Group Assignment - Source Code

Due May 19, 2019 by 23:59 **Points** 20 **Submitting** a file upload **File Types** zip
Available until May 28, 2019 at 13:00

This assignment was locked May 28, 2019 at 13:00.

COMP5347 Web Application Development

Group Assignment - Data Analytics Web Application

Introduction

In this assignment, you will work as a group to build a small data analytic web application. Each group must have three members. You will demonstrate that you are able to design and implement a typical three-tier web application. You will also demonstrate that your application can communicate with third party web site through published web services.

You will be given a data set, containing some historical data. You will need to query and compute various summary statistics at the server-side and present the results on a web page. You may also need to download latest data from the original data source and append that to the data set.

Dataset

The main dataset contains a number of files in JSON formats storing revision histories of Wikipedia articles. The dataset was created by querying **Wikipedia API (https://www.mediawiki.org/wiki/API:Main_page)** . The articles are selected from **Featured Articles (https://en.wikipedia.org/wiki/Wikipedia:Featured_articles)** .

Each file stores the entire revision history of an article up to 24 March 2019 (around 1:00 PM Sydney time). The file is named after the article's title. Each revision is stored as a JSON object, with many properties. The whole file contains an array of many revision objects. Not all properties will be used in the assignment. Below are the relevant ones:

- **title**: stores the title of the article

- **timestamp**: stores the date and time a revision was made
- **user**: stores the user name or IP address of the user that made the revision
- **anon**: the presence of the field indicates that the revision is made by anonymous users.

Explanation of other properties can be found from corresponding MediaWiki API document at **this page (https://www.mediawiki.org/wiki/API:Revision)**. Below are examples of two typical revision objects (Australia.json):


```
[
  {
       "revid": 889196604,
       "parentid": 888459794,
       "minor": true,
       "user": "SS49",
       "userid": 33136685,
       "timestamp": "2019-03-24T03:26:51Z",
       "size": 198222,
       "sha1": "2b8eefacfbeb07bb85b124065a3760e4dccfb1ce",
       "parsedcomment": "…",
       "title": "Australia"
   },
   …
   {
       "revid": 381546472,
       "parentid": 381467622,
       "minor": false,
       "user": "86.161.49.158",
       "anon": true,
       "userid": 0,
       "timestamp": "2010-08-28T19:26:02Z",
       "size": 144127,
       "sha1": "d5116fb48689c8acd28afbafb94fc32dc4f6b8f5",
       "parsedcomment": "…",
       "title": " Australia "
   },
…
```


Both revision examples shown above are from an article with title “Australia”. The first revision is made by a registered user with name “SS49”. The second revision example is made by an anonymous user as indicated by the presence of property ‘anon’. The value of ‘user’ property is an IP address.

The dataset also contains several extra text files: *admin_*.txt* and *bot.txt*. Wikipedia has 4 administrator types including active administrators, semi-active administrators, inactive administrators and former administrators. Each of admin text files (*admin_active.txt*, *admin_semi_active.txt*, *admin_inactive.txt*, and *admin_former.txt*) contains a list of all administrators in English Wikipedia depending on the admin type. Administrators can perform special actions on wiki pages, some of which are recorded as revisions. The *bot.txt* contains a list of all bot users in English Wikipedia. The bot users have registered names but are not human editors. They are scripts designed for automatic tasks, such as fixing grammatical errors or detecting vandalism. Many bot user’s actions would result in new revisions. From the provided files, you

may notice that several bot users are also administrators since they received administrator privilege, so they are able to perform special actions.

Functional Requirements

Main/Landing Page

The Landing page should display key information about the application, ***ArticleInsight***, in the form of text and images including description of the available functionality with some sample analytics graphs that can be generated through the application. Also, it should provide two options; Sign-up and Login. Users cannot access or use available statistic functions until they create a valid account and login.

All users must *sign-up* before they can see and interact with the application's functionality. So, your application should allow users to create an account first. The user must provide first name and last name, email address (username), and password. You need to do appropriate data validation to ensure valid data is entered before creating an account. The sign-up and login functions must be secure in terms of sign-up and login. You do not need to implement any verification for the sign-up process. Once all data is entered correctly, an account should be created and maintained in the database.

Once an account is created successfully, a user should be able to login using their email address and password. Users should be able to see and interact with the analytics functionality only upon successful login (you must manage the login sessions properly). Users should also be able to logout from your web application.

Article Analytics Functionality

Your application should compute various analytics at overall data set level and at individual article level.

Overall Analytics

For overall analytics, you need to find out and display the following as text:

- Titles of the two articles with highest number of revisions. This is the default behavior.
- Titles of the two articles with lowest number of revisions. This is the default behavior.
- The user should be provided with a way to change the number of articles for highest and lowest number of revisions, the same number should be applied to both categories.
- The article edited by largest group of registered users. Each wiki article is edited by a number of users, some making multiple revisions. The number of unique users is a good indicator of an article's popularity.
- The article edited by smallest group of registered users.

- The top 2 articles with the longest history (measured by age). For each article, the revision with the smallest timestamp is the first revision, indicating the article's creation time. An article's age is the duration between now and its creation time.
- Article with the shortest history (measured by age).

Beside the above functionality, you also need to provide some visual analytics including:

- A bar chart of revision number distribution by year and by user type across the whole dataset. Figure 1 shows an example.
- A pie chart of revision number distribution by user type across the whole data set. Figure 2 shows an example. When users click or users' mouse hovers to the administrator part of the pie chart, you must show the distribution of each administrator type (active, semi-active, inactive, and former).

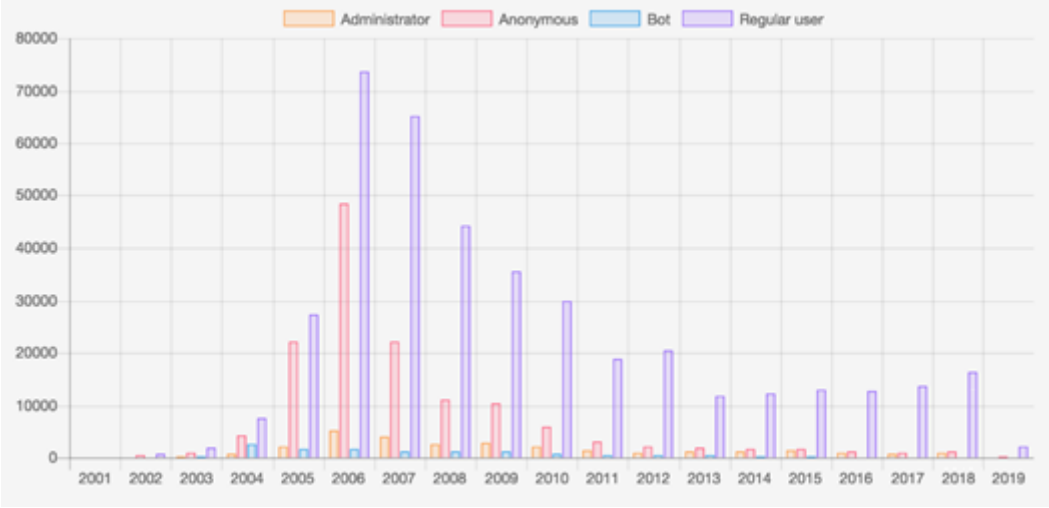

Figure 1: Example bar chart showing overall yearly revision number distribution

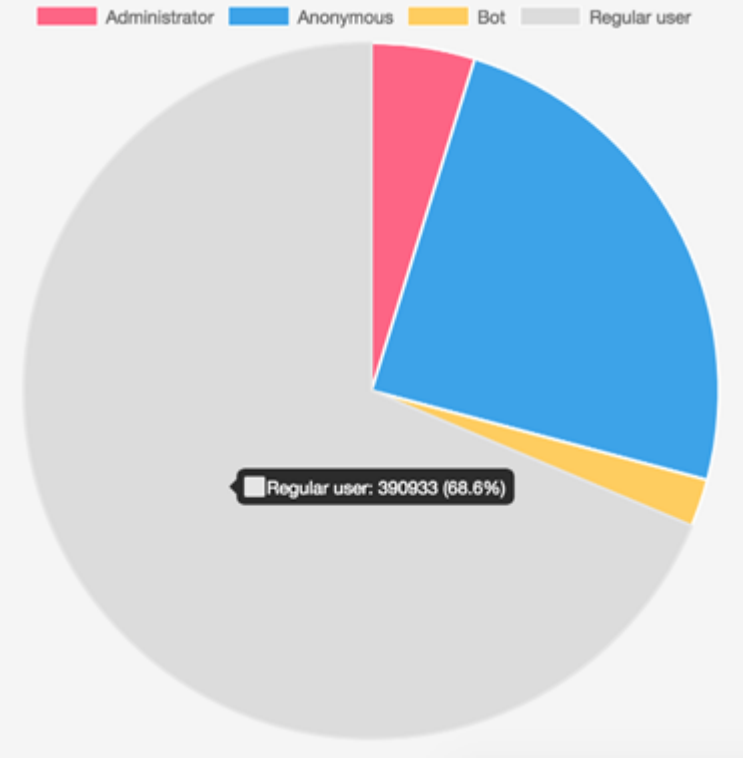

Figure 2: Example pie chart of revision number distribution by user type

We are interested in four types of users: anonymous, administrator, bot and regular user. Revisions made by anonymous users are indicated by the “anon” field in the revision JSON object. Revisions without “anon” field can be made by the other three types of users. You will need to compare the user name with the names in the provided text files to determine if a user is administrator, bot or just regular ones.

The text summary should always be displayed on the page in “Overall” state. You should provide a way for an end user to switch between the two charts. This could be a button or a link.

Individual Article Analytics

The individual page statistics should provide a mechanism, for instance, a simple drop-down list, for your application’s end user to select an article from a list of all available article titles in the data set. You should also use other more user-friendly mechanisms or add features, such as total number of revisions, next to an article title, to assist with the selection. You may also allow end users to enter the first few letters and use autocomplete feature to quickly locate the title of interest.

Once an end user selects an article, your application needs to check if the history of that article in the database is up to date. We consider histories less than one day old as up to date. For instance, if a user has selected article “Australia” at 8:00pm on 21 March 2019 and the latest revision of “Australia” in the database was made on 10:00pm, 20 March 2019; the history is considered as up to date and you do not need to do anything. However, if the latest revision of “Australia” was made on 10:00am, 20 March 2019, the history is considered as not up to date. You need to query MediaWiki API to pull all possible new revisions made after last update.

There should be a message indicating if a data pulling request is made and if so, how many new revisions have been downloaded. It is possible that a data pulling request is made, but the article has no new revision to be downloaded.

For the selected article, display the following summary information:

- The title
- The total number of revisions
- The top 5 regular users ranked by total revision numbers on this article, and the respective revision numbers.

You also need to produce three charts:

- A bar chart of revision number distributed by year and by user type for this article.
- A pie chart of revision number distribution based on user type for this article.
- A bar chart of revision number distributed by year made by one or a few of the top 5 users for this article.

For the last chart, make sure you provide a way to select the users.

You should also provide year filter (i.e., "*From:*" and "*To:*") selection. If users change the year filter, both summary information and charts must be generated based on data from this range of years only.

Figure 3 shows an example of the yearly revision distribution using data in file "*Germany.json*". Figure 4 shows a pie chart of user type distribution for article "Germany". Figure 5 shows a bar chart of the year revision distribution of user "Horst-schlaemma" for article "Germany". These figures are just some illustrations, you can use different color or design to show the chart. The data shown in these figures may not reflect the actual data provided in the json files. The charts should not be displayed in one page, you should provide a mechanism for end users to switch among charts.

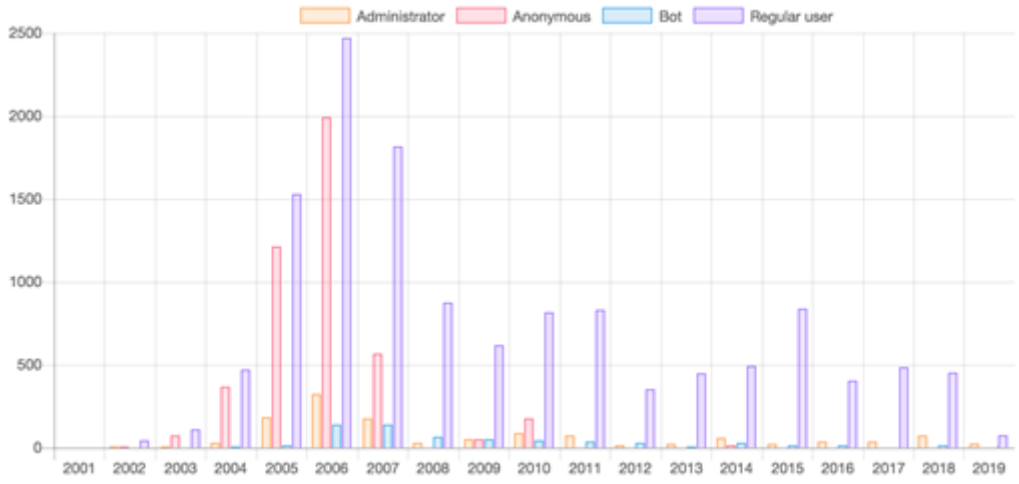

Figure 3: Example bar chart showing yearly revision number distribution

Figure 4: Example Pie chart showing user type distribution

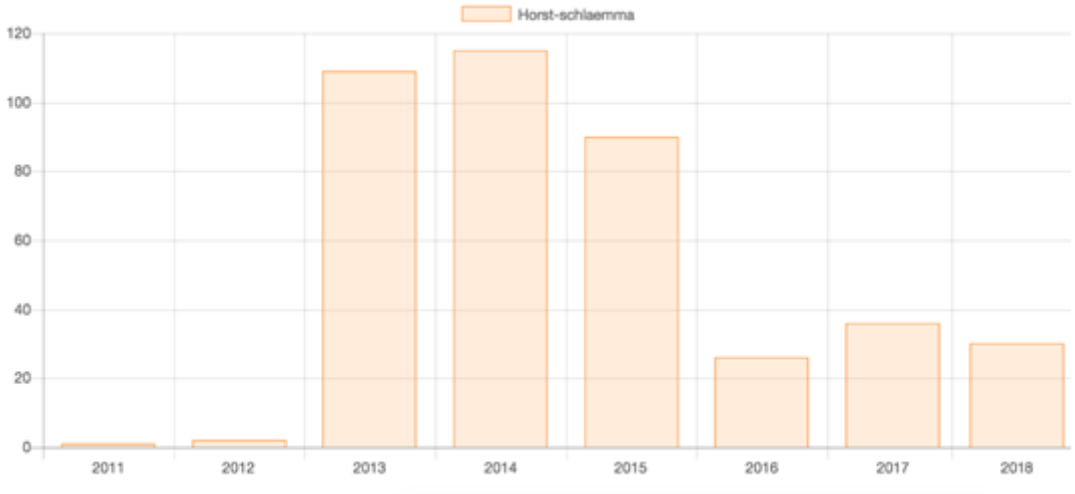

Figure 5: Example single user yearly revision distribution

Author Analytics

In this page, you should enable the end user to display all articles that are changed (or have revisions made) by a specific author. To do so, you should allow the end user to an author name in a free text format.

You should display the articles' names along with number of revisions made by that author next to it. The end user should also be able to select to see time stamps of all revisions made to an article, if that author made more than revision to an article he is attributed with.

Design and Implementation Requirements

Your application should operate on a single page, with all communications between client and server need to be asynchronous. For simplicity, it is allowed to implement the main/landing page functions in a separate page. The other functions (overall, individual and author statistics) must be implemented within a single page (following SPA principles).

You should design your own layout. You must use the MVC pattern to structure your application and interaction among components. The design and UI interfaces should be user-friendly and intuitive.

You should use JavaScript to implement both front and back end of the application. The back-end application should use Node.js framework. The back-end storage system should be MongoDB. You can use other popular JavaScript libraries not covered in this course.

Your application should show good performance when running any functionality. You should consider techniques for optimizing application performance including communications among tiers and database design and implementation. Your design and implementation should not be specific for the provided dataset. You should consider a dynamic design and implementation that work properly and scale with any dataset.

We have created an organization on Sydney Github, you can access it from **this link (https://github.sydney.edu.au/orgs/SOFT5347-Web-Application-Development/dashboard)**. You will also receive a Github invitation via email to join this organization. You must create a new repository for your project and use this Github repository to maintain your code and project. We will use Github to verify each group member's contributions.

Dataset

Dataset for development purpose can be downloaded here: **dataset**

Application Demo

Each group must demo their application during week 13 lab. Note the demo will use another dataset that conforms to structure of the dataset included in this assignment. A demo set will be provided All team members must prepare and participate in the demo. Each team member will be required to demo one part of the application, and they should be able to answer any question about the application design and implementation. In preparation for your demo:

- Expect to discuss edge cases
- Adhere to good user design and user experience principles
- Each member should have complete understanding of how the overall system works. i.e. Someone worked delicately on the frontend must also be able to answer database questions. It is highly recommended to follow the bus factor
- Following good software design principles is recommended
- If you are using external libraries not discussed in the tutorials, discuss it with your tutor. Your tutor maybe unfamiliar with the library you're using. Expect to answer questions about the usage of the library in your assignment. e.g. reasons for using the library and alternative solutions.

Group Member Contribution

If members of your group do not contribute sufficiently you should alert your tutor as soon as possible. The course instructor has the discretion to scale the group's mark for each member as follows:

Level of Contribution	**Proportion of final grade received**
No contribution	0%
Partial/poor contribution	30%-60%
Major contribution	100%

Deliverable and Submission Guideline

Each team must make two submissions through Canvas:

- Application source code: submit a zip file with all your project files to this particular page submission
- Technical report: submit a pdf file of your report on Canvas to **the other assignment page about the technical report submission**

Marking Guide

- Application Design and Implementation (80%)
  - Functional Requirements
    - Main Page
    - Article Analytics – Overall
    - Article Analytics – Individual
    - Article Analytics – Author
  - Design and Implementation Requirements
- Technical Report (20%)

Note: any assignment updates will be made in Canvas and announced via Ed.

Academic honesty

While the University is aware that the vast majority of students and staff act ethically and honestly, it is opposed to and will not tolerate academic dishonesty or plagiarism and will treat all allegations of dishonesty seriously.

Further information on academic honesty, academic dishonesty, and the resources available to all students can be found on the academic integrity pages on the current students website: **https://sydney.edu.au/students/academic-integrity.html (https://sydney.edu.au/students/academic-integrity.html)** .

Further information for on research integrity and ethics for postgraduate research students and students undertaking research-focussed coursework such as Honours and capstone research projects can be also be found on the current students website: **https://sydney.edu.au/students/research-integrity-ethics.html (https://sydney.edu.au/students/research-integrity-ethics.html)** .

Compliance statement

In submitting this work, I acknowledge I have understood the following:

- I have read and understood the University of Sydney's **Academic Honesty in Coursework Policy 2015 (https://sydney.edu.au/policies/showdoc.aspx?recnum=PDOC2012/254&RendNum=0)** .
- The work is substantially my own and where any parts of this work are not my own I have indicated this by acknowledging the source of those parts of the work and enclosed any quoted text in quotation marks.
- The work has not previously been submitted in part or in full for assessment in another unit unless I have been given permission by my unit of study coordinator to do so.
- The work will be submitted to similarity detection software (Turnitin) and a copy of the work will be retained in Turnitin's paper repository for future similarity checking. Note: work submitted by postgraduate research students for research purposes is not added to Turnitin's paper repository.
- Engaging in plagiarism or academic dishonesty in coursework will, if detected, lead to the University commencing proceedings under the **Academic Honesty in Coursework Policy 2015 (https://sydney.edu.au/policies/showdoc.aspx?recnum=PDOC2012/254&RendNum=0)** and the **Academic Honesty Procedures 2016 (http://sydney.edu.au/policies/default.aspx?mode=glossary&word=Academic+honesty)** .
- Engaging in plagiarism or academic dishonesty in research-focussed work will lead to the University commencing proceedings under the **Research Code of Conduct 2013 (https://sydney.edu.au/policies/showdoc.aspx?recnum=PDOC2013/321&RendNum=0)** and the **Academic Honesty Procedures 2016 (http://sydney.edu.au/policies/default.aspx?mode=glossary&word=Academic+honesty)** .
- Engaging another person to complete part or all of the submitted work will, if detected, lead to the University commencing proceedings against me for potential student misconduct under the **University of Sydney (Student Discipline) Rule 2016 (http://sydney.edu.au/policies/showdoc.aspx?recnum=PDOC2017/441&RendNum=0)** .

COMP5347_Demo_Group_Assignment_Rubric

Criteria	Ratings			Pts
(1) Landing Page - General Include web application name: ArticleInsight	**0.25 pts** **Full Marks**	**0.1 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.25 pts
(2) Landing Page - General Users cannot access or use available statistic functions until they create a valid account and login. All users must sign-up/login before they can see and interact with the application's functionality	**0.25 pts** **Full Marks**	**0.1 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.25 pts
(3) Landing Page - Registration A link to register/sign-up a new user page/pop-up from landing page	**0.25 pts** **Full Marks**	**0.1 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.25 pts
(4) Landing Page - Registration Provide a form that contains first name, last name, email address (as username), and password.	**0.25 pts** **Full Marks**	**0.1 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.25 pts
(5) Landing Page - Registration Provide a validation either automatic or when the submit button is clicked for email field.	**0.25 pts** **Full Marks**	**0.1 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.25 pts
(6) Landing Page - Registration Provide a validation either automatic or when the submit button is clicked for password field (cannot be empty and should be hidden from users – using asterisk ***)	**0.25 pts** **Full Marks**	**0.1 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.25 pts
(7) Landing Page - Login A link to login page/pop-up from landing page	**0.25 pts** **Full Marks**	**0.1 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.25 pts
(8) Landing Page - Login User should be able to login successfully (use the previous registered user)	**1.0 pts** **Full Marks**	**0.4 pts** **Partial Marks**	**0.0 pts** **No Marks**	1.0 pts
(9) Landing Page - Login A page with access to the main wiki analytic features are available after login	**0.25 pts** **Full Marks**	**0.1 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.25 pts
(10) Landing Page - Login It should handle session well after login.	**0.5 pts** **Full Marks**	**0.2 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.5 pts

Criteria	Ratings			Pts
(11) Landing Page - Logout A link to logout	**0.25 pts** **Full Marks**	**0.1 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.25 pts
(12) Landing Page - Logout User should be able to logout successfully (use the previous registered user)	**0.25 pts** **Full Marks**	**0.1 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.25 pts
(13) Landing Page - Logout It should handle session well after logout.	**0.5 pts** **Full Marks**	**0.2 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.5 pts
(14) Overall Analytics - Title summaries Shows titles of the two articles with highest number of revisions (2 items by default)	**0.25 pts** **Full Marks**	**0.1 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.25 pts
(15) Overall Analytics - Title summaries Shows titles of the two articles with lowest number of revisions (2 items by default).	**0.25 pts** **Full Marks**	**0.1 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.25 pts
(16) Overall Analytics - Title summaries Show titles edited by largest group of registered users (2 items by default).	**0.25 pts** **Full Marks**	**0.1 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.25 pts
(17) Overall Analytics - Title summaries Show titles edited by smallest group of registered users (2 items by default).	**0.25 pts** **Full Marks**	**0.1 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.25 pts
(18) Overall Analytics - Title summaries 18. Show titles with the longest history measured by age (2 items by default).	**0.25 pts** **Full Marks**	**0.1 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.25 pts
(19) Overall Analytics - Title summaries Show titles with the shortest history measured by age (2 items by default).	**0.25 pts** **Full Marks**	**0.1 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.25 pts
(20) Overall Analytics - Title summaries User is able to change the number of articles for highest and lowest number of revisions; the same number should be applied to all of above.	**0.25 pts** **Full Marks**	**0.1 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.25 pts
(21) Overall Analytics - Overall charts A bar chart of revision number distribution by year and by user type across the whole dataset	**1.0 pts** **Full Marks**	**0.4 pts** **Partial Marks**	**0.0 pts** **No Marks**	1.0 pts

Criteria	Ratings			Pts
(22) Overall Analytics - Overall charts A pie chart of revision number distribution by user type across the whole data set.	**1.0 pts** **Full Marks**	**0.4 pts** **Partial Marks**	**0.0 pts** **No Marks**	1.0 pts
(23) Overall Analytics - Overall charts If user's mouse hovers or clicks to the administrator part of the pie chart, it must show the distribution of each administrator type (active, semi-active, inactive, and former)	**0.5 pts** **Full Marks**	**0.2 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.5 pts
(24) Overall Analytics - Overall charts User is able to switch between the two charts (this could be a button or a link).	**0.25 pts** **Full Marks**	**0.1 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.25 pts
(25) Individual Article Analytics - General User is able to select an article from a list of all available article titles in the dataset if it is a drop-down menu. The page has use autocomplete feature to quickly locate the title of interest for searching an article if it's an input text. If the title is not found, it must show an error message or pop-up.	**0.5 pts** **Full Marks**	**0.2 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.5 pts
(26) Individual Article Analytics - General The total number of revisions will be shown in the drop-down list beside the title. If the page uses text input, it should show the number of revisions somewhere beside the input	**0.5 pts** **Full Marks**	**0.2 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.5 pts
(27) Individual Article Analytics - Charts After selecting one article, it must show the following information: Total number of revisions, Top 5 regular users, A bar chart of revision number, A pie chart of revision number, and A bar chart based on top users	**1.25 to >1.0 pts** **Full Marks**	**1.0 to >0.0 pts** **Partial Marks**	**0.0 pts** **No Marks**	1.25 pts
(28) Individual Article Analytics - Charts For the last bar chart, user should be able to select one or more of top 5 users to show the graph. Note: If the selection is drop-down then it will only show one user at a time. If the selection is checkbox, user should be able to select more than 1	**0.5 pts** **Full Marks**	**0.2 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.5 pts

Criteria	Ratings			Pts
(29) Individual Article Analytics - Charts If selecting article that is not up-to-date with no new revisions, it must show that the page is loading data from Wikipedia API but there is no available data	**0.25 pts** **Full Marks**	**0.1 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.25 pts
(30) Individual Article Analytics - Charts If selecting article that is not up-to-date with new revisions available, it must show that the page is loading data from Wikipedia API and inform that N revisions have been collected	**0.25 pts** **Full Marks**	**0.1 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.25 pts
(31) Author Analytics - General User is able to select an author name in a free text format (input text). The page is able to handle boundary cases	**0.5 pts** **Full Marks**	**0.2 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.5 pts
(32) Author Analytics - General After inputting author name, it must show this list of articles with the number of revisions made	**0.5 pts** **Full Marks**	**0.2 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.5 pts
(33) Author Analytics - General User must be able to select an article from the previous list either from drop down or any appropriate selection input. After selected, it will show all timestamps revisions made for that particular article	**0.5 pts** **Full Marks**	**0.2 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.5 pts
(34) Design and Implementation Requirements The application must operate on a single page (SPA principles), with all communications between client and server need to be asynchronous no reload/refresh page except the main/landing that includes login/register.	**1.0 pts** **Full Marks**	**0.4 pts** **Partial Marks**	**0.0 pts** **No Marks**	1.0 pts
(35) Design and Implementation Requirements The page has reasonably good UI and be user-friendly user interaction.	**1.0 pts** **Full Marks**	**0.4 pts** **Partial Marks**	**0.0 pts** **No Marks**	1.0 pts
(36) Design and Implementation Requirements The page has good performance when running any functionality, e.g., loading the data quick and reasonably fast. The latency when switch between overall page and individual page should not be too long (less than one minute). When switch between different charts, the latency should be smaller (several seconds)	**1.0 pts** **Full Marks**	**0.4 pts** **Partial Marks**	**0.0 pts** **No Marks**	1.0 pts

Criteria	Ratings			Pts
(37) Design and Implementation Requirements The website adapts MVC design pattern, explain how the page implements MVC	**1.0 pts** **Full Marks**	**0.4 pts** **Partial Marks**	**0.0 pts** **No Marks**	1.0 pts
(38) Understanding and Teamwork The group used Github appropriately, everyone has access the github and did several commits	**0.25 pts** **Full Marks**	**0.1 pts** **Partial Marks**	**0.0 pts** **No Marks**	0.25 pts
(39) Understanding and Teamwork Able to answer questions related to code understanding and teamwork	**2.0 pts** **Full Marks**	**0.8 pts** **Partial Marks**	**0.0 pts** **No Marks**	2.0 pts
				Total Points: 20.0