our own locality (Tromsø). A more robust and dynamic approach would be needed to expand the solution globally. This will give birth to more challenges such as authentication of users across countries and taking into account the regional regulations.

The vast Majority of internet traffic comes from mobile. According to statistics, since 2017, Over 50% of this data is consistently observed to be found on mobile devices even when tablets are excluded (statista, 2021). This is why, the application should ideally be a mobile one.

Storing large amount of data is something that can also be further explored. Cloud resources provide solutions such as S3 for front-end deployments and Elastic Beanstalk for back-end solutions. Such resources can be used to store users data alongside the above mentioned consent forms in order to deal with all legal liabilities.

Enabling sensory devices to sync up with the application would allow users to transfer all their data to a single place where the application would keep a track of it and also process it for further feedback and tracking.

The smart nudging app for healthier transportation choices aim to introduce a new method of searching for journeys that provide not only an up to date information regarding different journeys, but also allow users to track their physical activeness and any variation in their overall health that can be observed with the external factors. The utility of such application would only become common practice when the consequences of lack of physical activity is realized by the masses. For that, we believe it requires more effort to spread awareness among the general public. The solution we proposed would only become mainstream once the problem is recognized by the majority.

Conclusion

We contributed to this project by proposing a solution in the form of an application that would use contemporary technologies while relying on personal information of its users, their activities and incorporates different factors that are involved when taking a journey. Our research also borders on behavioral aspects of users when it comes to different nudges and how they respond to them by performing a limited test. Furthermore, we suggested how the conventional way of travel and selecting journeys can be transformed into health oriented personal goal that would not only benefit the users but would also allow them to track their 'gains'. We believe that by allowing users to have access to their profiles that tracks their travel activities and corresponding data regarding time saved, calories burned and steps taken etc, would make travelling even more enjoyable and health friendly. However, the research also suggests that some prior knowledge of users and their preferences is crucial towards performing an effective nudge. In conclusion, factors influential towards travel coupled with users personal information and history can allow us to prompt users to not only travel healthy but make a healthy habit out of it.

Bibliography

- Aggarwal, S., 2018. Modern web-development using reactjs. *International Journal of Recent Research Aspects* 5 (1), 2349–7688.
- Andersen, A., Karlsen, R., Yu, W., 2018. Green transportation choices with iot and smart nudging. In: *Handbook of Smart Cities*. Springer, pp. 331–354.
- Andrew McAfee, E. B., October, 2012. Big data: The management revolution).
- Bhogal, J., Choksi, I., 2015. Handling big data using nosql. In: *2015 IEEE 29th International Conference on Advanced Information Networking and Applications Workshops*. IEEE Computer, pp. 393–398.
URL <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7096207&tag=1>
- Bonnet, L., Laurent, A., Sala, M., Laurent, B., Sicard, N., 2011. Reduce, you say: What nosql can do for data aggregation and bi in large repositories. In: *2011 22nd International Workshop on Database and Expert Systems Applications*. IEEE, pp. 483–488.
- Brenn, M., 2018. 83% of u.s. adults drive frequently; fewer enjoy it a lot. *Website Journal*.
- Brewer, E., February 2012. Cap twelve years later: how the rules have changed.
- Burckhardt, S., 2014. Principles of eventual consistency. In: *Foundations and Trends in Programming Languages*. Vol. 1. pp. 48–58.
URL <http://dx.doi.org/10.1561/25000000011>
- C. Schneider, M. W., vom Brocke, J., Jul 2018. Digital nudging: Guiding online user choices through interface design 61 (7), 67–73.
- Cattell, R., December, 2010. Scalable sql and nosql data stores) 39 (4).
- Chandra, D. G., 2015. Base analysis of nosql database. *Future Generation Computer Systems* 52, 13–21.
- Chasseur, C., Li, Y., Patel, J. M., 2013. Enabling json document stores in relational systems. In: *WebDB*. Vol. 13. pp. 14–15.
- Chaudhuri, S., Dayal, U., 1997. An overview of data warehousing and olap technology. *ACM Sigmod record* 26 (1), 65–74.
- C.L. Philip Chen, C.-Y. Z., January ,2014. Data-intensive applications, challenges, techniques and technologies: A survey on big data.
- Fedosejev, A., 2015. *React. js essentials*. Packt Publishing Ltd.
- Gallagher, J., 2015. Inactivity 'kills more than obesity. *Website Journal*.

-
- Grolinger, K., Higashino, W. A., Tiwari, A., Capretz, M. A., 2011. Data management in cloud environments: Nosql and newsql data stores. *Journal of Cloud Computing: Advances, Systems and Applications* 2 (1), 22.
URL <https://christof-strauch.de/nosqltdbs.pdf>
- Győrödi, C., Győrödi, R., Pecherle, G., Olah, A., 2015. A comparative study: MongoDB vs. mysql. In: 2015 13th International Conference on Engineering of Modern Electric Systems (EMES). IEEE, pp. 1–6.
- Hahn, E., 2016. *Express in Action: Writing, building, and testing Node.js applications*. Manning Publications,.
- Hayes, B., 2008. *Cloud computing* 51 (7), 9–11.
- Ibrahim Abaker Targio Hashem, Ibrar Yaqoob, N. B. A. S. U. K., July, 2014. The rise of big data on cloud computing: Review and open research issues.
- Ibrahim Abaker Targio Hashema, Ibrar Yaqoob, N. B. A. S. M. A. G. S. K., January 2015. The rise of big data on cloud computing: Review and open research issues 47, 98–115.
- Inmon, W. H., 2005. *Building the data warehouse*. John wiley & sons.
- Ishwarappa, A. J., 2015. A brief introduction on big data 5vs characteristics and hadoop technology.
- James Manyika, Michael Chui, B. B. J. B. R. D. C. R. A. H. B., May, 2011. Big data: The next frontier for innovation, competition, and productivity.
- Jaseena, K., David, J. M., 2014. Issues, challenges, and solutions: big data mining. *CS & IT-CSCP* 4 (13), 131–140.
- Katarina Segersthl, H. O.-K., 2017. Distributed user experience in persuasive technology environments, international conference on persuasive technology.
- Knex js, 2020. Knex js. <http://knexjs.org/#changelog>, [Online; accessed 3-November-2020].
- Li, K., Li, G., 2018. Approximate query processing: What is new and where to go? *Data Science and Engineering* 3 (4), 379–397.
- Makridis, I., August, 2018. Big data analytics and knowledge discovery through location-based social networks (Ibsn).
- Markus Weinmann, Christoph Schneider, J. v. B., 2016. Digital nudging, business information systems engineering 58 (6), 433–436.
- McAfee, A., Brynjolfsson, E., Davenport, T. H., Patil, D., Barton, D., 2012. Big data: the management revolution. *Harvard business review* 90 (10), 60–68.
- Mehra, R., Lodhi, N., Babu, R., 2015. Column based nosql database, scope and future. In: *International Journal of Research and Analytical Reviews*. Vol. 2. pp. 105–113.
- Memcached, 2020. "in-memory key value store". Accessed: 2020-04-20.
URL <http://memcached.org/>
- Michael Armbrust, Armando Fox, R. G. A. D. J. R. K. A. K. G. L. D. P. A. R. I. S. M. Z., 2010. A view of cloud computing 53 (4), 50–58.
- Miller, R. B., 1968. Response time in man-computer conversational transactions. In: *Proceedings of the December 9-11, 1968, fall joint computer conference, part I*. pp. 267–277.

-
- Mimmi Castmo, R. P., June, 2018. The alliance between digital nudging persuasive design.
- Moniruzzaman, A., Hossain, S. A., 2013. Nosql database: New era of databases for big data analytics-classification, characteristics and comparison. arXiv preprint arXiv:1307.0191.
- Mozafari, B., 2017. Approximate query engines: Commercial challenges and research opportunities. In: Proceedings of the 2017 ACM International Conference on Management of Data. pp. 521–524.
- Mustaque Ahamad, R. K., September, 1999. Scalable consistency protocols for distributed services.
- Oinas-Kukkonen, H., Harjumaa, M., 2009. Persuasive systems design: Key issues, process model, and system features,” communications of the association for information systems 24, 485–500.
- openweathermap, 2020. open weathermap api. <https://openweathermap.org/api>, [Online; accessed 7-December-2020].
- Paul C. Zikopoulos, Chris Eaton, D. d. T. D. G. L., 2012. Understanding Big Data Analytics for Enterprise Class Hadoop and Streaming Data. Mc Graw-Hill Companies.
- Press, G., March, 2016. Cleaning big data: Most time-consuming, least enjoyable data science task, survey says.
- Prusty, N., 2015. Learning ECMAScript 6. Packt Publishing Ltd.
- Pl Kraft, Filip Drozd, E. O. K., June 2008. Digital therapy: Addressing willpower as part of the cognitive-affective processing system in the service of habit change.
- reactjs.org, 2020. React Js introducing jsx.
URL <https://reactjs.org/docs/introducing-jsx.html>
- Redis, 2020. In-memory key value store. Accessed: 2020-04-20.
URL <http://redis.io/>
- R.H. Thaler, C. S., 2008. Nudge: Improving decisions about health, wealth and happiness.
- Richard H. Thaler, Cass R. Sunstein, J. P. B., 2013. Choice Architecture. IN The Behavioral Foundations of Public Policy, pages 428-439. Princeton University Press.
- Richard H. Thaler, Cass R. Sunstein, J. P. B., April, 2010. Choice architecture.
- Rouse, M., Oct 2018. What is olap (online analytical processing)? - definition from whatis.com.
URL <https://searchdatamanagement.techtarget.com/definition/OLAP>
- Russom, P., 2011. Big Data Analytics, TDWI best practices report, pages 1-34. Vol. 21. The Data Warehousing Institute (TDWI).
- Russom, P., et al., 2011. Big data analytics. TDWI best practices report, fourth quarter 19 (4), 1–34.
- Seth Gilbert, N. A. L., 2012. Perspectives on the cap theorem.
- Shanthy Mendis, Tim Armstrong, D. B. F. B. J. L. C. M. S. M. V. P. L. R. V. D. C. E. S. G. S., 2014. Global status report on noncommunicable diseases 2014.
-

-
- Signicat, 2021. Signicat. <https://www.signicat.com/en>, [Online; accessed 26-May-2020].
- Sir, V., July, 2016. Relational database management system).
- statista, 2021. statista. <https://www.statista.com/statistics/277125/share-of-website-traffic-coming-from-mobile-devices/#:~:text=Mobile%20accounts%20for%20approximately%20half,since%20the%20beginning%20of%202017.>, [Online; accessed 27-May-2020].
- Strauch, C., Sites, U.-L. S., Kriha, W., 2011. Nosql databases.
URL <https://christof-strauch.de/nosql dbs.pdf>
- Team, T. B. I., 2015-2016. The behavioural insights team.
- Thomas H. Davenport, P. B., Bean, R., Fall ,2012. How big data is different.
- Tilkov, S., Vinoski, S., 2010. Node. js: Using javascript to build high-performance network programs. IEEE Internet Computing 14 (6), 80–83.
- Tobias Mirsch, Christiane Lehrer, R. J., 2017a. Digital nudging: Altering user behavior in digital environments.
- Tobias Mirsch, Christiane Lehrer, R. J., 2017b. Digital nudging: Altering user behavior in digital environments. Paper.
- Tromskortet, 2020. Tromskortet. <https://www.tromskortet.no>, [Online; accessed 13-December-2020].
- Valter Balegas, Sergio Duart, e. C. F. R. R. N. P., April, 2015. Putting consistency back into eventual consistency.
- Vaughan, A., 2019. Can you shop with your dna? New Scientist 244 (3256), 17.
URL <http://www.sciencedirect.com/science/article/pii/S0262407919321475>
- Wael M.S. Yafooz, Siti Z.Z. Abidin, N. O. Z. I., December 2013. Managing unstructured data in relational databases).
- Watson, H. J., 2014. Tutorial: Big data analytics: Concepts, technologies, and applications. Communications of the Association for Information Systems 34 (1), 65.
- Weber, E. J. J. . S. B. S. . B. G. C. D. . C. F. . D. G. G. . G. H. . R. P. L. . J. W. P. . E. P. . D. S. . B. W. . E. U., 2012. Beyond nudges: Tools of a choice architecture.
- Youssra, R., Sara, R., 2018. Big data and big data analytics: concepts, types and technologies. Int J Res Eng 5 (9), 524–528.

