Pulse Infographics

A Project Report
Submitted in partial fulfilment of the
Requirements for the award of the Degree of

BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)

By

Joel Jose
TIT2122040

Under the esteemed guidance of
Mrs. Biju Ramesh

DEPARTMENT OF INFORMATION TECHNOLOGY
SIES COLLEGE OF ARTS, SCIENCE & COMMERCE
(AUTONOMOUS)
SION (W), MUMBAI, 400022
MAHARASHTRA

2021

PROFORMA FOR THE APPROVAL PROJECT PROPOSAL

Roll No**:** TIT2122040

1. Name of the Student — Mr. Joel Jose
2. Title of the Project — Pulse Infographics
3. Name of the Guide — Mrs. Biju Ramesh
4. Teaching experience of the Guide — 18 years
5. Is this your first submission? — Yes ☑ No ☐

Joel Jose
Signature of the Student
Date: **05/10/2021**

Ms. Biju Ramesh
Signature of the Guide
Date: **05/10/2021**

Ms. Sudha. B
Signature of the Coordinator
Date: **05/10/2021**

SIES COLLEGE OF ARTS, SCIENCE & COMMERCE
(AUTONOMOUS)
(Affiliated to University of Mumbai)
SION (W), MUMBAI-400022

DEPARTMENT OF INFORMATION TECHNOLOGY

SIES College of Arts, Science & Commerce
RISE WITH EDUCATION
Sion (West), Mumbai – 400022.
(Autonomous)

CERTIFICATE

This is to certify that the project entitled, **"Pulse Infographics"**, is Bona fide work of **Mr. Joel Jose** bearing Seat No: **TIT2122040 s**ubmitted in partial fulfilment of the requirements for the award of degree of **BACHELOR OF SCIENCE in INFORMATION TECHNOLOGY.**

Ms. Biju Ramesh
Internal Guide

Ms. Sudha B.
Coordinator

Date: 05/10/2021

Ms. Biju Ramesh

External Examiner

Abstract

The Project at hand is made with respect to real-time systems present in our day-to-day world. It brings around various fields where information is necessary to a single platform and presents the information in a neat and user-readable format. It cultivates and presents the user with various checkpoints and shares necessary insight with respect to the data that was collected.

This project is intended with the notion of representing information in the graphical perspective, as to give visual insights to the field. It also aims at bringing multiple relevant fields with different methods of execution in a single landing site. This makes the entire project fragmented with individual components that can be altered if necessary. The graphical implementation is done with several checkpoints to further make the information subject to analysis. The intended representation is done with various modules and different implementation methods.

To summarize, it is safe to say the project is simply a data collection, analysis, assessment, and representation model. It helps with informatics and infographics, as the name suggests and is simply implemented with varying methods of implementation.

ACKNOWLEDGEMENT

The completion of the project at hand was possible only with the guidance and help of the faculty of SIES college. The direction and accessible help that was conferred towards us, helped us work under guided timelines. I would like to acknowledge and express my gratitude towards my project guide Mrs. Biju Ramesh and Mrs. Sudha B., H.O.D. of IT Department, SIES College of Arts, Science & Commerce (Autonomous)

I would also like to accredit my friends, family members, and other group members for their support and encouragement throughout my project.

DECLARATION

I hereby declare that the project entitled, "**Pulse Infographics**" done at **SIES COLLEGE OF ARTS, SCIENCE AND COMMERCE (AUTONOMOUS)**, has not been in any case duplicated to submit to any other university for the award of any degree. To the best of my knowledge other than me, no one has submitted to any other university.

The project is done in partial fulfilment of the requirements for the award of degree of **BACHELOR OF SCIENCE (INFORMATION TECHNOLOGY)** to be submitted as final semester project as part of our curriculum.

Joel Jose

Name and Signature of the Student

Table of Content

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

1.1 Background:

Infographics is the representation of data, information or knowledge in the graphical format. Information is always best represented visually, which helps others to understand and analyse trends or characteristics of the field from which the information is collected. It can become a tool to further derive useful analytics and help a user to predict or forecast a behaviour of a system.

There already exists free information available for different fields on the internet. Using this information, we can derive a model that accurately describes and educates us with respect to understanding trends. Information analysis helps a person to make informed decisions which further helps to predict and hypothesize conditions that will benefit them as well as the society.

The project simply is designed to bring various fields where information has become an irreplaceable resource for analysis and accurately displays it using the graphical notations. The fields that have been taken into consideration are all in respect to information sharing and are trending with respect to economic impacts in the society. The name pulse suggests that the information fields are as important as the pulse of the human heart, indicating life. Hence, the project simply takes into consideration the representation of information that is of the utmost importance.

1.2 Objectives:

The project aims to bring a unified front in data analysis and visualization regarding data science. It gives users different live information to understand the implications of that field. Users also get to review the web application's interface and other aspects by providing their feedback using a part of the application's feedback form. The objective is simply to represent information from those fields which have the highest importance with respect to the economic and societal aspect.

To sum it up the project aims at infographic representation of data from economically sensitive fields of science and technology. Hence it is purely of the intention to provide information to those seeking graphical representation and for studying better ways to represent the information by taking feedback.

1.3 Purpose, Scope, Applicability

1.3.1 purpose

Analysing as well as interpreting data has become exceedingly difficult in terms with respect to time consumption. It almost is a frustrating endeavour as it becomes

difficult to make sense out of raw and numeric data. Research data almost is always available since the dawn of internet and hence there exists several places from which we can fetch and gain information. Using this information, we can make a model that accurately describe and makes sense of the data in the visual format. Updating data using requests from a server will ensure it is always real time and new.

There already exists many already available such information displaying websites and apps, but none of them cover all the fields at the same time. Hence the true purpose is to do exactly a convergence of this representation and analytics of data.

1.3.2 Scope:

The entire project is a fragmented framework, meaning it is a project with multi-server execution interface. Each server will independently cover the workflow that it is assigned to do, using the framework in which it is built. Once the execution starts the server will cover one-page workflow and keep refreshing and inputting information for the graphical display. The multi web-app is compatible with all browsers supporting the Mozilla web framework.

The project has many checkpoints and has the scope to change and expand infinitely if the servers are supported and working. The message entry from feedback will be secured using smtp requests and managed in a database. With respect to entity relationships there is no interconnection of data for the different databases and hence the entire project is disconnected making it an advantage as well as a disadvantage.

1.3.3 Applicability:

The application of real time data scraping, and visualisation is as much important as the internet's primary goal that is to connect people virtually and share information globally. The entire project applies the concept of net scraping and information utilisation to make a service better and more available to the network of people using it. The more people use a service the better the service gets by constantly being subjected to feedback. Infographic application has almost become the primary focus of today's data science platform and is necessary for future benefaction.

Chapter 2: Survey of Technologies

2.1 Introduction:

A web application is application software that runs on a web server, unlike computer-based software programs that are run locally on the operating system of the device. Web applications are accessed by the user through a web browser with an active network connection.

The following project at hand is made in regard with various web-based architectures and frameworks available for different programming languages. The Frameworks used are mainly those which offer server-side solutions to handling requests and are integrated with database if necessary. The languages used for the following projects differ with respect to individual projects, and are completely independent in regard with the server-based architecture that they run in. Hence, there are multiple programming languages used in this project along with differing frameworks and architectures.

In the following session we will see various frameworks and technologies that can be used for both front end and backend development.

2.2 Front-End Development:

Front-end web development is known as the client-side web development and is the practice of producing interactable interface for a website or web application, so that the user can see and interact with them directly. The challenge associated with front end development is that the tools and techniques used to create the front end of a website change constantly with the programming languages and frameworks that are used.

Let us see the languages used for the front-end development in this project.

2.2.1 Hyper Text Markup Language (HTML)

HTML is used extensively for the structuring of web pages. It helps with adding components such as tables, links, buttons, and content that is to be displayed using the specified tags. All websites and web applications make use of html file, even if html is not used directly while programming the interface.

Advantages:

- HTML is widely used.
- Every browser supports HTML Language.
- Easy to learn and use.
- HTML is light weighted and fast to load.
- Loose syntax (although, being too flexible won't suit standards).
- HTML is easy enough to write

- HTML is that it is easy to code even for novice programmers.
- HTML is increasingly used for data storage as like XML syntax.
- Free – You need not buy any software.
- HTML is present in every window by default, so we don't have to buy it.
- HTML has many tags and attributes which can short your line of code.

Disadvantages:

- It cannot produce dynamic output alone since it's a static language.
- Making the structure of HTML documents becomes tough to understand.
- Errors can be costly.
- It can create only static and plain pages so if we'd like dynamic pages then HTML isn't useful.
- Required to write a lot of code for just creating a simple webpage.
- We must check up the deprecated tags and confirm not to use them to appear because another language that works with HTML has replaced the first work of the tag, and hence the opposite language needs to be understood and learned.
- Security features offered by HTML are limited.
- If we need to write down long code for creating a webpage then it produces some complexity.
- HTML can create only static and plain pages so if we'd like dynamic pages then HTML isn't useful.
- Security features are not good at HTML.

2.2.2 Cascading Style Sheets (CSS)

CSS is used along with html either internally or externally to make the website look more appealing to the users, and in turn makes the website user friendly. The layout as well as the colour and size of different html components can be easily manipulated using CSS. Bootstrap that is precompiled CSS code is also used to manipulate html components. Bootstrap based code is available on websites and can be added to our html using the class name for the given tag.

Advantages of CSS:

- It is less complex therefore the effort is significantly reduced.
- It helps to form spontaneous and consistent changes.
- CSS changes are device friendly. With people employing a batch of various range of smart devices to access websites over the web, there's a requirement

for responsive web design.

- It has the power for re-positioning. It helps us to determine the changes within the position of web elements who are there on the page.
- These bandwidth savings are substantial figures of insignificant tags that are indistinct from a mess of pages.
- Easy for the user to customize the online page
- It reduces the file transfer size.

Disadvantages of CSS:

- With CSS, what works with one browser might not always work with another. The web developers need to test for compatibility, running the program across multiple browsers.
- There exists a scarcity of security.
- Browser compatibility (some styles sheets are supported, and some are not).
- CSS works differently on different browsers. IE and Opera supports CSS as different logic.
- There might be cross-browser issues while using CSS.
- There are multiple levels which creates confusion for non-developers and beginners.

2.2.3 Java Script (JS)

JS is used for both front-end as well as backend development. This project also makes use of JS for the same. JS is used for front-end development mainly to handle client-side requests and interactions with the web page. It also helps with making the website more feature rich and enhances the user's experience.

JS can be used to handle server-based requests and events such as rendering and fetching data from the user also since JS is used in almost all web apps it becomes an excellent language for backend development. We can also integrate JS to use different API's and make the process of coding simpler for developers.

Advantages of JavaScript

- Speed. Client-side JavaScript is very fast because it can be run immediately within the client-side browser. Unless outside resources are required, JavaScript is unhindered by network calls to a backend server.
- Simplicity. JavaScript is relatively simple to learn and implement.
- Popularity. JavaScript is used everywhere on the web.

- Interoperability. JavaScript plays nicely with other languages and can be used in a huge variety of applications.
- Server Load. Being client-side reduces the demand on the website server.
- Gives the ability to create rich interfaces.

Disadvantages of JavaScript

- Client-Side Security. Because the code executes on the users' computer, in some cases it can be exploited for malicious purposes. This is one reason some people choose to disable JavaScript.
- Browser Support. JavaScript is sometimes interpreted differently by different browsers. This makes it somewhat difficult to write cross-browser code.

2.2.4 React JS (R-JS)

React JS is an open-source frontend framework that is based on JavaScript, developed by Facebook, and best known for its virtual DOM feature. Virtual DOM helps with making the HTML and CSS part of the code easily modifiable and makes it easier for programmers to make changes to the website without having to physically change the already existing layout and design. The virtual DOM feature makes react a very viable and preferred choice for frontend development.

React has been recently used in many corporations as it is a very powerful tool as well as makes the process of making dynamic websites and web apps easier. React also is very secure and makes use of all the latest ES6 JS components.

Advantage of ReactJS

- Easy to Learn and Use
- Creating Dynamic Web Applications Becomes Easier
- Reusable Components
- Performance Enhancement
- The Support of Handy Tools
- Known to be SEO Friendly
- The Benefit of Having JavaScript Library
- Scope for Testing the Codes

Disadvantage of ReactJS

- The high pace of development
- Poor Documentation
- View Partition
- JSX as a barrier

2.2.5 Python Streamlit

Streamlit is a Python library-based framework that allows people to build frontend user interface for our machine learning and data science apps by writing all the code in Python. Beautiful user interface can easily be designed through numerous components from the library. It is mainly used in machine learning projects or projects that display graphical content.

Streamlit is extensively used for data science projects and hence the web apps that are using streamlit have an advantage in regard with the interface of the web app. Streamlit web apps have a certain level of abstraction that makes programming and developing apps very simple.

Streamlit's Features:

- Free and open source
- Build apps in a dozen lines of Python with a simple API
- No call-backs
- No hidden state
- Works with TensorFlow, Keras, PyTorch, Pandas, Numpy, Matplotlib, Seaborn, Altair, Plotly, Bokeh, Vega-Lite, and more

2.2.6 Angular JS

AngularJS is an open-source web application framework. It was originally developed in 2009 and is now maintained by Google. AngularJS is a structural framework for dynamic web applications. It lets you use HTML as your template language and lets you extend HTML's syntax to express your application components clearly and succinctly. Its data binding and dependency injection eliminate much of the code you currently must write. And it all happens within the browser, making it an ideal partner with any server technology.

Advantages of Angular:

- MVC Architecture implementation
- Enhanced Design Architecture
- Many supported Modules

Disadvantages of Angular:

- Limited SEO options
- Angular is verbose and complex
- Steep learning curve

2.3 Back-End Development:

Backend Development is also known as server-side development. It is everything that the users don't see and contains behind-the-scenes activities that occur when performing any action on a website. It focuses primarily on databases, backend logic, APIs, and Servers. Backend development languages handle the 'behind-the-scenes' functionality of web applications. It's code that connects the web to a database, manages user connections, and powers the web application itself. Backend development works in tandem with the front end to deliver the final product to the end user.

Let us see the languages used for the back-end development.

2.3.1 PHP

PHP is a popular general-purpose scripting language that is especially suited to web development. Fast, flexible, and pragmatic, PHP powers everything from your blog to the most popular websites in the world.

Advantages:

- Most important advantage of PHP is that it's open source and freed from cost. It is often downloaded anywhere and readily available to use for event of web applications.
- It is platform independent. PHP based applications can run on any OS like UNIX, Linux and windows.
- It has less learning curve because it is straightforward and straightforward to use. If a private knows C programming can easily work on PHP.
- It is more stable from a few years with assistance of providing continuous support to various versions.
- It helps in reusing an equivalent code and no got to write lengthy code and sophisticated structure for event of web applications.
- It helps in managing code easily.
- It has powerful library support to use various function modules for data representation.

Disadvantages:

- It is not that secure since it is open source, because the ASCII text file are often easily available.
- It is not suitable for giant content-based web applications.
- It has a weak type, which can cause incorrect data and knowledge to user.
- PHP frameworks got to learn to use PHP built-in functionalities to avoid writing additional code.

- Using more features of PHP framework and tools cause poor performance of online applications.
- reference documentation, there are easier programming languages for web apps.
- It's highly tough to manage because, it's not competent modular. It already imitates the features of Java language.

2.3.2 Python Flask

Flask is a popular Python web framework, meaning it is a third-party Python library used for developing web applications. Flask is a micro web framework written in Python. It is classified as a microframework because it does not require tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions.

There are many modules or frameworks which allows to build your webpage using python like bottle, Django, flask etc. But the real popular ones are Flask and Django. Django is easy to use as compared to Flask, but Flask provides you the versatility to program with.

Various features of flask:

- WSGI: Web Server Gateway Interface (WSGI) has been adopted as a standard for Python web application development. WSGI is a specification for a universal interface between the web server and the web applications.
- Werkzeug: It is a WSGI toolkit, which implements requests, response objects, and other utility functions. This enables building a web framework on top of it. The Flask framework uses Werkzeug as one of its bases.
- jinja2: jinja2 is a popular templating engine for Python. A web templating system combines a template with a certain data source to render dynamic web pages.

2.3.3 Django

Django is python-based Framework that is used to develop complete web-apps that includes front-end as well as back-end. For front-end you can use HTML, CSS, bootstrap, JavaScript. Great thing is that Django supports Jinja templates (if you are not aware of Jinja templates. Please Google…). With the help of Django, you can develop dynamic web-apps. As Python has tons of library with good readability and for the rest of the things Django is there to help you out. Another great thing is that you can develop REST API with the help of Django REST Framework and further that REST API can be consumed by any client-side apps, let it be web-app, android, or iOS app. So basically, REST API is used to serve common backend or business logic to every client-side app.

Advantages Of Django

Fast: This has been designed in a way to help the developers make an application as fast as possible. From idea, production to release, Django helps in making it both cost effective and efficient. Thus, it becomes an ideal solution for developers having a primary focus on deadlines.

Fully Loaded: It works in a way that includes dozens of extras to help with user authentication, site maps, content administration, RSS feeds and much more such things. These aspects help in carrying out the web development process completely.

Secure: When you are doing it in Django, it is ensured that developers don't commit any mistakes related to security. Some of the common mistakes include SQL injection, cross-site request forgery, clickjacking and cross-site scripting. To effectively manage usernames and passwords, the user authentication system is the key.

Scalable: To meet the heaviest traffic demand, the benefits of Django framework can be seen. Therefore, the busiest sites use this medium to quickly meet the traffic demands.

Versatile: Content management, scientific computing platforms, and even big organizations, all these aspects are very efficiently managed using Django.

Disadvantages of Django:

Uses routing pattern specific to its URL

Django is too monolithic

Everything is based on Django ORM

Components get deployed together

Knowledge of full system is required to work.

2.3.4 Ruby on rails

Ruby on Rails is an open-source software used to build web applications. Rails is a framework used to create websites using the general-purpose programming language Ruby. Ruby ranks amongst the top ten programming languages predominantly because of the vogueishness of Rails.

As we know that most of the languages like Java, HTML, CSS, etc. do not cover the front end and back end. They either only for the back end or for the front end but Ruby on Rails is used for both front end back end, it is like a complete package to develop a web application.

Advantages of Ruby on Rails

Tooling: Rails provides tooling that helps us to deliver more features in less time.

Libraries: There's a 3rd party module(gem) for just about anything we can think of.

Code Quality: Ruby code quality significantly higher than PHP or NodeJS equivalents.

Test Automation: The Ruby community is big into and test automation and testing.

Large Community: Ruby is large in the community.

Productivity: Ruby is incredibly fast from another language. Its productivity is high.

Disadvantages of Ruby on Rails

Runtime Speed: The run time speed of Ruby on Rails is slow as compared to NodeJS and Golang.

Lack of Flexibility: As we know that Ruby on Rails is ideal for standard web applications due to its hard dependency between components and models. But when it comes to adding unique functionality and customization in apps it is challenging.

Boot Speed: The boot speed is also a drawback of ROR. Due to the dependence upon the number of gem dependencies and files, it takes some time to start which can obstruct the developer performance.

Documentation: To find good documentation is hard for the less popular gems and for libraries that make heavy use of mixing.

Multithreading: Ruby on Rails supports multithreading, but some IO libraries do not support multithreading because they keep hold of the global interpreter lock. So it means if you are not careful enough, your request will get queued up behind the active requests, and you will face performance issues.

Active Record: Due to the access use of Active records in the ROR and hard dependency, the domain becomes tightly coupled to your persistence mechanism.

2.4 Database languages, Integrations, and frameworks:

Database Management Languages:

Database languages are used to read, update and store data in a database. There are several such languages that can be used for this purpose; one of them is SQL (Structured Query Language).

Types of DBMS languages:

DQL (Data Query Language) is used to fetch the information from the database which is already stored there.

DDL (Data Definition Language) is used to define table schemas.

DCL (Data Control Language) is used for user & permission management. It controls the access to the database.

DML (Data Manipulation Language) is used for inserting, updating, and deleting data from the database.

Frameworks:

2.4.1 SQL Server

SQL Server is a relational database management system, or RDBMS, developed and marketed by Microsoft. SQL Server is built on top of SQL, a standard programming language for interacting with the relational databases. A database server is a computer program that provides database services to other programs or computers, as defined by the client-server model. Therefore, a SQL Server is a database server that implements the Structured Query Language (SQL). There are many different versions of Microsoft SQL Server, catering for different workloads and demands. A data centre version is tailored to higher levels of application support and scalability, while the Express version is a scaled down, free edition of the software.

Advantages

Security Features Are Better

Lower Cost of Ownership

Optimized data storage

Data recovery support

2.4.2 Oracle Database

Oracle database is an RDMS system from Oracle Corporation. The software is built around the relational database framework. It allows data objects to be accessed by users using SQL language. Oracle is a completely scalable RDBMS architecture which is widely used all over the world. Oracle is one of the biggest vendors in the IT market and the shorthand name of its flagship RDBMS product, that was formally called Oracle Database. The database allows you to rerun actual production workloads, including online user and batch workloads, in test environments. It can be used for read-write, reporting, testing, or backups, reducing the load on the primary database.

2.4.3 MySQL

MySQL was created by a Swedish company MySQL AB. The features are like support to cross-platform, stored procedures, triggers, cursors, data definition language, ACID compliance, SSL support, views updatable, partitioning, Indexing, select, commit grouping, Unicode support and many more. There are certain limitations in My SQL. In MySQL, Triggers are limited to only one action per timing. It means only one trigger can be executed on the table if any event happens on the table. Triggers cannot be defined on views as well. The other limitation is

MySQL does not follow the full SQL standards. MySQL uses the 'MySQL dump' backup tool, which supports backing up of data from all the storage engines. The other MySQL back up software program is 'XtraBackup', which is open source. MySQL can be run on Cloud as well as Amazon and Microsoft Azure. MySQL can be used as a service.

2.4.4 PostgreSQL

PostgreSQL is a powerful, open-source object-relational database system. It is free and open-source relational database management system, maintained by PostgreSQL Global Development Group and its prolific community. PostgreSQL seems to be more universal. It is widely available on multiple operating systems: FreeBSD, HP-UX, Linux, NetBSD, OpenBSD, OS X, Solaris, Unix, Windows. PostgreSQL has user-defined functions in proprietary language PL/pgSQL or with common languages like Perl, Python, Tcl etc. PostgreSQL can be driven entirely from the command line. PostgreSQL has a better concurrency management system. It handles very well the case where multiple processes can access and modify shared data at the same time

2.5 Technologies used in this project:

We will discuss all the programming languages and web frameworks used in this project in detail in the following section. We get to see all these languages and frameworks being used in different parts of the individual projects in this infographics project. Also, we will look at how these languages work with respect to keeping the project fragmented and independently working with the server that they communicate with.

Frontend technologies used in this project:

This project uses html in almost all the web apps that have been created, as most of the browsers support html syntax and it is easy to modify and work with. Some projects don't need html as the underlying frameworks such as streamlit, which takes care of the html components by itself. It also uses CSS everywhere where html components have been used and makes use of bootstrap as well for the designing. The project uses JS for the crypto-graph web app and for the landing site. JS has been used for both server-side event handling as well as for loading data from API. It uses react mainly for the covid tracker web app as well as the news web app. Python streamlit is used for stocks as well as the bioinformatics web application.

Backend technologies used in this project:

The landing site and the cryptograph web app uses php for server-based functionality and integrations. This project uses Flask framework for almost all the web apps that are made in python. The project also implements different integrations using the Flask framework such as database integration and smtp request handling. Some projects use JS, react and Streamlit for backend as well.

This project uses SQLite for the database. SQL is a simple query language that helps to store and retrieve information in and from tables using English based queries written. Since this project is for infographics, it mainly uses the database connection to store data from the user.

Chapter 3: Requirement and Analysis

3.1 Problem Definition:

Visual representation of information helps with the annotation associated with the concept of deriving meaningful insights from raw numeric values. The raw data for the visual representation is often cultivated through various warehouses of information. This storage affiliated system is characterized by its ability to derive an understanding of the underlying paradigm. Infographic representation of information helps with the case study of various stakeholder's resources and forecasting their outcomes. The entire model of graphic annotation is based on understanding trends and improving relations to get the desired yield from the underlying system.

Infographics often uses algorithmic methods to derive the forecast of information from a system and projects this forecasted dataset onto the graphical representation. The advantage of this is that it helps users with making informed decisions that may help them. Though often these forecasts are not able to capture in enough information for processing and end up being inaccurate, they assist with an overall purely computational insight for decision making.

Infographics is not only used for research purpose but also for portraying information in a much more readable and easily understandable format. The current infographic representations are very attractive and makes the interface stand out in which they are implemented. Simple and straightforward notations can give meaningful insights about the system as well, hence using checkpoints and markers are very essential in infographic projects.

3.2 Requirements Specification:

Informatic representation primarily requires a set of data or values from a reliable source. The data should also be of numeric type or else working with the data will need additional computation, which will increase the processing and resource management specifications. The computed set of data should contain checkers or fields so that it can be managed according to the graphical representation. Huge sets of data can create a latency in processing and will require massive storage. Using new and fresh datasets is essential for keeping the application or user interface relevant.

Redundant information can create unreliable graphs and hence should be scraped out to increase the precision of processing. Freely available API's can make the task of processing simple, yet the entire project will in turn be contingent on whether the API is working or whether the API is reliable. The processed data set from an API will only be reliable enough for a prediction model or to exhibit it directly in the form of a graph. Analysis is essential for infographics project hence to make analysis easier the graphical notations should include proper notations. Using modern graphical representation methods are also very important as it is more appealing to the users.

3.3 Planning and Scheduling:

3.3.1 Planning:

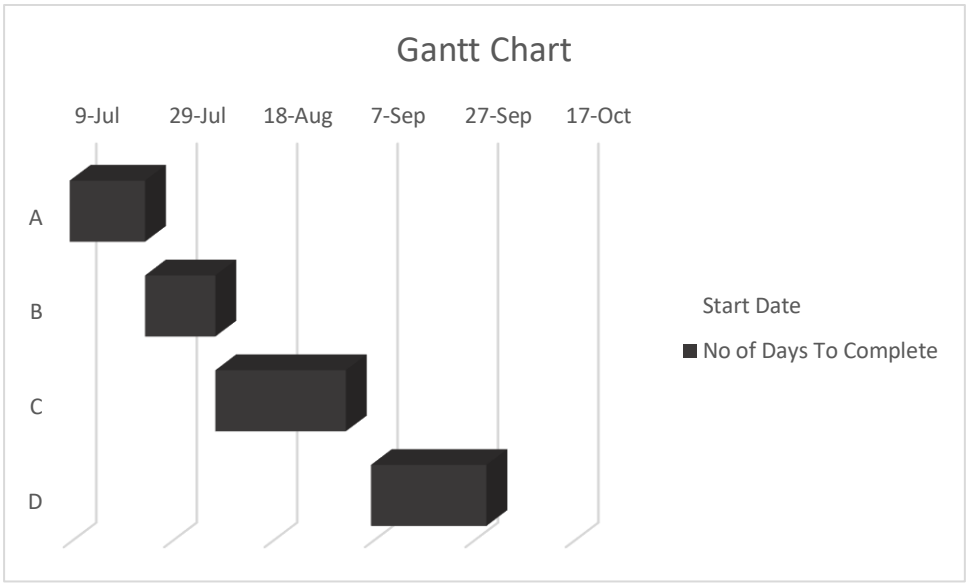
The most important task in infographics is managing the data that we work with for the representation. To work with the data, we need to exactly know all its fields as well as how it should be represented. Storing, retrieving, and processing information should all be done in a sequential format that makes the formatting effective.

Proper execution of these steps needs a supporting framework. Along with the framework we need to create a simple user-friendly interface for the representation. The project also requires us to select the fields for which we create and represent the data in graphical formats. The individuality of all fields makes it necessary to work with them sequentially and use a bottom-top approach. Even though we need to work regarding the bottom-top approach we need to have a good foresight of the final result.

3.3.2 Scheduling:

Gantt Chart:

Gantt Chart is used to visually display the scheduling of the project phases in which the activities are broken down and displayed on a chart which makes it is easy to understand and interpret. It is commonly used for tracking project schedules. They help one to access how long a project should take and helps to monitor a project progress once it's underway. The below shown Gantt Chart is for documentation (Chapter 1 to Chapter 4) which shows the time taken to complete each Chapter.



PERT Diagram:

PERT is the abbreviation for Project Evaluation Review Technique. It is a technique used for planning, scheduling, organizing, coordinating tasks within a project. It is a method to analyse the tasks involved in completing a given project, mainly the time needed to complete each task and to identify the minimum time needed to complete the total project. As this chart uses critical path analysis which helps to create schedules that will make the project to go in a smoother way. It is also useful to measure future consequences of activities in the project. The below PERT diagram shows the schedule of the Chapters.

Chapter 1: A, Chapter 2: B, Chapter 3: C, Chapter 4: D

Chapter 1: A **Chapter 2: B** **Chapter 3: C** **Chapter 4: D**

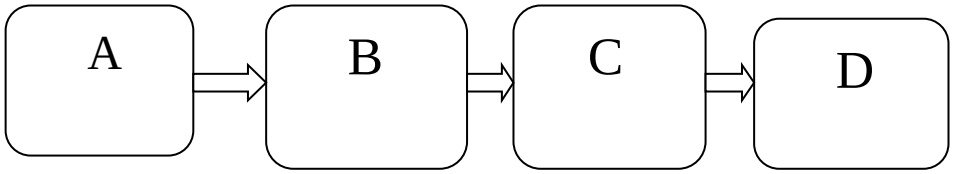


Figure 3.2: Activity - (A)

Activity	**Precedence**	**Duration**
A	-	15 days
B	A	14 days
C	B	26 days
D	C	23 days

Table 3.1: Critical Activity - (A)

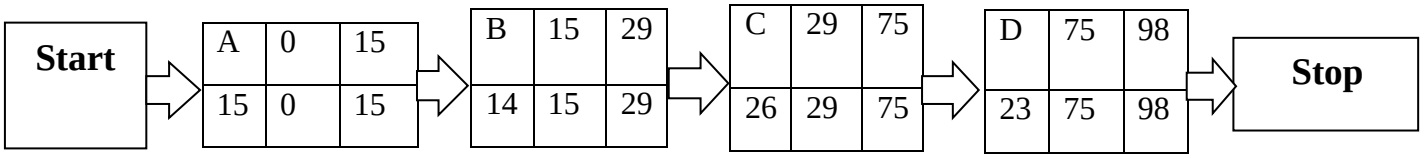


Figure 3.3: Critical Activity of - (A)

Slack = 0 for all the activities.

Therefore, Critical Path = A-B-C-D

3.4 Software and Hardware Requirements:

3.4.1 Software:

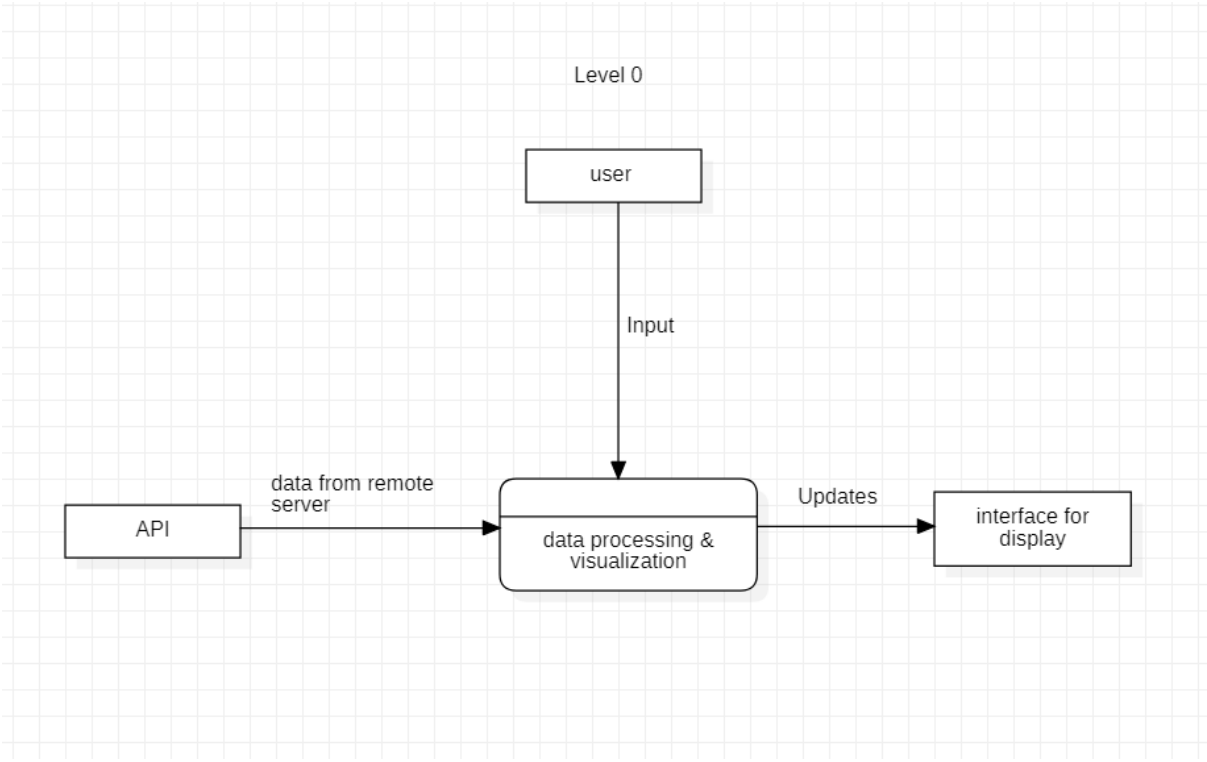
1. Any OS(Linux/windows/macOS)
2. Any IDE for coding (Visual studio code/Atom)
3. Python (Flask and streamlit)
4. React (Nodejs)
5. SQLite & DB browser
6. Browser with Html5 & ES6 support
7. PHP version 6 and above

3.4.2 Hardware:

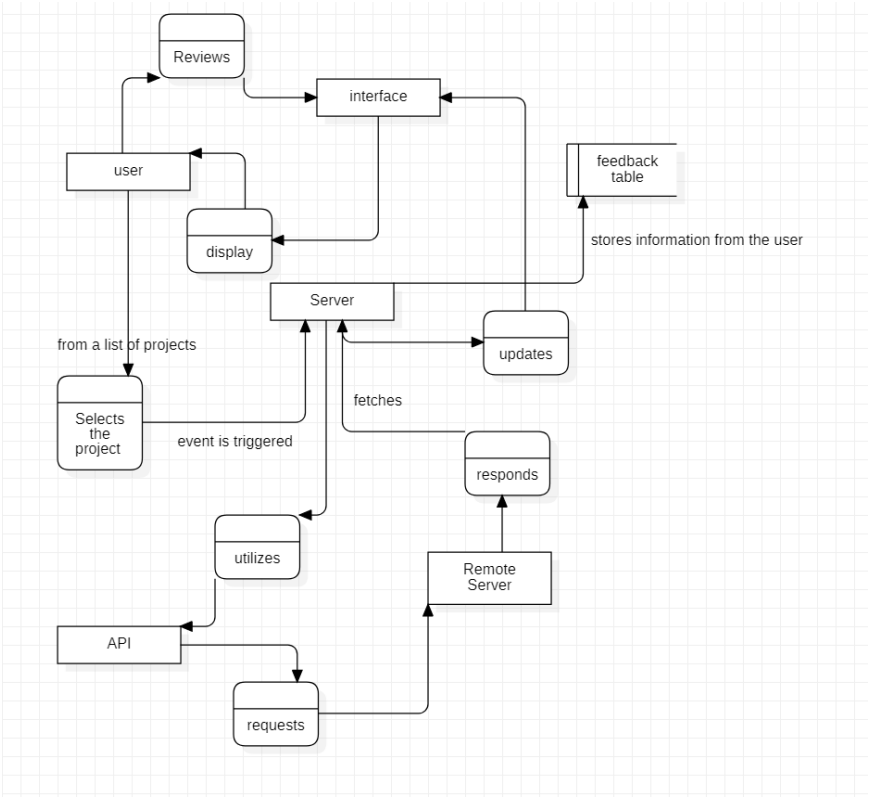
1. Laptop or PC
2. Any Processor (Intel/AMD) with 2 cores and 2ghz core frequency or more
3. At least 4GB ram (8/16 GB is preferred)
4. At least 250 GB local storage.

3.5 Conceptual Model

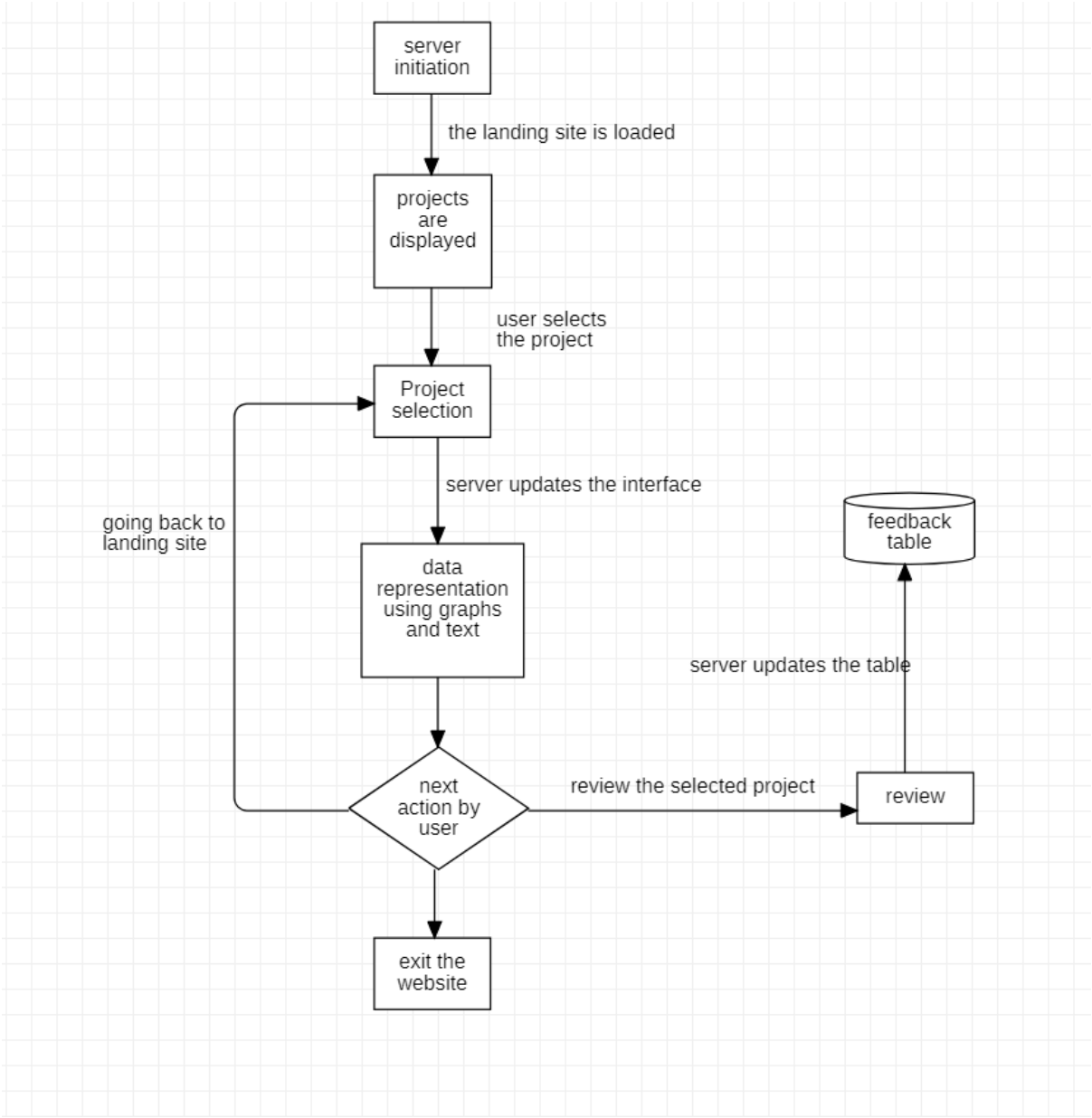
3.5.1 Data Flow Diagram



Level 1



3.5.2 Flow Chart



Chapter 4: System Design

4.1 Basic Modules:

The Project at hand is an accumulation of small data science web deployable projects

Hence it contains the following modules in it

1. **Landing site**: The landing site contains a list of all the projects as well as an option to review the project. The landing site contains the abstract of the project in it's about section
2. **The deployable web apps**: These are of different types and are relevant to their fields. Their main functionality is to offer infographic representation of live data from remote servers.
3. **Review page**: The Review page contains a feedback form that posts the user details as well as the reviewed information to the feedback table and uses smtp to store it in mail trap

4.2 General Issues:

The Project has a fragmented layout and design, making it discontinuous and it requires high maintenance. Since there are no user credentials involved in this project the security aspect is uncompromised, but it implements multiple frameworks and hence is dependant on those frameworks to keep working when it's deployed.

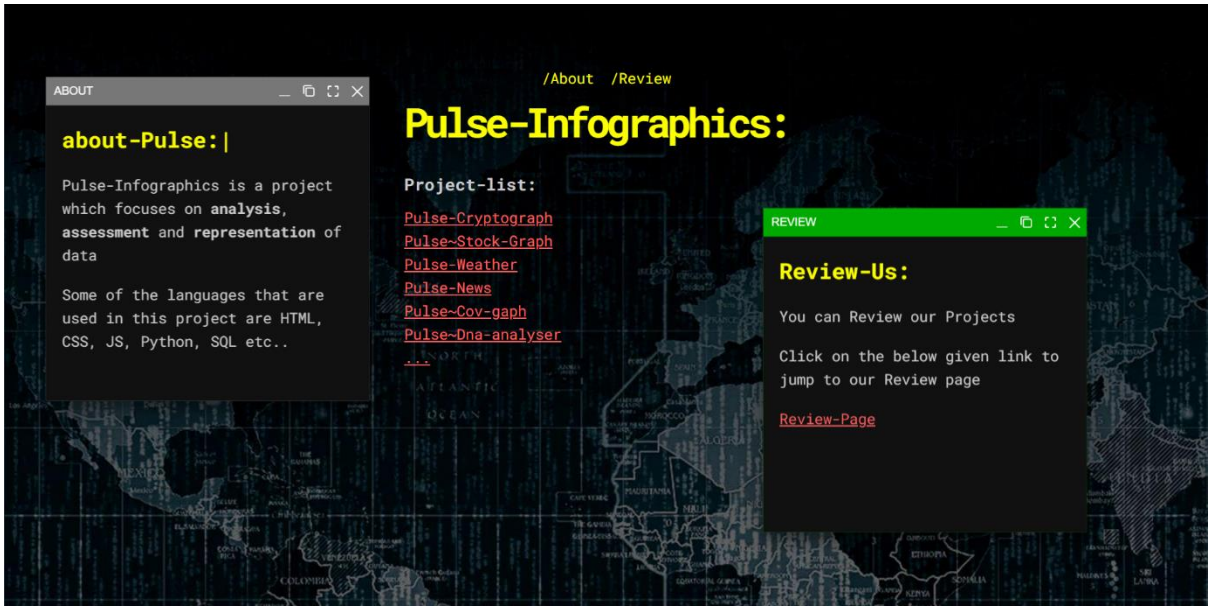
Any change in the framework can result in an entire application being subjected to failure including the dependence on open and free Web API's. The moment a company or organisation discontinues or deprecates its data provision the application will no longer be able to process and display data.

Multi-server implementation makes the project resource heavy and implementing it requires higher system and hardware requirements. The Use of free API's can lead to bad requests and responses from the remote server as it is subject to exploitation and poor maintenance.

This entire project is in regard with Data Science and learning and hence has very little economic or organisational value and can only be associated regarding data visualisation and representation.

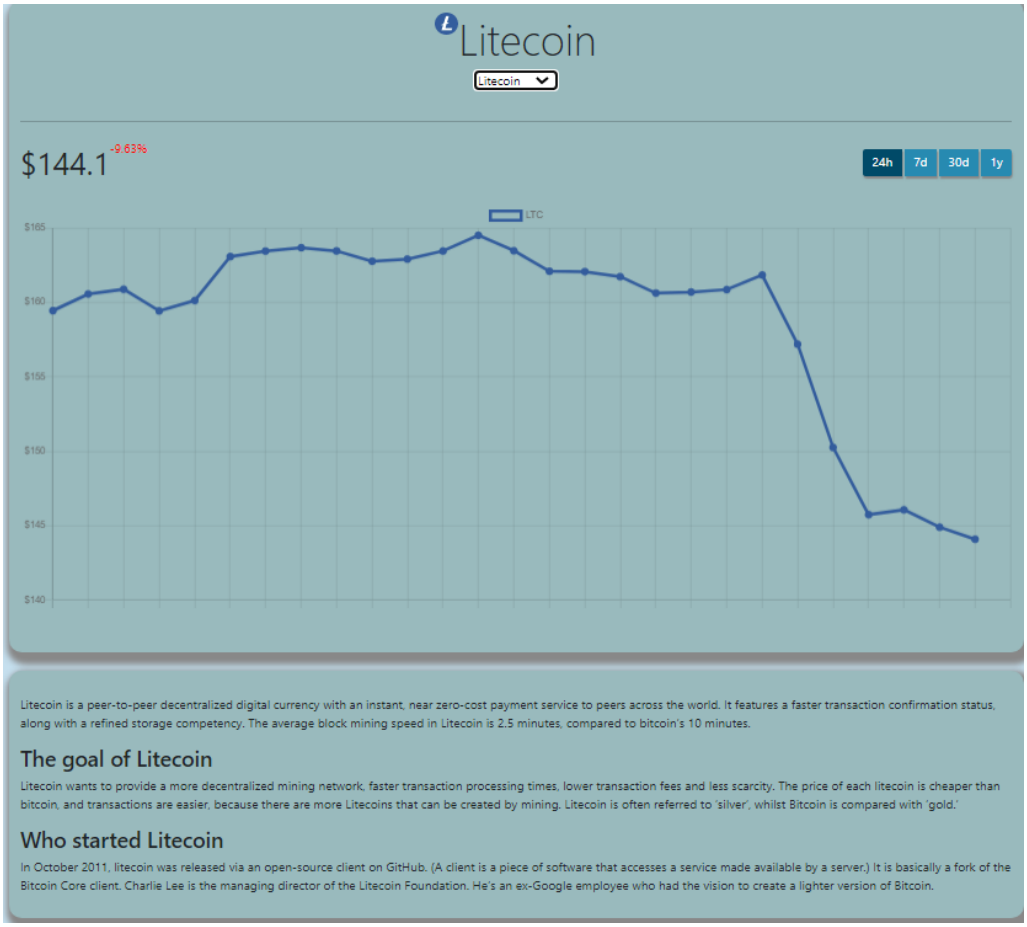
4.3 User Interface Designs:

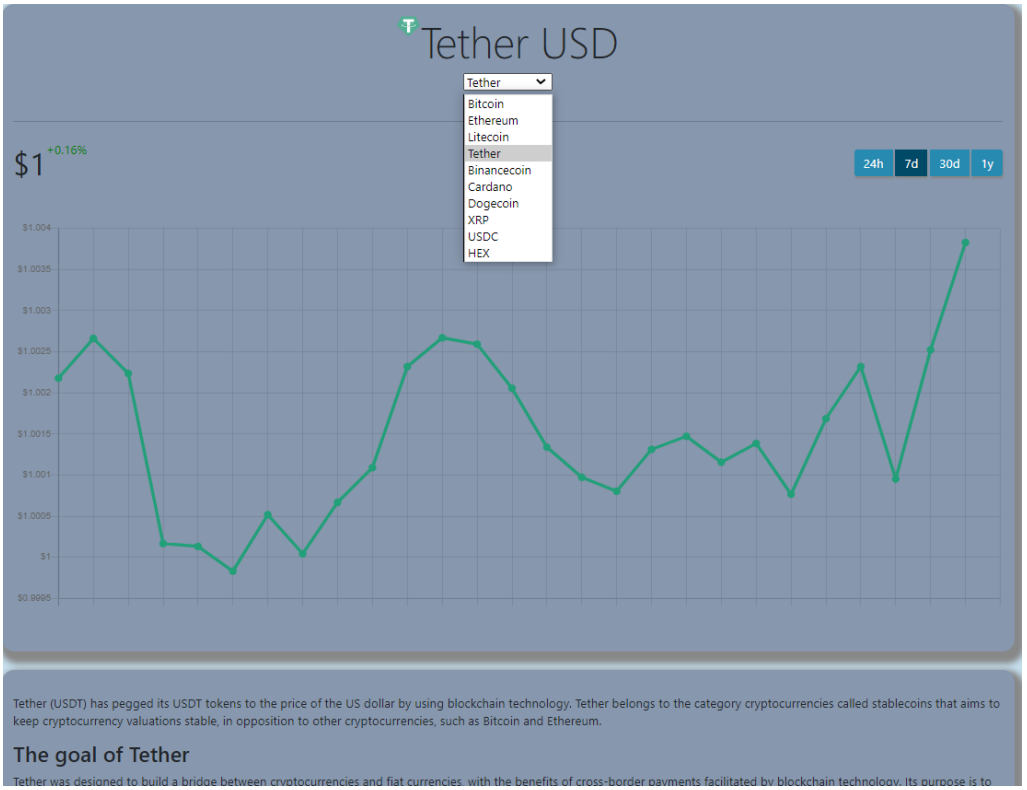
Landing site:



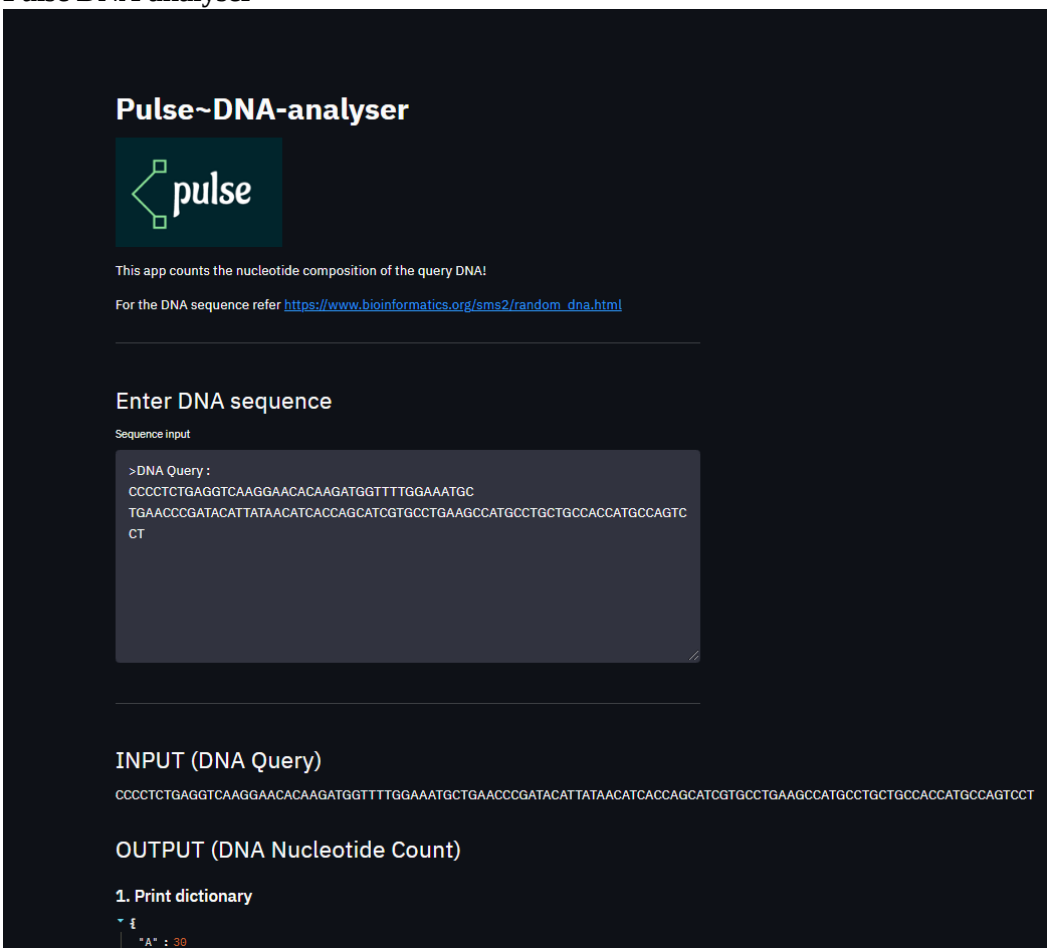
1. **User interface for the deployable web apps from the list**

Pulse cryptograph

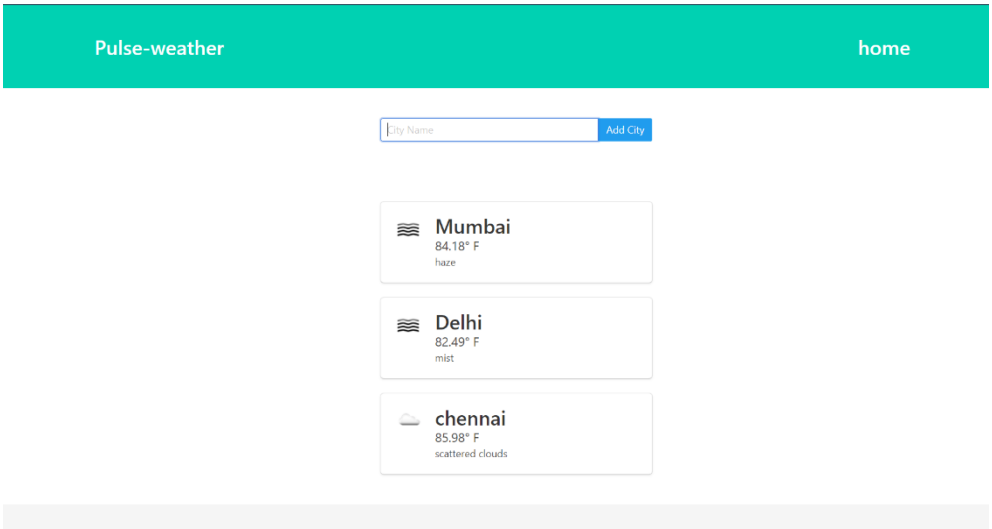




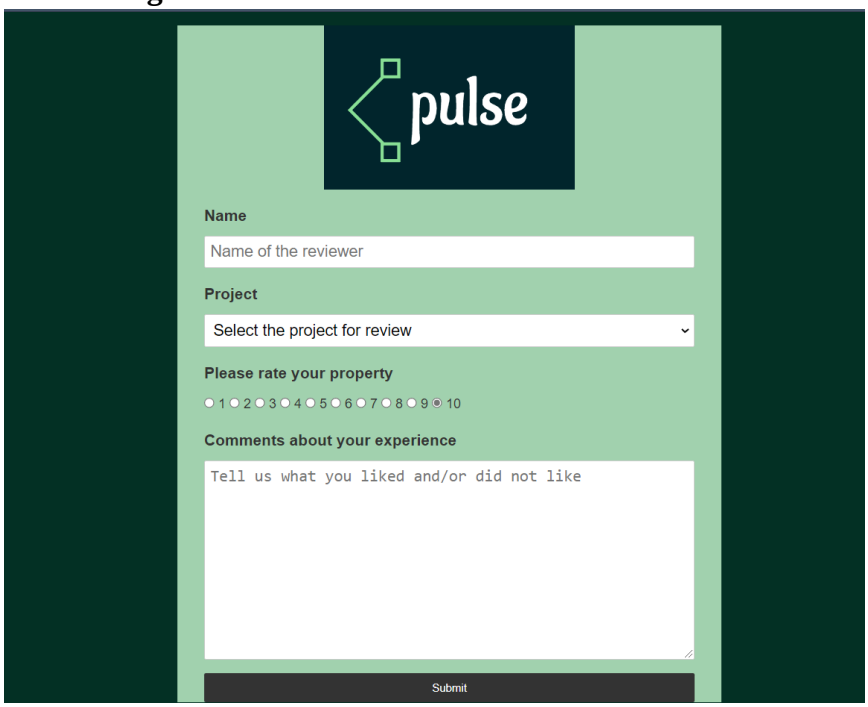
Pulse DNA analyser



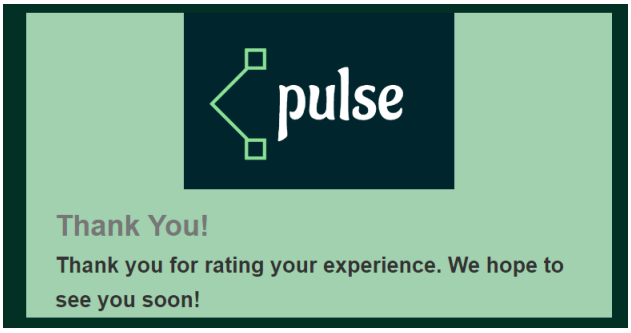
Pulse weather



Review Page



After submitting



References:

https://datadialogs.ischool.berkeley.edu/2015/schedule/running-agile-data-science-teams

https://www.edureka.co/blog/data-science-projects/

https://rockcontent.com/blog/source-code-the-5-rules-of-researching-and-sourcing-infographics/

https://data-flair.training/blogs/data-science-project-ideas/

https://stackoverflow.com/

SIES COLLEGE OF ARTS,SCIENCE & COMMERCE(AUTONOMOUS)
(Affiliated to University of Mumbai)
SION(W), MUMBAI-400022

DEPARTMENT OF INFORMATION TECHNOLOGY

SIES College of Arts, Science & Commerce
RISE WITH EDUCATION
Sion (West), Mumbai – 400022.
(Autonomous)

CERTIFICATE

This is to certify that the project entitled, **"Pulse Infographics "**, is bonafied work of **Joel Jose** bearing Seat. No: (**TIT2122040**) and the documentation of the project is submitted online in Microsoft Teams in partial fulfillment of the requirements for the award of degree of BACHELOR OF SCIENCE in INFORMATION TECHNOLOGY from University of Mumbai.

28/3/22

Internal Guide
Examiner

S.I.E.S. COLLEGE OF ARTS, SC. & COM.+ MUMBAI 400 022

28/3/22

External

Date: 28/03/2022

College Seal

TABLE OF CONTENTS

List of Tables

List of Figures

Modifications made under Chapter 1, 2, 3 4

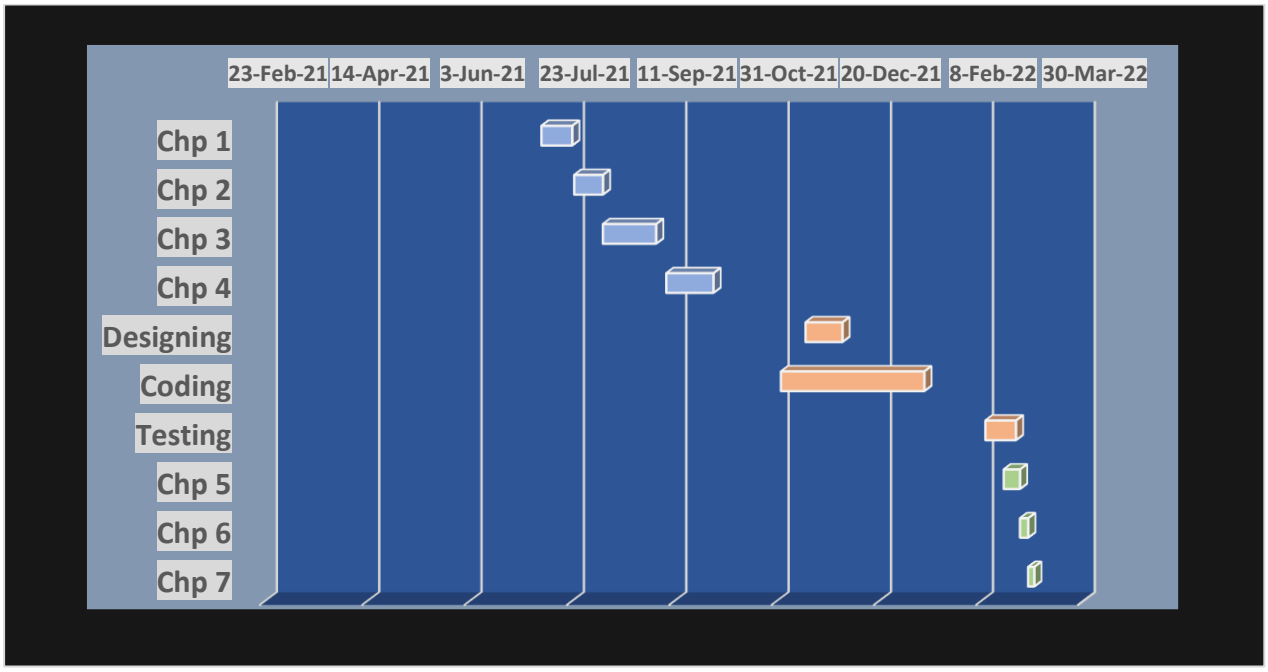
change in chapter 2:

Frontend technologies used in this project:

This project uses html in almost all the web apps that have been created, as most of the browsers support html syntax and it is easy to modify and work with. Some projects don't need html as the underlying frameworks such as streamlit, which takes care of the html components by itself. It also uses CSS everywhere where html components have been used and makes use of bootstrap as well for the designing. The project uses JS for the crypto-graph web app and for the landing site. JS has been used for both server-side event handling as well as for loading data from API. It uses react mainly for the expense tracker web app as well as the Pulse~advisor web app. Python streamlit is used for stocks graph as well as the DNA analyser web application.

Gantt Chart

There are changes in the Gantt Chart. The modified Gantt chart is given below:



PERT:

There are additions in the PERT diagrams earlier. The following changes are made:

Chapter 1: **A** Chapter 3: **C** Designing: **E** Testing: **G** Chapter 6: **I**

Chapter 2: **B** Chapter 4: **D** Coding: **F** Chapter 5: **H** Chapter 7: **J**

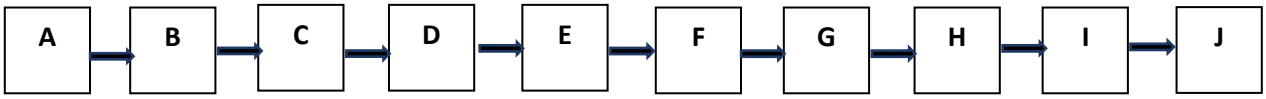


Figure 3.2: Activity

Activity	Precedence	Duration
A	-	2 weeks
B	A	2 weeks
C	B	4 weeks
D	C	3 weeks
E	D	3 weeks
F	E	10 weeks
G	F	2 weeks
H	G	1 weeks
I	H	1 week
J	I	1 week

Slack=0 for all the activities. Therefore, Critical Path = A-B-C-D-E-F-G-H-I-J.

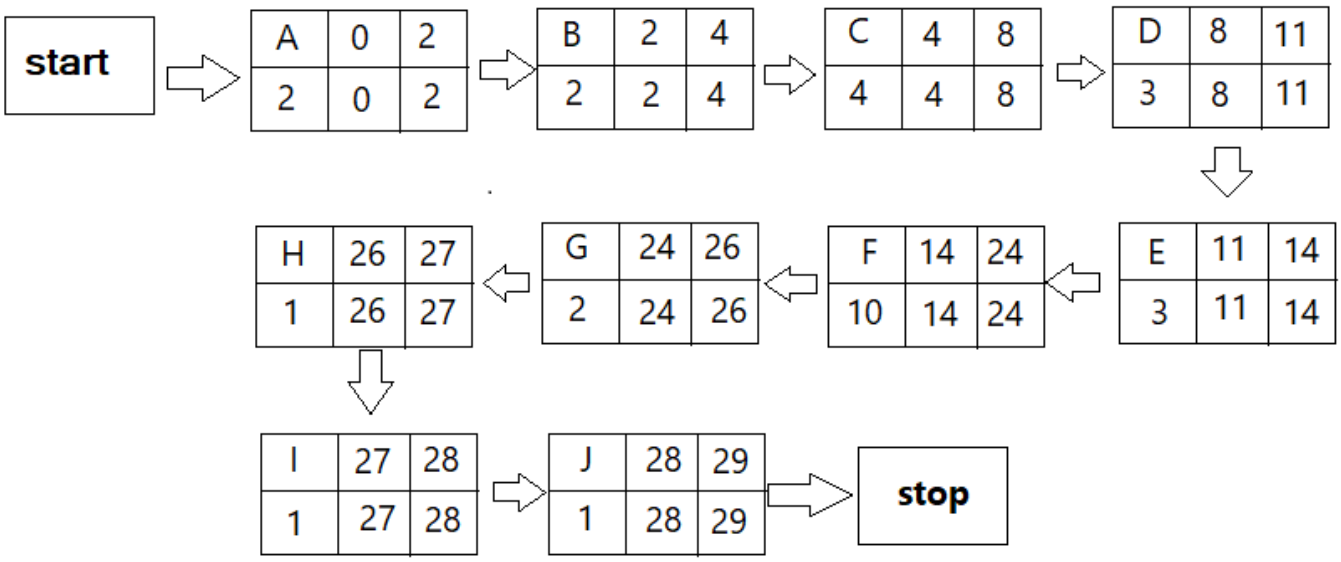


Figure 3.3: Critical Activity

Chapter 5: Implementation and Testing

5.1 Implementation Approaches

The project consists of several data visualisation and data handling web applications. Each of the web applications are created independently of the others. The individuality helps with adding components or features onto the project without affecting the entirety of the already implemented parts. Since this application is more oriented towards the end user, it contains features that help users navigate and use the different services easily in different tabs concurrently.

The visualisations help users to understand context as well as interpret different aspects of the data. The use of API's also makes the application relevant with respect to real time data. The different tickers and options help users decide and make informed decision.

5.2 Coding Details and Code Efficiency

Code:

Landing page for the project:

Landing.html

```
<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="UTF-8" />

    <meta http-equiv="X-UA-Compatible" content="IE=edge" />

    <meta name="viewport" content="width=device-width, initial-scale=1.0" />

    <link rel="stylesheet" href="landing.css" />

    <title>Welcome</title>

    <link rel="icon" href="pulse-icon.ico">


  </head>

  <body>
```

```
<div class="container">
  <nav>
    <ul>
      <li id="about">/About</li>
      <li id="contact">/Review</li>
    </ul>
  </nav>

  <main>
    <h1>Pulse-Infographics:<span class="cursor">|</span></h1>
    <h3>Project-list:</h3>
    <ul>
      <li>
        <a
          href="cryptograph/pulse.html"
          target="_blank"
          >Pulse-Cryptograph</a
        >
      </li>
      <li>
        <a href="http://localhost:8501/" target="_blank"
          >Pulse~Stock-Graph</a
        >
      </li>
      <li>
```

```
  <a href="http://localhost:8502/" target="_blank"
    >Pulse~Dna-analyser
  </a>
</li>
<li>
  <a href="http://127.0.0.1:5001/" target="_blank"
    >Pulse-Weather</a
  >
</li>
<li>
  <a href="http://127.0.0.1:5002/" target="_blank"
    >Pulse-News</a
  >
</li>
<li>
  <a href="http://localhost:3000/" target="_blank"
    >Pulse~Advisor</a
  >
</li>
<li>
  <a href="http://localhost:3001/" target="_blank"
    >Pulse~Expense-Tracker</a
  >
</li>
```

```
        <li>
          <a href="" target="_blank"
            >...</a
          >
        </li>
      </ul>
    </main>
  </div>

  <div class="hidden">
    <div id="about-content">
      <h2>about-Pulse:<span class="cursor">|</span></h2>
      <p>Pulse-Infographics is a project which focuses on <b>analysis</b>, <b>assessment</b> and <b>representation</b> of data</p>
      <p>
        Some of the languages that are used in this project are HTML, CSS, JS, Python, SQL etc..
      </p>
    </div>

    <div id="contact-content">
      <h2>Review-Us:<span class="cursor">|</span></h2>
      <p>You can Review our Projects</p>
      <p>Click on the below given link to jump to our Review page</p>
```

```
        <a href="http://127.0.0.1:5003/ ">Review-Page</a>
      </div>
    </div>

    <script src="winbox.js"></script>
    <script src="main.js"></script>
  </body>
</html>
```

landing.css

```
@import
url('https://fonts.googleapis.com/css2?family=Roboto+Mono:wght@300;400;700&display=s
wap');

li:hover{
  color:#00ff2a;
}

a:hover{
  color: #00ff2a;
}

* {
  box-sizing: border-box;
  padding: 0;
  margin: 0;
}
```

```
:root {
  --text-color: #fbff00;
}

body {
  font-family: 'Roboto Mono', sans-serif;
  color: #ccc;
  font-size: 18px;
  line-height: 1.6;
  background-image: url('1.jpg');
}

h1 {
  color: var(--text-color);
  font-size: 50px;
  letter-spacing: -5px;
  margin-bottom: 20px;
}

h2 {
  color: var(--text-color);
}

h3 {
  margin-bottom: 10px;
}

ul {
  list-style-type: none;
```

```
}

a {
  color: #ff6363;
  text-decoration: underline;
}

p {
  margin: 20px 0;
}

nav {
  width: 30%;
}

nav ul {
  display: flex;
  justify-content: space-around;
  align-items: center;
}

nav ul li {
  color: var(--text-color);
  cursor: pointer;
}

.container {
  max-width: 600px;
  margin: auto;
  height: 70vh;
```

```
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
}

.wb-body {
  background: #111;
  padding: 20px;
}

.hidden {
  display: none;
}

.cursor {
  font-weight: 700;
  animation: 1s blink step-end infinite;
}

@keyframes blink {
  from,
  to {
    color: transparent;
  }

  50% {
    color: var(--text-color);
  }
}
```

Main.js

```
const about = document.querySelector('#about')
const contact = document.querySelector('#contact')
const aboutContent = document.querySelector('#about-content')
const contactContent = document.querySelector('#contact-content')

about.addEventListener('click', () => {
  const aboutBox = new WinBox({
    title: 'ABOUT',
    // modal: true,
    width: '400px',
    height: '400px',
    top: 50,
    right: 50,
    bottom: 50,
    left: 50,
    mount: aboutContent,
    onfocus: function () {
      this.setBackground('#00aa00')
    },
    onblur: function () {
      this.setBackground('#777')
    },
  })
})

contact.addEventListener('click', () => {
  const contactBox = new WinBox({
    title: 'REVIEW',
```

```
    width: '400px',
    height: '400px',
    top: 150,
    right: 50,
    bottom: 50,
    left: 250,
    mount: contactContent,
    onfocus: function () {
      this.setBackground('#00aa00')
    },
    onblur: function () {
      this.setBackground('#777')
    },
  })
})
```

Winbox.js

```
;(function () {
  'use strict'
  var e,
    h = document.createElement('style')
  h.innerHTML =
    "@keyframes fade-in{0%{opacity:0}to{opacity:.85}}.winbox.modal:after,.winbox.modal:before{content:"}.winbox{position:fixed;left:0;top:0;background:#0050ff;box-shadow:0 14px 28px rgba(0,0,0,.25),0 10px 10px rgba(0,0,0,.22);transition:width .3s,height .3s,transform .3s;transition-timing-function:cubic-bezier(.3,1,.3,1);will-change:transform,width,height;contain:layout size;text-align:left;touch-action:none}.max,.no-shadow{box-shadow:none}.wb-header,.winbox iframe{position:absolute;width:100%}.wb-header{left:0;top:0;height:35px;color:#fff;overflow:hidden}.wb-body,.wb-n,.wb-s{position:absolute;left:0}.wb-n,.wb-s{height:10px}.wb-body{right:0;top:35px;bottom:0;overflow:auto;-webkit-overflow-scrolling:touch;overflow-scrolling:touch;will-change:contents;background:#fff;margin-top:0!important;contain:strict}.wb-title{font-family:Arial,sans-serif;font-size:14px;padding-left:10px;cursor:move;line-height:35px;white-space:nowrap;overflow:hidden;text-
```

overflow:ellipsis}.wb-n{top:-5px;right:0;cursor:n-resize}.wb-e{position:absolute;top:0;right:-5px;bottom:0;width:10px;cursor:w-resize}.wb-s,.wb-se,.wb-sw{bottom:-5px}.wb-s{right:0;cursor:n-resize}.wb-w,.winbox.modal:before{position:absolute;top:0;bottom:0}.wb-w{left:-5px;width:10px;cursor:w-resize}.wb-ne,.wb-nw,.wb-sw{width:15px;height:15px;position:absolute}.wb-nw{top:-5px;left:-5px;cursor:nw-resize}.wb-ne,.wb-sw{cursor:ne-resize}.wb-ne{top:-5px;right:-5px}.wb-sw{left:-5px}.wb-se{position:absolute;right:-5px;width:15px;height:15px;cursor:nw-resize}.wb-icon{float:right;height:35px;max-width:100%;text-align:center}.wb-icon *{display:inline-block;width:30px;height:100%;background-position:center;background-repeat:no-repeat;cursor:pointer;max-width:100%}.no-close .wb-close,.no-full .wb-full,.no-header .wb-header,.no-max .wb-max,.no-min .wb-min,.no-resize .wb-body~div,.winbox.min .wb-body>*,.winbox.min .wb-full,.winbox.min .wb-min,.winbox.modal .wb-full,.winbox.modal .wb-max,.winbox.modal .wb-min{display:none}.wb-min{background-image:url();background-size:14px auto;background-position:center bottom 11px}.wb-max{background-image:url();background-size:17px auto}.wb-close{background-image:url();background-size:15px auto}.wb-full{background-image:url();background-size:16px auto}.winbox.max .wb-body~div,.winbox.max .wb-title,.winbox.min .wb-body~div,.winbox.modal .wb-body~div,.winbox.modal .wb-title{pointer-events:none}.winbox.min .wb-title{cursor:default}.max .wb-body{margin:0!important}.winbox

iframe{height:100%;border:0}.winbox.modal:before{left:0;right:0;background:inherit;border-radius:inherit}.winbox.modal:after{position:absolute;top:-100vh;left:-100vw;right:-100vw;bottom:-100vh;background:#0d1117;animation:fade-in .2s ease-out forwards;z-index:-1}.no-animation{transition:none}.no-header .wb-body{top:0}.no-move:not(.min) .wb-title{pointer-events:none}"

var k = document.getElementsByTagName('head')[0]

k.firstChild ? k.insertBefore(h, k.firstChild) : k.appendChild(h)

var q = document.createElement('div')

q.innerHTML =

' '

function r(a, b, c, g) {

a.addEventListener(b, c, g || !1 === g ? g : !0)

}

function t(a) {

a.stopPropagation()

a.cancelable && a.preventDefault()

}

function w(a, b, c) {

c = '' + c

a['_s_' + b] !== c && (a.style.setProperty(b, c), (a['_s_' + b] = c))

}

var x = document.documentElement,

y = [],

A = 0,

B,

C,

F,

G,

K,

```
  L,
  M
function O(a, b) {
  if (!(this instanceof O)) return new O(a)
  B || Q()
  this.g = q.cloneNode(!0)
  this.body = this.g.getElementsByClassName('wb-body')[0]
  var c, g
  if (a) {
    if (b) {
      var f = a
      a = b
    }
    if ('string' === typeof a) f = a
    else {
      if ((g = a.modal)) var u = (c = 'center')
      var z = a.id
      var H = a.root
      f = f || a.title
      var D = a.mount
      var d = a.html
      var I = a.url
      var l = a.width
      var m = a.height
      u = a.x || u
      c = a.y || c
      var E = a.max
      var n = a.top
      var p = a.left
      var v = a.bottom
```

```
    var J = a.right
    B = a.index || B
    var W = a.onclose
    var X = a.onfocus
    var Y = a.onblur
    var Z = a.onmove
    var aa = a.onresize
    b = a.background
    var P = a.border
    var N = a['class']
    b && this.setBackground(b)
    P && w(this.body, 'margin', P + (isNaN(P) ? '' : 'px'))
  }
}
this.setTitle(f || '')
a = L
f = M
n = n ? R(n, f) : 0
v = v ? R(v, f) : 0
p = p ? R(p, a) : 0
J = J ? R(J, a) : 0
a -= p + J
f -= n + v
l = l ? R(l, a) : (a / 2) | 0
m = m ? R(m, f) : (f / 2) | 0
u = u ? R(u, a, l) : p
c = c ? R(c, f, m) : n
B = B || 10
this.g.id = this.id = z || 'winbox-' + ++A
this.g.className =
```

```
    'winbox' +
    (N ? ' ' + ('string' === typeof N ? N : N.join(' ')) : '') +
    (g ? ' modal' : '')
  this.x = u
  this.y = c
  this.width = l
  this.height = m
  this.top = n
  this.right = J
  this.bottom = v
  this.left = p
  this.max = this.min = !1
  this.j = W
  this.l = X
  this.i = Y
  this.o = Z
  this.m = aa
  E ? this.maximize() : this.move().resize()
  this.focus()
  D ? this.mount(D) : d ? (this.body.innerHTML = d) : I && this.setUrl(I)
  ba(this)
  ;(H || document.body).appendChild(this.g)
}
O['new'] = function (a) {
  return new O(a)
}
function R(a, b, c) {
  'string' === typeof a &&
    ('center' === a
      ? (a = ((b - c) / 2) | 0)
```

```
      : 'right' === a || 'bottom' === a
      ? (a = b - c)
      : ((c = parseFloat(a)),
        (a =
          '%' === (" + c !== a && a.substring((" + c).length))
            ? ((b / 100) * c) | 0
            : c)))
  return a
}
function Q() {
  var a = document.body
  a[(G = 'requestFullscreen')] ||
    a[(G = 'msRequestFullscreen')] ||
    a[(G = 'webkitRequestFullscreen')] ||
    a[(G = 'mozRequestFullscreen')] ||
    (G = '')
  K =
    G &&
    G.replace('request', 'exit')
      .replace('mozRequest', 'mozCancel')
      .replace('Request', 'Exit')
  r(window, 'resize', function () {
    L = x.clientWidth
    M = x.clientHeight
    S()
  })
  L = x.clientWidth
  M = x.clientHeight
}
function ba(a) {
```

```
T(a, 'title')
T(a, 'n')
T(a, 's')
T(a, 'w')
T(a, 'e')
T(a, 'nw')
T(a, 'ne')
T(a, 'se')
T(a, 'sw')
r(a.g.getElementsByClassName('wb-min')[0], 'click', function (b) {
  t(b)
  a.minimize()
})
r(a.g.getElementsByClassName('wb-max')[0], 'click', function (b) {
  t(b)
  a.focus().maximize()
})
G
  ? r(a.g.getElementsByClassName('wb-full')[0], 'click', function (b) {
      t(b)
      a.focus().fullscreen()
    })
  : a.addClass('no-full')
r(a.g.getElementsByClassName('wb-close')[0], 'click', function (b) {
  t(b)
  a.close()
  a = null
})
r(
  a.g,
```

```
    'click',
    function () {
      a.focus()
    },
    !1
  )
}
function U(a) {
  y.splice(y.indexOf(a), 1)
  S()
  a.removeClass('min')
  a.min = !1
  a.g.title = ''
}
function S() {
  for (var a = y.length, b = 0, c, g; b < a; b++)
    (c = y[b]),
      (g = Math.min((L - 2 * c.left) / a, 250)),
      c
        .resize((g + 1) | 0, 35, !0)
        .move((c.left + b * g) | 0, M - c.bottom - 35, !0)
}
function T(a, b) {
  function c(d) {
    t(d)
    a.min
      ? (U(a), a.resize().move().focus())
      : (w(a.g, 'transition', 'none'),
        (z = d.touches) && (z = z[0])
          ? ((d = z), r(window, 'touchmove', g), r(window, 'touchend', f))
```

```
        : (r(window, 'mousemove', g), r(window, 'mouseup', f)),
      (H = d.pageX),
      (D = d.pageY),
      a.focus())
}
function g(d) {
  t(d)
  z && (d = d.touches[0])
  var I = d.pageX
  d = d.pageY
  var l = I - H,
    m = d - D,
    E
  if ('title' === b) {
    a.x += l
    a.y += m
    var n = (E = 1)
  } else {
    if ('e' === b || 'se' === b || 'ne' === b) {
      a.width += l
      var p = 1
    } else if ('w' === b || 'sw' === b || 'nw' === b)
      (a.x += l), (a.width -= l), (n = p = 1)
    if ('s' === b || 'se' === b || 'sw' === b) {
      a.height += m
      var v = 1
    } else if ('n' === b || 'ne' === b || 'nw' === b)
      (a.y += m), (a.height -= m), (E = v = 1)
  }
  if (p || v)
```

```
    p && (a.width = Math.max(Math.min(a.width, L - a.x - a.right), 150)),
      v &&
        (a.height = Math.max(Math.min(a.height, M - a.y - a.bottom), 35)),
      a.resize()
  if (n || E)
    n && (a.x = Math.max(Math.min(a.x, L - a.width - a.right), a.left)),
      E && (a.y = Math.max(Math.min(a.y, M - a.height - a.bottom), a.top)),
      a.move()
  H = I
  D = d
 }
 function f(d) {
  t(d)
  w(a.g, 'transition', '')
  z
   ? (window.removeEventListener('touchmove', g, !0),
     window.removeEventListener('touchend', f, !0))
   : (window.removeEventListener('mousemove', g, !0),
     window.removeEventListener('mouseup', f, !0))
 }
 var u = a.g.getElementsByClassName('wb-' + b)[0],
  z,
  H,
  D
 r(u, 'mousedown', c)
 r(u, 'touchstart', c, { passive: !1 })
}
e = O.prototype
e.mount = function (a) {
 this.unmount()
```

```
  a.h || (a.h = a.parentNode)
  this.body.textContent = ''
  this.body.appendChild(a)
  return this
}
e.unmount = function (a) {
  var b = this.body.firstChild
  if (b) {
    var c = a || b.h
    c && c.appendChild(b)
    b.h = a
  }
  return this
}
e.setTitle = function (a) {
  a = this.title = a
  this.g.getElementsByClassName('wb-title')[0].firstChild.nodeValue = a
  return this
}
e.setBackground = function (a) {
  w(this.g, 'background', a)
  return this
}
e.setUrl = function (a) {
  this.body.innerHTML = '<iframe src="' + a + '"></iframe>'
  return this
}
e.focus = function () {
  F !== this &&
    (w(this.g, 'z-index', B++),
```

```
    this.addClass('focus'),
    F && (F.removeClass('focus'), F.i && F.i()),
    (F = this),
    this.l && this.l())
  return this
}
e.hide = function () {
  return this.addClass('hide')
}
e.show = function () {
  return this.removeClass('hide')
}
e.minimize = function (a) {
  C && V()
  !a && this.min
    ? (U(this), this.resize().move())
    : !1 === a ||
      this.min ||
      (y.push(this),
      S(),
      (this.g.title = this.title),
      this.addClass('min'),
      (this.min = !0))
  this.max && (this.removeClass('max'), (this.max = !1))
  return this
}
e.maximize = function (a) {
  if ('undefined' === typeof a || a !== this.max)
    this.min && U(this),
      (this.max = !this.max)
```

```
        ? this.addClass('max')
            .resize(
              L - this.left - this.right,
              M - this.top - this.bottom,
              !0
            )
            .move(this.left, this.top, !0)
        : this.resize().move().removeClass('max')
  return this
}
e.fullscreen = function (a) {
  if ('undefined' === typeof a || a !== C)
    this.min && (this.resize().move(), U(this)),
      (C && V()) || (this.body[G](), (C = !0))
  return this
}
function V() {
  C = !1
  if (
    document.fullscreen ||
    document.fullscreenElement ||
    document.webkitFullscreenElement ||
    document.mozFullScreenElement
  )
    return document[K](), !0
}
e.close = function () {
  this.min && U(this)
  this.j && this.j()
  this.unmount()
```

```
  this.g.parentNode.removeChild(this.g)
  F === this && (F = null)
}
e.move = function (a, b, c) {
  'undefined' === typeof a
    ? ((a = this.x), (b = this.y))
    : c ||
      ((this.x = a ? (a = R(a, L - this.left - this.right, this.width)) : 0),
      (this.y = b ? (b = R(b, M - this.top - this.bottom, this.height)) : 0))
  w(this.g, 'transform', 'translate(' + a + 'px,' + b + 'px)')
  this.o && this.o(a, b)
  return this
}
e.resize = function (a, b, c) {
  'undefined' === typeof a
    ? ((a = this.width), (b = this.height))
    : c ||
      ((this.width = a ? (a = R(a, L - this.left - this.right)) : 0),
      (this.height = b ? (b = R(b, M - this.top - this.bottom)) : 0))
  w(this.g, 'width', a + 'px')
  w(this.g, 'height', b + 'px')
  this.m && this.m(a, b)
  return this
}
e.addClass = function (a) {
  this.g.classList.add(a)
  return this
}
e.removeClass = function (a) {
  this.g.classList.remove(a)
```

```
    return this
  }
  window.WinBox = O
}.call(this))
…….
```

Cryptograph:

Pulse.html

```
<!DOCTYPE html>
<html>

<head>
  <link rel="icon" href="images for web//pulse-icon.ico">
    <title>Pulse-cryptograph</title>
    <meta name="description" content="An example of a simple home page screen.">
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

    <link  href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.0-beta2/dist/css/bootstrap.min.css"
rel="stylesheet"
        integrity="sha384-
BmbxuPwQa2lc/FVzBcNJ7UAyJxM6wuqIj61tLrc4wSX0szH/Ev+nYRRuWlolflfl"
crossorigin="anonymous">
    <link rel="stylesheet" href="pulse.css">

    <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.0/jquery.min.js"></script>
    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"
        integrity="sha384-
B4gt1jrGC7Jh4AgTPSdUtOBvfO8shuf57BaghqFfPlYxofvL8/KUEfYiJOMMV+rV"
        crossorigin="anonymous"></script>
    <script src="https://cdn.jsdelivr.net/npm/chart.js@2.9.4/dist/Chart.min.js"
```

```
    integrity="sha384-
zNy6FEbO50N+Cg5wap8IKA4M/ZnLJgzc6w2NqACZaK0u0FXfOWRRJOnQtpZun8ha"
    crossorigin="anonymous"></script>
  <script src="script.js"></script>
</head>

<body>
  <div class="wrapper">
    <nav class="navbar navbar-dark mb-5">

      <a class="navbar-brand" href="#">
        Pulse-Crypto-Graph
      </a>

      <a class="navbar-brand2" href="http://cors-
anywhere.herokuapp.com/corsdemo">Server</a>
    </nav>
    <div class="container">
      <div class="justify-content-center">
        <div class="justify-content-center">
          <!-- coin svg goes below -->
          <img src="#" width="30" height="30" class="d-inline-block align-top" alt="">
          <!-- coin name appended below -->
          <h1 class="display-4 d-inline-block" id="currency">Bitcoin</h1>
        </div>
        <div>
          <select id="cryptoList" class="custom-select custom-select-lg mb-3">
            <option value="0">Bitcoin</option>
            <option value="1">Ethereum</option>
            <option value="2">Litecoin</option>
            <option value="3">Tether</option>
```

```
                <option value="4">Binancecoin</option>
                <option value="5">Cardano</option>
                <option value="6">Dogecoin</option>
                <option value="7">XRP</option>
                <option value="8">USDC</option>
                <option value="9">HEX</option>
            </select>
        </div>
        <hr class="my-4">
    </div>
    <div>
        <div class="d-flex justify-content-between align-items-center">
            <div>
                <!-- dynamic current price -->
                <p class="changeFont d-inline-block mb-0">$</p>
                <p id="currentPrice" class="changeFont d-inline-block mb-0"> </p>
                <p id="percentChange" class="d-inline-block align-top"></p>
            </div>
            <div class="btn-group d-inline-block align-center" id="timeframeButtons">
                <!--Timeframe radio buttons-->
                <input type="radio" name="options" value="24h" class="btn-check" id="24h">
                <label class="btn btn-success" for="24h">24h</label>
                <input type="radio" name="options" value="7d" class="btn-check" id="7d">
                <label class="btn btn-success" for="7d">7d</label>
                <input type="radio" name="options" value="30d" class="btn-check" id="30d">
                <label class="btn btn-success" for="30d">30d</label>
                <input type="radio" name="options" value="1y" class="btn-check" id="1y">
                <label class="btn btn-success" for="1y">1y</label>
            </div>
```

```
            </div>
            <!-- Graph within canvas element -->
            <canvas class="my-4 w-100" id="myChart" width="900" height="380"></canvas>
          </div>
        </div>
        <div id="infoContainer" class="container mt-4">
          <!-- Crypto information -->
        </div>
    </div>
</body>

</html>
```

Pulse.css

```
.navbar-brand2{
  padding-right:15px;
  color:#fcd204;
}
.navbar-dark .navbar-brand {
    color: #fcd204;
}
a{
   color: #9c02c5;
   text-decoration: underline;
}

.container > div div{
    display: flex;
    justify-content: center;
}
```

```
.container {
    border-radius: 1em;
    box-shadow: 5px 8px 8px 8px #888888;
    padding: 15px;
    background-color: rgb(135, 151, 173);
}
#infoContainer {
    min-height: 200px;
}
.navbar-brand {
    margin-left: 15px;
}
.btn-group label:hover{
    background-color: rgb(117, 25, 89);
}
.btn-group label {
    background-color:rgb(39, 138, 177);
    border: none;
    box-shadow: 1px 3px 2px #888888;
}
.btn-group input[type="radio"]:checked + label {
    background-color: rgb(0, 74, 104);
}
.navbar {
    background-color: rgb(50, 97, 117);



}
.wrapper {
    background-color: rgb(197, 223, 238);
```

```
    padding-bottom: 24px;
    min-width: 350px;
    min-height: 1000px;
}
.changeFont {
    font-size: 2.5rem;
}


/* Media Queries */
@media only screen and (max-width:768px) {

}
@media only screen and (max-width: 500px) {
    .changeFont {
        font-size: 2rem;
    }
    .btn-group label {
        padding: .3rem .5rem;
    }
}
@media only screen and (max-width: 400px) {
    .changeFont {
        font-size: 1.5rem;
    }
    .btn-group label {
        padding: .25rem .3rem;
    }
}
```

Script.js

```
$(document).ready(function () {
  // Define currency codes
  'use strict';
  const bitcoin = "Qwsogvtv82FCd"
  const ethereum = "razxDUgYGNAdQ"
  const litecoin = "D7B1x_ks7WhV5"
  const tetherusd = "HIVsRcGKkPFtW"
  const binancecoin ="WcwrkfNI4FUAe"
  const cardano = "qzawljRxB5bYu"
  const dogecoin ="a91GCGd_u96cF"
  const xrp = "-l8Mn2pVlRs-p"
  const usdc = "aKzUVe4Hh_CON"
  const hex ="9K7m6ufraZ6gh"
  const filecoin ="ymQub4fuB"
  var uuid = bitcoin;
  // Define timeframe code
  var time = "24h";
  // Define empty chart globally, then call getCoinData to populate default chart (btc)
  let myChart;
  getCoinData(uuid, time);
  // ######
  // Functions
  function getCoinData(currency, timeframe) {
     console.log("getCoinData Success");
     var baseUrl = "https://api.coinranking.com/v2/coin/" + currency + "?timePeriod=" + timeframe;
     var proxyUrl = "https://cors-anywhere.herokuapp.com/";
     var apiKey = "coinranking97dd0079db2ec447bc4af83adc4c2d94efc925f76570c087"
     $(`#${timeframe}`).prop("checked", true).css("border", "4px solid green");
     fetch(`${proxyUrl}${baseUrl}`, {
```

```
        method: 'GET',
        headers: {
            'Content-Type': 'application/json',
            'x-access-token': `${apiKey}`,
            'Access-Control-Allow-Origin': "*"
        }
    })
        .then((response) => {
            if (response.ok) {
                response.json().then((json) => {
                    console.log("getCoinDataResponse Success");
                    handlerFunction(json.data);
                })
            }
        })
}
function handlerFunction(data) {
    console.log(data);
    if (myChart) {
        console.log("destroying old chart.");
        myChart.destroy();
        $("#currentPrice").empty();
        $("img").attr("src", "#");
        $("#percentChange").empty();
        $("#infoContainer").empty();
    }
    let coinsData = data.coin;
    // Add the selected currency's name and icon, along with current price.
    var price = Math.round((parseFloat(coinsData.price) + Number.EPSILON) * 100) / 100;
    $("#currency").text(coinsData.name);
```

```
    $("img").attr("src", coinsData.iconUrl);
    $("#currentPrice").text(price);

    // ######
    // Add percent change over specified time period.
    // ######

    var change = Math.round((parseFloat(coinsData.change) + Number.EPSILON) * 100) / 100;
    $("#percentChange").text(change)
    if (change > 0) { //Checks if the change is positive or negative, then assigns color to the text
        $("#percentChange").css("color", "green").prepend("+").append("%");
    }
    else {
        $("#percentChange").css("color", "red").append("%");
    }
    var description = `<p>${coinsData.description}</p>`;
    $("#infoContainer").append(description);

    // ######
    // Graph
    // ######

    var ctx = document.getElementById('myChart')
    myChart = new Chart(ctx, {
        type: 'line',
        data: {
            labels: ['', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', '', ''],
            datasets: [{
                data: coinsData.sparkline,
```

```
            label: coinsData.symbol,
            lineTension: 0,
            backgroundColor: 'transparent',
            borderColor: coinsData.color,
            borderWidth: 4,
            pointBackgroundColor: coinsData.color,
        },
        ]
    },
    options: {
        scales: {
            yAxes: [{
                ticks: {
                    beginAtZero: false,
                    callback: function (value, index, values) {
                        return '$' + value;
                    }
                }
            }]
        },
        legend: {
            display: true,
        }
    }
  })
}
// ######
// Currency type selector
$('#cryptoList').change(function () {
  var selectedValueCurrency = parseInt($(this).val());
```

```
//Depends on Value 0-2 respective function gets called.
switch (selectedValueCurrency) {
    case 0:
        console.log("radio BTC success");
        uuid = bitcoin;
        getCoinData(uuid, time);
        break;
    case 1:
        console.log("radio ETH success");
        uuid = ethereum;
        getCoinData(uuid, time);
        break;
    case 2:
        console.log("radio LTC success");
        uuid = litecoin;
        getCoinData(uuid, time);

    case 3:
         console.log("radio Teather success");
         uuid = tetherusd;
         getCoinData(uuid, time);
         break;

    case 4:
         console.log("radio binancecoin success");
         uuid = binancecoin;
         getCoinData(uuid, time);
         break;
```

```
 case 5:
      console.log("radio cardano success");
      uuid = cardano;
      getCoinData(uuid, time);
      break;

 case 6:
      console.log("radio dogecoin success");
      uuid = dogecoin;
      getCoinData(uuid, time);
      break;

case 7:
      console.log("radio xrp success");
      uuid = xrp;
      getCoinData(uuid, time);
      break;

case 8:
      console.log("radio usdc success");
      uuid = usdc;
      getCoinData(uuid, time);
      break;

case 9:
      console.log("radio hex success");
      uuid = hex;
      getCoinData(uuid, time);
      break;
```

```
        case 10:
            console.log("radio filecoin success");
            uuid = filecoin;
            getCoinData(uuid, time);
            break;
    }
  });
  // ######
  // Timeframe selector
  $('input:radio[name=options]').on("click", function () {
    if (time != $("input[name=options]:checked").val()) {
      time = $("input[name=options]:checked").val();
      console.log(time);
      getCoinData(uuid, time);
    }
  })
})
```

Stock graph:

Main.py

```
import streamlit as st
from datetime import date
from PIL import Image
import yfinance as yf
from plotly import graph_objs as go
```

```
START = "2018-01-01"
TODAY = date.today().strftime("%Y-%m-%d")

st.title('Pulse~Stock-graph')
image = Image.open('2.png')
st.image(image, width=200)
st.write("""
This app shows the stockmarket opening and closing prize using the yfinance API!
***
""")

stocks = ('GOOGL', 'AAPL', 'MSFT', 'SBI', 'RELI','ICBK','IDEA.NS','FB','IBM','YESBANK.NS')
selected_stock = st.selectbox('Select dataset for prediction', stocks)

@st.cache
def load_data(ticker):
    data = yf.download(ticker, START, TODAY)
    data.reset_index(inplace=True)
    return data


data_load_state = st.text('Loading data...')
data = load_data(selected_stock)
data_load_state.text('Loading data... done!')

st.subheader('Raw data')
st.write(data.tail())


# Plot raw data
```

```
def plot_raw_data():
    fig = go.Figure()
    fig.add_trace(go.Scatter(x=data['Date'], y=data['Open'], name="stock_open"))
    fig.add_trace(go.Scatter(x=data['Date'], y=data['Close'], name="stock_close"))
    fig.layout.update(title_text='Time Series data with Rangeslider', xaxis_rangeslider_visible=True)
    st.plotly_chart(fig, width=2500, height=2000)



plot_raw_data()
```

Dna analyser:

Pulse~DNA-analyser.py

```
import pandas as pd
import streamlit as st
import altair as alt
from PIL import Image

im = Image.open("pulse-icon.ico")
st.set_page_config(
    page_title="Pulse~DNA-analyser",
    page_icon=im,
    layout="wide",
)

image = Image.open('2.png')

st.image(image, width=200)

st.write("""
This app counts the nucleotide composition of the query DNA!
```

```
\nFor the DNA sequence refer https://www.bioinformatics.org/sms2/random_dna.html
***
""")

#st.sidebar.header('Enter DNA sequence')
st.header('Enter DNA sequence')

sequence_input = ">DNA Query :\nCCCCTCTGAGGTCAAGGAACACAAGATGGTTTTGGAAATGC\nTGAACCCGATACATTATAACATCACCAGCATCGTGCCTGAAGCCATGCCTGCTGCCACCATGCCAGTCCT"

#sequence = st.sidebar.text_area("Sequence input", sequence_input, height=250)
sequence = st.text_area("Sequence input", sequence_input, height=250)
sequence =sequence.upper()
sequence = sequence.splitlines()
sequence = sequence[1:] # Skips the sequence name (first line)
sequence = ''.join(sequence) # Concatenates list to string

st.write("""
***
""")

## Prints the input DNA sequence
st.header('INPUT (DNA Query)')
sequence

## DNA nucleotide count
st.header('OUTPUT (DNA Nucleotide Count)')

### 1. Print dictionary
```

```
st.subheader('Dictionary')
def DNA_nucleotide_count(seq):
  d = dict([
          ('A',seq.count('A')),
          ('T',seq.count('T')),
          ('G',seq.count('G')),
          ('C',seq.count('C'))
          ])
  return d

X = DNA_nucleotide_count(sequence)

#X_label = list(X)
#X_values = list(X.values())

X

### 2. Print text
st.subheader('Print text')
st.write('There are  ' + str(X['A']) + ' adenine (A)')
st.write('There are  ' + str(X['T']) + ' thymine (T)')
st.write('There are  ' + str(X['G']) + ' guanine (G)')
st.write('There are  ' + str(X['C']) + ' cytosine (C)')

### 3. Display DataFrame
st.subheader('DataFrame')
df = pd.DataFrame.from_dict(X, orient='index')
df = df.rename({0: 'count'}, axis='columns')
df.reset_index(inplace=True)
df = df.rename(columns = {'index':'nucleotide'})
```

```
st.write(df)

### 4. Display Bar Chart using Altair
st.subheader('Bar chart')
p = alt.Chart(df).mark_bar().encode(
    x='nucleotide',
    y='count'
)
p = p.properties(
    width=alt.Step(80)  # controls width of bar.
)
st.write(p)
```

weather:

app.py

```
import requests
from flask import Flask, render_template, request
from flask_sqlalchemy import SQLAlchemy



app = Flask(__name__)
app.config['DEBUG'] = True
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///weather.db'

db = SQLAlchemy(app)



class City(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    name = db.Column(db.String(50), nullable=False)
```

```
@app.route('/', methods=['GET', 'POST'])
def index():
    if request.method == 'POST':
        new_city = request.form.get('city')

        if new_city:
            new_city_obj = City(name=new_city)

            db.session.add(new_city_obj)
            db.session.commit()

    cities = City.query.all()

    url = 'http://api.openweathermap.org/data/2.5/weather?q={}&units=imperial&appid=271d1234d3f497eed5b1d80a07b3fcd1'

    weather_data = []

    for city in cities:
        r = requests.get(url.format(city.name)).json()

        weather = {
            'city': city.name,
            'temperature': r['main']['temp'],
            'description': r['weather'][0]['description'],
            'icon': r['weather'][0]['icon'],
        }
```

```
        weather_data.append(weather)

    return render_template('weather.html', weather_data=weather_data)
```

weather.html

```
<!DOCTYPE html>
<html lang="en">

<head>

    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="icon" href="../static/pulse-icon.ico">
    <title>Pusle-Weather</title>

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/bulma/0.6.2/css/bulma.css" />
</head>

<body style="background-color:#aedeab;">
    <section class="hero is-primary", style="background-color:#010a01;">
        <div class="hero-body">
            <div class="container">
                <h1 class="title">
                    Pulse-weather <a href="http://localhost/landing.html", style="padding-left:1000px;">home</a>
                </h1>

            </div>
        </div>
```

```
</section>
<section class="section">
  <div class="container">
    <div class="columns">
      <div class="column is-offset-4 is-4">
        <form method="POST">
          <div class="field has-addons">
            <div class="control is-expanded">
              <input class="input" name="city" type="text" placeholder="City Name">
            </div>
            <div class="control">
              <button class="button is-info">
                Add City
              </button>
            </div>
          </div>
        </form>
      </div>
    </div>
  </div>
</section>
<section class="section">
  <div class="container">
    <div class="columns">
      <div class="column is-offset-4 is-4">
        {% for weather in weather_data %}
        <div class="box">
          <article class="media">
            <div class="media-left">
              <figure class="image is-50x50">
```

```
                        <img src="http://openweathermap.org/img/w/{{ weather.icon }}.png"
alt="Image">
                    </figure>
                </div>
                <div class="media-content">
                    <div class="content">
                        <p>
                            <span class="title">{{ weather.city }}</span>
                            <br>
                            <span class="subtitle">{{ weather.temperature }}° F</span>
                            <br> {{ weather.description }}
                        </p>
                    </div>
                </div>
            </article>
        </div>
        {% endfor %}
    </div>
  </div>
 </div>
 </section>
 <footer class="footer">
 </footer>
</body>

</html>
```

News app:

app.py

```
from flask import Flask, render_template
from newsapi import NewsApiClient
```

```
app = Flask(__name__)



@app.route('/')
def Index():
    newsapi = NewsApiClient(api_key="b0f75ce660c0466a9a98c2478f8abb62")
    topheadlines = newsapi.get_top_headlines(sources="business-insider")
    s= 'the-times-of-india',  'bbc-news', 'abc-news','al-jazeera-english', 'ars-technica', 'associated-press', 'bleacher-report', 'bloomberg' ,
    articles = topheadlines['articles']


    desc = []
    news = []
    img = []
    links = []


    for i in range(len(articles)):
        myarticles = articles[i]


        news.append(myarticles['title'])
        desc.append(myarticles['description'])
        img.append(myarticles['urlToImage'])
        links.append(myarticles['url'])


    mylist = zip(news, desc, img, links)


    return render_template('index.html', context=mylist)
```

```
if __name__ == "__main__":
    app.run(debug=True)
```

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <link rel="icon" href="../static/pulse-icon.ico">

    <title>Pulse-News </title>

    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">
</head>
<body style="background-color:#bacbe6;">

<div class="jumbotron" style="background-color:#4d6a99;">

    <h1 style="color:#f9f7fa">

        <img src="../static/2.png" alt="" width="200px" height="100px">
   NEWS

    </h1>

</div>
```

```
<div class="container">

    {% for new, des, i, ln in context %}

    <img src="{{i}}" alt="">
    <h3>News: </h3> {{new}}
    <h4>Description: </h4> {{des}}
    <h4>Link: <a href="{{ln}}">click here</a></h4>

    {% endfor %}



</div>



</body>
</html>
```

Pulse advisor:

Api..

Index.js

```
import axios from 'axios';



export const getPlacesData =  async(type, sw, ne) => {
  try{
     const {data: { data }} = await axios.get(`https://travel-advisor.p.rapidapi.com/${type}/list-in-boundary`, {
       params: {
         bl_latitude: sw.lat,
         tr_latitude: ne.lat,
```

```
        bl_longitude: sw.lng,
        tr_longitude: ne.lng,


      },
      headers: {
        'x-rapidapi-host': 'travel-advisor.p.rapidapi.com',
        'x-rapidapi-key': '5d83b37ab4mshb19dd7cab78f99ap11603bjsn3800472eb862'
      }
      // 'KJwZZIJSFimshuivMSVGaiYzkRomp15f2vKjsnK4bKzuUzVLzA' mine-> '3e8cd46085mshc9704f3d361f170p12c181jsn101bb1cf8161' welper-> '5d83b37ab4mshb19dd7cab78f99ap11603bjsn3800472eb862'
    });
    return data;
  }catch(error){
    console.log(error);
  }
}
```

Components:

1. Header:

   Header.jsx

```
import react, {useState} from 'react';
import { Autocomplete } from '@react-google-maps/api';
import { AppBar, Toolbar, Typography, InputBase, Box, Link  } from '@material-ui/core';
import SearchIcon from '@material-ui/icons/Search';
import useStyles from './styles.js';
const Header = ({onLoading, onPlaceChanged}) => {
    const classes = useStyles();


    return(
     <AppBar position='static'>
        <Toolbar className={classes.toolbar}>
           <Typography variant='h5' className={classes.title}>
              Pulse~Advisor
```

```
            </Typography>
            <Box display='flex'>
            <Typography variant='h6' className={classes.title}>
              <a href="http://localhost/landing.html" className={classes.a}> Where to?
</a>
            </Typography>
            {/* <Autocomplete onLoad={onLoading} onChange={onPlaceChanged}>
            <div className={classes.search}>
             <div className={classes.searchIcon}>
               <SearchIcon />
             </div>
             <InputBase placeholder="Search…" classes={{ root: classes.inputRoot,
input: classes.inputInput }}/>
            </div>
          </Autocomplete> */}
            </Box>
          </Toolbar>

      </AppBar>
    );
}

export default Header;
```

2. List

List.jsx

```
import React, { useState, useEffect, createRef } from 'react';

import { CircularProgress, Grid, Typography, InputLabel, MenuItem, FormControl, Select
} from '@material-ui/core';


import PlaceDetails from '../PlaceDetails/PlaceDetails';

import useStyles from './styles.js';


const List = ({ places, type, setType, rating, setRating, childClicked, isLoading }) => {


  const classes = useStyles();

  const[elRefs, setElRefs] = useState([]);
```

```
useEffect(()=>{

    setElRefs((refs) => Array({places}.length).fill().map((_, i) => refs[i] || createRef()));
},[places]);
return(
 <div className={classes.container}>
    <Typography variant='h4'>Explore Restaurants, Hotels and Places around you</Typography>
     {isLoading ? (
    <div className={classes.loading}>
     <CircularProgress size="5rem" />
    </div>
  ) : (
    <>
     <FormControl className={classes.formControl}>
        <InputLabel>Rating</InputLabel>
        <Select value={rating} onChange={(e)=> setRating(e.target.value)}>
           <MenuItem value={0}>ALL</MenuItem>
           <MenuItem value={3}>Above 3.0</MenuItem>
           <MenuItem value={4}>Above 4.0</MenuItem>
           <MenuItem value={4.5}>Above 4.5</MenuItem>

        </Select>
        </FormControl>

        <FormControl className={classes.formControl}>
        <InputLabel>Type</InputLabel>
        <Select value={type} onChange={(e)=> setType(e.target.value)}>
           <MenuItem value="restaurants">Restaurants</MenuItem>
           <MenuItem value="hotels">Hotels</MenuItem>
           <MenuItem value="attractions">Attractions</MenuItem>
```

```
          </Select>
          </FormControl>
          <Grid container spacing={3} className={classes.list}>
          {places?.map((place, i) => (
            <Grid ref={elRefs[i]} key={i} item xs={12}>
              <PlaceDetails selected={Number(childClicked) === i} refProp={elRefs[i]} place={place} />
            </Grid>
          ))}
        </Grid>
        </>
    )}
    </div>
  );
}

export default List;
```

3. Map:

Map.jsx

```
import React from 'react';
import GoogleMapReact from 'google-map-react';
import { Paper, Typography, useMediaQuery } from '@material-ui/core';
import LocationOnOutlinedIcon from '@material-ui/icons/LocationOnOutlined';
import Rating from '@material-ui/lab/Rating';


import useStyles from './styles.js';

const Map = ({setCoordinates, setBounds, coordinates, places, setChildClicked}) => {
  const classes = useStyles();
```

```
    const isDesktop = useMediaQuery('(min-width:600px)');
    return(
     <div className={classes.mapContainer}>
        <GoogleMapReact bootstrapURLKeys={{key:'AIzaSyDcsfrMw6eYvxgmb-I93AR8T3B4UkHzbwk' }}
          defaultCenter={coordinates}
          center={coordinates}
          defaultZoom={14}
          margin={[50, 50, 50, 50]}
          options={''}
          onChange={(e)=> {
            console.log(e);
            setCoordinates({ lat: e.center.lat, lng: e.center.lng  });
            setBounds({ne: e.marginBounds.ne, sw: e.marginBounds.sw});

          }}
          onChildClick={(child)=> setChildClicked(child)}
          >

          {places?.map((place, i) => (
            <div
            className={classes.markerContainer}
            lat={Number(place.latitude)}
            lng={Number(place.longitude)}
            key={i}
           >
            {
              !isDesktop ? (
                <LocationOnOutlinedIcon color="primary" fontSize='large'/>
              ): (
```

```
            <Paper elevation={3} className={classes.paper}>
              <Typography className={classes.typography} variant='subtitle2' gutterBottom>
                {place.name}
              </Typography>
              <img className={classes.pointer} src={place.photo ? place.photo.images.large.url : 'https://www.foodserviceandhospitality.com/wp-content/uploads/2016/09/Restaurant-Placeholder-001.jpg'} alt={place.name}/>

              <Rating name="read-only" size="small" value={Number(place.rating)} readOnly />
            </Paper>
            )
          }
          </div>
        ))}
      </GoogleMapReact>
    </div>
  );
}

export default Map;
```

4. Place details:

PlaceDetails.jsx

```
import React from 'react';
import { Box, Typography, Button, Card, CardMedia, CardContent, CardActions, Chip } from '@material-ui/core';
import LocationOnIcon from '@material-ui/icons/LocationOn';
import PhoneIcon from '@material-ui/icons/Phone';
import Rating from '@material-ui/lab/Rating';
```

```
import useStyles from './styles';

const PlaceDetails = ({place, selected, refProp }) => {
    const classes = useStyles();

    if(selected) refProp?.current?.scrollIntoView({behaviour: "smooth", block: "start" })
    return(
     <Card elevation={6}>
        <CardMedia style={{height:350}}
        image={place.photo ? place.photo.images.large.url : 'https://www.foodserviceandhospitality.com/wp-content/uploads/2016/09/Restaurant-Placeholder-001.jpg'}
         title={place.name}
            />
            <CardContent>
                <Typography gutterBottom variant="h5">{place.name}</Typography>
                <Box display="flex" justifyContent="space-between">
                <Rating name="read-only" value={Number(place.rating)} readOnly />
                    <Typography gutterBottom variant='subtitle1'>Out of {place.num_reviews} reviews </Typography>

                </Box>

                <Box display="flex" justifyContent="space-between">
                    <Typography variant='subtitle1'>Price</Typography>
                    <Typography gutterBottom variant='subtitle1'>{place.price_level}</Typography>

                </Box>

                <Box display="flex" justifyContent="space-between">
```

```
                <Typography variant='subtitle1'>Ranking</Typography>
                <Typography gutterBottom variant='subtitle1'>{place.ranking}</Typography>


            </Box>
            {place?.awards?.map((award) => (
        <Box display="flex" justifyContent="space-between" my={1} alignItems="center">
          <img src={award.images.small} alt='award' />
          <Typography variant="subtitle2" color="textSecondary">{award.display_name}</Typography>
        </Box>
      ))}
      {place?.cuisine?.map(({ name }) => (
        <Chip key={name} size="small" label={name} classNameName={classes.chip} />
      ))}
      {place.address && (
        <Typography gutterBottom variant="body2" color="textSecondary" classNameName={classes.subtitle}>
          <LocationOnIcon />{place.address}
        </Typography>
      )}
      {place.phone && (
        <Typography variant="body2" color="textSecondary" classNameName={classes.spacing}>
          <PhoneIcon /> {place.phone}
        </Typography>
      )}
          </CardContent>
          <CardActions>
```

```
        <Button size="small" color="primary" onClick={() => window.open(place.website, '_blank')}>
          Website
        </Button>
      </CardActions>
    </Card>
  );
}

export default PlaceDetails;
```

App.js

```
import react, {useEffect, useState} from "react";
import Header from "./components/Header/Header";
import List from "./components/List/List";
import Map from "./components/Map/Map";
import {getPlacesData} from './api/index';
import PlaceDetails from "./components/PlaceDetails/PlaceDetails";
import { CssBaseline, Grid } from "@material-ui/core";

const App = () => {
  const [places, setPlaces] = useState([]);
  const [filteredPlaces, setFilteredPlaces] = useState([]);
  const [coordinates, setCoordinates] = useState({});
  const [bounds, setBounds] = useState({});
  const [childClicked, setChildClicked] = useState(null);
  const [isLoading, setIsLoading] = useState(false);
  const [type, setType] = useState('restaurants');
  const [rating, setRating] = useState('');
  const[autocomplete, setAutocomplete] = useState(null);
```

```
useEffect(()=>{
  navigator.geolocation.getCurrentPosition(({ coords: {latitude, longitude} }) => {
     setCoordinates({lat: latitude, lng: longitude});
  })
}, []);

useEffect(()=>{
   const filteredPlaces = places?.filter((place)=> place.rating > rating );
   setFilteredPlaces(filteredPlaces);

},[rating]);


useEffect(()=>{
   if (bounds.sw && bounds.ne) {
      setIsLoading(true);
   getPlacesData(type, bounds.sw, bounds.ne)
   .then((data)=>{
      console.log(data);
       setPlaces(data?.filter((place)=> place.name && place.num_reviews > 0));
       setFilteredPlaces([]);
       setIsLoading(false);
   });
}
},[bounds, type]);


const onLoading = (autoC) => setAutocomplete(autoC);

const onPlaceChanged = () => {
```

```
  const lat = autocomplete.getPlace().geometry.location.lat();
  const lng = autocomplete.getPlace().geometry.location.lng();

  setCoordinates({lat, lng});
};
return(
    <>
    <CssBaseline />
    <Header onPlaceChanged={onPlaceChanged} onLoading={onLoading} />
    <Grid container spacing={3} style={{width: '100%'}}>

        <Grid item xs={12} md={4} >
        <List
        isLoading={isLoading}
        childClicked={childClicked}
        places={filteredPlaces.length ? filteredPlaces : places}
        type={type}
        setType={setType}
        rating={rating}
        setRating={setRating}
        />
        </Grid>

        <Grid item xs={12} md={8} >
            <Map
             setCoordinates={setCoordinates}
             setBounds={setBounds}
             coordinates={coordinates}
             places = {filteredPlaces.length ? filteredPlaces : places}
             setChildClicked={setChildClicked}
```

```
                />
            </Grid>

        </Grid>



        </>
    );
}

export default App;
```

index.js

```
import react from "react";
import reactDom from "react-dom";
import App from './App'

reactDom.render(<App />, document.getElementById('root'));
```

Expense Tracker:

Components:

1. Details

Details.jsx

```
import React from 'react';
import {Card, CardHeader, CardContent, Typography} from '@material-ui/core';
import { Doughnut } from 'react-chartjs-2';
import useStyles from './styles';
import useTransactions from '../../useTransactions';
```

```
const Details = ({title}) => {
   const classes = useStyles();
   const {total, chartData} = useTransactions(title);
   return (
      <Card className={title === 'Income' ? classes.income : classes.expense}>
        <CardHeader title={title} />
        <CardContent>
          <Typography variant="h5">${total}</Typography>
          <Doughnut data={chartData} />
        </CardContent>
      </Card>
    );
   };

export default Details
```

2. Main:
   1. Form

Form.jsx

```
import React from 'react';
import {Card, CardHeader, CardContent, Typography} from '@material-ui/core';
import { Doughnut } from 'react-chartjs-2';
import useStyles from './styles';
import useTransactions from '../../useTransactions';
const Details = ({title}) => {
    const classes = useStyles();
    const {total, chartData} = useTransactions(title);
    return (
       <Card className={title === 'Income' ? classes.income : classes.expense}>
         <CardHeader title={title} />
         <CardContent>
           <Typography variant="h5">${total}</Typography>
```

```
            <Doughnut data={chartData} />
          </CardContent>
        </Card>
      );
    };

export default Details
```

2. List

List.jsx

```
import React, {useContext} from 'react'
import {List as MUIList, ListItem, ListItemAvatar, ListItemText, Avatar, ListItemSecondaryAction, IconButton, Slide} from '@material-ui/core';
import {Delete, MoneyOff} from '@material-ui/icons';
import useStyles from './styles';
import { ExpenseTrackerContext } from '../../../context/context';
const List = () => {
  const {deleteTransaction, transactions} = useContext(ExpenseTrackerContext)

  const classes = useStyles();

  return (
    <MUIList dense={false} className={classes.list}>
      {transactions.map((transaction)=> (
        <Slide direction="down" in mountOnEnter unmountOnExit key={transaction.id}>
          <ListItem>
            <ListItemAvatar>
              <Avatar className={transaction.type === 'Income' ? classes.avatarIncome : classes.avatarExpense}>
                <MoneyOff/>
              </Avatar>
            </ListItemAvatar>
```

```
                <ListItemText primary={transaction.category} secondary={`$${transaction.amount} - ${transaction.date}`} />
                <ListItemSecondaryAction>
                <IconButton edge="end" aria-label="delete" onClick={() => deleteTransaction(transaction.id)}>
                    <Delete/>
                  </IconButton>
                </ListItemSecondaryAction>
              </ListItem>
            </Slide>
          ))}
      </MUIList>
    )
}

export default List
```

main.jsx

```
import { ExpenseTrackerContext } from '../../context/context';
import React, {useContext} from 'react';
import{Card, CardHeader, CardContent, Typography, Grid, Divider} from '@material-ui/core';
import useStyles from './styles';
import Form from './Form/Form';
import List from './List/List';

const Main = () => {
    const {balance} = useContext(ExpenseTrackerContext);
    console.log(balance);
    const classes = useStyles();
    return (
```

```
    <Card className={classes.root}>
      <CardHeader title="Pulse~ Expense-Tracker"/>
      <CardContent>
        <Typography align="center" variant='h5'>
          Total Balance ${balance}</Typography>
        <Typography variant='subtitle1' style={{lineHeight : '1.5em', marginTop: '20px'}}>
          {/* {Infocard} */}
          Try saying: Add income for $100 in category Salary for Monday...
        </Typography>
        <Divider />
        <Form/>
      </CardContent>
      <CardContent className={classes.cardContent}>
        <Grid container spacing={2}>
          <Grid item xs={12}>
            <List/>
          </Grid>
        </Grid>
      </CardContent>

    </Card>
   );
}

export default Main
```

Constants:

Categories.js

```
const incomeColors = ['#123123', '#154731', '#165f40', '#16784f', '#14915f', '#10ac6e', '#0bc77e', '#04e38d', '#00ff9d'];
```

```
const expenseColors = ['#b50d12', '#bf2f1f', '#c9452c', '#d3583a', '#dc6a48', '#e57c58', '#ee8d68', '#f79d79', '#ffae8a', '#cc474b', '#f55b5f'];

export const incomeCategories = [
  { type: 'Business', amount: 0, color: incomeColors[0] },
  { type: 'Investments', amount: 0, color: incomeColors[1] },
  { type: 'Extra income', amount: 0, color: incomeColors[2] },
  { type: 'Deposits', amount: 0, color: incomeColors[3] },
  { type: 'Lottery', amount: 0, color: incomeColors[4] },
  { type: 'Gifts', amount: 0, color: incomeColors[5] },
  { type: 'Salary', amount: 0, color: incomeColors[6] },
  { type: 'Savings', amount: 0, color: incomeColors[7] },
  { type: 'Rental income', amount: 0, color: incomeColors[8] },
];

export const expenseCategories = [
  { type: 'Bills', amount: 0, color: expenseColors[0] },
  { type: 'Car', amount: 0, color: expenseColors[1] },
  { type: 'Clothes', amount: 0, color: expenseColors[2] },
  { type: 'Travel', amount: 0, color: expenseColors[3] },
  { type: 'Food', amount: 0, color: expenseColors[4] },
  { type: 'Shopping', amount: 0, color: expenseColors[5] },
  { type: 'House', amount: 0, color: expenseColors[6] },
  { type: 'Entertainment', amount: 0, color: expenseColors[7] },
  { type: 'Phone', amount: 0, color: expenseColors[8] },
  { type: 'Pets', amount: 0, color: expenseColors[9] },
  { type: 'Other', amount: 0, color: expenseColors[10] },
];

export const resetCategories = () => {
  incomeCategories.forEach((c) => c.amount = 0);
```

```
  expenseCategories.forEach((c) => c.amount = 0);
};
```

Context:

Context.jsx

```
import React, {useReducer, createContext} from 'react';
import contextReducer from './contextReducer';

const initalState = JSON.parse(localStorage.getItem('transactions')) || [];

export const ExpenseTrackerContext = createContext(initalState);
export const Provider = ({children})=>{
   const [transactions, dispatch] = useReducer(contextReducer, initalState);
   const deleteTransaction = (id) => {
      dispatch({ type: 'DELETE_TRANSACTION', payload: id });
     };

     const addTransaction = (transaction) => {
      dispatch({ type: 'ADD_TRANSACTION', payload: transaction });
     };
     const balance = transactions.reduce((acc, currVal) => (currVal.type === 'Expense' ? acc - currVal.amount : acc + currVal.amount), 0);
     console.log(transactions);
   return(
      <ExpenseTrackerContext.Provider value={{deleteTransaction, addTransaction, transactions, balance}}>
         {children}
      </ExpenseTrackerContext.Provider>
   )
}
```

ContextReducer.jsx

```
const contextReducer = (state, action) => {
  let transactions;

  switch (action.type) {
    case 'DELETE_TRANSACTION':
      transactions = state.filter((transaction) => transaction.id !== action.payload);

      localStorage.setItem('transactions', JSON.stringify(transactions));

      return transactions;
    case 'ADD_TRANSACTION':
      transactions = [action.payload, ...state];

      localStorage.setItem('transactions', JSON.stringify(transactions));

      return transactions;
    default:
      return state;
  }
};

export default contextReducer;
```

Utils:

formatDate.js

```
export default (date) => {
    const d = new Date(date);
    let month = `${d.getMonth() + 1}`;
    let day = `${d.getDate()}`;
    const year = d.getFullYear();
```

```
    if (month.length < 2) { month = `0${month}`; }
    if (day.length < 2) { day = `0${day}`; }

    return [year, month, day].join('-');
  };
```

App.js

```
import { PushToTalkButton, PushToTalkButtonContainer } from '@speechly/react-ui';
import React from 'react';
import Details from './components/Details/Details';
import Main from './components/Main/Main';
import {Grid} from '@material-ui/core';
import useStyles from './styles'



const App = () => {
    const classes = useStyles();
    return (
        <div>
            <Grid classname={classes.grid}container spacing={0} alignItems="center" justify="center" style={{height:'100vh'}}>
            <Grid item xs={12} sm={3} >
                <Details title="Income"/>
            </Grid>
            <Grid item xs={12} sm={3} >
              <Main />
            </Grid>
            <Grid item xs={12} sm={3} >
                <Details title="Expense"/>
```

```
            </Grid>
            </Grid>
            <PushToTalkButtonContainer>
                <PushToTalkButton />
            </PushToTalkButtonContainer>
        </div>
    );
}

export default App
```

index.js

```
import React from 'react';
import ReactDom from 'react-dom';
import App from './App';
import { Provider } from './context/context';
import { SpeechProvider } from '@speechly/react-client';
import './index.css'

ReactDom.render(<SpeechProvider appId="f533a0d0-53de-40be-8dd5-cb583174d75c" language="en-US"> <Provider><App /> </Provider> </SpeechProvider>, document.getElementById('root'));
```

useTransactions.js

```
import { useContext } from 'react';
import { ExpenseTrackerContext } from './context/context';

import { incomeCategories, expenseCategories, resetCategories } from './Constants/categories';
```

```
const useTransactions = (title) => {
  resetCategories();
  const { transactions } = useContext(ExpenseTrackerContext);
  const rightTransactions = transactions.filter((t) => t.type === title);
  const total = rightTransactions.reduce((acc, currVal) => acc += currVal.amount, 0);
  const categories = title === 'Income' ? incomeCategories : expenseCategories;

  rightTransactions.forEach((t) => {
    const category = categories.find((c) => c.type === t.category);

    if (category) category.amount += t.amount;
  });

  const filteredCategories = categories.filter((sc) => sc.amount > 0);

  const chartData = {
    datasets: [{
      data: filteredCategories.map((c) => c.amount),
      backgroundColor: filteredCategories.map((c) => c.color),
    }],
    labels: filteredCategories.map((c) => c.type),
  };

  return { filteredCategories, total, chartData };
};

export default useTransactions;
```

COCOMO

COCOMO (Constructive Cost Model) is a regression model based on LOC, i.e., **number of Lines of Code**. It is a procedural cost estimate model for software projects and often used as a process of reliably predicting the various parameters associated with making a project such as size, effort, cost, time and quality. It was proposed by Barry Boehm in 1970 and is based on the study of 63 projects, which make it one of the best-documented models.
Different models of COCOMO have been proposed to predict the cost estimation at different levels, based on the amount of accuracy and correctness required. All of these models can be applied to a variety of projects, whose characteristics determine the value of constant to be used in subsequent calculations.

- Organic
- Semi detached
- Embedded

Mode	**"A" variable**	**"B" variable**	**"C" variable**	**"D" variable**	**KLOC**
Semi-Detached	3	1.12	2.5	0.35	3.5
Effort (person-months)		**Duration (months)**		**Staffing**	
10.5		5.6931		1.844	

Table 5.1 COCOMO Model

Staffing=effort/duration, effort=a*KLOC^{b}, KLOC=lines of code (thousands)
duration=c*effort^d

In this project functions are defined to generalise the use of multiple lines of code wherever possible.
In this project all the functions defined, or modules used are separate from each other and are saved in different files, which makes it easy for the developer to navigate in between them. Separate directories are created for separate views or pages as well as different projects for the same purpose.
Use of predefined libraries are followed to minimize the lines of codes.

5.3 Testing Approach

Testing is very essential for a system to work smoothly. It helps us find any bugs or errors and rectify them. Generally different models such as category partition testing and state machine-based testing are used to implement testing.

In the project, category partition testing is used. Both the approaches of functional testing and user acceptance testing are appropriate, but user acceptance testing is focused more on in the project. White Box testing method is implemented for testing the different web applications.

5.3.1 Test Cases:

Test Case 1: Landing site functionality

Test Case Id	Test Case Description	Test Steps	Expected Result	Actual Result	Status
1	Entering the landing site and use the different links to observe the different web apps	Click on each project link and see whether you're redirected to them	Redirected to respective web app page	Redirected to the respective web apps	Pass
2	Click on about option	Click on the about option	The about content box appears	The about content box appears	Pass
3	Click on review option	Click on the review option	The review content box appears	The review content box appears	Pass
4	Click on the Review us link	Click on the Review us link	Redirected to Review page	Redirected to Review page	Pass

Test Case 2: Cryptograph, stocks, DNA analyser application functionality

Test Case Id	Test Case Description	Test Steps	Test Data	Expected Result	Actual Result	Status
1	Select different crypto currencies from the drop down	Click on the dropdown and select another currency	None	The graph as well as content shown is changed to the respective selection	The graph as well as content shown is changed to the respective selection	Pass
2	Select different time stamps	Click on the different buttons for different time stamps	None	The graph shows data according to the time stamp selected	The graph shows data according to the time stamp selected	Pass
3	Select different companies for stock graphs	Click on the drop-down list and select a company	None	The data should be displayed for the company selected and show the graph	The data should be displayed for the company selected and show the graph	Pass
4	Enter a different DNA sequence	Remove the default DNA query and enter a	DNA sequence	The bar chart should show case the new counts of ATGC	The bar chart shows case the new counts of ATGC	Pass

	and click shift enter	new DNA sequence and hit enter		respectively as well as the counts for each is displayed on the data displayed	respectively as well as the counts for each is displayed on the data displayed	

Test Case 3: Weather app, news app, expense tracker

Test Case Id	Test case description	Test steps	Expected result	Actual result	Status
	Add a new city name (for weather app)	Click on the input box and enter the city for which you want weather details and click on the add button	The new city appears in the list along with a weather information as well as a db update	The new city appears in the list along with a weather information as well as a db update	Pass
	Latest News should be displayed when loaded (news app)	None	News details should be updated	Details are updated	Pass
	Add income/expense manually(expense tracker)	Add an expense/income field and submit a transaction	The doughnut chart is updated as well as the transaction is added to the list of all transactions	The doughnut chart is updated as well as the transaction is added to the list of all transactions	Pass
	Add income/expense verbally (expense tracker)	Speak holding the voice button and make a transaction	The doughnut chart is updated as well as the transaction is added to the list of all transactions	The doughnut chart is updated as well as the transaction is added to the list of all transactions	Pass
	Delete a transaction	Remove the transaction from the list of transactions or verbally make a query to remove a transaction	The transaction is removed, and the doughnut chart is updated	The transaction is removed, and the doughnut chart is updated	Pass

Test Case 4: Pulse advisor and review web applications:

Test Case Id	Test case description	Test steps	Expected result	Actual result	Status
	Filter by reviews	Click on the dropdown and select a different review	Only hotels/places or restaurants with filtered reviews should load	Only hotels/places or restaurants with filtered reviews should load	Pass
	Select hotels or places instead of restaurants	Click on the dropdown and select either hotel or places option	Only the selected category loads	Only the selected category loads	Pass
	Move around the map by dragging	Select and move the cursor on the map	The places and cards get automatically updated based on the moving or clicking on the map	The places and cards get automatically updated based on the moving or clicking on the map	Pass
	Submit a review (Review app)	Select and submit a review	The review is processed and stored in a database along with a mail sent to the admin about the review details. Showing the thank you page to the user	The review is processed and stored in a database along with a mail sent to the admin about the review details. Showing the thank you page to the user	Pass

5.4 Modifications and Improvements

Testing phase usually leads to the discovery of certain bugs and errors. The different web applications didn't have any bugs after going through the testing phase.

Chapter 6: Results and Discussions

6.1 Test Reports

The main purpose of a test case is to ensure if different features within an application are working as expected. It helps to test the validation of the system. In the following section we will see the different modules associated with the project

The modules of this project are as follows:

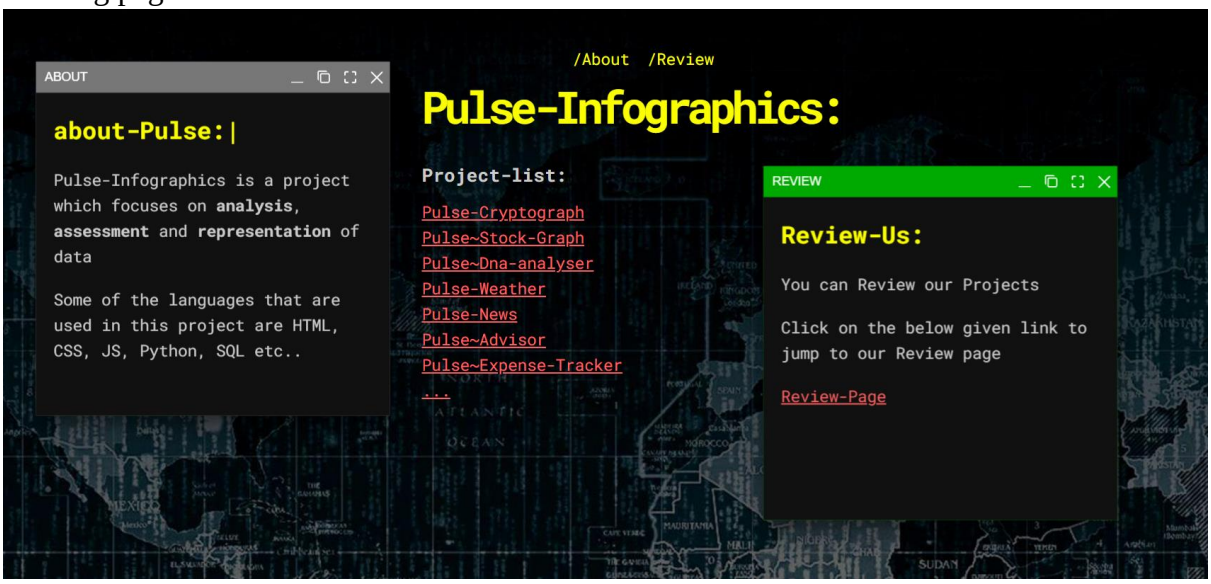
- Landing page
- Review page
- Project-list.
  The project list further contains different web applications as an entire module which is independent of the others

    a. Pulse-Cryptograph (a graphical visualisation of the value of various crypto currencies)
    b. Pulse~Stock-graph (a graphical visualisation on the opening and closing prizes of different companies)
    c. Pulse~DNA-analyser (a graphical visualisation based on DNA query input by the user)
    d. Pulse-Weather (a information displayed about the weather of certain cities that the user inputs)
    e. Pulse-News (headlines along with headers displayed for latest news fetched over the internet)
    f. Pulse-Advisor (a travel advisor for the user that recommends places based on the user's geo location and displays them on the map)
    g. Pulse~Expense-Tracker (An expense tracker that visualises the users income and expense and can be altered manually as well as with the help of speech recognition)

6.2 User Documentation

Pulse infographics provides users with visual data which makes it easier to interpret as well as understand different trends associated with the project. The different modules and components are showcased below.

- Landing page



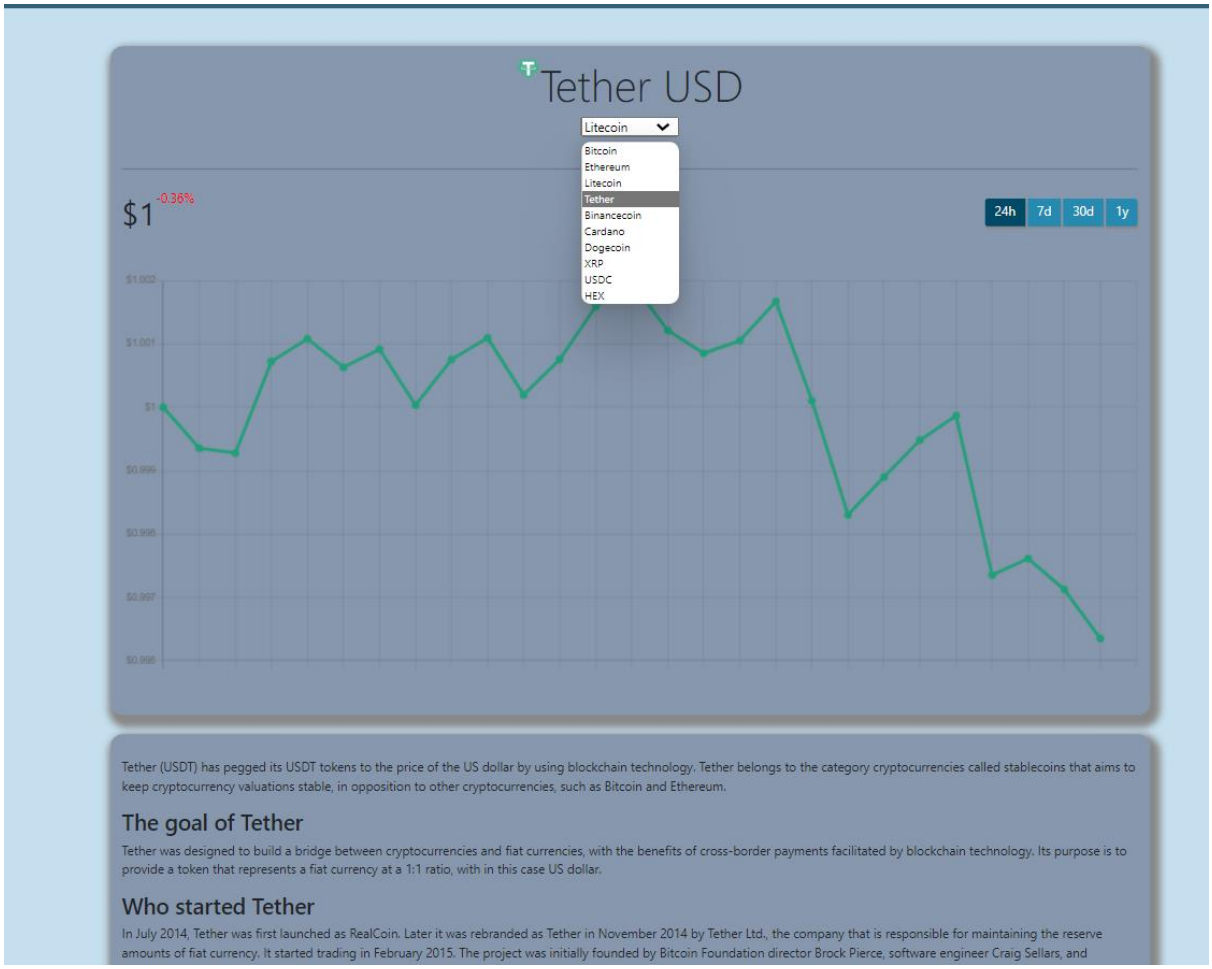
The landing site contains 3 parts:

h. The about window that shows a brief description about the pulse infographics project.
i. The Review window that contains a link that redirects the user to the review page.
j. The project list that contains a list of all the web applications that showcase different types of information and are made for different types of data representation.

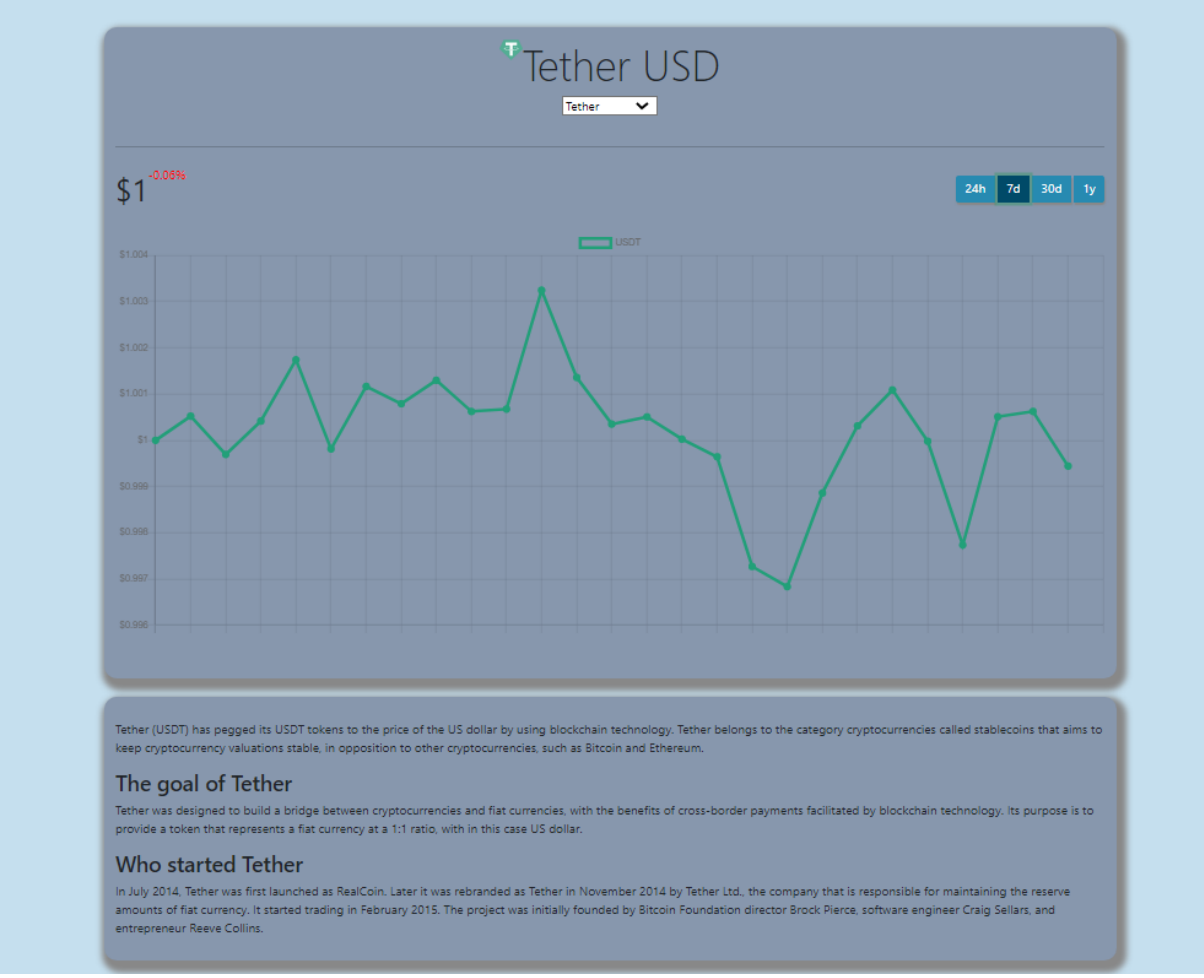
The user can further close or open the review and about pulse window clicking on the window panel options or just clicking on the about or review buttons. He/she can also open the different projects by clicking on them. The review page can be accessed by clicking on the review-page link.

- Pulse-cryptograph page

The pulse cryptograph web application displays Realtime crypto values against the USD and showcases the value graph where a user can see the trend of the graph as well as choose different time period for which the graph must be displayed. There is a content part below the graph that displays the relevant content associated with the current crypto currency that is selected.

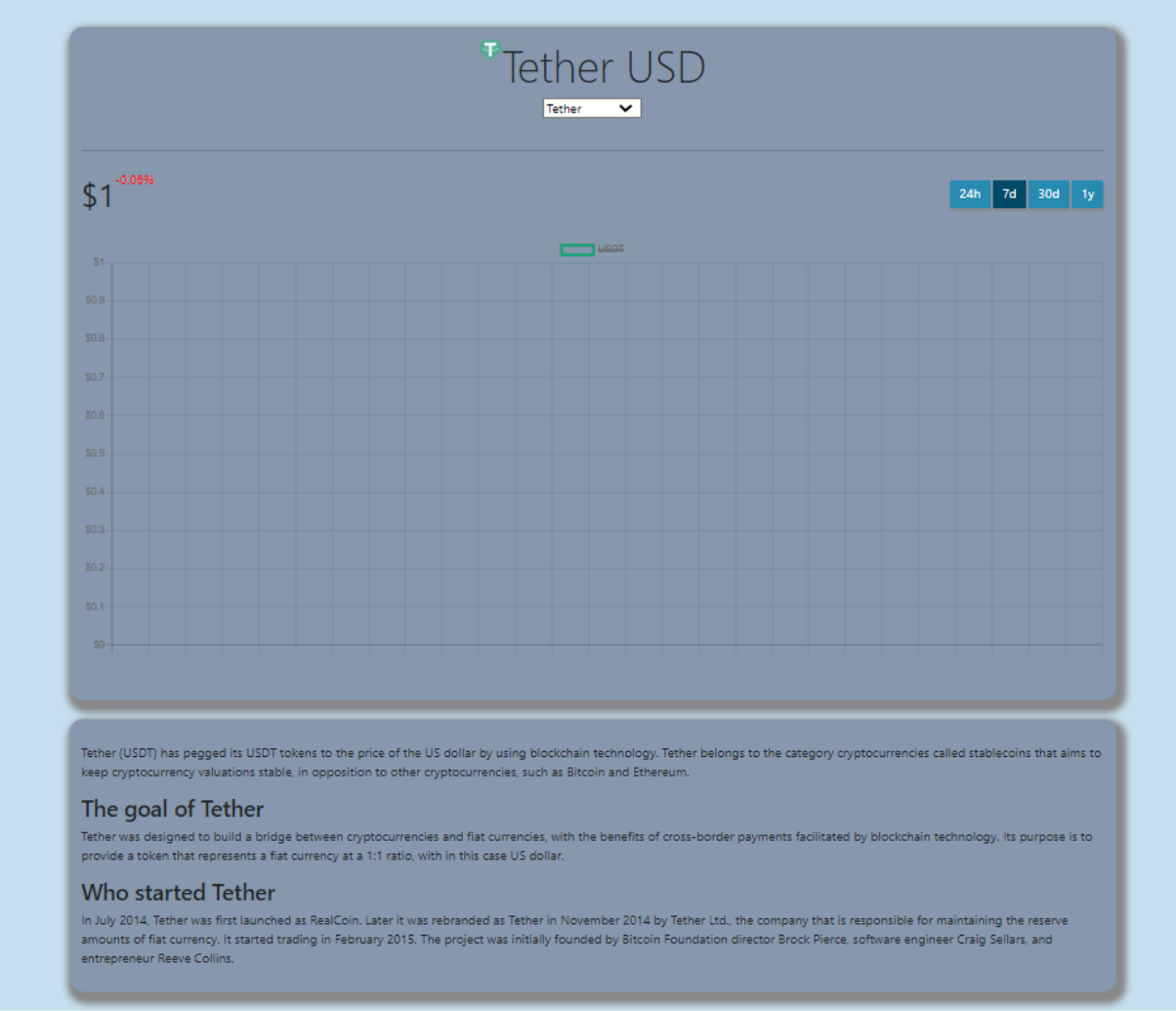


On changing the selection of the crypto currency, we can see that the graph and content change accordingly.



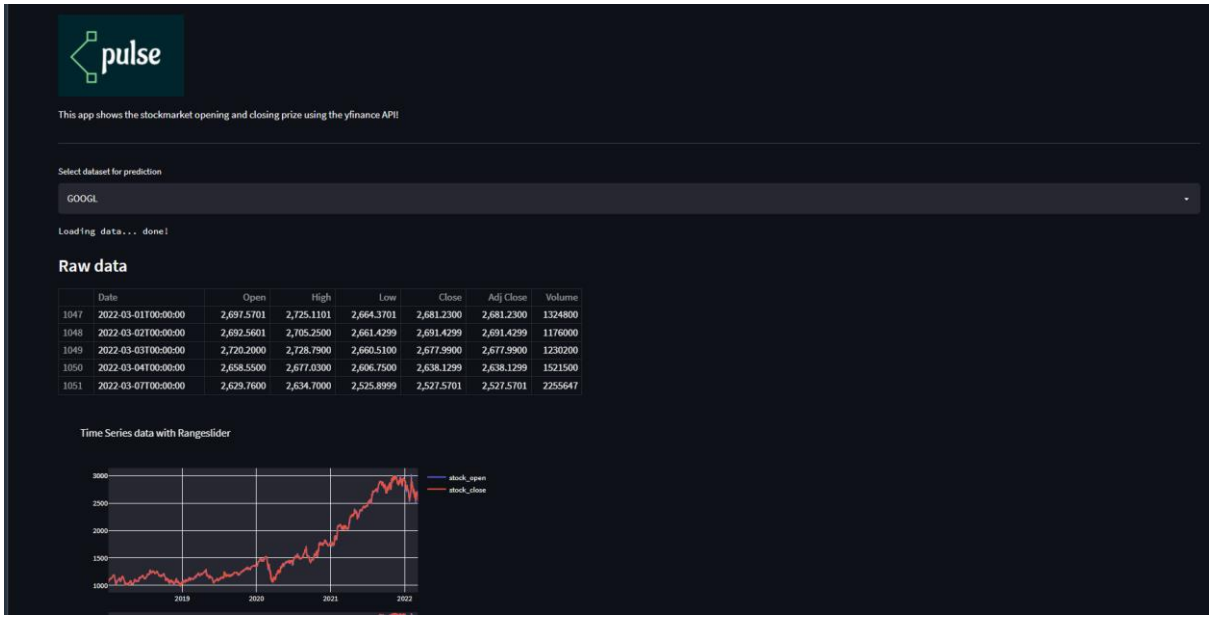
We can see that the graph changes in accordance to the radio button group containing the different timelines

We also see that the content is not affected by that and only one of the buttons can be clicked at a time and it turns dark blue.

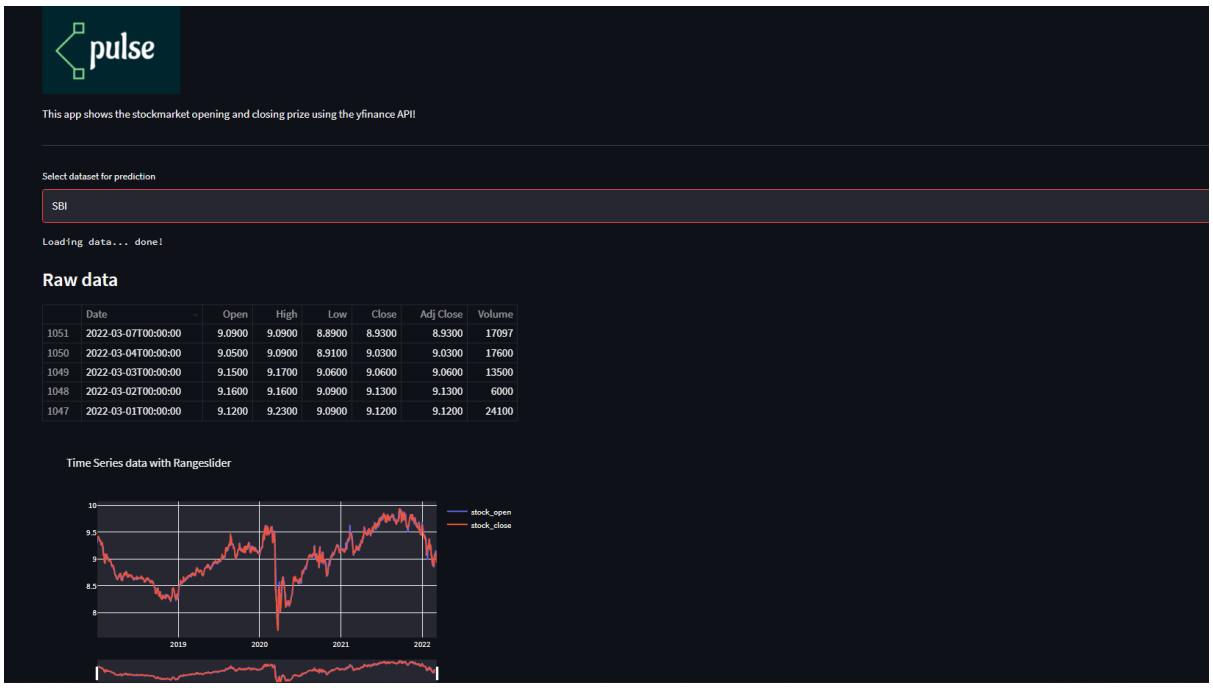


A user can also deselect the graph and make it invisible by clicking on the box button placed at the centre of the graph.

- Pulse~Stock-graph

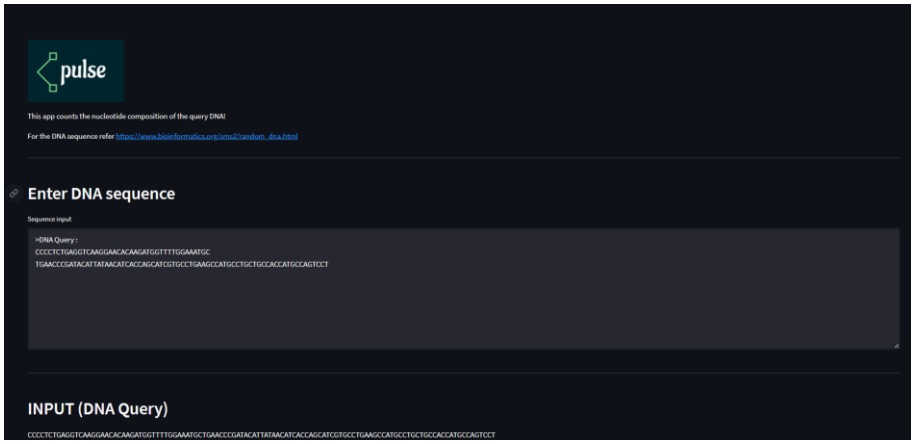


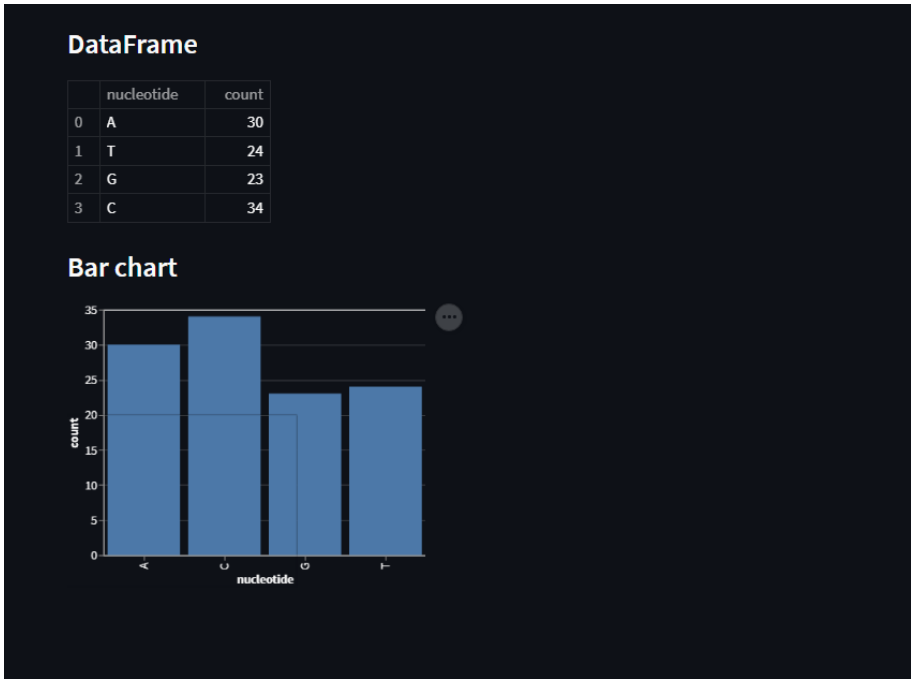
The pulse stock graph displays the opening and closing prices of stock along with the dataset. The user can change the stock for which he needs the stock prizes by selecting from the drop-down list.



We can see the data along with the graph changes accordingly when the company is changed.

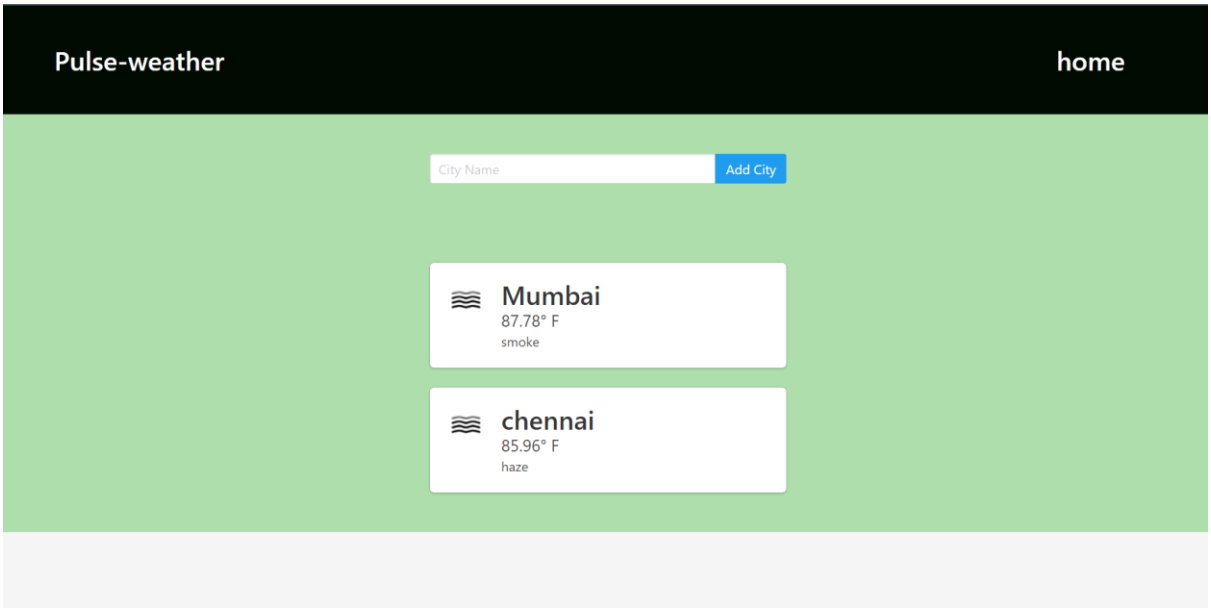
- Pulse~Dna-analyser





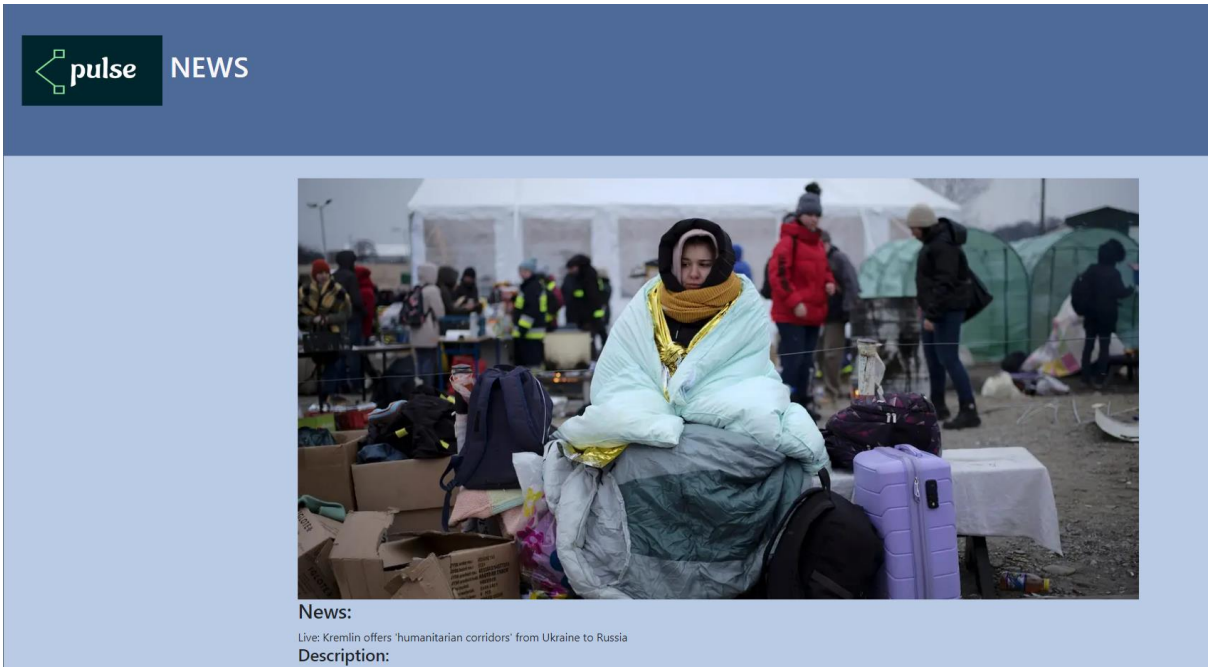
The Pulse DNA analyser showcases the count of different nucleotide based on an input DNA sequence and plots a bar chat accordingly. The user has the ability to change the input sequence and observe the changes in the plot as well as the data.

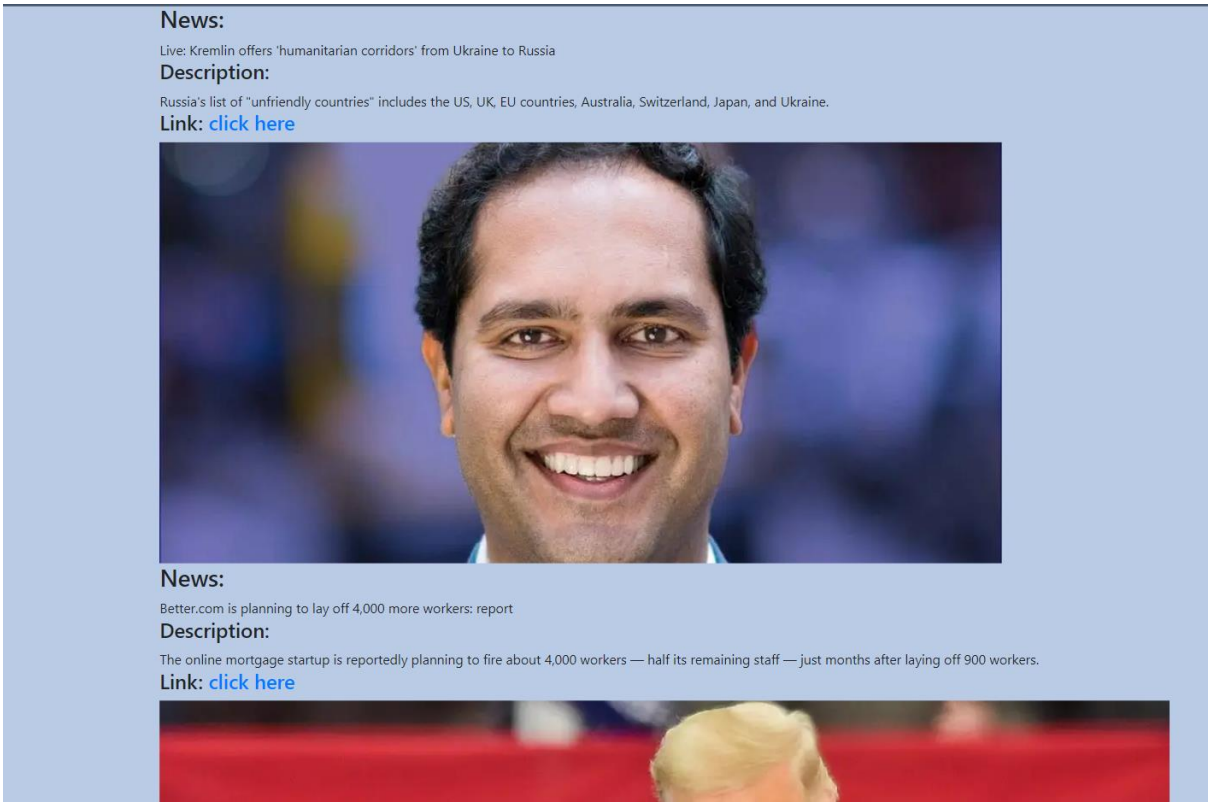
- Pulse-Weather



The pulse weather app showcases the weather for different cities along with the symbology associated for the weather. The user can add another city for which he wants to know the weather as well and the app updates the webpage.

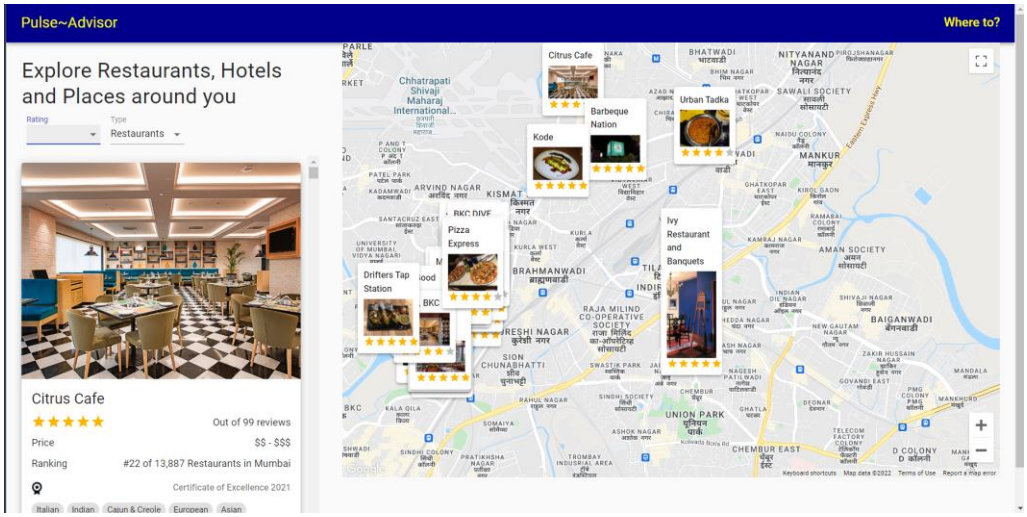
- Pulse-News

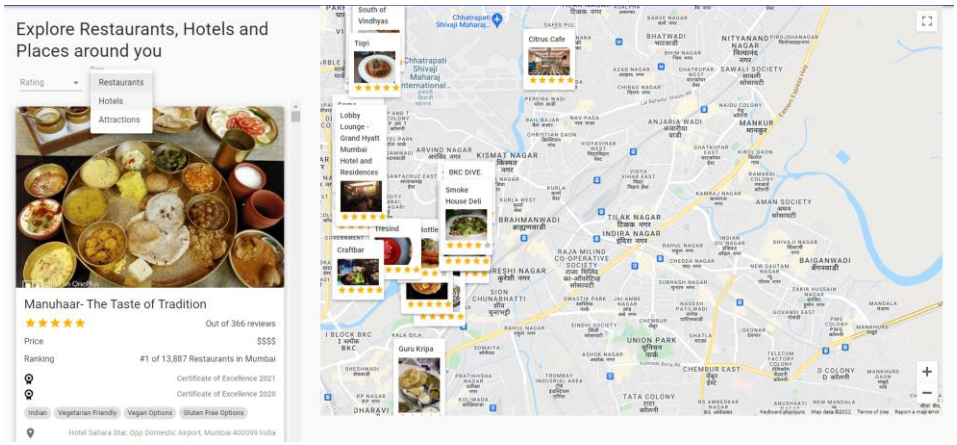


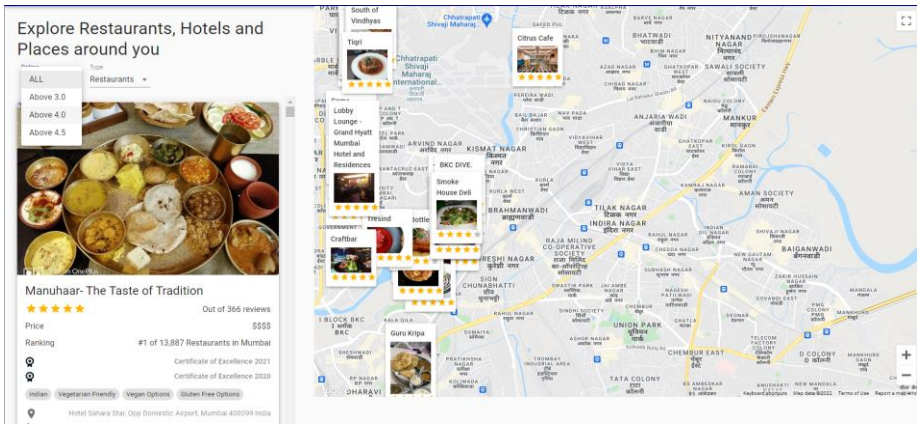


The pulse news app dynamically loads the latest news and showcases them accordingly. The user can click on the link and read the news article from the relevant news website as well.

- Pulse~Advisor



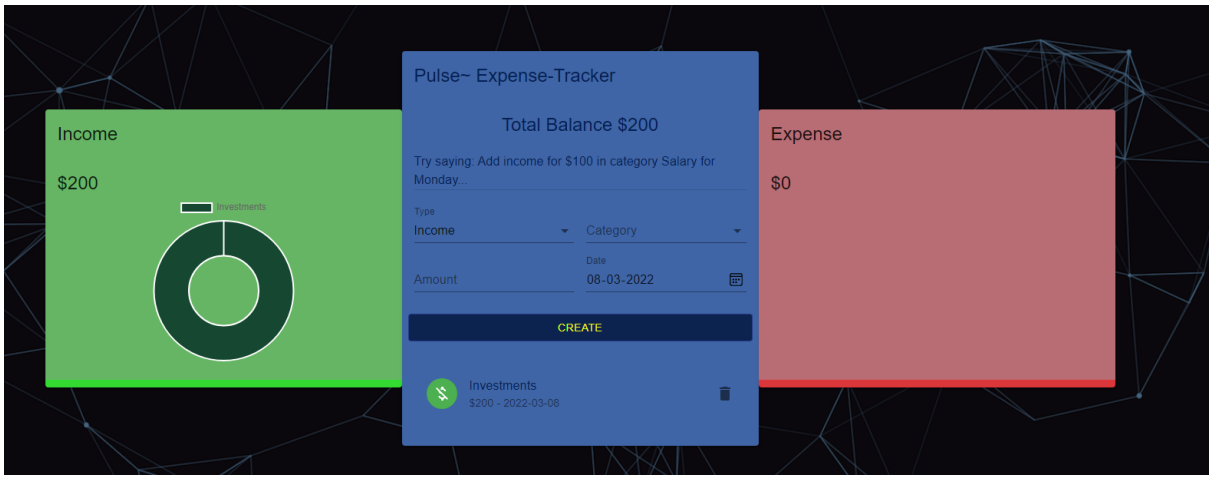




The pulse advisor is a travel advisor that helps users explore places around them by showcasing different places around the user and showcasing the map and cards associated with the place. It also showcases different information about the place using cards. The user can use the map or even filter places or according to ratings

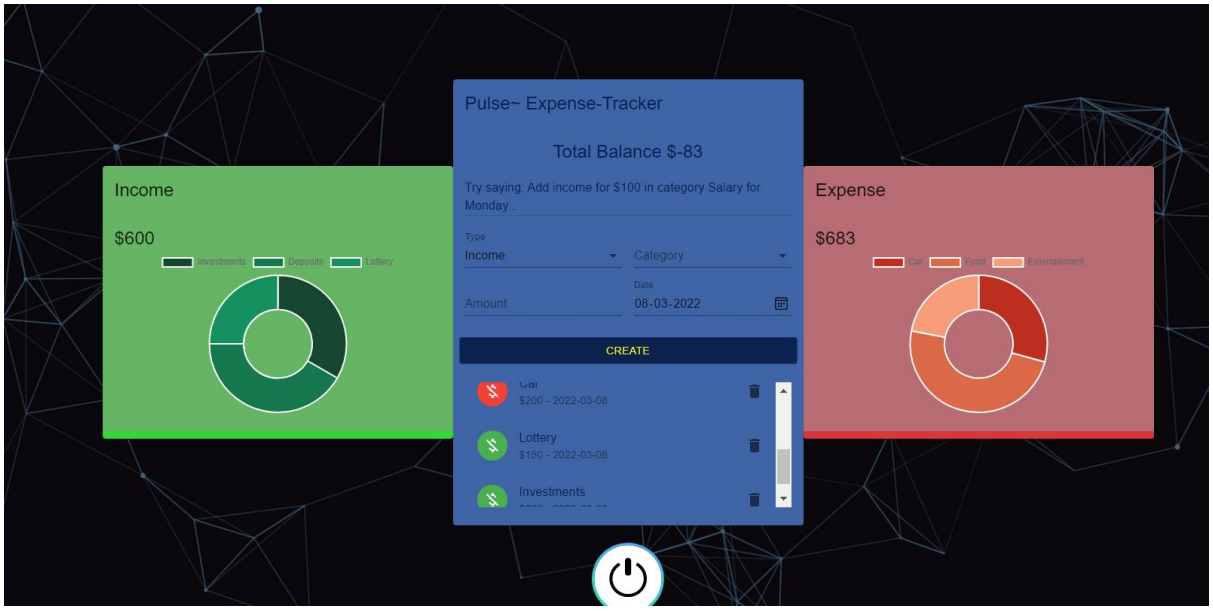
- Pulse~Expense-Tracker

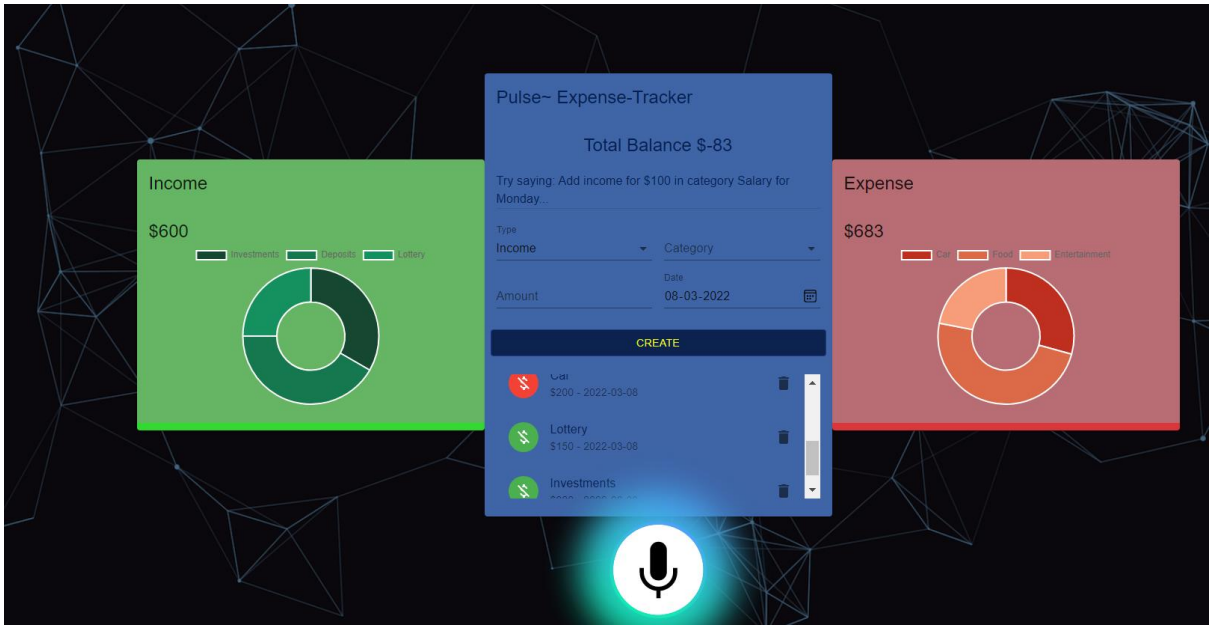
The pulse expense tracker keeps a record of the user's expense and income along with the display of a doughnut chart that is developed after a transaction is recorded



Here the user has added the income for 200 dollars and a doughnut chart is added

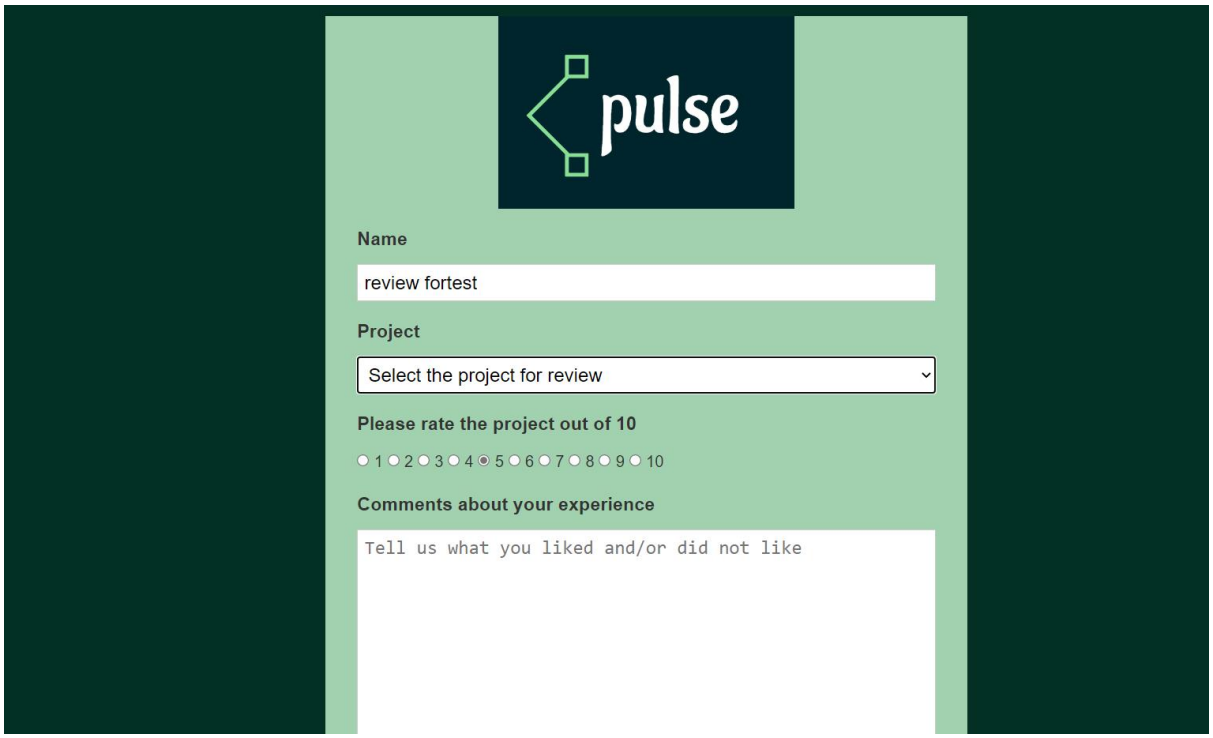
The user can add and delete a transaction manually or by using the mic and talk in it and we can see the balance



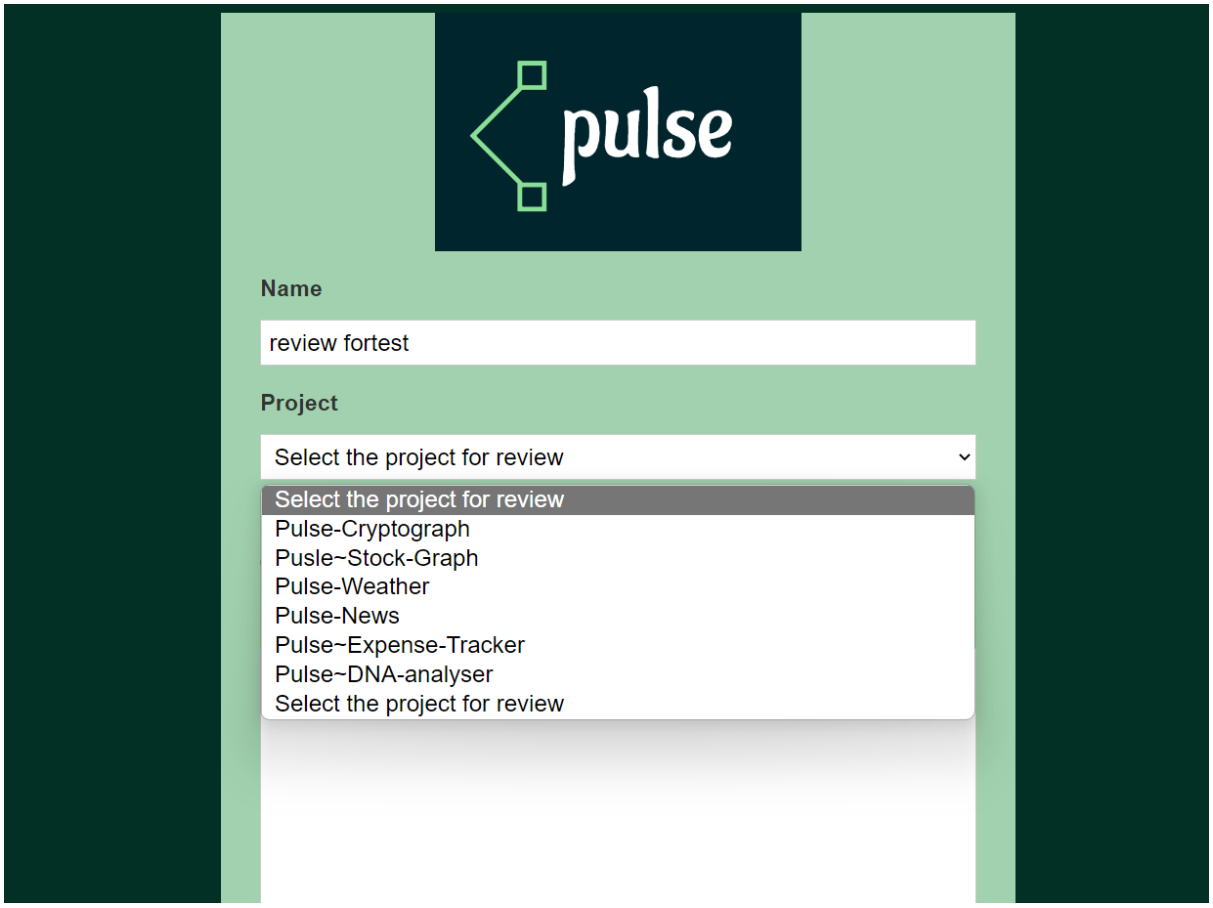


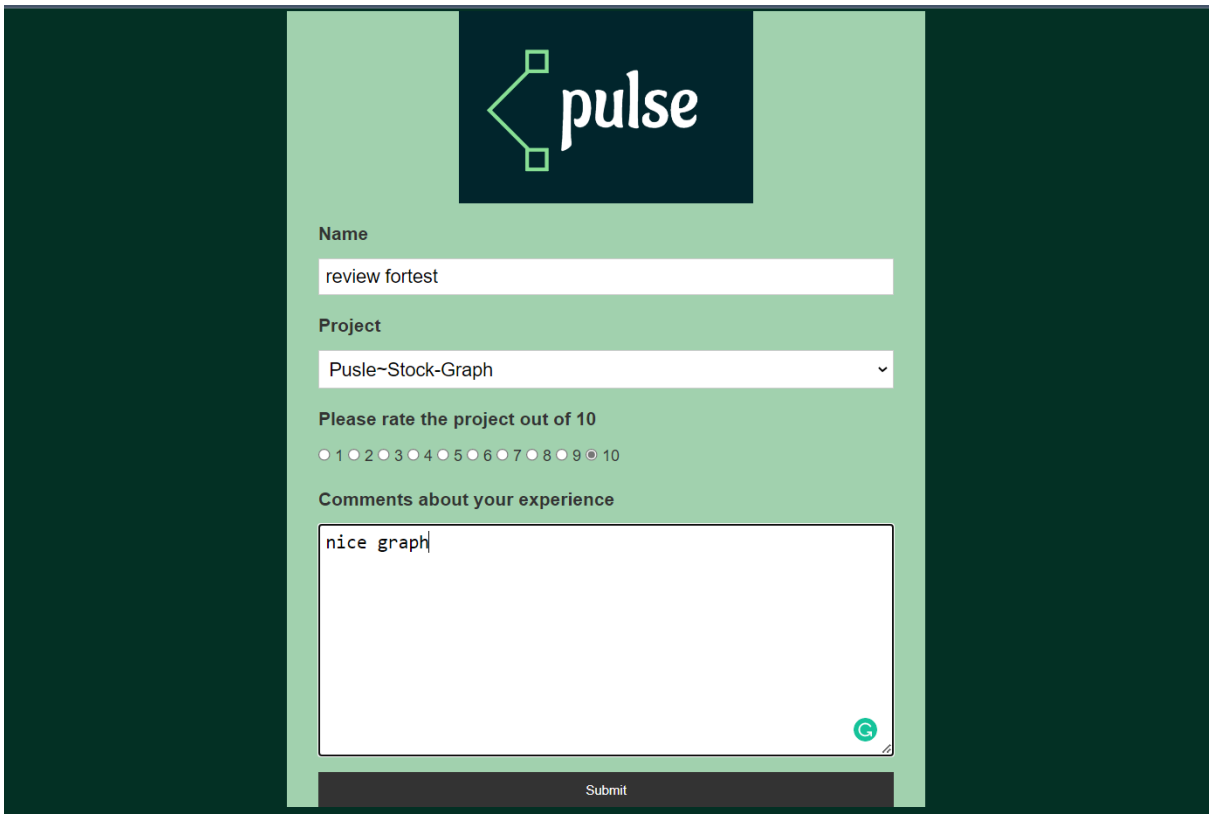
This is the animation that loads when the user speaks holding the mic button.

- Pulse-Review

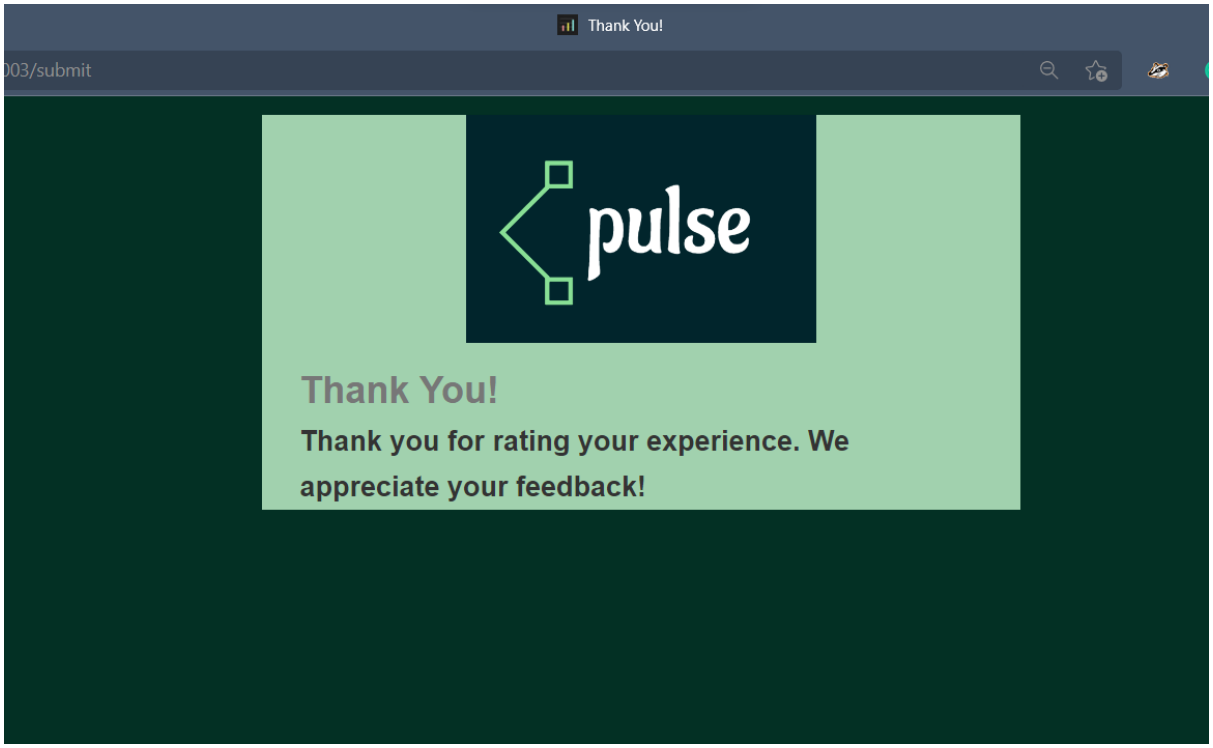


The user can put up his name and then select one of the projects from the dropdown and give a rating

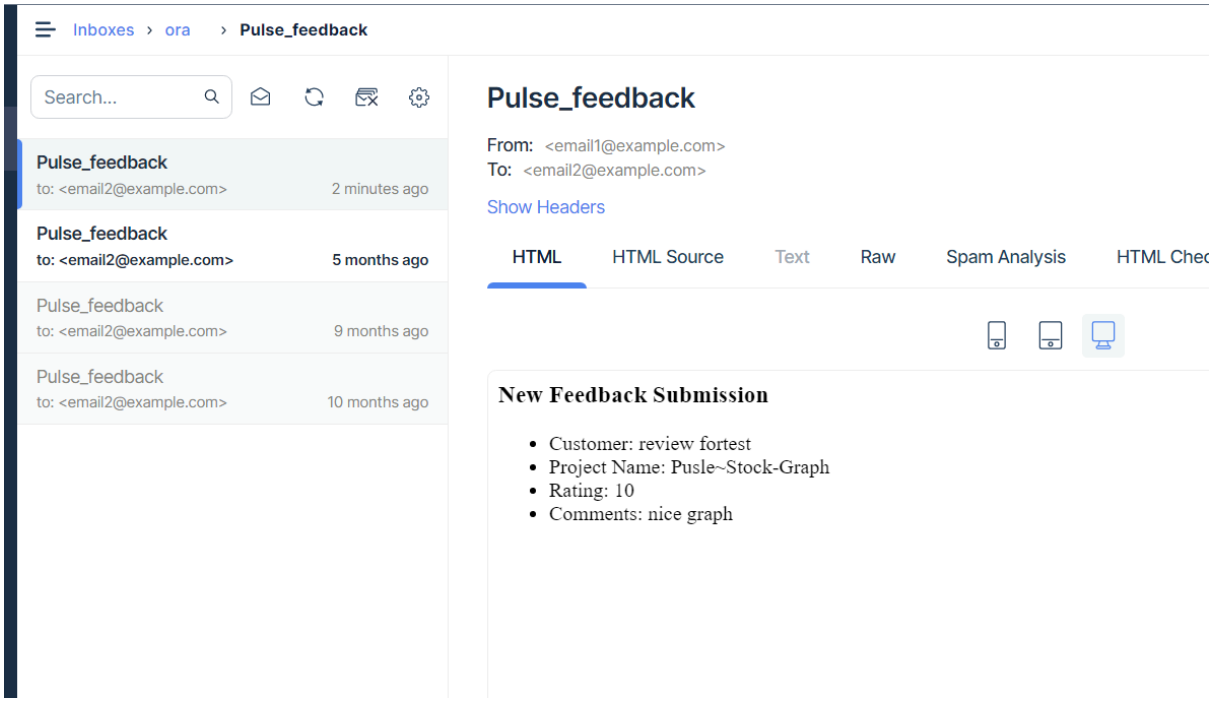

pulse
Name
review fortest
Project
Select the project for review
Select the project for review
Pulse-Cryptograph
Pusle~Stock-Graph
Pulse-Weather
Pulse-News
Pulse~Expense-Tracker
Pulse~DNA-analyser
Select the project for review


pulse
Name
review fortest
Project
Pusle~Stock-Graph
Please rate the project out of 10
1 2 3 4 5 6 7 8 9 10
Comments about your experience
nice graph
Submit

After submission the user is greeted by the thankyou page



The admin receives a mail of the rating as well the review is stored in a table.



Chapter 7: Conclusion

7.1 Significance of the System

This project is an accumulation of small projects that aim to provide information to the users by using graphical and visual analogy. The project brings in information from different relevant fields in our day-to-day life and provides a way to access them freely without discrimination.

It aims to provide users with different information-based services that are already available on the internet, with the ease of navigating them under the same belt. It also tries to embed as many information sources as possible, that provide relevant and up to date information.

7.2 Limitations of the System

Almost every web application under the project uses a different framework and server to function. Though this makes the project more flexible for development, it ends up using more resources than what a standard project would utilize. It also makes the individual projects rely on the underlying framework and sources from which information is being gathered.

7.3 Future Scope of the Project

The project is fully functional as well as has the capability to expand massively, since more web applications can be added, whenever the developer decides on a new web project. The best part about the project being that the developer doesn't have to particularly stick to any framework for the development. The already made web applications can be made to have extra features based on the capabilities of the API's that have been used.

References

https://datadialogs.ischool.berkeley.edu/2015/schedule/running-agile-data-science-teams

https://www.edureka.co/blog/data-science-projects/

https://rockcontent.com/blog/source-code-the-5-rules-of-researching-and-sourcing-infographics/

https://data-flair.training/blogs/data-science-project-ideas/

https://stackoverflow.com/

https://www.youtube.com/channel/UCmXmlB4-HJytD7wek0Uo97A

https://www.freecodecamp.org/