**Software Engineering Standards**

**System Requirements and Analysis Specification (RAS)**

**Social Network Analysis with Deep Learning**

**Version 1.2**

Document Number: RAS-001

Project Team Number: A9

Project Team Members: Anish Malhorta (am7161)

Chun Zen Marc Tse (czt210)

Mehmed Mladenov (mm7277)

**REVIEW AND APPROVALS**

|  |  |  |  |
| --- | --- | --- | --- |
| **Printed Name and**  **Title** | **Function**  **(Author,**  **Reviewer,**  **Approval)** | **Date** | **Signature** |
| Anish Malhorta | Author | 2/25/2018 | Anish Malhorta |
| Chun Zen Marc Tse | Author | 2/25/2018 | Chun Zen Marc Tse |
| Mehmed Mladenov | Author | 2/25/2018 | Mehmed Mladenov |
| Professor | Reviewer | 2/25/2018 | Professor |
| Anish Malhorta | Author | 3/6/2018 | Anish Malhorta |
| Chun Zen Marc Tse | Author | 3/6/2018 | Chun Zen Marc Tse |
| Mehmed Mladenov | Author | 3/6/2018 | Mehmed Mladenov |
| Professor | Reviewer | 3/6/2018 | Professor |
| Anish Malhorta | Author | 4/12/2018 | Anish Malhorta |
| Chun Zen Marc Tse | Author | 4/12/2018 | Chun Zen Marc Tse |
| Mehmed Mladenov | Author | 4/12/2018 | Mehmed Mladenov |
|  |  |  |  |

**REVISION LEVEL**

|  |  |  |
| --- | --- | --- |
| **Date** | **Revision Number** | **Purpose** |
| Feb 25, 2018 | Version 1.0 | Initial Release |
| March 6, 2018 | Version 1.1 | Fixed Defects from V 1.0, Added Section 7 and 8 |
| April 13, 2018 | Version 1.2 | Fixed Defects from V 1.2, Added Sections 9-13 |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Table of Contents

1. INTRODUCTION1

1.1 Purpose1

1. SCOPE1

2.1 Identification1

2.2 Bounds1

2.3 Objectives1

1. OVERALL system overview2

3.1 Context Diagram2

3.2 Additional Descriptive Items2

1. Document Overview3
2. Reference Documents4
3. business Requirements4

6.1 Technology4

6.2 Economy4

6.3 Regulatory and Legal4

6.4 Market Considerations4

6.5 Risks and Alternatives5

6.6 Human Resources and Training5

1. specific requirements (descriptive functional requirements)6

7.1 Functional Descriptive Detailed Requirements6

7.2 Requirement Use Cases6

7.3 Use Cases Diagrams7

1. non-functional descriptive detailed requirements (non-functional requirements defintions)10
2. Analysis10

9.1 Component (Component/Package/Subsystem) Architecture10

9.2 Component Architecture Diagram11

9.3 Component Descriptions11

9.4 Class Diagrams13

9.5 Events13

9.6 Activity/State (Scenario) Section14

9.7 State Logic22

9.8 Behavior22

1. System test plan requirements23
2. Qualification provisions23
3. requirements traceability24
4. evolution of the ras24
5. rationale24
6. notes25
7. appendices25

Schedule Tracking25

Defect Tracking27

Dictionaries28

1. **INTRODUCITON**
   1. **Purpose**

The purpose of this document is mainly to define the domain, requirements and analysis for the project and the entirety of its development cycle. The document provides an overview of the objective of our team’s software and other facts pertaining to the use of the software, as well as the technical requirements for the software to be developed and operated successfully.

The intended audience for this document is mainly but not limited to, the client of the software product, the management level personnel involved with the software team, developers of the software, and the testers and end-users of the software.

1. **SCOPE**

There is a lack of simple, efficient and modularized support for determining the efficacy of marketing campaigns by operating on different data sets and performance metrics. The system will address these concerns by allowing the plugging-in of various data sets and the standards used to determine their success, as well as predict future performance. The system will provide an easy-to-use interface for this purpose, which will show results of big data analytics in visual formats, such as GIS maps, charts and spreadsheets. The system, and the algorithm that supports it, will be fine-tuned to support modularity for a variety of data sets that may be used with it as well as the performance indicators used to measure potential success. Such a system would have tremendous benefit for many organizations, simplifying the process of using Deep Learning greatly while also reducing the cost of implement such techniques.

* 1. **Identification**

Social Network Analysis with Deep Learning Requirements and Analysis Specification Version 1.1.

* 1. **Bounds**

The system boundary includes the software of the social network analysis itself, the boundary between end-users (researchers, data scientists, marketers in companies) and the software and the boundary between the software and external databases and data metrics from other companies (social media companies, digital marketing companies, in house data from client).

* 1. **Objectives**

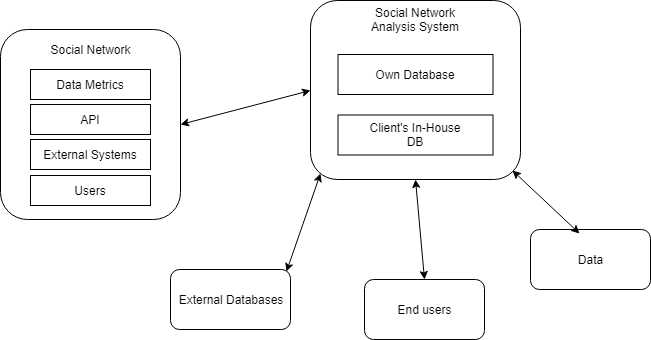
The highest priority objectives are the optimization of the algorithm necessary to perform analytics on data sets, as well as the actual representation of the analysis. As such, the visual component of the system will be crucial – making certain that the customer has the several interchangeable way to view the analytics they will use the system for. Making the system simple to use will also be a high priority deliverable, with a functional and self-descriptive dashboard being necessary. Another important, but lower priority objective will be modularization of the algorithm itself. Being able to use different data sets is an important long-term goal of the system, but not before the structure has been created for a general use case.

Because the system will require constant testing of the algorithm itself on various data sets, the project will run incrementally at each step of production, with texting and requirements analysis being done at every stage of development for maximum productivity and time-line benefit.

Initial Deliverables:

* Project Proposal: 2/08/18
* Software Requirements and Analysis Specification (RAS):
  + - Business & Project Definition: 2/20/18
    - Software Requirements: 3/06/18
    - Complete: 3/20/18
    - Software Analysis Specification: 4/17/18
* Software Project Management Plan (SPMP): 4/05/18
* Software Design Document: 5/01/18
* Presentation: 4/26/18 – 5/03/18

1. **OVERALL SYSTEM OVERVIEW**
   1. **Context Diagram**

****

* 1. **Additional Descriptive Items**

Product Functions

* Collect large amounts of data on social network
* Analyze the sentiment of general public towards a specific topic
* Predict a trend towards a topic

User Characteristics

* Product will be used by marketing teams of firms, data scientists and researchers
* End-users can range from being not highly proficient with computers to extremely technical, but in general should know how to interact with a well-designed UI
* End-users is expected to know how to read and input data

Constraints

* If the software product is not allowed to input data from social network companies
* Computer processing power is not enough to compute the result of a large amount of data
* Public databases that are available for data mining may not have sufficient types of data
* Development language can be exploited to create a security loophole

Assumptions and Dependencies

* Availability and quality of the data types to be processed
* The optimization of analysis and prediction algorithm

Requirement Subsets

* User interface and general user-friendly control mechanism could be delayed until the algorithm of processing the data is done

1. **Document Overview**

The document is organized in such a way to first provide a top-level description and analysis of the need for this system as well as the general solutions being used for implementation in section 1 to 3. In section 6, the document further delves into the business applications of the system – the economic and technological drivers utilized by the system in order to be effective. The functional and non-functional requirements of the system are detailed in section 7 and 8 respectively and the analysis of these requirements is provided below in section 9. The manner of system testing in section 10 is described for the purpose of finding defects. In section 11 and 12, the document describes its own examination and validity, as well as the validity of each individual artifact stemming from its original functional requirement. In section 13 and 14, the document details the iterative approach taken in designing the system chronologically as it evolves, with new revisions of this document and their differences being described later. The appendix serves as a method of tracking schedule and defects encountered when designing the system.

1. **Reference Documents**

“Team A9 Project Proposal” Social Network Analysis with Deep Learning, A9, Version 2.0, February 16, 2018.

“Team A9 Software Project Management Plan” Social Network Analysis with Deep Learning, A9, Version 1.0, April 3, 2018.

1. **Business Requirements**
   1. **Technology**

* Big data analytics helps support the business objective in easily scalable metrics for clients to improve upon their strategies. Being able to analyze large quantity of data can help gauge a better picture of consumers’ information and data related to them.
* Machine learning helps give a direction for clients in terms of how they can improve upon their strategies. Being able to predict a trend within social media can help clients create buzz-worthy marketing campaigns and generate useful leads for business.
  1. **Economics**
* Main financial income driver for the system is the motivation for clients to use the software to find the right targeted audience for their products/services, information about their consumers in order to optimize marketing strategies and fine tune companies’ products/services.
* Mode of financial income can be a subscription/service based model, where we act as a “consultant” and provide valuable insights and metrics to companies based on data sets available and optimize marketing strategies for a monthly or annual fee.
  1. **Regulatory and Legal**
* Privacy laws
* System manipulation and fraud
  1. **Market Considerations**
* Established names in the field, for example Google, who already does ad/marketing analytics and provides data to companies.
* Lots of competitions, especially because of the recent rise of popularity in big data and machine learning
* Lots of investors and VCs to provide funding for startups in the field, because of the popularity
  1. **Risks and Alternatives**

|  |  |
| --- | --- |
| Business Risk: | Limited access to data from companies |
| Probability: | Medium |
| How discovered: | Other companies limit use of data |
| Responsible Party: | Management, Legal |
| Status: | Not found yet |
| Mitigation Plan: | Deal making, benefits to companies |

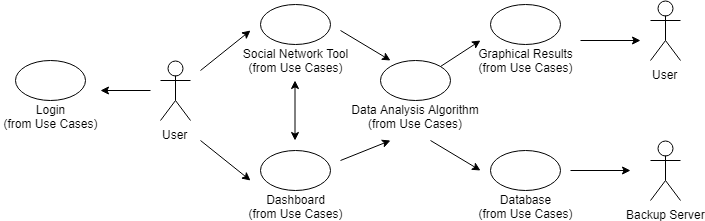
|  |  |
| --- | --- |
| Operational Risk: | Project Delays |
| Probability: | Medium |
| How discovered: | Schedule Tracking |
| Responsible Party: | Developers, PM |
| Status: | Not found yet |
| Mitigation Plan: | Have meetings to follow up with dev |

|  |  |
| --- | --- |
| Technology Risk: | Not enough quality data |
| Probability: | Medium |
| How discovered: | After getting results from data analysis |
| Responsible Party: | Developers |
| Status: | Not found yet |
| Mitigation Plan: | Extract more specific data types |

|  |  |
| --- | --- |
| Economic Risk: | Market too niche |
| Probability: | Low |
| How discovered: | Not many people willing to switch to system |
| Responsible Party: | Designers |
| Status: | Not found yet |
| Mitigation Plan: | Adopt more traditional elements |

* 1. **Human Resources and Training**
* Developers should have knowledge in data science, machine learning and statistics
* UI developers should have knowledge in front-end development and UI/UX design

1. **Specific Requirements (Descriptive Functional Requirements)**
   1. **Functional Descriptive Detailed Requirements**
2. System must support user login
   1. Passwords will be system generated
   2. Passwords can be changed using 2-stage authentication
   3. It will authenticate user credentials
3. The system must display system menu of projects
   1. User can create a project
   2. User can update a project
   3. User can delete a project
4. Data Analysis Algorithm
5. Database
   1. **Requirement Use Cases**
6. **Login**
7. User enters username and password
8. System authenticates credentials
9. If fails, system prompts user to retry. If users fail too many times, system will trigger mechanisms to ensure user is not bot (captcha for example)
10. Users can reset credentials in case forgotten.
11. **Dashboard**
12. Users can manage projects
    1. Open existing projects
    2. Create new projects
    3. Delete existing projects
13. Users can manage data to be used for analysis
    1. Upload custom data sets
    2. Create new data sets
    3. Organize and edit data sets
14. Users can access options to configure UI, output results, etc.
15. Users can access visual representations of data
16. **Social Networks Tool**
17. Users can search for tags and parameters to build a data set
18. Users can search through demographics of social network for data sets
19. Save data sets
20. **Graphical Results**
21. Users can export results to other file types
22. Users can save results
23. **Data Analysis Algorithm**
24. Users’ data sets is processed
25. If fails, prompts users to adjust data sets
26. Outputs results and predict trend
27. **Database**
28. Database is used for storing user credentials, projects, analysis parameters and results
29. System admin can access database to edit information inside database
    1. **Use Case Diagrams**



**Use Case Descriptions**

|  |  |  |  |
| --- | --- | --- | --- |
| **Login** | | | |
| **Description** | Primary access point for user to use the system | | |
| **Pre-Condition** | Permission to gain access | | |
| **Flows** | **Basic or Normal Flows** | 1 | User passing a lock mechanism to gain access |
| 2 |  |
| 3 |  |
|  | **Alternative Flows** | 1 |  |
| 2 |  |
| 3 |  |
| **Post Conditions** | Allows users to either create new projects or gain access to saved projects, or create new data sets | | |
| **Special Requirements** | None | | |
| **Extension Points** | Leads to Dashboard or Social Network Tool | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Dashboard** | | | |
| **Description** | Allows users to upload custom data sets/Begin new project/Continue from saved projects | | |
| **Pre-Condition** | Have already passed through login mechanism | | |
| **Flows** | **Basic or Normal Flows** | 1 | Delete a project |
| 2 | Create a blank project |
| 3 | Open past project |
|  | **Alternative Flows** | 1 | Import custom data sets from Social Network Tool |
| 2 | Go to Social Network Tool to create data sets |
| 3 |  |
| **Post Conditions** | Construct clear configurable metrics to users for analysis | | |
| **Special Requirements** | None | | |
| **Extension Points** | Leads to Social Network Tool or Data Analysis Algorithm | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Social Networks Tool** | | | |
| **Description** | Allows user to create tags that provide search parameters for social networks to build a data set | | |
| **Pre-Condition** | Have already passed through login mechanism/Created a blank project and attempt to build data sets | | |
| **Flows** | **Basic or Normal Flows** | 1 | Transition from dashboard to create data set for use |
| 2 | Modify data sets for projects |
| 3 |  |
|  | **Alternative Flows** | 1 | Failed to build data, retry |
| 2 |  |
| 3 |  |
| **Post Conditions** | Data set is built for projects to use/Failed to build data sets, requires more specific search parameters | | |
| **Special Requirements** | Search parameters must be feasible for social network API | | |
| **Extension Points** | Transition to Data Analysis Algorithm or Dashboard | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Data Analysis Algorithm** | | | |
| **Description** | Make use of tags and data sets to generate trends and useful metrics for users | | |
| **Pre-Condition** | Project has been created to go through analysis | | |
| **Flows** | **Basic or Normal Flows** | 1 | Transition from dashboard/social network tool to do analysis |
| 2 |  |
| 3 |  |
|  | **Alternative Flows** | 1 | Data analysis has failed |
| 2 | Back to social network tool/dashboard to change metrics |
| 3 |  |
| **Post Conditions** | Notification to let users know whether analysis has failed or not/Provide instructions | | |
| **Special Requirements** | Data sets are specific enough to generate useful metrics | | |
| **Extension Points** | Transition to Graphical Results and Database | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Graphical Results** | | | |
| **Description** | Provide visual data based on various insights, tags and demographics | | |
| **Pre-Condition** | Data analysis algorithm has been run | | |
| **Flows** | **Basic or Normal Flows** | 1 | Data analysis produces visual data based on metrics input |
| 2 | Transition back to dashboard to edit project for other analyzes |
| 3 | Allows users to download results |
|  | **Alternative Flows** | 1 |  |
| 2 |  |
| 3 |  |
| **Post Conditions** | Graphical results need to be visually interpretable/Lets users know when analysis fails | | |
| **Special Requirements** | Data analysis has been executed correctly | | |
| **Extension Points** | Back to users | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Database** | | | |
| **Description** | Store data sets, projects, analysis parameters and results | | |
| **Pre-Condition** | Data analysis algorithm has been run | | |
| **Flows** | **Basic or Normal Flows** | 1 | Data analysis produces results to be saved |
| 2 | Users save data to database |
| 3 |  |
|  | **Alternative Flows** | 1 | Automatically save projects when user is in dashboard |
| 2 |  |
| 3 |  |
| **Post Conditions** | Provides indication that data has been saved | | |
| **Special Requirements** | Have access to backup server | | |
| **Extension Points** | Back to users | | |

1. **Non-Functional Requirements Definitions**
2. **System Capabilities**

The platform will be cloud based and saved on our servers, which would make it available for users with any hardware characteristics to have access.

1. **Security**

The platform will provide the users with the ability to control the access to their own data. This includes controlling who may view and alter data uploaded to the platform. The database would be encrypted with Advanced Encryption Standard (AES), which is one of the standard encryption algorithms by the U.S. Government. Also, extra layer of protection will be added by providing users with Secure Hash Algorithm-2 (SHA-2). Two-factor authentication (2fa), will be added as optional configuration for users with sensitive data uploaded to our database. Supported 2fA third party apps will be limited to *Google Auth, Authy and Duo*, since these are the most trusted authentication apps.

1. **Performance/Availability**

The database would not save any parameters for developing visual results. These parameters will be input by the user and/or calculated real time in order to save space on the platform servers, which would allow increased space usage for detailed searches, with the cost of time. However, the platform uses several of the fastest sorting algorithms with worst case time complexity of O(n log n), which compensates the time used by the parameters.

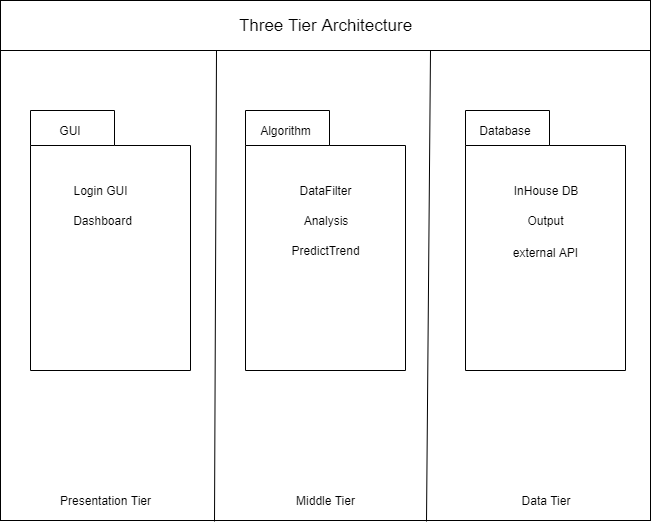
1. **Recoverability/Backup**

The user would be provided with storage on our cloud servers to save their data. The platform will also automatically make daily backups of this data on an offline server and provide users with a private key or a secrete seed, so they can retrieve their data securely on other account in case of hack. If the data is retrieved on another account, it would be deleted from the old one. If any of these keys are lost, the data becomes irretrievable. The backups will be synchronized daily and be available for downloads on users’ devices.

1. **Analysis**
   1. **Component (Component/Package/Subsystem) Architecture**

Three tier architecture is used for the design of system of application – Presentation, Middle and Data tier. Presentation tier includes all components of the system that directly interacts with the user. GUIs such as login and dashboard is included in the Presentation tier. Middle tier includes all component of the system that processes information passed from users and directs to the database. Algorithms such as data analysis, prediction of trend, and data filter is included in the Middle tier. Data tier includes all components of the system that deals with the management and direct control of data in own database, as well as the handling of external data from APIs of external sources. In-house database, external API, output is included in the Data tier.

* 1. **Component Architecture Diagram**



* 1. **Component Descriptions**

*Login GUI*

The login will provide access to the dashboard, which will serve as the main primary point of access for the client in order to use the Social Network Analysis System. The login screen will require the users to enter their user credentials, and after validated, will be transported to the Dashboard.

*Dashboard*

The dashboard will provide a clear mechanism for the user to upload custom data sets in a new project to satisfy the purpose of modularity. Users can create new projects or delete existing projects. The dashboard will allow the user to continue analysis on a previous project and will clearly present the user configurable metrics for an analysis they have ran on a set of data. The dashboard will allow the user to access visual representations of data.

*DataFilter*

The filter should be applied to data collected from external sources through external APIs, as well as the

*Analysis*

The algorithm will make use of tags and demographics set by the user. The algorithm must provide non-trivial analysis based on trends that may not be prevalent to the user. The algorithm must provide various perspectives with which the user can view the data in the ‘Graphical Results’ and ‘Dashboard’ contexts.

*PredictTrend*

The algorithm will be built adaptively in order to use deep-learning techniques to predict a trend after analysis.

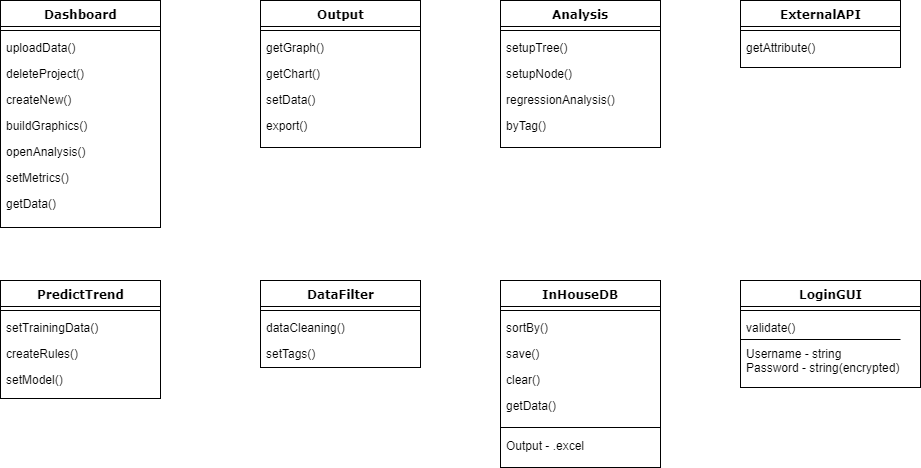
*In-house DB*

The database will store user credentials, project information, analysis parameters and the results of running the algorithm. The database will not store graphical results built in the ‘Graphical Results’ context in order to save space and time, as an optimization trade-off. The charts and visual representations of analytics will be built in real time with the idea that users will be editing the points of focus in this analysis for each project multiple time.

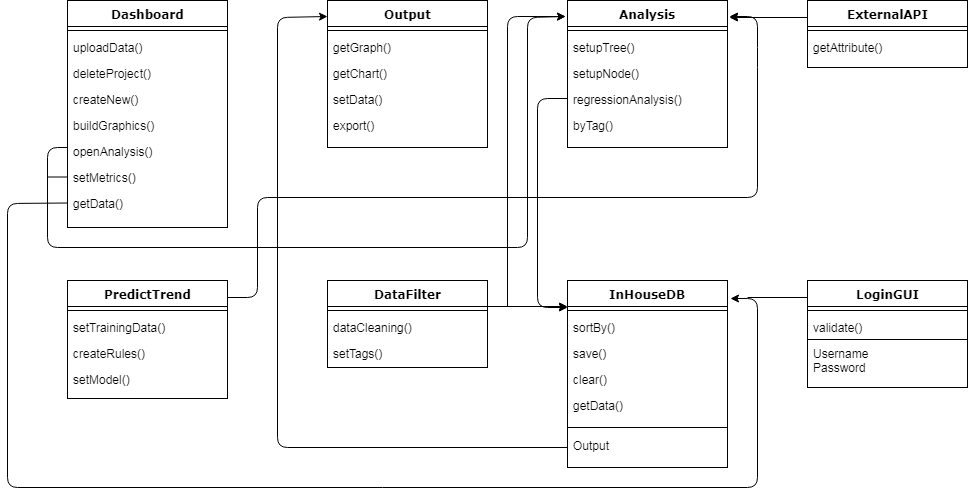
*External API*

Tags will be searched for throughout widely-used Social Media platforms such as Twitter, Facebook and Instagram through API integration. User will then be provided with demographics they can use to organize data, such as age, location, density. Each query resulting from specification of a tag or demographic can be added to iteratively build a customized data set for analysis.

* 1. **Class Diagrams**
     1. **Individual Class Diagrams**



* + 1. **Class Relationship/Interaction Diagrams**

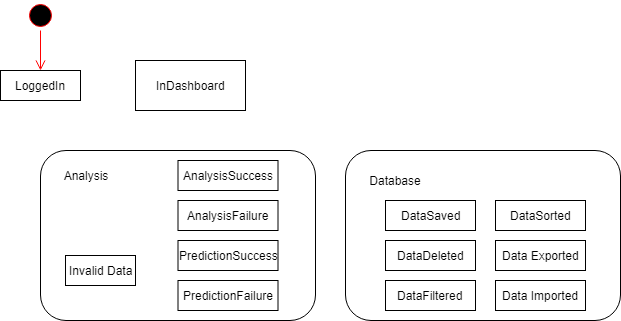


* 1. **Events**
     1. **Motives**

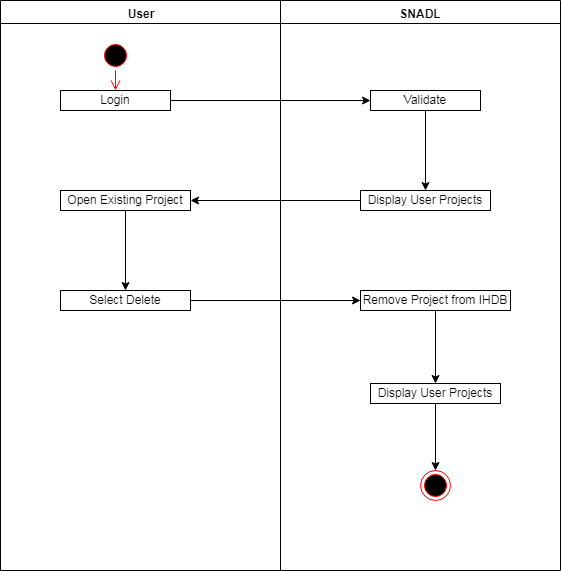
*Run analysis on a data set*

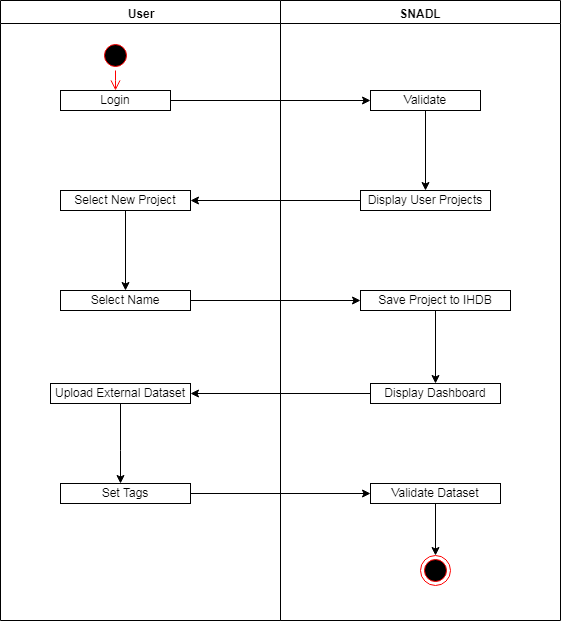
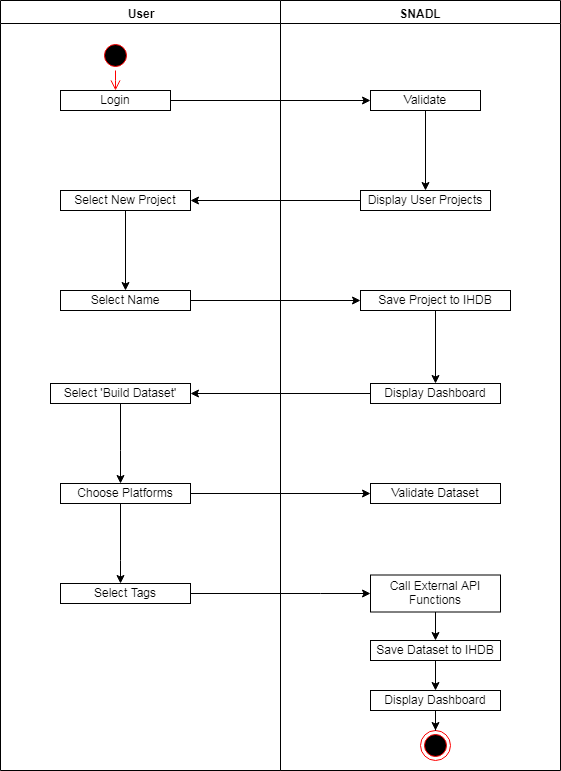
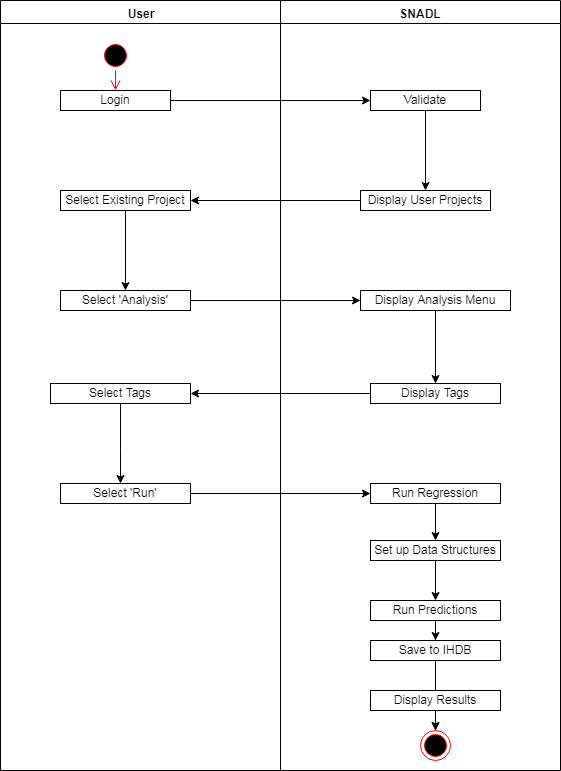
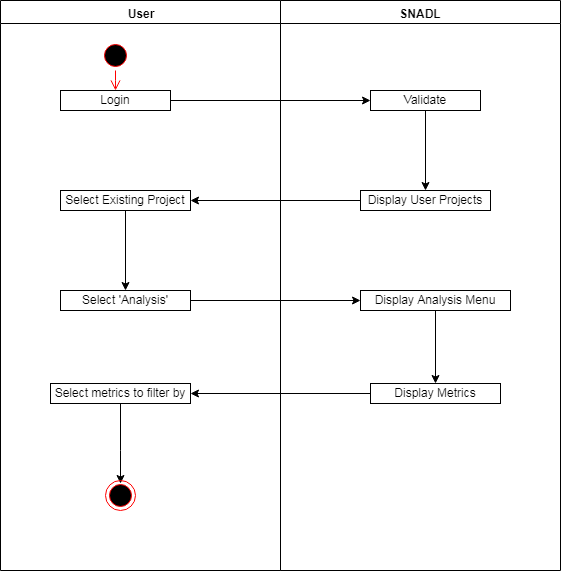
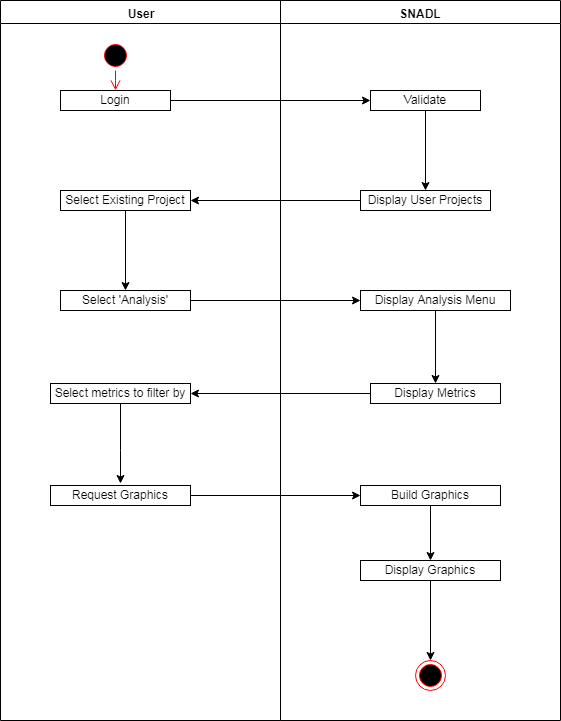
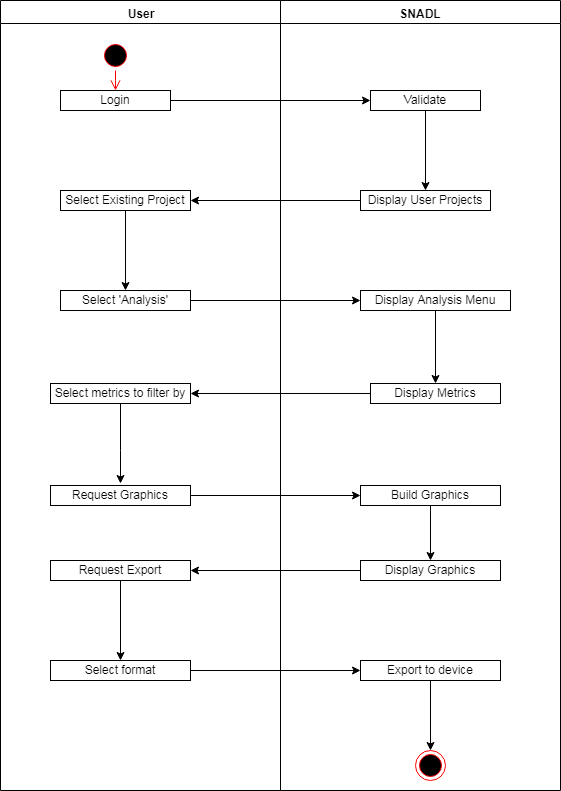
The purpose of the motive is to extrapolate information of users on social media platform based on data tags provided by user of system, as well as obtaining a prediction of trend. The user first creates or opens project, then the user should provide metrics for the system to analyze. Then the system runs the algorithm for data analysis and prediction. The algorithm outputs results. The results are saved to the database, and a graphical representation of the result is shown in the GUI. The user can then choose to export the result into other formats.

* + 1. **Event Diagrams**

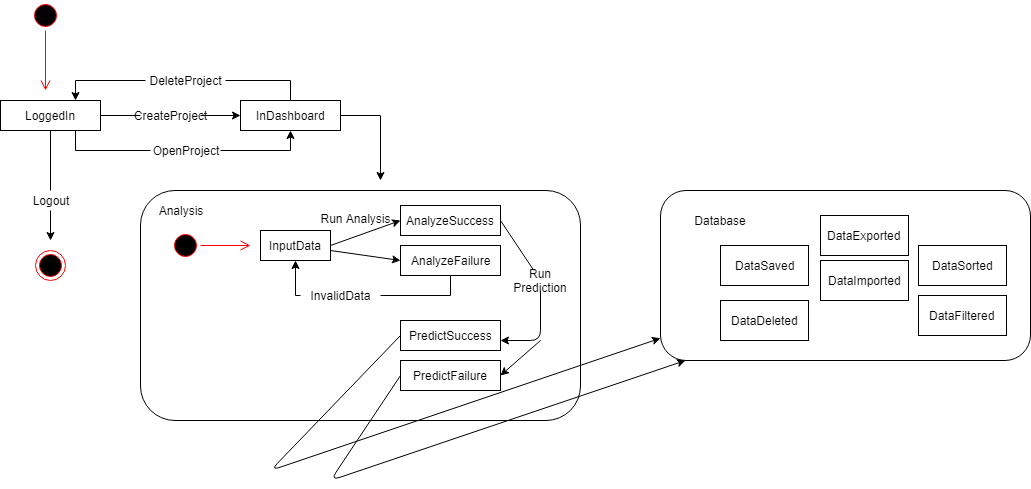


* 1. **Activity/State (Scenario) Section**

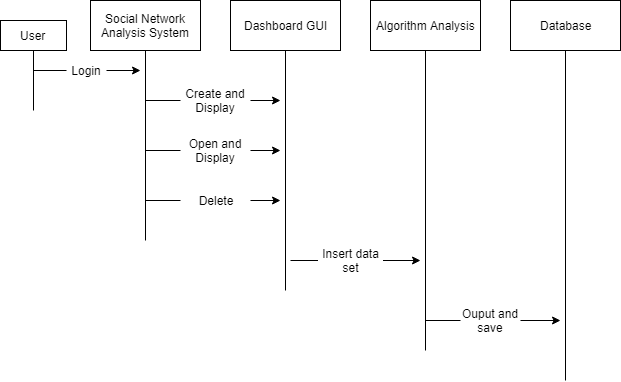


* 1. **State Logic**



* 1. **Behavior**
     1. **Sequence Diagrams**



* + 1. **Collaboration Diagrams**

To be completed in design.

1. **System Test Play Requirements**

Verification and validation will be done with unit and acceptance tests. Verification is done between team members through daily SCRUM meetings, followed by formal weekly meetings. Prototyping of application and modeling of algorithm used in application will be heavily used for the verification process of development. At the beginning of each iteration of development of application, a prototype of the core priority of development should be created in order to verify the feasibility of the application. The prototyping process should not take longer than twelve hours, and in extreme cases where development is lagging behind schedule, prototyping process should not be used. Instead, a less labor intensive method such as traceability of metrics is applied. Our customers will help in the validation process with validating if requirements and tasks are being satisfied. This can be done in weekly formal meetings, which will include demonstrations of baseline iteration of core function of product and project status.

Testing of the validity of the results of the algorithm will be done using historical data in order to verify analysis results. This will be the basis of testability in which previous marketing or advertisement campaigns will be reviewed holistically ad algorithmically with the results compared. Other features of the SNADL, such as the graphics building section can be reviewed similarly by verifying against historical data. The remaining features require simple testability, in that they can be easily validated by test use of their functionalities, such as the GUI.

The configuration manager will be responsible for quality assurance in verifying that deliverables meet the requirements specified in this document.

1. **Qualification Provisions**

This document will be reviewed for quality through the configuration management plan, referenced in the Software Project Management Plan. Weekly forma meetings occur in which the configuration evaluation, identification and release management is handled by all members of the team, and supervised by the Configuration Manager in order to ensure quality in this document. Each requirement is examined for traceability, correctness, ambiguity, completeness and consistency in order to ensure quality software with minimal defects that remains on schedule. The customer maintains an important role in verifying this document in reviews alongside the Configuration Manager at every baseline.

The types of testing used to verify quality in this document are:

1. Peer review
2. Inspections
3. Walkthroughs
4. Unit Testing

Peer reviews involve all members of the team reviewing the document before its release during the weekly formal meetings. The process should be a collaborative effort to find as many defects as possible in the document. Internal and external inspections of the document will also be employed to ensure quality in the document. Internal inspections involve members of team ensuring that the document meets IEEE standards, as well as ensuring the document meets the requirements of document set forth by the RAS guideline provided by Software Engineering class of NYU Tandon. External inspections involve communications with and inspections by customers to ensure that all requirements of document is met. Inspections by TAs and professor of Software Engineering class will also be conducted after each milestone of document to ensure that the standard is properly met.

1. **Requirements Traceability**

All components created during the project development can be traced from the initial source. In the first release (Version 1.0) of the Requirements and Analysis Specification (RAS), the source for traceability was the project proposal. But with every modified version of the RAS document, more sources of traceability is provided. These release changes will be indicated in Section 2.1.

1. **Evolution of the RAS**

The Requirement and Analysis Specification (RAS) document will be updated at every release. The initial release could contain sections which are not applicable at the certain document version, but with every newly created version of the documentation, these sections will be modified. This could also affect the sections which already have information about the project. Each release will be stored in GitHub repository, which would allow traceability and will prevent any data loss. Since all the data will be stored in a cloud server repository, this will allow developers to trace if any error in the new releases. Every release should be noted by a unique version number that increments with each modification. (Version = x.y, where x is only changing with major updated and y is indicating the number of modifications and release of the current x).

1. **Rationale**

*None*

1. **Notes**

*None*

1. **Appendices**

**Schedule Tracking**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (Individual or Team) | Estimated | Actual | Difference |
| RAS Business & Project Definition | Anish | 5 Hours | 6.5 Hours | 1.5 Hours |
|  | Mark | 3 Hours | 6.5 Hours | 3.5 Hours |
|  | Mehmed | 7 Hours | 6.5 Hours | -0.5 Hours |
|  | Summary (People Hours) | 15 Hours | 19.5 Hours | 4.5 Hours |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (Individual or Team) | Estimated | Actual | Difference |
| RAS Software Requirements | Anish | 7 Hours | 5 Hours | 2 Hours |
|  | Mark | 8 Hours | 5 Hours | 3 Hours |
|  | Mehmed | 7 Hours | 5 Hours | 2 Hours |
|  | Summary (People Hours) | 22 Hours | 15 Hours | 7 Hours |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (Individual or Team) | Estimated | Actual | Difference |
| RAS Complete | Anish | 12 Hours | 9 Hours | 3 Hours |
|  | Mark | 10 Hours | 7 Hours | 3 Hours |
|  | Mehmed | 11.5 Hours | 7 Hours | 4.5 Hours |
|  | Summary (People Hours) | 33.5 Hours | 23 Hours | 9.5 Hours |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (Individual or Team) | Estimated | Actual | Difference |
| Software Analysis Specification | Anish | 10 Hours |  |  |
|  | Mark | 10 Hours |  |  |
|  | Mehmed | 9 Hours |  |  |
|  | Summary (People Hours) | 29.5 Hours |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (Individual or Team) | Estimated | Actual | Difference |
| SPMP | Anish | 12 Hours | 13 Hours | 1 Hour |
|  | Mark | 13 Hours | 14 Hours | 1 Hour |
|  | Mehmed | 11 Hours | 11.5 Hours | .5 Hour |
|  | Summary (People Hours) | 36 Hours | 38.5 Hours | 2.5 Hours |

**Cumulative**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (Individual or Team) | Estimated | Actual | Difference |
|  | Anish | 32 Hours |  |  |
|  | Mark | 32 Hours |  |  |
|  | Mehmed | 32 Hours |  |  |
|  | Summary (People Hours) | 96 Hours |  |  |

**Defect Tracking**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (Individual or Team) | Estimated | Actual | Difference |
| RAS Business & Project Definition | Anish | 7 | 12 | 5 |
|  | Mark | 8 | 12 | 4 |
|  | Mehmed | 6 | 12 | 6 |
|  | Summary (Avg.) | 7 | 12 | 4 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (Individual or Team) | Estimated | Actual | Difference |
| RAS Software Requirements | Anish | 6 | 12 | 6 |
|  | Mark | 12 | 12 | 0 |
|  | Mehmed | 5 | 12 | 7 |
|  | Summary (Avg.) | 8 | 12 | 6.5 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (Individual or Team) | Estimated | Actual | Difference |
| RAS Complete | Anish | 16 |  |  |
|  | Mark | 16 |  |  |
|  | Mehmed | 14 |  |  |
|  | Summary (Avg.) | 15 |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (Individual or Team) | Estimated | Actual | Difference |
| Software Analysis Specification | Anish | 8 |  |  |
|  | Mark | 12 |  |  |
|  | Mehmed | 11 |  |  |
|  | Summary (Avg.) | 10 |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (Individual or Team) | Estimated | Actual | Difference |
| SPMP | Anish | 11 | 2 | 9 |
|  | Mark | 5 | 2 | 3 |
|  | Mehmed | 9 | 2 | 7 |
|  | Summary (Avg.) | 8 | 2 | 6 |

**Cumulative**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Artifact or Deliverable | Who (Individual or Team) | Estimated | Actual | Difference |
|  | Anish | 48 |  |  |
|  | Mark | 53 |  |  |
|  | Mehmed | 45 |  |  |
|  | Summary (Avg.) | 49 |  |  |

**Dictionaries**

Classes

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Description | Methods | Attributes |
| Dashboard | GUI of Dashbaord | uploadData() |  |
| deleteProject() |
| createNew() |
| buildGraphics() |
| openAnalysis() |
| setMetrics() |
| getData() |
| LoginGUI | GUI of Login | validate() | Username |
| Password |
| Analysis | Algorithm for data anlysis | setupTree() |  |
| setupNode() |
| regressionAnalysis() |
| byTag() |
| ExternalAPI | APIs of social networks | getAttributes() |  |
| PredictTrend | Algorithm for prediction | setTrainingData() |  |
| createRules() |
| setModel() |
| InHouseDB | Database | sortBy() | Data |
| save() |
| clear() |
| getData() |
| Output | Result of algorithm | getGraph() |  |
| getChart() |
| setData() |
| export() |
| DataFilter | Algorithm for managing data | dataCleaning() |  |
| setTags() |

Methods

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Description | Class | Arguments |
| uploadData | Upload custom data | Dashboard | Data |
| deleteProject | Delete existing project |  |
| createNew | Create new project |  |
| buildGraphics | Build graphics from output | Output |
| openAnalysis | Transition to Analysis |  |
| setMetrics | Set Data metrics |  |
| getData | Get data metrics | Data Metrics |
| validate | Validate user credentials | LoginGUI | Username |
| Password |
| setupTree | Set up tree data structure | Analysis | Data Metrics |
| setupNode | Set up node data structure | Data Metrics |
| regressionAnalysis | Analyze by regression | Data Metrics |
| byTag | Analyze by tag | Tags |
| getAttributes | Get data attributes | ExternalAPI |  |
| setTrainingData | Set training data for algo. | PredictTrend | Data Metrics |
| createRules | Create rules for algo. |  |
| setModel | Set up deep learning model |  |
| sortBy | Sort data | Database | Tags |
| save | Save data | Data |
| clear | Clear data | Data |
| getData | Get output data | Output | Output |
| getGraph | Get output graph | Output |
| getChart | Get output chart | Output |
| setData | Set output data | Output |
| export | Export to other format | Output |
| dataCleaning | Trim data | DataFilter | Data |
| setTags | Set data tags | Tags |

Attributes

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Name | Description | C/S | Type | Size | Attribute | Attribute | Attribute |  | R/W |
| Username | Username to login | S | string |  |  |  |  |  | R,W |
| Password | Password to login | S | string |  |  |  |  |  | R,W |
| Output | Output of algo. | C |  |  | Data |  |  |  | R |
| Data Metrics | Used for data analysis | C |  |  | Data |  |  |  | R,W |
| Tags | Used to set rules for algo. | C |  |  | Data |  |  |  | R,W |
| Data | Generic container of data | C |  |  | Strings | Int | Double |  | R,W |

Relationship

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Name | Description | From class | To class | Optional/Mandatory | Cardinality |
| Imports To | Data from API imports to analysis | ExternalAPI | Analysis | Optional | Many to 1 |
| Open | Go from one class to another | Dashboard | Analysis | Mandatory | 1 to 1 |
| LoginGUI | Dashboard | Mandatory | 1 to 1 |
| Validates | Validate credentials of LoginGUI with data in InHouseDB | InHouseDB | LoginGUI | Mandatory | 1 to 1 |
| Generate | Run analysis and generate outputs | Analysis | Output | Mandatory | 1 to Many |

Key Events

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Name | Description | Motive | Action | Pre-conditions | Post-conditions | State Changes |
| LoggedIn | User is logged in | To use system | Log in thrugh username and password | User Credential validated | Transition to dashboard | From login screen |
| InDashboard | User is in dashboard | To access system's features | Create, open, delete project | Is logged in | Transition to analysis | From loggedIn |
| AnalyzeSuccess | Data analysis is successful | To analyze data | Input data for analysis, and click "analyze" | Valid data is inputted | Transition to data prediction | From dashboard |
| AnalyzeFailure | Data analysis is not successful | To analyze data | Input data for analysis, and click "analyze" | Invalid data is inputted | Prompts user to re-enter data | From dashboard |
| PredictSuccess | Data prediction is successful | To predict trend | Input data for analysis, and click "analyze" | Valid data is inputted, transitioned from analysis | Transition to output | From dashboard |
| PredictFailure | Data prediction is not successful | To predict trend | Input data for analysis, and click "analyze" | Data metrics is not sufficient, transitioned from analysis | Alert user of prediction failure | From dashboard |
| DataSaved | Data is saved to database | Prevent data loss | System automatically saves results to database or user edit database manually | Data is valid for saving | Alert user of success of action | From data prediction |
| DataDeleted | Data is deleted from database | Remove data | User selects which data to delete | Data is in database | Alert user of success of action | From data prediction |
| DataExported | Data from database is exported to other formats | Convert data | User selects which data to be exported, as well as format of the exported data | Exported format can be supported | Alert user of success of action | From database |
| DataImported | Data from external APIs is imported to database | Use data from external sources | User imports data through API | Data is valid to be imported | Alert user of success of action | From APIs |