



International School

## CAPSTONE PROJECT 2

CMU-SE-451 / CMU-IS-451 / CMU-CS-451



## EXPERT-DRIVEN SMART DASHBOARD APPLICATION

### Members:

Vo Van Hoa  
Pham Van Tin  
Ky Huu Dong  
Tran Thi Thanh Kieu

### Mentor:

PhD Nguyen Thanh Binh

Date:

01 - Mar - 2021

01 - Jun - 2021

PROJECT INFORMATION			
<b>Project Acronym</b>	EDSDA		
<b>Project Title</b>	Expert-Driven Smart Dashboard Application		
<b>Project Web URL</b>	<a href="https://sda-research.ml/">https://sda-research.ml/</a>		
<b>Start Date</b>	01 - Mar - 2021		
<b>End Date:</b>	02 - Jun - 2021		
<b>Lead Institution</b>	International School, Duy Tan University		
<b>Project Mentor</b>	Ph.D Binh, Thanh Nguyen		
<b>Scrum Master</b>	Hoa, Vo	hoavo.dng@gmail.com	0935.193.182
<b>Team Members</b>	Tin, Pham Van	tinphamvan123@gmail.com	0932.535.175
	Dong, Ky Huu	kyhuudong@gmail.com	0898.246.980
	Kieu, Tran Thi Thanh	thanhkieutran391@gmail.com	0358.583.251



## CAPSTONE PROJECT 2

CMU-SE-451 / CMU-IS-451 / CMU-CS-451

### PROJECT PROPOSAL

Version 2.1

Date: 1 - Mar - 2021

## EXPERT-DRIVEN SMART DASHBOARD APPLICATION

#### Submitted by

Vo Van Hoa  
Pham Van Tin  
Ky Huu Dong  
Tran Thi Thanh Kieu

#### Approved by

#### Capstone Project 2 - Mentor:

Name

Signature

Date

Binh, Thanh Nguyen \_\_\_\_\_

A handwritten signature in blue ink, appearing to read 'Binh, Thanh Nguyen'.

\_\_\_\_\_ 16 - Mar - 2020

#### Proposal Review Panel Representative:

Name

Signature

Date

PROJECT INFORMATION			
<b>Project Acronym</b>	EDSDA		
<b>Project Title</b>	Expert-Driven Smart Dashboard Application		
<b>Project Web URL</b>	<a href="https://sda-research.ml/">https://sda-research.ml/</a>		
<b>Start Date</b>	01 - Mar - 2021		
<b>End Date:</b>	02 - Jun - 2021		
<b>Lead Institution</b>	International School, Duy Tan University		
<b>Project Mentor</b>	Ph.D Binh, Thanh Nguyen		
<b>Scrum Master</b>	Hoa, Vo	hoavo.dng@gmail.com	0935.193.182
<b>Team Members</b>	Tin, Pham Van	tinphamvan123@gmail.com	0932.535.175
	Dong, Ky Huu	kyhuudong@gmail.com	0898.246.980
	Kieu, Tran Thi Thanh	thanhkieutran391@gmail.com	0358.583.251

DOCUMENT INFORMATION			
<b>Document Title</b>	Project Proposal		
<b>Author(s)</b>	Team C2SE.06		
<b>Role</b>	[EDSDA] Proposal_v2.1		
<b>Date</b>	01 - Mar - 2021	Filename	[EDSDA] 001 Project Proposal
<b>URL</b>	<a href="https://github.com/sdateamdtu2020/SDA-v2.0">https://github.com/sdateamdtu2020/SDA-v2.0</a>		
<b>Access</b>	Project and CMU Program		

## REVISION HISTORY

Version	Person(s)	Date	Description	Approval
Draft	Hoa, Vo	15 - Jan - 2020	Initiate proposal	x
2.0	All members	07 - Mar - 2020	Finish content of the proposal	x
2.1	All members	16 - Mar - 2020	Update new context diagram & Table compare the different between ver1.0 & v2.0	x

# TABLE OF CONTENTS

PROJECT INFORMATION	1
DOCUMENT INFORMATION	1
REVISION HISTORY	2
<b>TABLE OF CONTENTS</b>	<b>3</b>
<b>1. INTRODUCTION</b>	<b>4</b>
1.1. PURPOSE OF DOCUMENT	4
1.2. PROJECT GOAL	4
1.3. BACKGROUND	4
<b>2. PROJECT DEFINITION</b>	<b>5</b>
2.1. USER NEEDS	5
2.2. NON-FUNCTIONAL REQUIREMENTS	6
2.3. FUNCTIONAL REQUIREMENTS	6
<b>3. CURRENT STATUS OF ART</b>	<b>7</b>
3.1. COMPARE WITH SIMILAR PRODUCTS ON THE MARKET	7
3.2. COMPARE WITH CAPSTONE 1 PRODUCT VERSION	8
<b>4. ENGINEERING APPROACH</b>	<b>10</b>
4.1. CONTEXT DIAGRAM	10
4.2. PROCESS DETAILS	10
4.3. TECHNICAL TO DEVELOP SYSTEM	12
4.4. THE REASON BEHIND THE TECHNOLOGY STACK	13
<b>5. TASK AND DELIVERABLES</b>	<b>14</b>
5.1. TASKS	14
5.2. DELIVERABLES	15
<b>6. PROJECT MANAGEMENT</b>	<b>16</b>
6.1. ABOUT SCRUM	16
6.2. WHY SCRUM	16
6.3. TOTAL COST ESTIMATE	17
6.4. TENTATIVE SCHEDULE	18
<b>7. PROJECT CONSTRAINTS</b>	<b>21</b>
<b>8. CONCLUSION</b>	<b>21</b>
<b>9. REFERENCE</b>	<b>22</b>

# 1. INTRODUCTION

## 1.1. PURPOSE OF DOCUMENT

- Define the business needs and problems in detail.
- Provide solutions for business needs.
- Provide an overview of resources, schedule, solution, and budget for the project.

The proposal merely introduces the project to the student development teams and provides the up-front information necessary for the team to develop a specification.

## 1.2. PROJECT GOAL

Expert-Driven Smart Dashboard - means a dashboard that is convenient for users to analyze and review data. It will include several datasets about real-time information of the environment, measurement data of air pollutants... EDSA will connect and analyze data from multiple sources in ways you've never imagined. Then reveal the insights you've been missing...in just a matter of minutes.

With smart suggestions and an intuitive visual interface, SDA makes it easy for any user to combine data and discover hidden insights in one place...without the usual scripting, coding, and IT hand-holding. With this dashboard, individuals or any subjects can take advantage of environmental data to be able to decide the best relevant policies.

Not only for the environmental expert, but EDSA also helps the user directly interact with their own data source by importing, automatically cleaning their data, and enabling the user to do everything with their own data in the easiest way.

## 1.3. BACKGROUND

Our environment is always changing. However, at the current rate of urbanization and industrialization, outside of the natural factors, the change of environment is mainly due to human factors. Emissions, population explosion, industrial solid waste, ... are the main causes leading to negative effects on the global environment. To address this at a holistic level, data analysis and aggregation are the first important tasks to be done.

However, analyzing and aggregating data from many different sources takes a lot of effort and money. To solve this problem, based on our knowledge of big data systems, we have built an intelligent data processing system that can be run on a website-platform with an intuitive and easy-to-use dashboard. This system is a prospective and useful tool for environmental experts and policymakers in Vietnam in particular, and worldwide in general. It will collect, analyze and synthesize data about all the factors that can affect the

environment, thereby helping users to come up with quick and accurate solutions to solve problems related to the environment.

In addition, data sources that open the environment in a particular aspect such as area emissions, soil erosion, etc. are often scarce and stored only in the user's local computer. Analytical tools of these types of data often require expertise using quite advanced technology, thus limiting interoperability with data stored in individuals. With this in mind, we built a subsystem where users could bring their data to EDSDA and analyze, aggregate, and visualize their data directly on the dashboard along with a wide range of system visualization tools.

## 2. PROBLEM DEFINITION

### 2.1. USER NEEDS

USER STORIES		
ID	Actor	Epic
1	As an environment expert	I want to drag a data cube from the cubes list and drop it onto the main content board so that I can choose the property of data that I need and visualize them on suitable kinds of charts and maps.
2	As an environment expert	I want to drag some data cubes from the cubes list and drop them onto the main content board so that I can choose the property of data and then connect these data cubes together for data operation and visualize it on suitable kinds of charts and maps.
3	As an environment expert	I want to view the data source of any data cubes so that I can verify the accuracy of the data.
4	As a user	I want to import my own data sources to the EDSDA database so that I can interact with my data directly with EDSA help.
5	As a user	I want to review my data sources before submitting them so that I can check the accuracy of my data.
6	As a user	I want to define the dimension and measure of my data source's data cubes by myself so that I can easily do OLAPing with my data source.

ID	Actor	Epic
7	As a user	I want when dragging and dropping data cubes with the same one or many dimensions, those dimensions will be automatically connected without manual manipulation, thereby saving time manually linking.
8	As a user	I want to publish the charts and data tables that I build on the dashboard so that I can use them for a variety of purposes including researching.

## 2.2. NON-FUNCTIONAL REQUIREMENTS

Below are the non-functional requirements that are being offered for the system :

- **Portability and compatibility:** The system is operated on a web-based platform and has the ability to run on any web browser.
- **Security:** Users can use the system without the fear of revealing personal information.
- **Availability:** The system can run continuity 24/24 a day.
- **Usability:** The system has a friendly and flexible user-interface and a great user experience.
- **Reliability:** The system has accurate and transparent data, functions that do exactly their job.

## 2.3. FUNCTIONAL REQUIREMENTS

Below are the functional requirements that are being offered for the system, which are the backbone of the project :

- Import data sources.
- Auto ETL
- Auto RDF Generation.
- Auto linking data.
- Drag data cubes from the cubes list.
- Drop data cubes onto the main content board.
- Connect the data cubes that have the relation between them.
- Merge the cubes to create a new data cube based on connected data cubes.
- Visualize the data cube as a line chart, a pie chart, a column chart, or a geographical map.

## 3. CURRENT STATUS OF ART

### 3.1. COMPARE WITH SIMILAR PRODUCTS ON THE MARKET

There are many research topics on data science applied to policy-making. Here are some researches that come from data scientist groups around the world.

1. Data science empowering the public (<https://www.waze.com/>): The Smart City dashboards in Rio de Janeiro, Brazil, were created to solve problems related to public transportation and traffic. For this, an infrastructure, a dashboard, and a data portal with more than three thousand datasets and seven APIs for real-time data use ([www.data.rio](http://www.data.rio)) were developed and used by the Center of Operations Rio (COR).
2. Interactive Dashboards (<http://www.data.rio>): Using Visual Analytics for knowledge Transfer and Decision Support ( Samar Al-Hajj, Ian Pike, Brian Fisher ) -A visual analytics dashboard that reflects the needs and preferences of injury stakeholders. The types of visualizations were selected to efficiently illustrate trends and patterns in injury data.
3. Triangulum City Dashboard (<https://moovitapp.com/>): An Interactive Data Analytic Platform for Visualizing SmartCity Performance ( Mina Farmanbar, Chunming Rong ).

It can be seen that data science is increasingly weaving and useful in all areas of life, from urban management to public policy management, and help in making decisions. But unfortunately, the world today is facing a vital problem, the environment. However, in today's market, there are very few data analysis platforms to help policymakers make the right decisions about environmental protection.

This is the reason that makes us build a smart dashboard called Smart Dashboard Application - SDA, the first environment data analysis dashboard in the market now. With Smart Dashboard Application, the users can easily choose and integrate any data that are satisfied with their needs and visualize it in many kinds of charts and maps, so that they - can consider the factors affecting the environment in the most comprehensive and intuitive way.

### 3.2. COMPARE WITH CAPSTONE 1 PRODUCT VERSION

In the first version, Smart Dashboard Application, we provide users with the following basic features:

- A static data warehouse of environmental data cubes.
- A drag & drop method for choosing data cubes to the dashboard.
- A manually linking data cubes method.
- Some visualization tools.

It can be seen that the two biggest disadvantages of the SDA version, are:

- The fixed data warehouse.
- The manual connection method by the user.

First, with the goal of targeting users who are not only environmental experts but also many other related aspects, it can be found that a fixed data warehouse is a rather limited thing, because of the specific aspects data are always updated and stored by experts on their personal computers, so data crawling tools will not be able to access them and enter the SDA system. This would be very detrimental for professionals to use SDA for specific data types. Realizing this, we have built **an intelligent data import system** that enables users to manually upload their flat data onto the system and interact with them as multidimensional data cubes.

The second thing, in the previous version, when users dropped data cubes on the dashboard and linked them together to aggregate data, they were forced to link the data cubes together by drawing connection lines. This takes a lot of time, even for first-time users of the system and for experienced users. To solve this problem, we have provided the system a method to **automatically detect data cubes with the same dimensions and connect them automatically**. This will improve the ability to attract users, the most important thing is to save time and manipulation of users, ensure efficiency and intelligence for the system.

SDA & EDSDA COMPARISON		
	Smart Dashboard Application	Expert-Driven Smart Dashboard Application
<b>Data source</b>	Fixed	Dynamic, user can import their own data sources
<b>Import data</b>	No	Yes
<b>ETL process</b>	Manual	Automatic
<b>RDF Data Cubes Generator</b>	Manual	Automatic
<b>Data Cubes suggestion</b>	No	Yes
<b>Hierarchy structure</b>	No	Yes
<b>Dimensions UI</b>	No	Yes
<b>Dimensions &amp; Measures suggesting &amp; selecting</b>	No	Yes
<b>Data Cubes automate linked</b>	No	Yes
<b>Export report</b>	No	Yes

## 4. ENGINEERING APPROACH

### 4.1. CONTEXT DIAGRAM

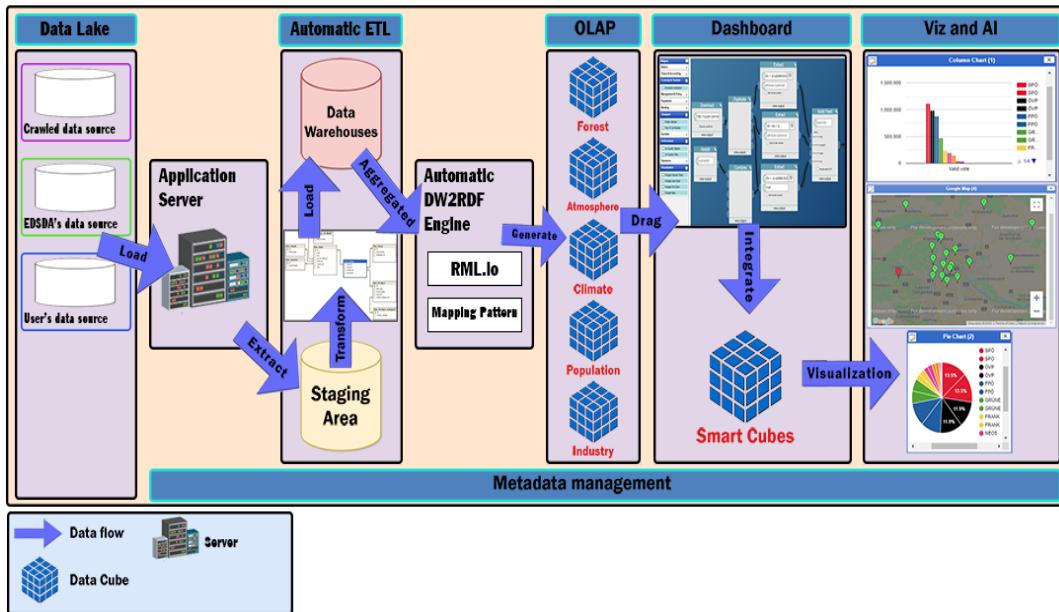


Figure 1: Context Diagram

### 4.2. PROCESS DETAILS

#### 1. Data Lake:

- Collect from environment open data platforms of the governments and NGO organizations.
- User imported data sources.
- Use web crawling techniques to crawl data from related environment websites.
- Data Format : CSV, JSON, XML.

#### 2. Application Server:

- Check for the changes of the data lake when the user imports new data sources.
- Handle Data Lake Exceptions if any errors occur.
- Trigger the Data Lake to Staging Area loading process.

### 3. Automatic ETL (Extract, Transform, Load)

#### 3.1. Extract:

In this step, we extract structured data after restructuring the raw data from the data lake. Mainly in this step, we focus on exacting data correctly from the structured data from the data lake.

#### 3.2. Transform:

Based on the business's constraints we have to transform the data to be compatible with our process to handle the data for displaying on the dashboard. Getting exacted data, transforming the data, and ready to load into the DW.

#### 3.3. Load:

Loading data into the target data warehouse database is the last step of the ETL process. We load all the data we have transformed into the staging areas of DW and based on the data we have loaded into staging areas we used to load the data from the staging areas to dimension tables, and fact tables.

#### 3.4. Perform automatic ETL:

Performing automatic ETL using tools for automating exacting data, transforming data, and loading data to DW. We use the available platform for scheduling to perform ETL at a specific time in a day. In addition, for performing ETL automatically we also build a data pipeline to make the data flow as we wanted.

### 4. Automatic DW2RDF Engine:

- Using RML.io with our custom features, it can generate RDF Data cubes automatically when users import new data sources to the data warehouse.
- Suitable with all data sources with a general mapping method.

### 5. OLAP

In this step, we store the data of the data warehouse as OLAP cubes. And then, for better query performance, data binding, and scalability, in addition to information transparency, we will automate converting OLAP cubes into RDF Data Cubes.

### 6. Dashboard

In this step, the user can drag any data cubes that appear as items on the sidebar and drop them onto the main content board, then connect between them and use the operator such as statistics merge, geo merge to build a new data cube that matches the user requirement.

## 7. Viz & AI

This step will perform the data cube which was created by the user with the form they want. It can be a map, a column chart, a line chart, or a pie chart.

### 4.3. TECHNICAL TO DEVELOP SYSTEM

Main Programming Language: Javascript, Python.

#### Data Crawling:

- Programming Language: Python, Javascript.
- Database: PostgreSQL.
- Library: Beautiful Soup 4, Selenium.

#### Data Warehouses:

- Programming Language: Python.
- Database: PostgreSQL.
- Library: Psycopg2, CSV.

#### Perform ETL automating:

- Tool for ETL process: Talend.
- Schedule for ETL process platform: Apache Airflow.

#### Data Cubes

- Programming Language: RDF-Graph, SPARQL.
- Tool for converting from Data Warehouse to RDF Data Cubes Storage: RML.io (with customized phase), MIRROR.
- Network Accessing: RDF-REST API.

#### Server:

- Programming Language: Javascript.
- Framework / Libraries: ExpressJS (NodeJS), enapso-graphdb-client.
- Operating System: Windows, Linux, macOS.
- Deployment Environment: Google Cloud with App Engine and SQL Services.
- Network Accessing: HTTP methods (POST, GET) via RESTful API.

#### Client:

- Programming language: HTML, CSS, Javascripts.
- Framework/Libraries: React, Redux, Material-UI, react-dnd, beautiful-react-diagrams, React Charts,...
- Deployment Environment: Google Firebase Hosting
- Web Browser: Chrome, Firefox, Microsoft Edge, Coccoc
- Network Accessing: World Wide Web (WWW), HTTP methods (POST, GET) via RESTful API with Axios.

## 4.4. THE REASON BEHIND THE TECHNOLOGY STACK

### **Data Warehouses:**

We choose PostgreSQL for the database. It tries to conform with the SQL standard where such conformance does not contradict traditional features or could lead to poor architectural decisions. Many of the features required by the SQL standard are supported, though sometimes with slightly differing syntax or function. Further moves towards conformance can be expected over time. As of the version 12 release in October 2019, PostgreSQL conforms to at least 160 of the 179 mandatory features for SQL:2016 Core conformance. As of this writing, no relational database meets full conformance with this standard.

### **Data cubes:**

We choose to use the RDF model because it allows describing resources (either digital or taken from the real world), by specifying the values of their properties. Thus, an RDF information unit is a triple s p o, with s, p, and o standing for the subject, property, and object, respectively, in RDF terminology. The RDF language is increasingly being used in order to export, share, and collaboratively author data in many settings. For instance, it serves as a metadata language to describe cultural artifacts in large digital libraries and to encode protein sequence data, as in the Uniprot data set. RDF is a natural target for representing heterogeneous facts contributed by millions of Wikipedia users, gathered within the DBpedia data source, as well as for the Linked Open Data effort, aiming at connecting and sharing collectively produced data and knowledge.

### **Server:**

We chose ExpressJS to build the webserver. It's a prebuilt NodeJS framework that makes creating server-side applications simple, fast, and flexible. NodeJS is powered by Google's V8 engine which means it's powerful and can handle a large number of requests without lapsing in dependability. Also, this means that this is a highly scalable choice when you



consider the Event Loop which manages all asynchronous operations allowing the program to continue to run as expected without stops.

#### **Client:**

We choose React to build the web client. React is developed by Facebook and right now, React and its ecosystems are the most popular stack for front-end developers.

React introduces a new way to interact with the browser: Virtual DOM. We can now build web UI a lot faster than using the old one, the traditional jQuery library. The UI has also become easier to develop and maintain. Besides that, React makes a huge impact on performance, every change on the client-side is now handled by the browser so that it will reduce server load. Together with Redux, React became an amazing tool: good performance, fast development and deployment speed, and a safe future. React has also got a huge and extremely active community that we can get help easily. And React-DnD is a set of React utilities to help you build complex drag and drop interfaces while keeping your components decoupled.

## 5. TASK AND DELIVERABLES

### 5.1. TASKS

WBS NUMBER	TASK TITLE
1	Preparation
2	Data Modeling
3	Physical Warehouse Design
4	Initiate Dynamical Data Warehouse
5	Finding Data Sources
6	Data processing
7	ETL Process Validating
8	Automate ETL Setting
9	Expanding database
10	Crawl data
11	Building Data Cubes Importing method
12	Building additional features
13	Common RDF Data Cubes Structure Designing

WBS NUMBER	TASK TITLE
14	Setting up Automate DW to RDF Process Tool & Environment
15	Implementing Automate DW2RDF Process
16	Setting up RDF Data Cubes Storing & SPARQL Endpoint
17	Validating RDF Data Cubes
18	Building SPARQL_Rest API
19	Building UI
20	Testing
21	Integrate
22	Deploy
23	Release

## 5.2. DELIVERABLES

No	Activities	Deliverables
1	Project Proposal	Project Proposal Document 2.1
2	Product Requirement	
3	Project Plan	
4	Product Backlog	
5	Architecture Document	
6	Database Design	
7	Interface Design	
8	Test Plan	
9	Test Case	
10	Acceptance Criteria	
11	Sprint Backlog & Burndown Chart	
12	Team Reflection	
13	Technologies Stack	

## 6. PROJECT MANAGEMENT

### 6.1. ABOUT SCRUM

Scrum is an agile method, so it follows the principles of Agile Manifesto ([see also Agile Manifesto](#)). In addition, Scrum operates on three core values, also known as the three pillars of Scrum, including Scrutiny, Inspection, and Adaptation.

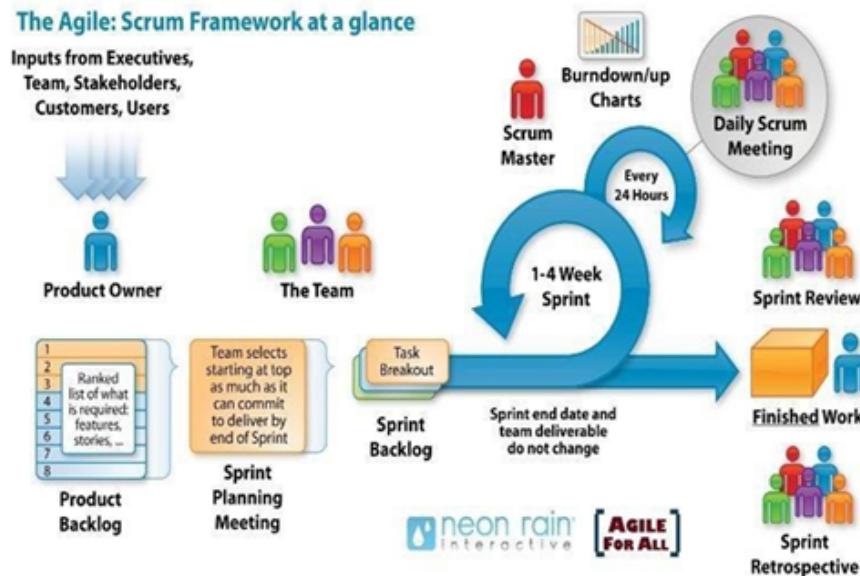


Figure 2: Scrum process

Based on the empirical process control theory, Scrum uses iterative and incremental algorithms to optimize efficiency and control risk. Scrum is simple, easy to learn, and has wide applicability. To be able to use Scrum, we need to understand and apply the elements that makeup Scrum include the core values (also known as the "three legs", or the three pillars of Scrum), roles, Events, and Scrum-specific artifacts.

### 6.2. WHY SCRUM

- Our team has 4 people
- The project will be continuously horizontally scaled up.
- There is only a short amount of time to finish the project.

So based on those constraints, we decided to choose SCRUM as the project lifecycle.

### 6.3. TOTAL COST ESTIMATE

COST DESCRIPTION	DETAIL
Salary	Duration: 16 weeks (112 days) Man-hour: 4hours/1day Salary: \$2/1hour Persons: 4 members Overtime cost per hour: 1.5\$/hour The salary of 1 person: \$1000 <b>Total: \$4000</b>
Laptop	Laptop for each member: \$800 <b>Total: \$3200</b>
Monitor	LG Monitor for each member: \$100 <b>Total: \$400</b>
VPS Engine server	VPS Engine server per month on Google Cloud: \$100 VPS Engine server for 3 months: 60\$ <b>Total: \$300</b>
Maintenance	Cost per month: \$400
Bugs fixing	Cost per bug: \$20
<b>Total</b>	Salary + Laptop + Monitor + VPS Engineer server <b>Total: \$7900</b>

## 6.4. TENTATIVE SCHEDULE

EDSDA - Project Timeline

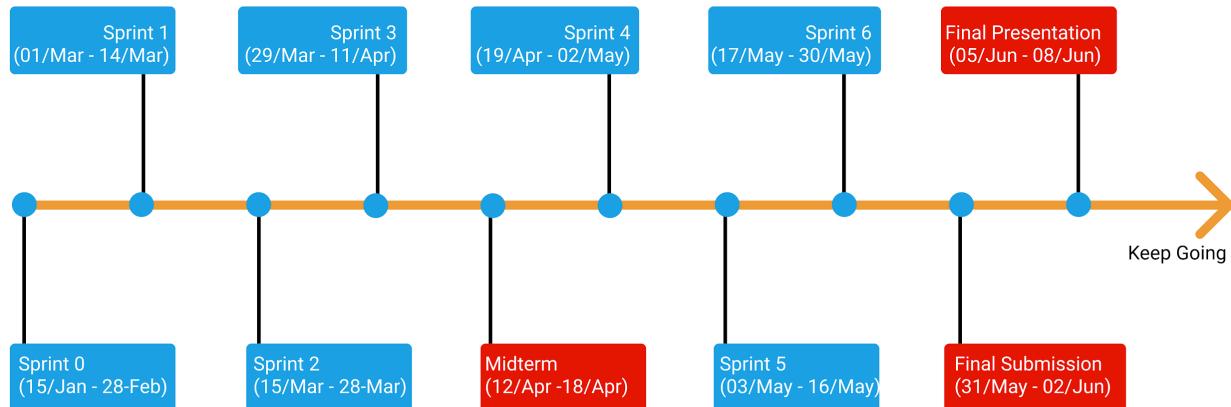


Figure 3: Project Timeline

WBS	Task	Start	End	Days
	<b>Sprint 1</b>	<b>01-Mar-21</b>	<b>14-Mar-21</b>	<b>14</b>
1	Crawl data			
2	Clean data			
3	Initiate RML process			
4	Initiate MIRROR			
5	Initiate GraphDB Database			
6	UI Design			
7	Initiate & config React project			
8	Testing			
	<b>Sprint 2</b>	<b>15-Mar-21</b>	<b>28-Mar-21</b>	<b>14</b>
1	Create DW			
2	ETL Process using tool			
3	Define common mapping for RML & MIRROR			
4	Implement API for importing data, getting data from GraphDB			

WBS	Task	Start	End	Days
5	Building UI			
6	Testing			
	<b>Sprint 3</b>	<b>29-Mar-21</b>	<b>11-Apr-21</b>	<b>14</b>
1	Getting, and Cleaning imported data			
2	ETL the imported data			
3	Describe common schema for RDF Data Cubes			
4	Run RML & MIRROR with sample data			
5	Integrate sample UI & API			
6	Building UI			
7	Testing			
	<b>Midterm</b>	<b>12-Apr-21</b>	<b>18-Apr-21</b>	<b>7</b>
1	Deploy version 0.1			
2	Review documentation			
	<b>Sprint 4</b>	<b>19-Apr-21</b>	<b>02-May-21</b>	<b>14</b>
1	Automating clean data			
2	Schedule for ETL data pipeline			
3	Customize RML & MIRROR			
4	Implement pipeline for automating RDF Data Cube generation			
5	Building UI			
6	Testing			
	<b>Sprint 5</b>	<b>03-May-21</b>	<b>16-May-21</b>	<b>14</b>
1	Handling new importing data to DW(Dimensions, Measures)			
2	Automating ETL for new importing data			
3	Integrate automate ETL & RDF generation			
4	Automating detect dimension & measure			



WBS	Task	Start	End	Days
5	Building UI			
6	Testing			
	<b>Sprint 6</b>	<b>17-May-21</b>	<b>30-May-21</b>	<b>14</b>
1	Automating crawling data from NGO, or any new defined data source.			
2	Cleaning data process for the new defined data source.			
3	Set up VPS for deploying			
4	Migrate GraphDB to VPS			
5	Building UI			
6	Integration			
7	Testing			
	<b>Final</b>	<b>31-May-21</b>	<b>08-Jun-21</b>	<b>9</b>
1	Review documentation			
2	Release / Deploy final version			
3	Presentation			



## 7. PROJECT CONSTRAINTS

CONSTRAINTS	CONSTRAINTS DESCRIPTION	GUIDELINES FOR ACCEPTANCE
Economic	In terms of cost, since it is a dashboard with not too many screens, the main issue lies in the cost of researching, implementing a fully automated backend system and frontend with smart methods. There is also the cost of renting a server and deploying a server that should be considered carefully	Design cost: Must be around \$100. Production cost: Should be under \$4000. Maintenance cost: Should be around \$500. Operation cost: Should be under \$1000
Ethical	Because this software analyzes data with most of the data being crawled, so pay attention to copyright and data security.	All crawl data sources must be open data and be allowed for public use.
Social and Global	The software is targeted to not only environmental experts but also data analysts in many other fields, so widespread popularity is very important.	The product needs to be developed in an optimal and user-friendly way to reach a wide range of users
Sustainability	Need to maintain the continuous operation of the system, so as not to affect the analysis of people	Development and maintenance work must be ensured to take place continuously, when issues are reported, it is necessary to focus on maintenance immediately. Server operation also needs to be ensured not to be interrupted.

## 8. CONCLUSION

The project will be finished in 13 weeks and divided into 6 sprints (2weeks/sprint), 1 week for midterm review, and 4 weeks for pre-sprint. It promises to be a convenient and useful tool for not only environmental experts and policymakers but also those who love to protect the environment. The confusion about a large data system and reviewing it in a general way to come up with precise planning will no longer be a problem, even for those who don't have technical expertise. The project will contribute to helping our society accurately and effectively fight against environmental damage and global warming.

## 9. REFERENCE

- Draw tool: <https://www.figma.com/>
- Software Development Standards for the Guidance and Control Software Project  
<https://sw-eng.larc.nasa.gov/>
- General Software Coding Standards and Guidelines  
[https://www.nws.noaa.gov/oh/hrl/developers\\_docs/General\\_Software\\_Standards.pdf](https://www.nws.noaa.gov/oh/hrl/developers_docs/General_Software_Standards.pdf)
- Scrum and best practices  
<https://docs.microsoft.com/en-us/azure/devops/boards/sprints/best-practices-scrum?view=azure-devops>
- The Scrum Guide <https://www.scrum.org/resources/scrum-guide>
- The ISO/IEC & IEEE/EIA Standard 12207, IEEE standards: IEEE-829 [3], IEEE-1008 [5], IEEE-1012
- Technologies Stack Document:  
[https://docs.google.com/document/d/1DnWvnATlfAxsXl8dq2-REEw9L7tW5cYyz\\_W0GR6n9U/edit?usp=sharing](https://docs.google.com/document/d/1DnWvnATlfAxsXl8dq2-REEw9L7tW5cYyz_W0GR6n9U/edit?usp=sharing)
- React documentation: <https://reactjs.org/docs/getting-started.html>
- Redux documentation: <https://redux.js.org/>
- Data science empowering the public: Data-driven dashboards for transparent and accountable decision-making in smart cities:  
<http://www.athanasiadis.info/assets/pdf/ems2018.pdf>
- Smart cities survey: Technologies, application domains and challenges for the cities of the future:  
[https://www.researchgate.net/publication/333684319\\_Smart\\_cities\\_survey\\_Technologies\\_application\\_domains\\_and\\_challenges\\_for\\_the\\_cities\\_of\\_the\\_future](https://www.researchgate.net/publication/333684319_Smart_cities_survey_Technologies_application_domains_and_challenges_for_the_cities_of_the_future)
- Interactive Dashboards: Using Visual Analytics for knowledge Transfer and Decision Support:  
[https://www.researchgate.net/publication/299633100\\_Interactive\\_Dashboards\\_Using\\_Visual\\_Analytics\\_for\\_knowledge\\_Transfer\\_and\\_Decision\\_Support](https://www.researchgate.net/publication/299633100_Interactive_Dashboards_Using_Visual_Analytics_for_knowledge_Transfer_and_Decision_Support)
- Dashboard technology-based solution to decision making:  
[https://www.researchgate.net/publication/277140671\\_DASHBOARD TECHNOLOGY BASED SOLUTION\\_TO\\_DECISION MAKING](https://www.researchgate.net/publication/277140671_DASHBOARD TECHNOLOGY BASED SOLUTION_TO_DECISION MAKING)
- Triangulum City Dashboard: An Interactive Data Analytic Platform for Visualizing Smart City Performance: <https://www.mdpi.com/2227-9717/8/2/250>
- Data Science of the Natural Environment: A Research Roadmap:  
<https://www.frontiersin.org/articles/10.3389/fenvs.2019.00121/full>
- IEEE: <https://www.ieee.org/>
- QBOAirbase: European Air Quality Database as an RDF Cube:  
<http://luisgalarraga.de/docs/qboairbase.pdf>

- Enhanced Living Environments:  
<https://link.springer.com/content/pdf/10.1007%2F978-3-030-10752-9.pdf>
- An Approach To Publish a Data Warehouse Content as Linked Data:  
<https://pdfs.semanticscholar.org/7913/6c3ad6f91aba1c9d186ede80fb1a1ec54704.pdf>



## CAPSTONE PROJECT 2

CMU-SE-451 / CMU-IS-451 / CMU-CS-451

### DESCRIPTION OF PRODUCT REQUIREMENTS FORM

Version 2.0

Date: 1 - Mar - 2021

### EXPERT-DRIVEN SMART DASHBOARD APPLICATION

#### Submitted by

Vo Van Hoa  
Pham Van Tin  
Ky Huu Dong  
Tran Thi Thanh Kieu

#### Approved by

##### Capstone Project 2 - Mentor:

Name

Signature

Date

Binh, Thanh Nguyen \_\_\_\_\_  \_\_\_\_\_ 26 - May - 2020

PROJECT INFORMATION			
<b>Project Acronym</b>	EDSDA		
<b>Project Title</b>	Expert-Driven Smart Dashboard Application		
<b>Project Web URL</b>	<a href="https://sda-research.ml/">https://sda-research.ml/</a>		
<b>Start Date</b>	01 - Mar - 2021		
<b>End Date:</b>	02 - Jun - 2021		
<b>Lead Institution</b>	International School, Duy Tan University		
<b>Project Mentor</b>	Ph.D Binh, Thanh Nguyen		
<b>Scrum Master</b>	Hoa, Vo	hoavo.dng@gmail.com	0935.193.182
<b>Team Members</b>	Tin, Pham Van	tinphamvan123@gmail.com	0932.535.175
	Dong, Ky Huu	kyhuudong@gmail.com	0898.246.980
	Kieu, Tran Thi Thanh	thanhkieutran391@gmail.com	0358.583.251

DOCUMENT INFORMATION			
<b>Document Title</b>	Requirements Description		
<b>Author(s)</b>	Team C2SE.06		
<b>Role</b>	[EDSDA] Requirements Description v2.0		
<b>Date</b>	01 - Mar - 2021	Filename	[EDSDA] 001 Requirements Description
<b>URL</b>	<a href="https://github.com/sdateamdtu2020/SDA-v2.0">https://github.com/sdateamdtu2020/SDA-v2.0</a>		
<b>Access</b>	Project and CMU Program		

## REVISION HISTORY

# **DESCRIPTION OF PRODUCT REQUIREMENTS**

Group: C2SE.06

Project: Expert-driven Smart Dashboard Application

Date: 06 - Mar - 2021

## **I. Short description of product ideas (less than 7 statements)**

- A very easy-to-use and user-friendly data analysis system.
- A dashboard helps the environmental expert analyze, aggregate, and visualize both the data sources on the internet and their data sources.
- Automated multidimensional data processing.
- Automated generate RDF Data Cube.
- Smart data linking.

## **II. Requirements**

High-level Functional Requirements	<ol style="list-style-type: none"><li>1. Users must be able to import their data sources</li><li>2. Users must be able to review and choose dimensions &amp; measures for their data sources</li><li>3. Users must be able to view all the data cubes that existed in the system.</li><li>4. The imported data must be listed on the data cubes list.</li><li>5. Users must be able to drag &amp; drop the data cubes from list to dashboard</li><li>6. The data operators must be always available.</li><li>7. The connectors between data cubes having the same dimensions must be connected automatically.</li><li>8. Users should be able to export their data visualization.</li></ol>
------------------------------------	---

Quality Attributes Requirements (example related to issues: Ease	1. Portability and compatibility: The system is operated on a web-based platform and can run on any web browser.
---	--

Use, Easy to Like, Easy to Learn, Easy to Understand, Easy to Buy / Yes, ...)	<p>2. Security: Users can use the system without the fear of revealing personal information.</p> <p>3. Availability: The system can run continuity 24/24 a day.</p> <p>4. Usability: The system has a friendly and flexible user-interface and a great user experience.</p> <p>5. Reliability: The system has accurate and transparent data, functions that do exactly their job.</p>
---	---

Operation Requirements (related to issues: Speed, Accuracy, Performance, Stability, Load Resistance, Scalability, Safety, ...)	<p>1. Adaptability: System must be able to import (CSV, XLSX files) and export (PDF, PNG files)</p> <p>2. Scalability: The database(DW) must be able to expand based on the data source, or business requirements.</p> <p>3. Accuracy: The UI visualizes exactly imported data from the user.</p> <p>4. Compatibility: The system will work on any web browser. The system will work on any operating systems (Linux, Window, and Mac)</p> <p>5. Maintainability: New versions of the system will be released every 4 months.</p>
---	---

Environment & Operation Requirements (related to issues: physical impacts on the environment, interaction with relevant or existing systems, conditions for product commercialization, ...)	<p>1. Cost must be controlled around \$3200 to have the best selling price for widely available in the market.</p> <p>2. Data copyright issues need to be transparent so that they do not violate the provisions of intellectual property rights when placed on the market.</p>
--	---

Requirements for Maintenance & Support	<p>1. Assets Management: Google Cloud VPS, 4 laptops, 1 PC, 1 tablet.</p> <p>2. Work Orders Management: PM has the right for leading all the tasks.</p> <p>3. Preventive Maintenance Management: All system features must be tested every week. The tester should report any issues immediately. Server maintenance should be done every 2 weeks.</p> <p>4. Report Management: Every meeting report must be stored.</p>
--	---

	Every maintenance report must be stored. The tester must have written documentation for any issues that occurred.
--	--

Security/ Safety Requirements (related to issues: conditions of use/access to products, personal freedom, inspection, ...)	<ol style="list-style-type: none"> <li>1. Constraint the import files (CSV, XLXS)</li> <li>2. Not directly interact with the database just only interact through the import method.</li> <li>3. The user is not able to attach queries directly with URLs.</li> <li>4. The user can not access GraphDB.</li> <li>5. Data cubes can not be edited by users.</li> <li>6. Prevent SQL injection in every field.</li> </ol>
---	---

Culture Requirements	<ol style="list-style-type: none"> <li>1. The system will operate in English</li> <li>2. The system focus on the environmental factors in Vietnam</li> <li>3. The system format will be UTF-8 for text, “DD-MM-YYYY” for date</li> </ol>
----------------------	--

Evaluate the complexity of engineering problems	<ol style="list-style-type: none"> <li>1. It is difficult to detect malicious code in the data source imported by the user.</li> <li>2. It is not yet possible to detect the data cubes with the same data.</li> <li>3. High complexity when integrating multiple platforms with many different programming languages at the same time.</li> <li>4. The invention of a method to automatically generate RDF Data Cubes from Data Warehouse is highly complex.</li> <li>5. Involving multiple data disciplines.</li> <li>6. Haven't found a method to simplify the data import process.</li> </ol>
---	---

Standard requirements	<ol style="list-style-type: none"> <li>1. Code standard: Python: PEP-8 Javascript: Airbnb React/JSX Style, Airbnb CSS-in-JavaScript Style. Java: Google Java Style.</li> <li>2. Design standard: Module Pattern &amp; Revealing Module Pattern.</li> </ol>
-----------------------	--

- |  |  |
|--|--|
|  | 3. Documentation standard: IEEE 802.22   |
|  | 4. ISO/IEC 25051:2006(TCVN 10540:2014)   |
|  | 5. Data ETL & Warehousing: Kimball's Data warehousing standard.                  |
|  | 6. Dashboard Design Standard: University of Southern Indiana Dashboard Standard. |



DUY TÂN  
UNIVERSITY

International School

## CAPSTONE PROJECT 2

CMU-SE-451 / CMU-IS-451 / CMU-CS-451

### PROJECT PLAN

Version 2.0

Date: 26 - Mar - 2021

## EXPERT-DRIVEN SMART DASHBOARD APPLICATION

#### Submitted by

Vo Van Hoa  
Pham Van Tin  
Ky Huu Dong  
Tran Thi Thanh Kieu

#### Approved by

##### Capstone Project 2 - Mentor:

Name

Signature

Date

Binh, Thanh Nguyen \_\_\_\_\_

A handwritten signature in blue ink, appearing to read 'Ng/Nguyn' followed by a stylized surname.

\_\_\_\_\_ 26 - Mar - 2021

PROJECT INFORMATION			
<b>Project Acronym</b>	EDSDA		
<b>Project Title</b>	Expert-Driven Smart Dashboard Application		
<b>Project Web URL</b>	<a href="https://sda-research.ml/">https://sda-research.ml/</a>		
<b>Start Date</b>	01 - Mar - 2021		
<b>End Date:</b>	01 - Jun - 2021		
<b>Lead Institution</b>	International School, Duy Tan University		
<b>Project Mentor</b>	Ph.D Binh, Thanh Nguyen		
<b>Scrum Master</b>	Hoa, Vo	hoavo.dng@gmail.com	0935.193.182
<b>Team Members</b>	Tin, Pham Van	tinphamvan123@gmail.com	0932.535.175
	Dong, Ky Huu	kyhuudong@gmail.com	0898.246.980
	Kieu, Tran Thi Thanh	thanhkieutran391@gmail.com	0358.583.251

DOCUMENT INFORMATION			
<b>Document Title</b>	Project Plan		
<b>Author(s)</b>	Team C2SE.06		
<b>Role</b>	[EDSDA] ProjectPlan_v2.0		
<b>Date</b>	01 - Mar - 2021	Filename	[EDSDA] 002 Project Plan
<b>URL</b>	<a href="https://github.com/sdateamdtu2020/SDA-v2.0">https://github.com/sdateamdtu2020/SDA-v2.0</a>		
<b>Access</b>	Project and CMU Program		

## REVISION HISTORY

Version	Person(s)	Date	Description
Draft	Hoa, Vo	01 - Mar - 2021	Initiate document
2.0	All members	14 - Mar- 2021	Finish content of the document
2.1	All members	25 - May - 2021	Update content, fix typo

# TABLE OF CONTENTS

PROJECT INFORMATION	1
DOCUMENT INFORMATION	1
REVISION HISTORY	2
<b>TABLE OF CONTENTS</b>	<b>4</b>
<b>I. PROJECT OVERVIEW</b>	<b>6</b>
1.1 PROJECT DESCRIPTION	6
1.2 SCOPE AND PURPOSE	6
1.3 ASSUMPTIONS AND CONSTRAINTS	7
1.4 PROJECT OBJECTIVES	9
1.5 CRITICAL DEPENDENCIES	12
1.6 PROJECT RISK	13
<b>II. PROJECT DEVELOPMENT APPROACH</b>	<b>14</b>
2.1 PROJECT PROCESS	14
2.2 REQUIREMENT CHANGE MANAGEMENT	17
2.3 PROJECT INTEGRATION STRATEGY	18
2.4 QUALITY MANAGEMENT	19
2.5 UNIT TESTING STRATEGY	23
2.6 INTEGRATION TESTING STRATEGY	23
2.7 SYSTEM TESTING STRATEGY	24
<b>III. ESTIMATE</b>	<b>24</b>
3.1 SIZE	24
3.2 EFFORT	25
3.3 SCHEDULE	26
3.4 RESOURCE	30
3.5 INFRASTRUCTURE	31
3.6 TRAINING PLAN	32
3.7 FINANCE	33
<b>IV. PROJECT ORGANIZATION</b>	<b>34</b>
4.1 ORGANIZATION STRUCTURE	34
4.2 PROJECT TEAM	35
4.3 EXTERNAL INTERFACES	35
<b>V. COMMUNICATION &amp; REPORTING</b>	<b>36</b>
5.1 REPORTING METHODOLOGY	36
5.2 COMMUNICATION METHODOLOGY	36

<b>VI. CONFIGURATION MANAGEMENT</b>	<b>37</b>
<b>VII. SECURITY ASPECTS</b>	<b>37</b>
<b>VIII. REFERENCE</b>	<b>37</b>

# 1. PROJECT OVERVIEW

## 1.1. PROJECT DESCRIPTION

<b>Project code</b>	C2SE.06	<b>Contract type</b>	Scientific Research
<b>Customer</b>	International School, Duy Tan University	<b>End-User</b>	Environmental Experts, Users
<b>Project Type</b>	External	<b>Scrum master</b>	Hoa, Vo Van
<b>Project Category</b>	Development & Maintenance	<b>Business domain</b>	Data Visualization, Environment,...
<b>Application type</b>	Web Application		

## 1.2. SCOPE AND PURPOSE

### 1.2.1. SCOPE

The backbone of the project :

- Import data sources.
- Auto ETL
- Auto RDF Generation.
- Auto linking data.
- Drag data cubes from the cubes list.
- Drop data cubes onto the main content board.
- Connect the data cubes that have the relation between them.
- Merge the cubes to create a new data cube based on connected data cubes.
- Visualize the data cube as a line chart, a pie chart, a column chart, or a geographical map.

Language:

- English

Duration: 13 weeks.

### 1.2.2. PURPOSE

Expert-Driven Smart Dashboard - means a dashboard that is convenient for users to analyze and review data. It will include several datasets about real-time information of the environment, measurement data of air pollutants... EDSDA will connect and analyze data from multiple sources in ways you've never imagined. Then reveal the insights you've been missing...in just a matter of minutes.

With smart suggestions and an intuitive visual interface, SDA makes it easy for any user to combine data and discover hidden insights in one place...without the usual scripting, coding, and IT hand-holding. With this dashboard, individuals or any subjects can take advantage of environmental data to be able to decide the best relevant policies.

Not only for the environmental expert, but EDSDA also helps the user directly interact with their own data source by importing, automatically cleaning their data, and enabling the user to do everything with their own data in the easiest way.

### 1.3. ASSUMPTIONS AND CONSTRAINTS

No	Description	Note
<b>Assumptions</b>		
1	Auto ETL and RDF generation may not be able to finish on time	Scope
2	The system will be uploaded to the web app for a public user acceptance test	External Interfaces
3	The system will be tested with pre-prepared standard data sources	Data Sources
<b>Constraints</b>		
1	EDSDA must be done and delivered by June 30th according to the customer demonstration deadline	Schedule
2	The project must conform to the system architecture specification and end-user requirements	Consistency
3	The project must comply with information security rules	Security

CONSTRAINTS	CONSTRAINTS DESCRIPTION	GUIDELINES FOR ACCEPTANCE
Economic	In terms of cost, since it is a dashboard with not too many screens, the main issue lies in the cost of researching, implementing a fully automated backend system and frontend with smart methods. There is also the cost of renting a server and deploying a server that should be considered carefully	Design cost: Must be around \$100. Production cost: Should be under \$4000. Maintenance cost: Should be around \$500. Operation cost: Should be under \$1000
Ethical	Because this software analyzes data with most of the data being crawled, so pay attention to copyright and data security.	All crawl data sources must be open data and be allowed for public use.
Social and Global	The software is targeted to not only environmental experts but also data analysts in many other fields, so widespread popularity is very important.	The product needs to be developed in an optimal and user-friendly way to reach a wide range of users
Sustainability	Need to maintain the continuous operation of the system, so as not to affect the analysis of people	Development and maintenance work must be ensured to take place continuously, when issues are reported, it is necessary to focus on maintenance immediately. Server operation also needs to be ensured not to be interrupted.

## 1.4. PROJECT OBJECTIVES

### 1.4.1. Standard Objectives

Metrics	Unit	Committed	Note
Start Date	dd-mmm-yy	01/Mar/21	
End Date	dd-mmm-yy	01/Jun/21	
Duration	elapsed days	91 days	
Maximum Team Size	Person	4	
Billable Effort	Person-day	\$10/day	
Calendar effort	Person-day	\$10/day	
Effort Usage	Person-day	364	

### 1.4.2. Specific Objectives

- Build a dashboard with drag-and-drop intelligence yet user-friendly.
- Implement a data warehouse that has a fully automated ETL method.
- Build a data import system and automatically clean data to help users automatically clean and interact with their own data source.
- The engine to automatically generate multidimensional RDF data cubes from the data warehouse must be built.
- The GraphDB database management system must operate continuously and allow data access up to 10000 rows.

### 1.4.3. Quality Attributes

ID	QA01
<b>Quality Attributes</b>	Performance
<b>Stimulus</b>	Uses the statistics merge operator for merging multiple data cubes.
<b>Source(s) of stimulus</b>	User
<b>Artifacts</b>	System
<b>Environment</b>	Normal mode
<b>System response</b>	The SDA returns a new data cubes as fast as possible
<b>Response measure(s)</b>	Within 3 seconds

**Table 3.3.1:** Quality Attributes: Performance

ID	QA02
<b>Quality Attributes</b>	Availability
<b>Stimulus</b>	The power is off while server is running
<b>Source(s) of stimulus</b>	Power
<b>Artifact</b>	During peak usage load
<b>Environment</b>	Hardware and software
<b>System response</b>	The system will use the cloud server to save the work so we don't need to worries about power incident occurred
<b>Response measure(s)</b>	All work always can be saved

**Table 3.3.2:** Quality Attributes: Availability

ID	QA03
<b>Quality Attributes</b>	Availability
<b>Stimulus</b>	Can't get specific data cubes when selecting it from the widget
<b>Source(s) of stimulus</b>	User
<b>Artifact</b>	System
<b>Environment</b>	Normal mode
<b>System response</b>	The system will log the fault immediately
<b>Response measure(s)</b>	Within immediately

**Table 3.3.2:** Quality Attributes: Availability

ID	QA03
<b>Quality Attributes</b>	Portability and compatibility
<b>Stimulus</b>	Open the browser and access <a href="http://sda-research.ml/">http://sda-research.ml/</a>
<b>Source(s) of stimulus</b>	User
<b>Artifact</b>	System
<b>Environment</b>	Normal mode
<b>System response</b>	The system can run on any web browser.
<b>Response measure(s)</b>	Within immediately

**Table 3.3.2:** Quality Attributes: Portability and compatibility

ID	QA03
<b>Quality Attributes</b>	Security
<b>Stimulus</b>	User access to EDSA and import a file
<b>Source(s) of stimulus</b>	User
<b>Artifact</b>	System
<b>Environment</b>	Normal mode
<b>System response</b>	The system will not track any individual information about the user
<b>Response measure(s)</b>	Within immediately

**Table 3.3.2:** Quality Attributes: Availability

## 1.5. CRITICAL DEPENDENCIES

No	Dependency	Expected delivery date	Note
1	Cloud Server	May 15th	put all the necessary modules on the server cloud to ensure the continuity of the system
2	GraphDB	April 30th	Store and retrieve linked data warehouses as RDF Data Cubes
3	Sample Data Sources	April 20th	Used to test import methods and automate data source ETL
4	Open data sources	April 10th	Exploiting open data sources of topics to create a stock of available data

## 1.6. PROJECT RISK

This part of the document contains several risks that could happen to the development team in the future. It also includes probability, severity, and mitigation strategy for each risk.

Risk	Definition	Probability	Severity	Mitigation Strategy
Lack of coding experiences	No one in team member work with Python, React, Express, Data Warehouses, Data Cubes, Crawl data	3	2	Each team member has to learn and help the other to learn quickly.
Source Code conflict	Problems while merging code between members to master branch	3	3	Each team member must resolve conflicts by using git merge CLI before merging to the master branch.
Member conflict	Team member maybe conflict with each other while discussing	3	2	Team building, playing board games to get everyone together.
Less equipment	No machine or hosting for deploying the server.	1	2	Try free hosting for deployment.
Time management	Every member has to go to work or school.	3	3	Overtime
Language barrier	Most of the documents the are in English, sometimes it hard to understand clearly the articles and the information	3	3	Improve the individual English skills while doing the project. Asking the mentor technology for specific

Probability		Severity	
1	Rarely happened.	0	Low damaged
2	Sometime happened	1	Medium damaged
3	Usually happened	2	Serious damaged

## 2. PROJECT DEVELOPMENT APPROACH

### 2.1. PROJECT PROCESS

#### 2.1.1. Reasons for selecting

- Our team has 4 people
- The project will be continuously horizontally scaled up.
- There is only a short amount of time to finish the project.

So based on those constraints, we decided to choose SCRUM as the project lifecycle.

#### 2.1.2. Agile Methodology

##### PRINCIPLE AND DIFFERENT STAGES

The SCRUM methodology relies on the incremental development of a software application while maintaining a completely transparent list of upgrade or correction demands to be implemented (backlog). It involves frequent deliveries, usually every four weeks, and the client receives a perfectly operational application that includes more and more features every time. This is why the method relies on iterative developments at a constant rhythm of 2-4 weeks. Upgrades can therefore be more easily integrated than when using a V-cycle.

This method requires four types of meetings:

- Daily meetings: the entire team meets for approximately 15 minutes every day in order to answer the following three questions, usually while standing: what did I do yesterday? What am I going to do today? Is there a cumbersome impediment today?
- Planning meetings: the entire team gathers to decide on the features that will make up the following sprint
- Work review meetings: during this meeting, every member presents what he has done during the sprint. They organize a demonstration of the new features or a presentation of the architecture. This is an informal meeting lasting for approximately 2 hours which is attended by the entire team.
- Retrospective meetings: at the end of each sprint, the team analyzes both successful and unsuccessful elements of their activity. During this meeting lasting between 15 and 30 minutes where everyone is invited and speaks on their own behalf, a vote of confidence is organized in order to decide on the improvements to be made.

The advantage of this method consists in reducing the documentation to the minimum in order to gain productivity. The idea is to write only the minimum documentation

which allows to save the history of the decisions taken on the project and to easily perform interventions on the software when it goes into the maintenance phase.

## AGILE - SCRUM ORGANISATION

The SCRUM methodology involves the following three main players:

- Product owner: In most projects, the product owner is the leader of the client's project team. He is the one who will define and prioritize the product features and choose the date and content of each sprint based on values (workloads) that the team communicates to him.
- Scrum Master: He is a genuine facilitator on the project as he makes sure that everyone works at their full potential by eliminating impediments and protecting the team from external interference. Moreover, he pays particular attention to the respect of the different SCRUM phases.
- Team: A team is typically made up of 4-10 people and groups together all the IT specialists who are necessary on a project, i.e. an architect, a designer, a developer, a tester, etc. The team is self-organizing and remains unchanged during an entire sprint.

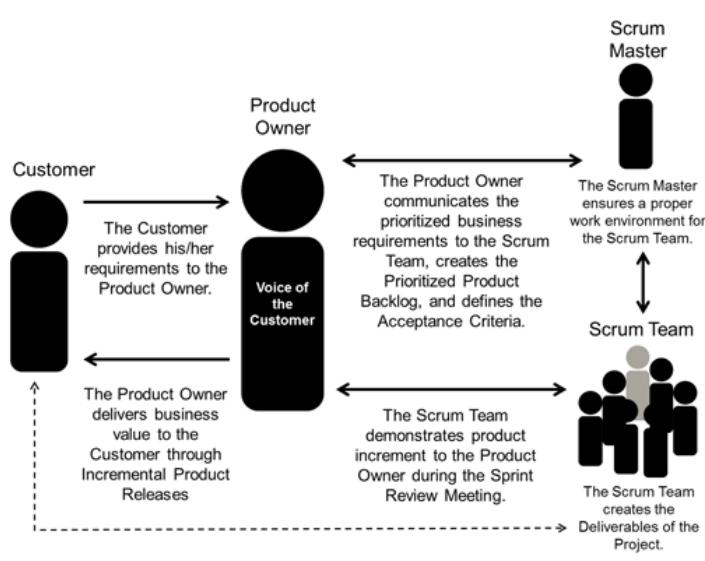


Figure 1: Scrum team members

## AGILE - SCRUM ADVANTAGES

Scrum differs from other development methods through its advantages which turn it into a pragmatic response to product owners' current needs:

Iterative and incremental method: this allows to avoid the "tunnel effect", i.e. the fact of seeing the result only at the final delivery, and nothing or almost nothing during the entire development phase, which is so frequent with V-cycle developments.

Maximum adaptability for product and application development: the sequential composition of the sprint content allows the addition of a modification or a feature that was not initially planned. This is precisely what renders this method "agile".

- Participatory method: every team member is asked to express his opinions and can contribute to all the decisions taken on the project. He is therefore more involved and motivated.
- Enhancing communication: by working in the same development room or being connected through different communication means, the team can easily communicate and exchange opinions on the impediments in order to eliminate them as early as possible.
- Maximizing cooperation: daily communication between the client and the team enables them to collaborate more closely.
- Increasing productivity: as it removes certain "constraints" of the classical methods, such as documentation or exaggerated formalization, SCRUM allows for increased team productivity. By adding to this the qualification of each module which allows determining an estimation, everyone can compare their performance to the average team productivity.

### 2.1.3. Scrum Process

Scrum is an agile method, so it follows the principles of Agile Manifesto ([see also Agile Manifesto](#)). In addition, Scrum operates on three core values, also known as Scrum, including Scrutiny, Inspection, and Adaptation.

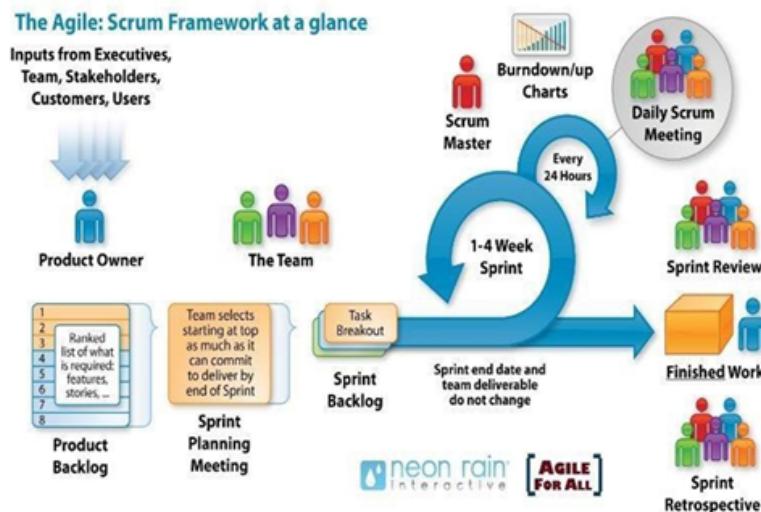


Figure 2: Scrum team members

Based on the empirical process control theory, Scrum uses iterative and incremental algorithms to optimize efficiency and control risk. Scrum is simple, easy to learn, and has wide applicability. To be able to use Scrum, we need to understand and apply the elements that makeup Scrum include the core values (also known as the "three legs", or the three pillars of Scrum), roles, Events, and Scrum-specific artifacts.

## 2.2. REQUIREMENT CHANGE MANAGEMENT

We use Ince's Change Process Model to handle the required changes. Ince's model focuses on how software configuration management relates to software change management. This model has two main sources of change requests, i.e. customer and development team as shown in Figure 1. In order for the change process to be initiated, a change request must be initiated in a software project. All such change requests are recorded in a change request note. The change control board then considers the suggested change. The change control board can reject the change (the change will not take place), batch the change (the change will take place but not immediately), or accept the change (the change is to be implemented at the earliest possible time). If the request for the change is successful, a change authorization note must be filled. After this, the change can be implemented and a system's documentation is modified. After implementation, the change is validated. Validation and test records are then produced to document the changes that have taken place. Finally, the configuration records are updated and the staff is informed about the new changes.

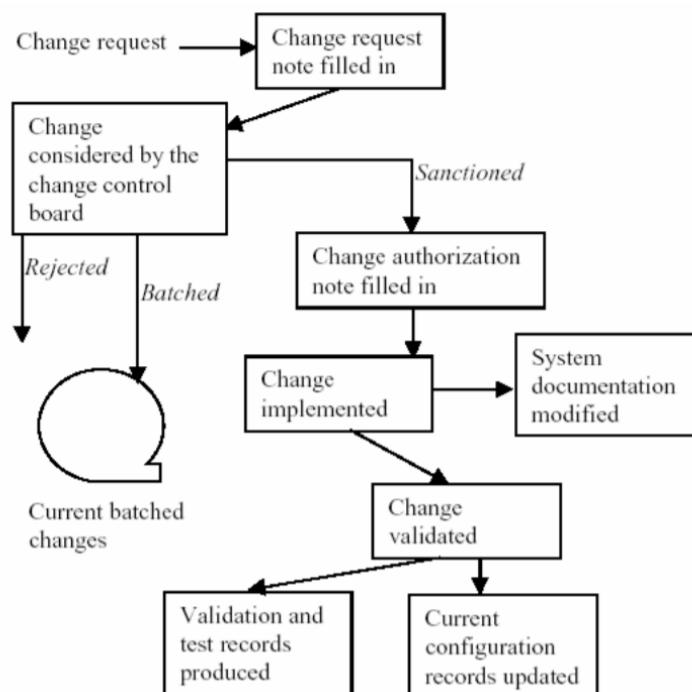


Figure 3: Ince's change process model

## 2.3. PRODUCT INTEGRATION STRATEGY

The project is integrated with Continuous Integration (CI) methods.

- **Develop** describes the practices necessary to implement stories and commit the code and components to version control.
- **Build** describes the practices needed to create deployable binaries and merge development branches into the trunk.
- **Test end-to-end** describes the practices necessary to validate the solution.
- **Stage** describes the practices necessary to host and validate the solution in a staging environment before production.

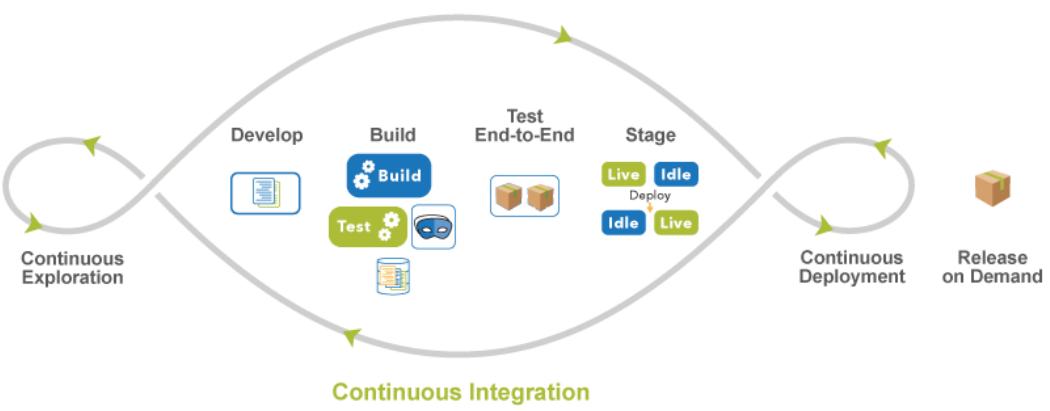


Figure 4: Continuous Integration

We decided to implement the CI method for EDSA with Bitbucket Pipeline. Bitbucket Pipelines is an integrated CI/CD service, built into Bitbucket. It allows users to automatically build, test, and even deploy code based on a configuration file in their repository.

## 2.4. QUALITY MANAGEMENT

### 2.4.1. Estimates of Defects to be detected

#### Pre-release review defects

Process	Planned found by review	Actual found by review
Requirement	<ul style="list-style-type: none"> <li>- Change business requirement</li> <li>- Change project architectural design requirements</li> <li>- Shortcomings in defining use cases and user stories</li> </ul>	<ul style="list-style-type: none"> <li>- Change business requirement</li> <li>- Change project architectural design requirements</li> <li>- Shortcomings in defining use cases and user stories</li> <li>- Change requirements on data processing and storage methods</li> </ul>
Amount of work	<ul style="list-style-type: none"> <li>- A lot of work causes overload for programmers who have a job</li> </ul>	<ul style="list-style-type: none"> <li>- A lot of work causes overload for programmers who have a job</li> </ul>
Programming tools	<ul style="list-style-type: none"> <li>- Inappropriate use of programming tools</li> </ul>	
Technologies stack	<ul style="list-style-type: none"> <li>- Inappropriate use of technologies stack</li> <li>- Some technologies are no longer supported</li> <li>- Lack of new technology</li> </ul>	<ul style="list-style-type: none"> <li>- Inappropriate use of technologies stack</li> <li>- Some technologies are no longer supported</li> <li>- Lack of new technology</li> <li>- Difficulty in accessing paid technologies</li> </ul>
Reference studies	<ul style="list-style-type: none"> <li>- Shortage of research articles related to the system</li> <li>- Articles often have a large academic volume and are difficult to apply in practice</li> </ul>	<ul style="list-style-type: none"> <li>- Shortage of research articles related to the system</li> <li>- Articles often have a large academic volume and are difficult to apply in practice</li> <li>- Difficult to access internal research works</li> </ul>

Process	Planned found by review	Actual found by review
Design	- Old version UI is hard to reuse to extend the functionality	- Old version UI is hard to reuse to extend the functionality - Completely change the UI of the system
Automatic processing method	- Lack of supporting technologies and tools	- Lack of supporting technologies and tools - Data processing results are not quite as expected
Coding	- Difficult to implement - Use different languages for each module	- Difficult to implement - Use different languages for each module - Dirty code
System integration	- The difficulty of creating a fully automated pipeline to run different modules - It is very difficult to integrate quickly because the system is built on many different programming languages and technologies	- The difficulty of creating a fully automated pipeline to run different modules - It is very difficult to integrate quickly because the system is built on many different programming languages and technologies
Deliver on time	- Large workload leads to late deadline - Difficulty in implementation causing slow progress	- Large workload leads to late deadline - Difficulty in implementation causing slow progress
Other	- Lack of money to maintain the cloud server and organize weekly meetings	- Lack of money to maintain the cloud server and organize weekly meetings
<b>Total</b>	<b>18</b>	<b>24</b>

### 2.4.2. Strategy for Meeting Quality Objectives

Strategy	Expected Benefits
Use standards and design patterns to implement, prevent unnecessary defects during system installation	5-10% reduction in defect injection rate and about 2% improvement in productivity
Public project to perform public user acceptance test, ensure test samples are verified with a large number of users to evaluate system performance and find potential defects	Find out most defects in UI/UX and data processing, take corrective measures as soon as possible, through which can improve the system quality in the best way
Write as many test cases as possible, make sure to cover all test cases	Ensure that the testing process is continuous without being forced at the end of the project, increasing the quality of each module of the system.
Use rolling wave planning to execute test cases continuously during development	Approximately 5% reduction in defect injection rate and 1% improvement in overall productivity
Use proven technologies, absolutely do not use tools that are no longer supported	Ensure the continuity and scalability of the system, avoiding the case of sudden system death when technologies stop supporting

### 2.4.3. Quality Control

Review Item	Type of Review	Reviewer	When
System Requirement Specification	One-person review	Scrum Master	Before the project kick-off meeting
Scope, Objective, and Goal	One-person review	Scrum Master	End of requirements 90%
System architecture specification	Group review	Scrum Master, Developer	After finishing System Requirement Specification
UI/UX Design	Group review	Scrum Master, Developer	After finishing the System architecture specification

Review Item	Type of Review	Reviewer	When
Project Plan, Scrum Schedule	Group review	Scrum Master, Developer, Tester	After finishing the System architecture specification
Resource allocation	One-person review	Scrum Master	After finishing Project Plan
Design document, object model	Group review	Scrum Master, Developer, Tester	End of 90% design
Database design	Group review	Scrum Master, Developer	After finishing object model design
Testing plan	Group review	Scrum Master, Tester	End of Sprint 0
Code	Group review	Scrum Master, Developer	After each module built

#### 2.4.4. Measurements Program

Data to be collected	Purpose	Responsible	When
Size: No. of KLOC// FP	Group review or One-person review	PM/SM	At the end of stages
Effort: No. person-day		Team members	Daily
Quality: No. defects detected		Reviewer, Tester	Right after the review/test
Schedule		PM/SM	Weekly and at the end of stages

## 2.5. UNIT TESTING STRATEGY

In this project, most of the functionality is highly specific, so we focus on using manual testing. Besides, there are still some functions that are repeated over and over again, for example:

- Import data source.
- Auto ETL process.
- Auto-generate RDF Data Cubes
- Export report.

We use automation tests with functions that are repeated above.

**Tool:**

- Manual Test: Google Sheet, Trello.
- Automation Test: Cypress.

There are principles while doing unit tests:

- For each class, there should be a test class that tests all the public methods.
- Tests cover at least positive tests and negative tests.
- Dependencies to other classes should be substituted by mock objects.
- Each test case covers exactly one functionality to achieve a quick bug fixing.
- For each abstract class, an abstract test class will be implemented. This abstract test class tests the implementation parts of the abstract class and outlines the correct use of the abstract class and the test classes to implement for the concrete classes.

**Requirements:**

- At least 70% of lines of code get coverage.
- Unit tests must be done once before the tester executes test cases in each sprint.
- Unit tests must be performed consistently, ensuring that there is no fluctuation in the number of unit tests per module.

## 2.6. INTEGRATION TESTING STRATEGY

Testers must design a test plan for integration testing with both Sub-component integration testing and Component integration testing because a component is typically composed of a set of sub-components and these sub-components consist of classes and packages.

- Sub-component integration testing: The integration tests between sub-components ensure compatibility of these software units across development cycles. An example of a sub-component integration test in the

Design Environment is the validation of outputs from the Modelling Library against the expected inputs of the Modelling Tool.

- Component integration testing: These tests will examine the compatibility of components' APIs according to a predefined set of rules. Furthermore, they examine the basic functionalities of a group of components as a sub-system. An example of such tests is between Design and Algorithm Optimization environments, where the outputs of the Modelling Tool will be tested against expected inputs of the Optimization Tool.

## 2.7. SYSTEM TESTING STRATEGY

For system testing, we use the following technique:

- Usability Testing: Test focus on user-friendly, easy-to-interact interface, flexibility in handling controls, and ability of the system to meet project objectives.
- Load Testing: Check whether the performance of the system is balanced and stable, whether the criterion is met in quality attribute or not.
- Functional Testing: provide missing functionality, functions list that testers think are likely to affect the product.

## 3. ESTIMATE

### 3.1. SIZE

#### 3.1.1. Lines of code:

Modules	LOC
UI	1300
Database Design	550
Data Clean Engine	900
ETL Process	1000
RDF Automate Generator	800
GraphDB API	600
CI/CD Pipelines	200
<b>Total</b>	<b>5350</b>

### 3.2. EFFORT

Planned Resources				
	Min man-days	Max man-days	Quantity	Position Title
Total Development	133	200	3	Software Engineer
Total QA	30.5	44	1	Tester
Total Data Science	40	45.5	2	Data Engineer
Total UI/UX Design	20	25	1	Designer
Total UI Implementation	40	45	1	Front-end Developer
Total BA	12	16	1	Business Analyst
Total PM	30	44	1	Project Manager
Total CI/CD	20	30	1	DevOps

### 3.3. SCHEDULE

#### 3.3.1. Project Milestone & Deliverables

##### Project Milestone

EDSDA - Project Timeline

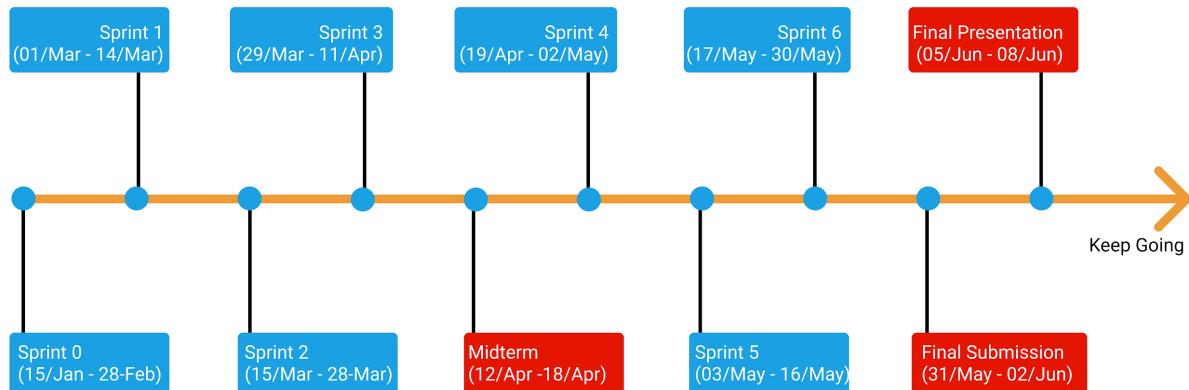


Figure 3.1: Project Timeline

##### DELIVERABLES

No	Activities	Deliverables
0	Requirements Description	Requirements Description 2.1
1	Project Proposal	Project Proposal Document 2.1
2	Product Requirement	Product Requirement Document 2.1
3	Project Plan	Project Plan Document 2.1
4	Product Backlog	Product Backlog Document 2.1
5	Architecture Document	Architecture Document Document 2.1
6	Database Design	Database Design Document 2.1
7	Interface Design	Interface Design Document 2.1
8	Test Plan	Test Plan Document 2.1
9	Test Case	Test Case Document 2.1
10	Acceptance Criteria	Acceptance Criteria Document 2.1
11	Sprint Backlog & Burndown Chart	Sprint Backlog & Burndown Chart Document 2.1

No	Activities	Deliverables
12	Team Reflection	Team Reflection Document 2.0
13	Technologies Stack	Technologies Stack Document 2.0

### 3.3.2. Work Breakdown Structure

WBS NUMBER	TASK TITLE
1	Preparation
2	Data Modeling
3	Physical Warehouse Design
4	Initiate Dynamical Data Warehouse
5	Finding Data Sources
6	Data processing
7	ETL Process Validating
8	Automate ETL Setting
9	Expanding database
10	Crawl data
11	Building Data Cubes Importing method
12	Building additional features
13	Common RDF Data Cubes Structure Designing
14	Setting up Automate DW to RDF Process Tool & Environment
15	Implementing Automate DW2RDF Process
16	Setting up RDF Data Cubes Storing & SPARQL Endpoint
17	Validating RDF Data Cubes
18	Building SPARQL_Rest API
19	Building UI
20	Testing
21	Integrate
22	Deploy
23	Release

### 3.3.3. Detailed Schedule

WBS	Task	Start	End	Days
	<b>Sprint 1</b>	<b>01-Mar-21</b>	<b>14-Mar-21</b>	<b>14</b>
1	Crawl data			
2	Clean data			
3	Initiate RML process			
4	Initiate MIRROR			
5	Initiate GraphDB Database			
6	UI Design			
7	Initiate & config React project			
8	Testing			
	<b>Sprint 2</b>	<b>15-Mar-21</b>	<b>28-Mar-21</b>	<b>14</b>
1	Create DW			
2	ETL Process using a tool			
3	Define a common mapping for RML & MIRROR			
4	Implement API for importing data, getting data from GraphDB			
5	Building UI			
6	Testing			
	<b>Sprint 3</b>	<b>29-Mar-21</b>	<b>11-Apr-21</b>	<b>14</b>
1	Getting, and Cleaning imported data			
2	ETL the imported data			
3	Describe common schema for RDF Data Cubes			
4	Run RML & MIRROR with sample data			
5	Integrate sample UI & API			
6	Building UI			
7	Testing			
	<b>Midterm</b>	<b>12-Apr-21</b>	<b>18-Apr-21</b>	<b>7</b>
1	Deploy version 0.1			
2	Review documentation			

WBS	Task	Start	End	Days
	<b>Sprint 4</b>	<b>19-Apr-21</b>	<b>02-May-21</b>	<b>14</b>
1	Automating clean data			
2	Schedule for ETL data pipeline			
3	Customize RML & MIRROR			
4	Implement pipeline for automating RDF Data Cube generation			
5	Building UI			
6	Testing			
	<b>Sprint 5</b>	<b>03-May-21</b>	<b>16-May-21</b>	<b>14</b>
1	Handling new importing data to DW(Dimensions, Measures)			
2	Automating ETL for new importing data			
3	Integrate automate ETL & RDF generation			
4	Automating detect dimension & measure			
5	Building UI			
6	Testing			
	<b>Sprint 6</b>	<b>17-May-21</b>	<b>30-May-21</b>	<b>14</b>
1	Automating crawling data from NGO, or any new defined data source.			
2	Cleaning data process for the new defined data source.			
3	Set up VPS for deploying			
4	Migrate GraphDB to VPS			
5	Building UI			
6	Integration			
7	Testing			
	<b>Final</b>	<b>31-May-21</b>	<b>08-Jun-21</b>	<b>9</b>
1	Review documentation			
2	Release / Deploy final version			
3	Presentation			

## 3.4. RESOURCE

### 3.4.1. Human Resources:

Position Titles	Quantity	Member
Backend Developer	2	Dong Ky, Hoa Vo
Frontend Developer	1	Tin Pham
Data Engineer	2	Dong Ky, Hoa Vo
Graphics Designer	1	Tin Pham
Business Analyst	1	Hoa Vo, Tin Pham
Tester	1	Kieu Tran
Project Manager	1	Hoa Vo
DevOps	1	Hoa Vo
<b>Total Human-Resources</b>	<b>10</b>	

### 3.4.2. Equipment

Name of equipment	Quantity
Laptop	4
Monitor	4
VPS Server	1
Tablet	1

### 3.5. INFRASTRUCTURE

Work/Product	Purpose	Expected Availability by	Note
<b>Development Environment</b>			
Win 10, Linux	Operating Systems	Initiation stage	
Postgresql	DBMS	Initiation stage	
Javascript	Development language for Web interface and API server	Initiation stage	
Python	Development language for data processing, Data warehouse generating, crawl data, and handling data sources.	Initiation stage	
<b>Hardware &amp; Software</b>			
Cloud server	2vCPUs, 1GB memory, 10GB storage, ubuntu v20.04 minimal		
Figma	UI Design & Architecture design		
<b>Other Tools</b>			
Github, bitbucket	Source version control	Initiation stage	
Cypress	Unit Test	Construction stage	
Googlesheet	Effort logging	Initiation stage	
Slack	Discussion	Initiation stage	
Trello	Task tracking	Initiation stage	
Google Office	Managing documents	Initiation stage	
Discord	Meeting online	Initiation stage	

### 3.6. TRAINING PLAN

Training Area	Participants	When, Duration	Waiver Criteria
<b>Technical</b>			
Python Language		7 days	If already trained
Javascript	Kieu	7 days	if already trained
<b>Business domain</b>			
Environment		7 days	
Data Visualization	Tin, Hoa, Dong	7 days	
<b>Process</b>			
Quality system	All members	3 hrs	If already trained
Configuration management	All members	2 hrs	If already trained for CC. For others, on-the-job training
Group review	All members	4 hrs	If already trained
Defect prevention	All members	4.5 hrs	Mandatory

#### 3.6.1. Compulsory training program:

- RDF Data Cubes Introduction courses.
- Data Warehouse in Advanced.
- Automate ETL tutorial.
- ReactJS DnD.
- ReactJS Flow.
- Redux.
- Testing Techniques Course.
- Automation Testing tutorial.
- How to apply CI/CD with Bitbucket Pipeline.

### 3.6.2. Additional training program:

- The way of working on a Scrum project.
- Release Planning Meeting
- Sprint Planning Meeting
- Daily Scrum Meeting
- Sprint Review Meeting
- Retrospective
- Scrum of Scrums

## 3.7. FINANCE

COST DESCRIPTION	DETAIL
Salary	Duration: 16 weeks (112 days) Man-hour: 5hours/1day Salary: \$2/1 hour Persons: 4 members Overtime cost per hour: 1.5\$/hour The salary of 1 person: \$1000 <b>Total: \$4000</b>
Laptop	Laptop for each member: \$800 <b>Total: \$3200</b>
Monitor	LG Monitor for each member: \$100 <b>Total: \$400</b>
VPS Engine server	VPS Engine server per month on Google Cloud: \$100 VPS Engine server for 3 months: 60\$ <b>Total: \$300</b>
Maintenance	Cost per month: \$400
Bugs fixing	Cost per bug: \$20
<b>Total</b>	Salary + Laptop + Monitor + VPS Engineer server <b>Total: \$7900</b>

## 4. PROJECT ORGANIZATION

### 4.1. ORGANIZATION STRUCTURE

Role	Responsibility	Name
Product Owner	Understand the user and customers with their needs. Collaborate with the development team. Manage the stakeholders. Describe the user experience and product features. Provides detailed user stories.	PhD Binh Thanh Nguyen
Scrum Master	Communicate the value of Scrum Teach the organization on Scrum to maximize business value Facilitate Sprint Planning, Daily Scrums, Sprint Reviews, and Retrospective Meetings Create the Task Board and Sprint Burndown Chart at the start of every Sprint Attend all Scrum meetings Preserve the integrity and spirit of the Scrum framework Maintain the focus of the Team Make the Team aware of impediments and facilitate efforts to resolve them Serve as a coach and mentor to members of the Team Respectfully hold the Team, Product Owner, and Stakeholders accountable for their commitments Continually work with the Team and business to find and implement improvements	Hoa, Vo Van
Secretary	Record the content of group meetings and activities of the member	Tin, Pham Van
Reviewer	Analysis of the functions and requirements of the product. Review documents related to the project	Hoa, Vo Van
Team Leader	DevOps, Back-end Dev: Server, RDF Data Cubes, B.A	Hoa, Vo Van
Team member	Frontend Developer, UI/UX Designer, DevOps	Tin, Pham Van
Team member	Back-end Dev: Database, Crawl	Dong, Ky Huu
Team member	Frontend Developer, Tester	Kieu, Tran Thi Thanh

## 4.2. PROJECT TEAM

Full Name	Email	Phone number	Role
Hoa, Vo	hoavo.dng@gmail.com	0935.193.182	Scrum master
Tin, Pham Van	tinphamvan123@gmail.com	0932.535.175	Team member
Dong, Ky Huu	kyhuudong@gmail.com	0898.246.980	Team member
Kieu, Tran Thi Thanh	thanhkieutran391@gmail.com	0358.583.251	Team member

## 4.3. EXTERNAL INTERFACES



Figure 5: UI Design of EDSA

## 5. COMMUNICATION & REPORTING

### 5.1. REPORTING METHODOLOGY

Audience/ Attendees	Topic/ Deliverable	Frequency	Method
- Product Owner - Scrum Master - Team Members	Project Progress Review	Weekly	Email, Skype
- Product Owner - Scrum Master - Team Members	Explicit Requirement	When needed	Email, Skype
- Mentor - Scrum master - Team members	Milestone review	End of each Milestone	Skype
- Scrum master - Team members	Daily tasks	Each day	Slack, Discord, Trello

### 5.2. COMMUNICATION METHODOLOGY

Type of Communication	Communication Schedule	Communication way	Who Initiates	Recipient
Status Report (Daily meeting)	Daily at 10AM	Slack	Scrum Master	Scrum Team
Schedule and Effort Tracking	Daily	Google Sheets, Trello	Scrum Master	Scrum Team
Work Review	Daily at 10AM	Trello	Scrum Master	Scrum Team
Work Report	Every Sunday	Slack or Face to Face	Scrum Master	Scrum Team
Project Review, ask problems	Every Monday (flexible)	Skype	Scrum Master	Scrum Team, Mentor
Ask & Review problems	Anytime	Face to face, Slack, Discord, Skype	Scrum's Member	Mentor and Scrum Team

## 6. CONFIGURATION MANAGEMENT

No	Tool	Content
1	Google Sheet	Track member activities. At the end of each day, team members will post on time log and Scrum Master will check.
2	Google Document	Track the changing of documents & manage versions of documents.
3	Bitbucket	Repositories for source code version management.
4	Weekly Meeting	Hold a meeting every week to assign tasks to each member. If there are some emergencies but we cannot sit together then we can use Discord to discuss online.
5	Document	All meetings must be documented and pictured.
7	Slack	Store document resources and designed components, Daily Scrum
8	Discord	Discuss online, stream and share problems
9	Github	Repositories for open-source code of the project

## 7. SECURITY ASPECTS

To deal with security risks, EDSA is precisely process-configured at every stage of the project.

- Concept and planning

The purpose of this stage is to define the application concept and evaluate its viability. This includes developing a project plan, writing project requirements, and allocating human resources.

- Prepare a list of security requirements for the project.
- Training sessions provide essential security knowledge ranging from basic threat awareness to in-depth information on secure development.

- Architecture and design

The purpose of this stage is to design a product that meets the requirements. This includes modeling the application structure and its usage scenarios, as well as choosing third-party components that can speed up development. The result of this stage is a design document.

- Secure design :The design document and subsequent updates are validated in light of the security requirements. Early design reviews assist in identifying features exposed to security risks before they are implemented.
- Implementation
  - This is the stage at which an application is actually created. This includes writing the application code, debugging it, and producing stable builds suitable for testing.
  - Guides and checklists remind programmers of typical mistakes to be avoided, such as storing unencrypted passwords. Enforcing secure coding principles eliminates many trivial vulnerabilities and frees up time for other important tasks.
  - Manual code reviews are still a must for building secure applications. Timely reviews help developers to flag and fix potential issues before they shift attention to other tasks.
- Testing and bug fixing

The purpose of this stage is to discover and correct application errors. This includes running automatic and manual tests, identifying issues, and fixing them.

We use the rolling wave testing method to perform continuous testing after each phase, which ensures that security issues will be detected and resolved as soon as possible.

- Release and maintenance

At this stage EDSA goes live, with many instances running in a variety of environments.

Before going live, we set up security for the cloud server, install SSL for hosting and related procedures to manage the cloud environment as closely as possible.

## 8. REFERENCE

[EDSDA]Proposal:

[https://docs.google.com/document/d/1i5YAMZ1wxw7DIX-KArDY1\\_1yXWSKtZ7MliIYvf\\_uAoo/](https://docs.google.com/document/d/1i5YAMZ1wxw7DIX-KArDY1_1yXWSKtZ7MliIYvf_uAoo/)

What is Scrum: <https://www.scrum.org/resources/what-is-scrum>

Design: <https://www.figma.com/>

Security Aspect:

<https://www.ptsecurity.com/ww-en/analytics/knowledge-base/how-to-approach-secure-software-development/>

Integrating Quality Management System into Software Development Processes:

<https://assist-software.net/blog/integrating-quality-management-system-software-development-processes>



## CAPSTONE PROJECT 2

CMU-SE-451 / CMU-IS-451 / CMU-CS-451

### PRODUCT BACKLOG AND USER STORY

Version 2.0

Date: 1 - Mar - 2021

### EXPERT-DRIVEN SMART DASHBOARD APPLICATION

#### Submitted by

Vo Van Hoa  
Pham Van Tin  
Ky Huu Dong  
Tran Thi Thanh Kieu

#### Approved by

##### Capstone Project 2 - Mentor:

Name

Signature

Date

Binh, Thanh Nguyen \_\_\_\_\_  \_\_\_\_\_ 26 - May - 2020

PROJECT INFORMATION			
<b>Project Acronym</b>	EDSDA		
<b>Project Title</b>	Expert-Driven Smart Dashboard Application		
<b>Project Web URL</b>	<a href="https://sda-research.ml/">https://sda-research.ml/</a>		
<b>Start Date</b>	01 - Mar - 2021		
<b>End Date:</b>	02 - Jun - 2021		
<b>Lead Institution</b>	International School, Duy Tan University		
<b>Project Mentor</b>	Ph.D Binh, Thanh Nguyen		
<b>Scrum Master</b>	Hoa, Vo	hoavo.dng@gmail.com	0935.193.182
<b>Team Members</b>	Tin, Pham Van	tinphamvan123@gmail.com	0932.535.175
	Dong, Ky Huu	kyhuudong@gmail.com	0898.246.980
	Kieu, Tran Thi Thanh	thanhkieutran391@gmail.com	0358.583.251

DOCUMENT INFORMATION			
<b>Document Title</b>	Product Backlog & User Story		
<b>Author(s)</b>	Team C2SE.06		
<b>Role</b>	[EDSDA] Product Backlog & User Story v2.1		
<b>Date</b>	01 - Mar - 2021	Filename	[EDSDA] 003 Product Backlog & User Story
<b>URL</b>	<a href="https://github.com/sdateamdtu2020/SDA-v2.0">https://github.com/sdateamdtu2020/SDA-v2.0</a>		
<b>Access</b>	Project and CMU Program		

## REVISION HISTORY

<b>Product</b>	Expert-Driven Smart Dashboard Application
<b>Active versions</b>	v2.0

<b>Scrum Master</b>	Hoa Vo Van
<b>Developer</b>	Tin Pham Van
<b>Developer</b>	Dong Ky Huu
<b>Developer</b>	Kieu Tran Thi Thanh

<b>Product Status</b>	In Development
<b>Sponsor</b>	Nguyen Thanh Binh

<b>Product Owner</b>	Nguyen Thanh Binh
<b>Project Champion</b>	Hoa Vo

## Projects

Id	Description	Start	End	Status	Remarks
Sprint 0	Initiate project requirement & design	15/Jan/21	28/Feb/21	Completed	
Sprint 1	Initiate infrastructure & data	01/Mar/21	14/Mar/21	Completed	
Sprint 2	Build Data Warehouse & RDF Storage	15/Mar/21	28/Mar/21	Completed	
Sprint 3	Implement data processing methods	29/Mar/21	11/Apr/21	Completed	
Sprint 4	Implement automatic methods	12/Apr/21	25/Apr/21	Completed	
Sprint 5	Handle importing data sources	26/Apr/21	09/May/21	Completed	
Sprint 6	Cloud server setting-up	10/May/21	23/May/21	Completed	
Final	Product delivery	31/May/21	08/Jun/21	Completed	

---

### Sprint Plan

Sprint	Integration	Start	End	Status	Story Points (Committed)	Story Points (Delivered)
Sprint 0	1	24/Jan/21	31/Jan/21	Completed	3	3
	2	31/Jan/21	02/Feb/21	Completed	25	23
	3	06/Feb/21	21/Feb/21	Completed	20	18
	4	22/Feb/21	28/Feb/21	Completed	25	27
Sprint 1	5	01/Mar/21	07/Mar/21	Completed	20	23
	6	08/Mar/21	14/Mar/21	Completed	21	23
Sprint 2	7	15/Mar/21	21/Mar/21	Completed	20	22
	8	22/Mar/21	28/Mar/21	Completed	25	28
Sprint 3	9	29/Mar/21	04/Apr/21	Completed	27	27
	10	05/Apr/21	11/Apr/21	Completed	23	26
Sprint 4	11	12/Apr/21	18/Apr/21	Completed	22	23
	12	19/Apr/21	25/Apr/21	Completed	25	27
Sprint 5	13	26/Apr/21	02/May/21	Completed	22	25
	14	03/May/21	09/May/21	Completed	21	22
Sprint 6	15	10/May/21	16/May/21	Completed	22	25
	16	17/May/21	23/May/21	Completed	25	27
Final	Final Submission	31/May/21	02/Jun/21	Completed		
Final	Final Presentation	05/Jun/21	08/Jun/21	Completed		

## Backlog

<b>Id</b>	<b>Epics</b>	<b>As a..</b>	<b>I want to ..</b>	<b>so that ..</b>
PB01	Drag & Drop Data Cubes	As an Environment expert	drag a data cube from the cubes list and drop it onto the main content board	I can choose the property of data that I need and visualize them on suitable kinds of charts and maps.
PB02	Drag & Drop Data Cubes & Visualize	As an environment expert	drag some data cubes from the cubes list and drop them onto the main content board	I can choose the property of data and then connect these data cubes together for using data operation and visualize it on suitable kinds of charts and maps.
PB03	View data cube	As an environment expert	I want to view the data source of any data cubes	I can verify the accuracy of the data.
PB04	Import a datacube	As a user	I want to import my own data sources to the EDSA database	I can interact with my data directly with EDSA help.
PB05	Data import view	As a user	I want to review my data sources before submitting them	I can check the accuracy of my data.
PB06	Define dimension, and measure properties	As a user	I want to define the dimension and measure of my data source's data cubes by myself	I can easily do OLAPing with my data source.
PB07	Export files	As a user	I want to export some report I have generated on the dashboard	I can use that exported file for my document and my analysis work

## Acceptance Criteria

Id	Acceptance Criteria	Priority	Status	Type	Sprint Id	Story Points (Est.)	Effort (Actual)
PB01	<ul style="list-style-type: none"> <li>- Display name of dimension, data operation, and type of visualization in sidebar</li> <li>- Dimensions and Data cubes are available in sidebar ready to drag</li> <li>- When onDrop, display exactly widget in mainboard</li> </ul>	5	Completed	Functional Criteria	3	50	13
PB02	<ul style="list-style-type: none"> <li>- Display name of dimension,datacube, data operation, and type of visualization in sidebar</li> <li>- Dimensions and Data cubes are available in sidebar ready to drag</li> <li>- When onDrop, display exactly widget in mainboard</li> <li>-Datacube, dimension, and the data visualization type must be connected together</li> </ul>	5	Completed	Functional	2	45	13
PB03	<ul style="list-style-type: none"> <li>-Data cube in the sidebar must available .</li> <li>-Data cube must connect to dimension to view the data from that dimension</li> </ul>	3	Completed	Functional	2	45	8
PB04	<ul style="list-style-type: none"> <li>- Upload file (.CSV, .XLXS is only accepted)</li> <li>-The uploaded files must follow the system's business constraint (format, content)</li> </ul>	4	Completed	Functional	5	43	18

## Acceptance Criteria

Id	Acceptance Criteria	Priority	Status	Type	Sprint Id	Story Points (Est.)	Effort (Actual)
PB05	<ul style="list-style-type: none"> <li>-Displaying exactly like excel format, uploaded file</li> <li>-The column of upload file can select by user</li> </ul>	5	Completed	Non-Functional	4	47	14
PB06	<ul style="list-style-type: none"> <li>-Dropdown with checkbox dimension values must contain dimension values from upload file</li> <li>-Dropdown with checkbox measure values must contain dimension values from upload file</li> <li>-User have to choose the value from checkbox to define the data cube (dimensions, and measures).</li> </ul>	4	Completed	Functional	5	43	12
PB07	-Must have at least one successfully generated visualization dashboard	1	Completed	Functional	6	47	13

ID	Name	Description	Actor	Priority	Trigger	Pre - Condition	Post - Condition	Basic Flow	Alternative Flow	Exception Flow	Business Rules	Non-Functional Requirement
1	View Data cubes	As a user, I want to see the list of the data cubes so that I can know exactly what data I have, how many data cubes available, and how to work with them.	Users	Must-Have	User wants to know the list of Data cubes	No	No	1. Access to the ESDA Web App through the internet. 2. At the right side of the web page shows widgets and the data cubes are available.	No	No	No	No
2	Drag Data cubes	As a user, I want to select one data cube so that I can drag that data cube into the active page and interact with other data cubes.	Users	Must Have	User wants to drag a specific data cube into the active page.	The dragged data cube must be available on the widgets. The dimensions of the dragged data cube must be valid.	The dragged data cube successfully displays on the active page.	1. Access to the ESDA Web App through the internet. 2. At the right side of the web page shows widgets and the data cubes are available. 3. Select one data cube and drag it into the middle of the active page.	No	No	No	No
3	Visualize Data cube	As a user, I want to visualize data cube in many different types so that I can see the data displaying different ways like charts, maps, and data tables then I can clearly understand the data and use it.	Users	Must Have	Users want to visualize the data cube.	The visualizing data cube must available on the widgets. The dimensions of selected data cube must be valid.	The visualizing data cube displaying on the active page.	1. Access to the ESDA Web App through the internet. 2. At the right side of the web page shows widgets and the data cubes, dimensions, and visualization types are available. 3. Select one at a time a data cube and drag it into the middle of the active page. 4. Select a dimension and drag it into active page you want to visualize. 5. Choose the visualize types and drag the visualize type into the active page. 6. Select the values of the data cube and dimension values. 7. Connect all available widgets together on the active page and then click the run button on visualizing widgets.	No	No	No	No
4	Select Data Cube	As a user, I want to select a new data cube to the active page so that I can display multiple data and connect them with the other data cubes.	Users	Must Have	User wants to select more data cubes.	The selected data cube must available on the widgets. The dimensions of selected data cube must be valid.	The selected data cube displaying on the active page.	1. Access to the ESDA Web App through the internet. 2. The right side of the web page shows widgets and the data cubes are available. 3. Select the data cube and drag it into the middle of the active page.	No	No	No	No
5	List two data cubes	As a user, I want to link two data cubes together so that I can display multiple data and connect them with the other data cubes.	Users	Must Have	Users want to link two data cubes together	The visualizing data cube must available on the widgets. Two data cubes or more are on the active page and available to link. Linking two or more data cubes must have the same dimension attributes. The dimensions of selected data cube must be valid.	The data cubes successfully linked together through the dimensions. The visualizing data cube displaying on the active page.	1. Access to the ESDA Web App through the internet. 2. The right side of the web page shows widgets and the data cubes, dimensions, and visualization types are available. 3. Select one at a time a data cube and drag it into the middle of the active page. 4. Select a dimension and drag it into the active page you want to visualize. 5. Choose the visualize types and drag the visualize type into the active page. 6. Select the values of the data cube and dimension values. 7. The second data cube, or the third data cube..., linking these data cubes together by connecting the widgets through dimensions widgets. 8. Connect all available widgets together on the active page and then click the run button on visualizing widgets.	No	No	No	No

ID	Name	Description	Actor	Priority	Trigger	Pre - Condition	Post - Condition	Basic Flow	Alternative Flow	Exception Flow	Business Rules	Non-Functional Requirement
6	List of Data Cubes	As a user, I want to create a new linked data cube so that I can manage the new data cube I have just created beside the previously created data cube to link them together.	Users	Must Have	Users want to create new linked data cubes.	The visualizing data cube must available on the widgets. Two data cubes or more are on the active page and available to link. Linking two or more data cubes must have the same dimension attributes. The dimensions of selected data cube must be valid.	The linked data cubes successfully linked together through the dimensions. The visualizing data cube displaying on the active page.	1. Access to the EDSDA Web App through the internet. 2. The right side of the web page shows widgets and the data cubes, dimensions, and visualization types are available. 3. Select one at a time a data cube and drag it into the middle of the active page. 4. Select a dimension and drag it into the active page you want to visualize. 5. Choose the visualize types and drag the visualize type into the active page. 6. Select the values of the data cube and dimension values. 7. The second data cube, or the third data cube..., linking these data cubes together by connecting the widgets through dimensions widgets. 8. Connect all available widgets together on the active page and then click the run button on visualizing widgets.	No	No	No	No
7	Visualize New Data cube	As a user, I want to visualize a new data cube in many different types so that I can view more data information about a new data cube.	Users	Must Have	Users wants to visualize a data cube.	-The visualizing data cube must available on the widgets. -The dimensions of selected data cube must be valid.	-The visualizing data cube displaying on the active page. -The new data display on the active page.	1. Access to the EDSDA Web App through the internet. 2. The right side of the web page shows widgets and the data cubes, dimensions, and visualization types are available. 3. Select one at a time a data cube and drag it into the middle of the active page. 4. Select a dimension and drag it into active page you want to visualize. 5. Choose the visualize types and drag the visualize type into the active page. 6. Select the values of the data cube and dimension values. 7. Connect all available widgets together on the active page and then click the run button on visualizing widgets.	No	No	No	No
8	Import Data Source	As a user, I want to import my own data source to EDSDA, so that I can interact with my data on EDSD	Users	Must Have	User want to import a data source	Data source must be in CSV or XLXS type	No	1. User accesses to EDSD homepage. 2. User clicks on Import button on top of the webpage. 3. User chooses data source file to be uploaded in the device. 4. User reviews the data source and chooses dimensions and measures for creating data cube. 5. User submit the data cube to system.	No	3b. Importing fail due to invalid data source. 4b. Returns to homepage	No	NFR1. When user chooses and imports a valid data source, the review screen must appear immediately. NFR2. When user submits the data cube structure, the page will reload within 2s.
9	Export report	As a user, I want to export my data charts as PDF, PNG, or JPG type	Users	Should have	User wants to capture their data chart	Data Cubes must be visualized or linked on the mainboard	No	1. Access to the EDSDA Web App through the internet. 2. At the right side of the web page shows widgets and the data cubes, dimensions, and visualization types are available. 3. Select one at a time a data cube and drag it into the middle of the active page. 4. Select a dimension and drag it into active page you want to visualize. 5. Choose the visualize types and drag the visualize type into the active page. 6. Select the values of the data cube and dimension values. 7. Connect all available widgets together on the active page and then click the run button on visualizing widgets. 8. Choose export Button on top of the board. 9. On the drop-down list, choose portable type of report ( PNG, PDF, JPG ). Then the report will automatically render and downloaded to user's device.	No	No	No	No

Activity / Process	Total budgeted Effort Usage (pd)	Total % budgeted Effort Usage (%)	Sprint 1		Sprint 2		Sprint 3		Sprint 4		Sprint 5		Sprint 6	
			No	%										
Requirement	0	100	1	0	2	0	3	0	4	0	5	0	6	0
Design	600	80	1	30	2	30	3	40	4	0	5	0	6	0
Coding	1820	100	1	20	2	20	3	20	4	20	5	10	6	10
Unit Testing	150	80	1	20	2	20	3	20	4	20	5	10	6	10
Testing	600	100	1	20	2	20	3	20	4	20	5	10	6	10
Deployment	100	100	1	0	2	0	3	0	4	0	5	20	6	80
Support for Acceptance Test	0	100	1	0	2	0	3	0	4	0	5	0	6	0
Project Planning	50	100	1	80	2	10	3	10	4	0	5	0	6	0
Project monitoring	400	80	1	20	2	20	3	20	4	20	5	10	6	10
Quality Assurance	200	100	1	20	2	20	3	20	4	20	5	10	6	10
Training	80	100	1	40	2	30	3	30	4	0	5	0	6	0



## CAPSTONE PROJECT 2

CMU-SE-451 / CMU-IS-451 / CMU-CS-451

### ARCHITECTURE DOCUMENT

Version 2.0

Date: 09 - May - 2021

### EXPERT-DRIVEN SMART DASHBOARD APPLICATION

#### Submitted by

Vo Van Hoa  
Pham Van Tin  
Ky Huu Dong  
Tran Thi Thanh Kieu

#### Approved by

##### Capstone Project 2 - Mentor:

Name

Signature

Date

Binh, Thanh Nguyen \_\_\_\_\_

A handwritten signature in blue ink, appearing to read 'Binh, Thanh Nguyen'.

\_\_\_\_\_ 26 - May - 2020

PROJECT INFORMATION			
<b>Project Acronym</b>	EDSDA		
<b>Project Title</b>	Expert-Driven Smart Dashboard Application		
<b>Project Web URL</b>	<a href="https://sda-research.ml/">https://sda-research.ml/</a>		
<b>Start Date</b>	01 - Mar - 2021		
<b>End Date:</b>	02 - Jun - 2021		
<b>Lead Institution</b>	International School, Duy Tan University		
<b>Project Mentor</b>	Ph.D Binh, Thanh Nguyen		
<b>Scrum Master</b>	Hoa, Vo	hoavo.dng@gmail.com	0935.193.182
<b>Team Members</b>	Tin, Pham Van	tinphamvan123@gmail.com	0932.535.175
	Dong, Ky Huu	kyhuudong@gmail.com	0898.246.980
	Kieu, Tran Thi Thanh	thanhkieutran391@gmail.com	0358.583.251

DOCUMENT INFORMATION			
<b>Document Title</b>	Architecture Document		
<b>Author(s)</b>	Team C2SE.06		
<b>Role</b>	[EDSDA] Architecture_Document_v2.1		
<b>Date</b>	01 - Mar - 2021	Filename	[EDSDA] 004 Architecture Document
<b>URL</b>	<a href="https://github.com/sdateamdtu2020/SDA-v2.0">https://github.com/sdateamdtu2020/SDA-v2.0</a>		
<b>Access</b>	Project and CMU Program		

## REVISION HISTORY

# TABLE OF CONTENTS

PROJECT INFORMATION	1
REVISION HISTORY	2
<b>TABLE OF CONTENTS</b>	<b>3</b>
<b>1. INTRODUCTION</b>	<b>5</b>
1.1. PURPOSE	5
1.2. DEFINITIONS, ACRONYMS AND ABBREVIATIONS	5
1.3. DOCUMENTS REFERENCES	5
<b>2. PROBLEM STATEMENT</b>	<b>6</b>
2.1. PROJECT OVERVIEW	6
2.2. BUSINESS DRIVERS	6
2.3. PROJECT GOAL	7
<b>3. ARCHITECTURE DRIVERS</b>	<b>7</b>
3.1. HIGH-LEVEL REQUIREMENTS	7
3.2. SYSTEM CONTEXT	8
3.3. QUALITY ATTRIBUTES	10
<b>4. CONSTRAINTS</b>	<b>12</b>
4.1. BUSINESS CONSTRAINTS	12
4.2. TECHNICAL CONSTRAINTS	12
<b>5. HIGH-LEVEL ARCHITECTURE</b>	<b>13</b>
5.1. COMPONENT AND CONNECTOR VIEW (C&C VIEW)	13
5.2. MODULE VIEW	17
5.2.1. Module View based on Web Application	17
5.2.2. Module View based on Server	21
5.3. ALLOCATION VIEW	23
<b>6. LOW-LEVEL ARCHITECTURE</b>	<b>24</b>
6.1. DATABASE DESIGN	24
6.1.1. Data warehouse architecture	24
6.1.2. Star schema, Fact Tables and Dimension Tables	25
6.2. DATA LAKE	25
6.3. ETL (EXTRACT, TRANSFORM, LOAD)	26
6.3.1. Base workflow	26
6.3.2. Import workflow	27
6.3.3. Data pipeline workflow	28
6.4. RDF DATA CUBES	29

6.4.1. RDF data cubes design	29
6.4.2. RDF data cubes architecture	29
6.5. RDF DATA CUBES AUTO GENERATOR	32
6.5.1. Traditional R2RML Methods	32
6.5.2. Automatic RDF Data Cubes Generator	33
6.6. APPLICATION APIs	34
<b>7. REFERENCES</b>	<b>34</b>

# 1. INTRODUCTION

## 1.1. PURPOSE

The purpose of the Architecture document is to:

- Define the architecture needs and technology in detail.
- Provide solutions for business needs.
- Provide overview about resources, schedule, solution and budget for the project.

The architecture merely introduces the project to the student development teams, and provides the up-front information necessary for the team to develop a specification.

## 1.2. DEFINITIONS, ACRONYMS AND ABBREVIATIONS

Acronyms	Definitions
EDSDA	Expert-driven Smart Dashboard Application
GUI	Graphical User Interface
SDK	Software Development Kit
ES	ECMAScript language

## 1.3. DOCUMENTS REFERENCES

No.	Reference
1	Product Backlog Document for EDSDA
2	Project Plan Document for EDSDA

## 2. PROBLEM STATEMENT

### 2.1. PROJECT OVERVIEW

Our environment is always changing. However, at the current rate of urbanization and industrialization, outside of the natural factors, the change of environment is mainly due to human factors. Emissions, population explosion, industrial solid waste, ... are the main causes leading to negative effects on the global environment. To address this at a holistic level, data analysis and aggregation are the first important tasks to be done.

However, analyzing and aggregating data from many different sources takes a lot of effort and money. To solve this problem, based on our knowledge of big data systems, we have built an intelligent data processing system that can be run on a website-platform with an intuitive and easy-to-use dashboard. This system is a prospective and useful tool for environmental experts and policymakers in Vietnam in particular, and worldwide in general. It will collect, analyze and synthesize data about all the factors that can affect the environment, thereby helping users to come up with quick and accurate solutions to solve problems related to the environment.

In addition, data sources that open the environment in a particular aspect such as area emissions, soil erosion, etc. are often scarce and stored only in the user's local computer. Analytical tools of these types of data often require expertise using quite advanced technology, thus limiting interoperability with data stored in individuals. With this in mind, we built a subsystem where users could bring their data to EDSA and analyze, aggregate, and visualize their data directly on the dashboard along with a wide range of system visualization tools.

### 2.2. BUSINESS DRIVERS

#### **Business problem:**

Our environment is always changing. However, at the current rate of urbanization and industrialization, outside of the natural factors, the change of environment is mainly due to human factors. Emissions, population explosion, industrial solid waste, ... are the main causes leading to negative effects on the global environment. To address this at a holistic level, data analysis and aggregation are the first important tasks to be done.

#### **Business need:**

Expert-driven Smart Dashboard Application have specific uses :

- Help users to come up with quick and accurate solutions to solve problems that are related to the environment.
- Help users represent data in many types of charts, diagrams and maps, thereby providing an overview to solve the problem.
- Help users can get the information and insights they need at a glance.
- Giving business predictions based on the chart, diagrams, and maps.

All the things above are based on the functionality of the Expert-driven Smart Dashboard Application system. EDSDA fully meets these requirements. Therefore, the development of EDSDA is very necessary and meaningful.

## 2.3. PROJECT GOAL

Expert-Driven Smart Dashboard - means a dashboard that is convenient for users to analyze and review data. It will include several datasets about real-time information of the environment, measurement data of air pollutants... EDSDA will connect and analyze data from multiple sources in ways you've never imagined. Then reveal the insights you've been missing...in just a matter of minutes.

With smart suggestions and an intuitive visual interface, SDA makes it easy for any user to combine data and discover hidden insights in one place...without the usual scripting, coding, and IT hand-holding. With this dashboard, individuals or any subjects can take advantage of environmental data to be able to decide the best relevant policies.

Not only for the environmental expert, but EDSDA also helps the user directly interact with their own data source by importing, automatically cleaning their data, and enabling the user to do everything with their own data in the easiest way.

## 3. ARCHITECTURE DRIVERS

### 3.1. HIGH-LEVEL REQUIREMENTS

(Refer to the Product Backlog document for EDSDA)

## 3.2. SYSTEM CONTEXT

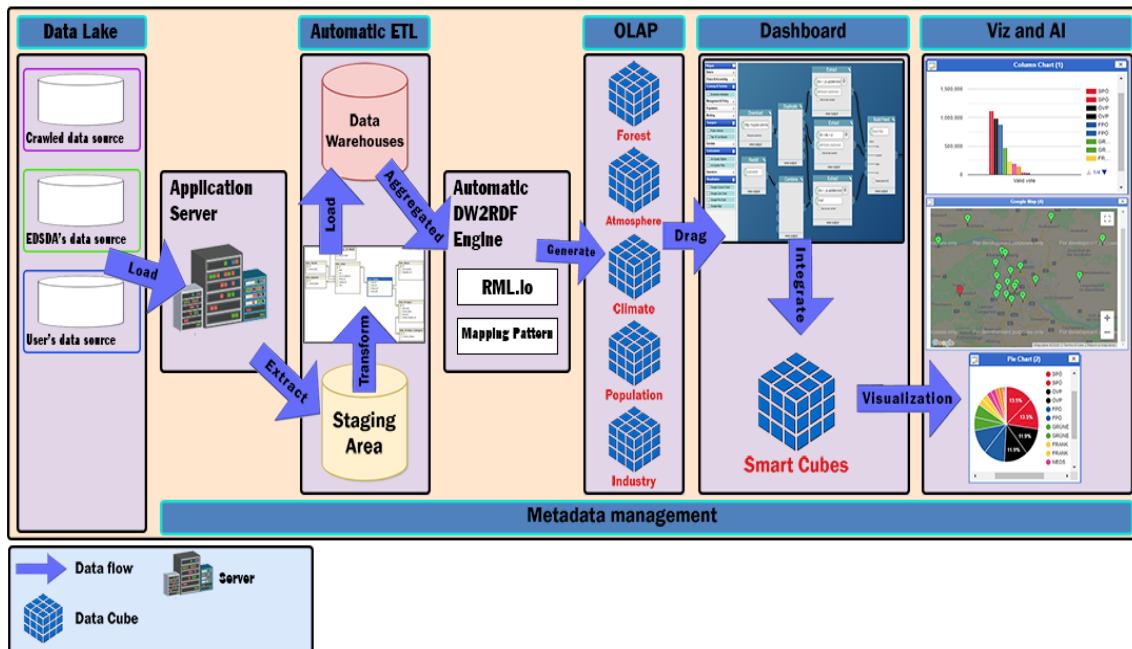


Figure 1: ESDA Context Diagram

### 1. Data Lake:

- Collect from environment open data platforms of the governments and NGO organizations.
- User imported data sources.
- Use web crawling techniques to crawl data from related environment websites.
- Data Format : CSV, JSON, XML.

### 2. Application Server:

- Check for the changes of the data lake when the user imports new data sources.
- Handle Data Lake Exceptions if any errors occur.
- Trigger the Data Lake to Staging Area loading process.

### 3. Automatic ETL (Extract, Transform, Load)

#### 3.1. Extract:

In this step, we extract structured data after restructuring the raw data from the data lake. Mainly in this step, we focus on exacting data correctly from the structured data from the data lake.

#### 3.2. Transform:

Based on the business's constraints we have to transform the data to be compatible with our process to handle the data for displaying on the dashboard. Getting exacted data, transforming the data, and ready to load into the DW.

### 3.3. Load:

Loading data into the target data warehouse database is the last step of the ETL process. We load all the data we have transformed into the staging areas of DW and based on the data we have loaded into staging areas we used to load the data from the staging areas to dimension tables, and fact tables.

### 3.4. Perform automatic ETL:

Performing automatic ETL using tools for automating exacting data, transforming data, and loading data to DW. We use the available platform for scheduling to perform ETL at a specific time in a day. In addition, for performing ETL automatically we also build a data pipeline to make the data flow as we wanted.

## 4. Automatic DW2RDF Engine:

- Using RML.io with our custom features, it can generate RDF Data cubes automatically when users import new data sources to the data warehouse.
- Suitable with all data sources with a general mapping method.

## 5. OLAP

In this step, we store the data of the data warehouse as OLAP cubes. And then, for better query performance, data binding, and scalability, in addition to information transparency, we will automate converting OLAP cubes into RDF Data Cubes.

## 6. Dashboard

In this step, the user can drag any data cubes that appear as items on the sidebar and drop them onto the main content board, then connect between them and use the operator such as statistics merge, geo merge to build a new data cube that matches the user requirement.

## 7. Viz & AI

This step will perform the data cube which was created by the user with the form they want. It can be a map, a column chart, a line chart, or a pie chart.

### 3.3. QUALITY ATTRIBUTES

ID	QA01
<b>Quality Attributes</b>	Performance
<b>Stimulus</b>	Uses the statistics merge operator for merging multiple data cubes.
<b>Source(s) of stimulus</b>	User
<b>Artifacts</b>	System
<b>Environment</b>	Normal mode
<b>System response</b>	The SDA returns a new data cubes as fast as possible
<b>Response measure(s)</b>	Within 3 seconds

**Table 3.3.1:** Quality Attributes: Performance

ID	QA02
<b>Quality Attributes</b>	Availability
<b>Stimulus</b>	The power is off while server is running
<b>Source(s) of stimulus</b>	Power
<b>Artifact</b>	During peak usage load
<b>Environment</b>	Hardware and software
<b>System response</b>	System will use the cloud server to save the work so we don't need to worries about power incident occurred
<b>Response measure(s)</b>	All work always can be saved

**Table 3.3.2:** Quality Attributes: Availability

ID	QA03
<b>Quality Attributes</b>	Availability
<b>Stimulus</b>	Can't get a specific data cubes when select it from widget
<b>Source(s) of stimulus</b>	User
<b>Artifact</b>	System
<b>Environment</b>	Normal mode
<b>System response</b>	System will log the fault immediately
<b>Response measure(s)</b>	Within immediately

**Table 3.3.2:** Quality Attributes: Availability

ID	QA03
<b>Quality Attributes</b>	Portability and compatibility
<b>Stimulus</b>	Open the browser and access to <a href="http://sda-research.ml/">http://sda-research.ml/</a>
<b>Source(s) of stimulus</b>	User
<b>Artifact</b>	System
<b>Environment</b>	Normal mode
<b>System response</b>	The system can run on any web browser.
<b>Response measure(s)</b>	Within immediately

**Table 3.3.2:** Quality Attributes: Portability and compatibility

ID	QA03
<b>Quality Attributes</b>	Security
<b>Stimulus</b>	User access to EDSA and import a file
<b>Source(s) of stimulus</b>	User
<b>Artifact</b>	System
<b>Environment</b>	Normal mode
<b>System response</b>	System will not track any individual information about the user
<b>Response measure(s)</b>	Within immediately

**Table 3.3.2:** Quality Attributes: Availability

## 4. CONSTRAINTS

### 4.1. BUSINESS CONSTRAINTS

- Project will be started on: 01 - Mar - 2021
- Project will be finished on: 02 - June - 2021
- Duration: 13 weeks

### 4.2. TECHNICAL CONSTRAINTS

Main Programming Language: Javascript, Python.

#### Data Crawling:

- Programming Language: Python, Javascript.
- Database: PostgreSQL.
- Library: BeautifulSoup 4, Selenium.

#### Data Warehouses:

- Programming Language: Python.
- Database: PostgreSQL.
- Library: Psycopg2, CSV.

#### Perform ETL automating:

- Tool for ETL process: Talend.
- Schedule for ETL process platform: Apache Airflow.

**Data Cubes**

- Programming Language: Java, SPARQL.
- Tool for converting from Data Warehouse to RDF Data Cubes Storage: RML.io (with customized phase).
- Network Accessing: RDF-REST API.

**Server:**

- Programming Language: Javascript.
- Framework / Libraries: ExpressJS (NodeJS), enapso-graphdb-client.
- Operating System: Windows, Linux, macOS.
- Deployment Environment: Google Cloud with App Engine and SQL Services.
- Network Accessing: HTTP methods (POST, GET) via RESTful API.

**Client:**

- Programming language: HTML, CSS, Javascripts.
- Framework/Libraries: React, Redux, Material-UI, React Flow, React Charts,...
- Deployment Environment: Google Firebase Hosting
- Web Browser: Chrome, Firefox, Microsoft Edge, Coccoc
- Network Accessing: World Wide Web (WWW), HTTP methods (POST, GET) via RESTful API with Axios.

## 5. HIGH-LEVEL ARCHITECTURE

### 5.1. COMPONENT AND CONNECTOR VIEW (C&C VIEW)

The diagram below shows the overview architecture including components and other related components. We have representations and behaviors for import components in the following sections

*References C&C View on attached page.*

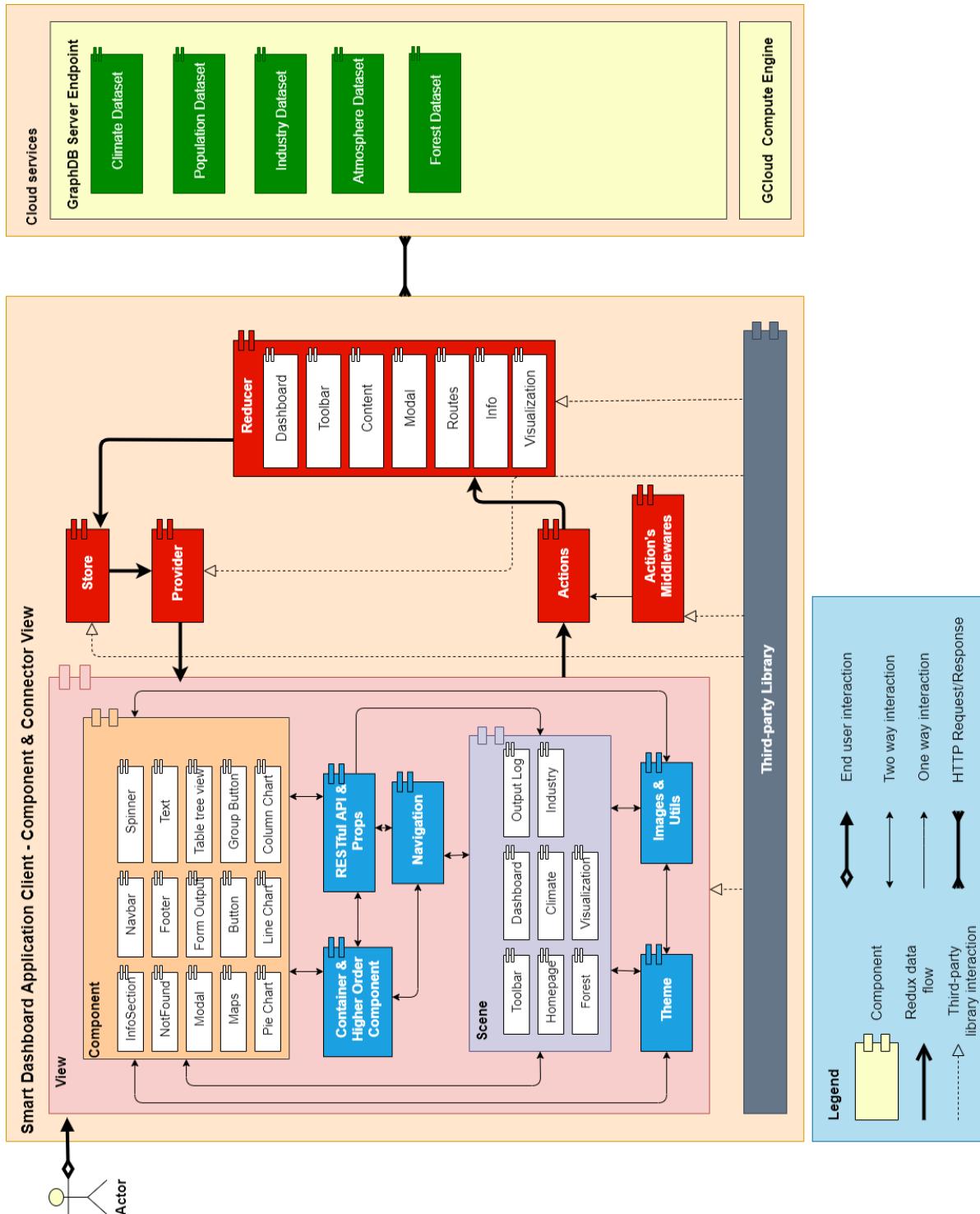


Figure 2: ESDA C&amp;C View

**Prose:**

When the end-user opens the application, GUI will appear. Users will interact with the View component. The View component includes some small components like Toolbar, Content, Description and Util, Scene, Navigation, Container and Higher Order component, and some other components (The View component will be detailed in Model View). If the User has any action, View will call the Actions component. Actions will format the requirement and send it to the Reducer. The Reducer will send a state tree to the Store. The Store will replace the old state tree with a new one. The Provider is a component that helps connect Store and View, it will get the new state from Store and send it back to View. Third party libraries will help the application run faster and reduce the code time.

Any time when the application needs to use data, it will call Cloud service. Cloud service includes Graph server endpoint. We place our server on GCloud and implement the application API on it.

Role & Responsibility	Description
<b>View</b>	
Component	Components let you split the app into independent, reusable pieces, and think about each piece in isolation.
Scene	The Scene transform represents the app in UI.
Container & Higher Order Component	Pattern that has proven to be very valuable for several React libraries
Restful API & Props	Provides a data in your API & render in child
Navigation	Provides an easy to use navigation solution,
Theme	Defining a set of styles.
Images & Utils	Storing images or utils library.
Store	The Store is the object that brings all together.
Provider	Make the store available to all container components in the application without passing it explicitly.
Actions	Actions are payloads of information that send data from your application to your store.
Actions's Middlewares	It provides a third-party extension point between dispatching an action, and the moment it reaches the reducer.

Role & Responsibility	Description
Reducer	Actions describe the fact that something happened, but don't specify how the application's state changes in response. This is the job of reducers.
Third-party Library	
GraphDB Server Endpoint	Build and ship our APIs faster and more consistently. Not having to worry about authentication, performance and status monitoring has reduced the time and effort we need to build great APIs
API:	Set of functions and procedures that allow the creation of applications which access the features or data of an operating system, application, or other service.

## 5.2. MODULE VIEW

### 5.2.1. Module View based on Web Application

References Module View on attached page.

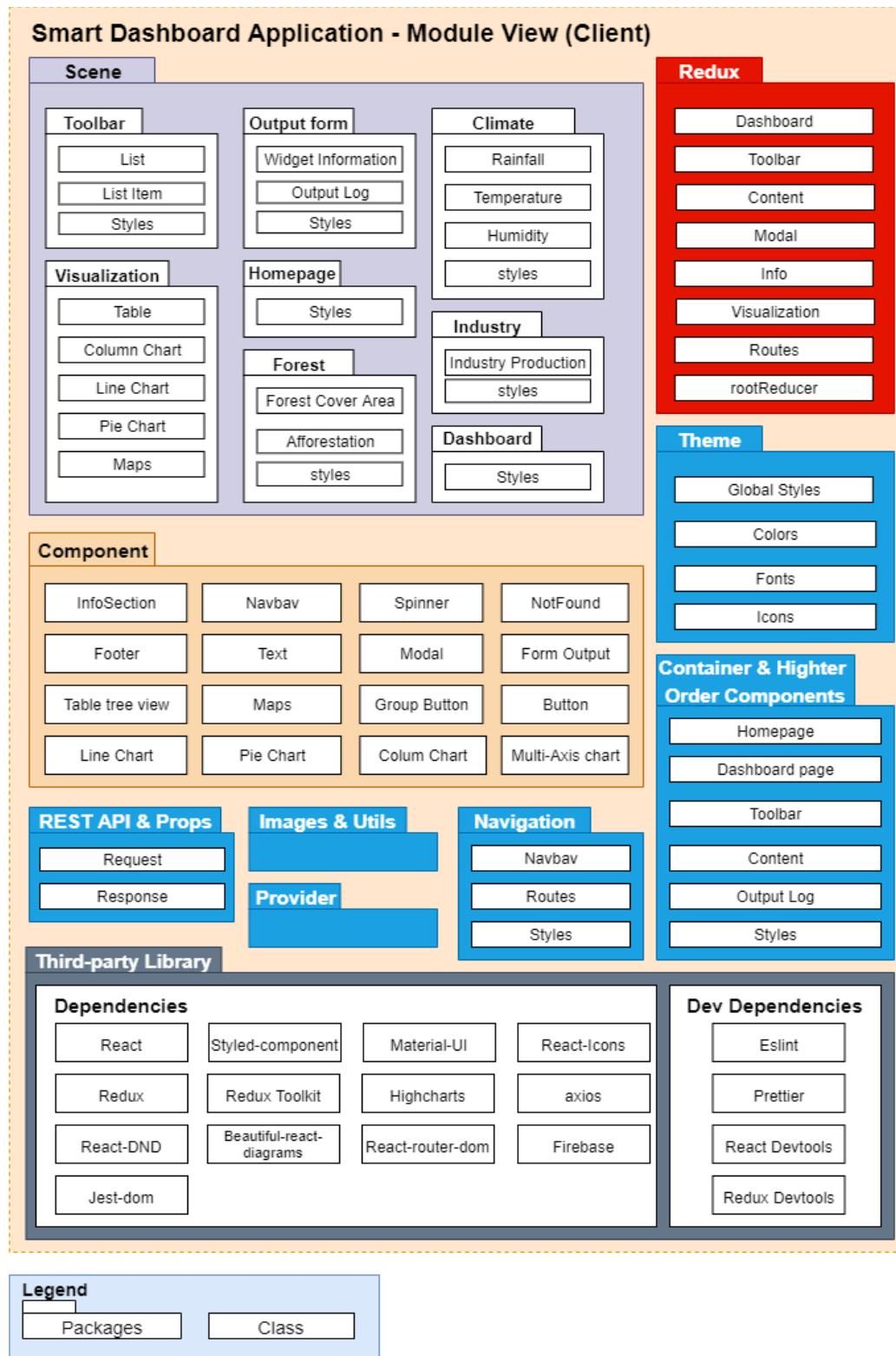


Figure 3: Module View Client

**Prose:**

The ESDA client application includes 10 packages that help the app run effectively.

The Scene package includes

Component package which has 16 classes which is often used and we custom it to fit our requirement. We also have Theme package to define the app format. It will define the app color, font, icon, and metric.

Navigation package contains a Navbar, Routes and its Style.

Redux package also has Dashboard, Toolbar, Content, Modal and root Reducer class to manage the state. Our Component and Higher Order Components is an advanced technique in React that is used in reusing components. Higher Order Components are not part of the React API. Specifically, a higher-order component is a function and it takes the argument as a component and returns a new component.

Images and Utils package contain the app image and functions to solve app general problems.

In third-party libraries, we have Dependencies and Devdependencies class.

Beside it, We have Dev Dependencies class like EsLint, Prettier, React Dev Tool and Redux Dev Tool to manage the code and make sure that our code follows the general format.

Finally, when the app wants to use data, it will connect to cloud service. Cloud service includes GraphDB and GCloud service

Role & Responsibility	Description
Styles	Defining styles
<b>Scene</b>	
Toolbar	Display list of datacube & dataset
Output form	Widget Log
User Guide	Help user can use app easily
Homepage	Home page of web
Dashboard	Higher Order Component that wrap toolbar, mainboard, Information Widget
Climate	DataCube
Industry	DataCube
Forest	DataCube
<b>Component</b>	

Role & Responsibility	Description
InfoSection	Section Information
Navbar	Navigation
Spinner	An UI when page is loading
NotFound	An UI when Page when 404 error
Footer	Footer
Form Input	A text field is used for form
Text	A general text is used for all text in this application
Modal	Modal Component is inherited Modal of Material-UI
Button	Button
Group Button	List of button
Custom Node	Widget of each datacube & visualization
Line Chart	Visualization data from data cube in chart
Column Chart	Visualization data from data cube in chart
Pie Chart	Visualization data from data cube in chart
Maps	Visualization data from data cube in maps
<b>Redux</b>	
Dashboard	A Redux store & reducer for handling state of Dashboard
Toolbar	A Redux store & reducer for handling state of Toolbar
Content	A Redux store & reducer for handling state of Content
Modal	A Redux store & reducer for handling state of Modal
Routes	Define routes of scenes for navigation
rootReducer	A combination of all defined reducer & third party reducer
<b>Theme</b>	
Global Styles	Defining styles
<b>Container &amp; Higher Order Components</b>	
Homepage	Responsible for handling homepage, about, contact routes
Dashboard page	Responsible for handling dashboard routes &
Toolbar	Handle list item & drag & drop from Toolbar to Content

Role & Responsibility	Description
Content	Handle widget node & connector in Mainboard
Outputlog	Handle widget information & data log
<b>Rest API &amp; Props</b>	
Request	Used to send request to get data from server
Response	Used to get data when requesting success, such as humidity, temperature,....
<b>Images &amp; Utils</b>	
<b>Provider</b>	
<b>Navigation</b>	
Navbar	A List of buttons such as: New, Help, Example1,...
Routes	Define routes of scenes for navigation
<b>Third-party Library</b>	
Dependencies	A list of dependencies that is required for operating the application
DevDependencies	A list of dependencies that is required for development environment

### 5.2.2. Module View based on Server

References Module View - Server on attached page.

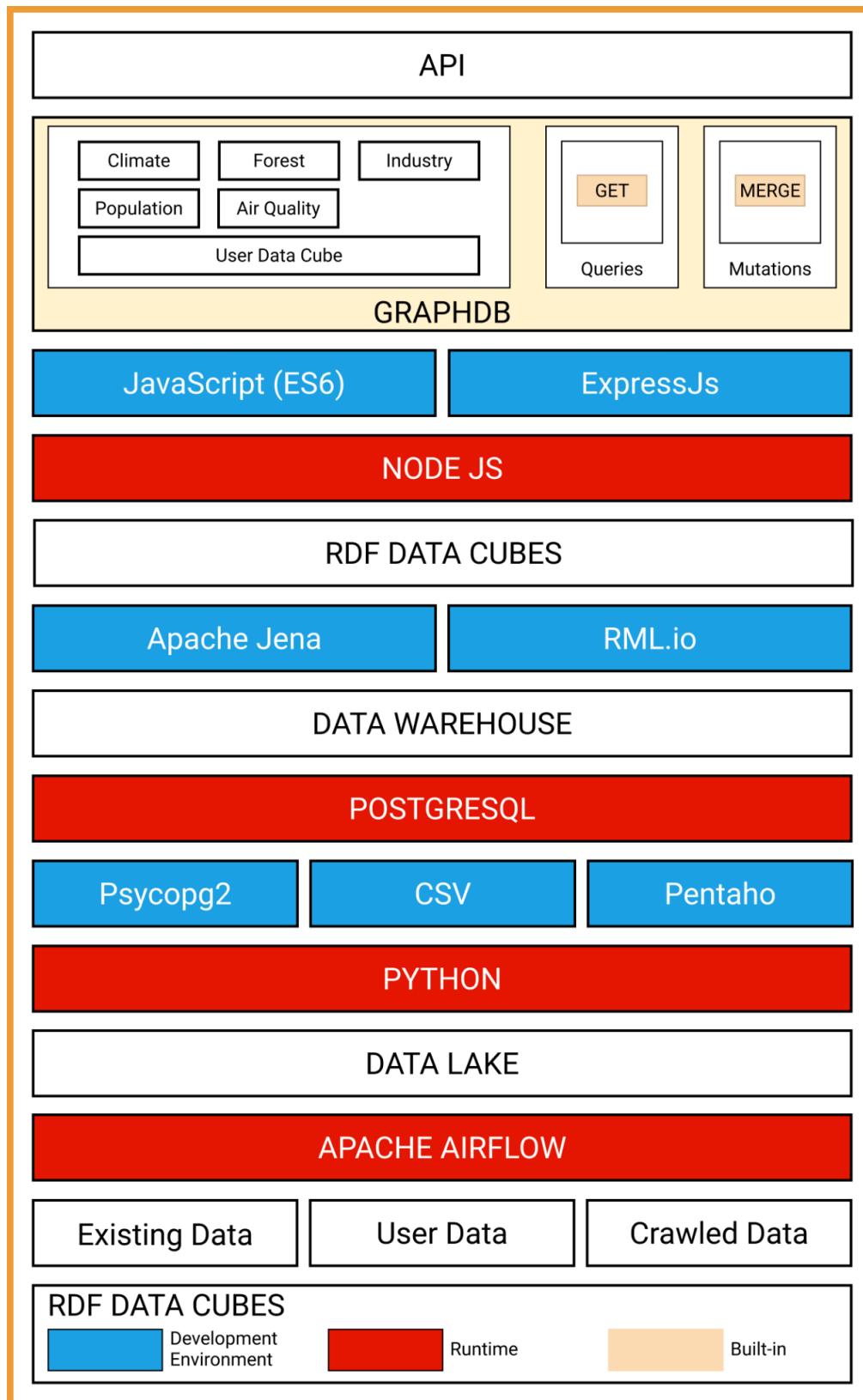


Figure 4: Module View - based on Server

**Prose:**

Role & Responsibility	Description
API	Execute the responses & requests
GraphDB	RDF Data Storing & Querying platform
Climate	Data of climate data cube ( humidity, rainfall, temperature values )
Forest	Data of forest data cube ( afforestation area, forest cover value )
Industry	Data of industry data cube
Population	Data of population data cube
Merge	Execute merging 2 or 3 or multi data cubes for creating a new data cube
Get	Get data of data cubes
Javascript ES6	Javascript ES6 Standard
ExpressJS	Library for programming API
NodeJS	Library for programming API
Apache Jena	Library for creating RDF Data Cube API
RML.io	Framework for building automatic Relational Data Warehouse to RDF Generator
Data Warehouse	Place where all the processed data is stored
PostgreSQL	Database system that help building the Data Warehouse
Psycopg2, CSV, Pentaho	Python frameworks that is used to programming auto ETL method
Python	Programming Language for ETL and importing the data sources
Data Lake	Place where storing all the non-processed data sources
Existing Data	Data sources of SDA
Crawled Data	Data sources that continuously updated with the crawler
User's Data	Data sources that are imported from user

### 5.3. ALLOCATION VIEW

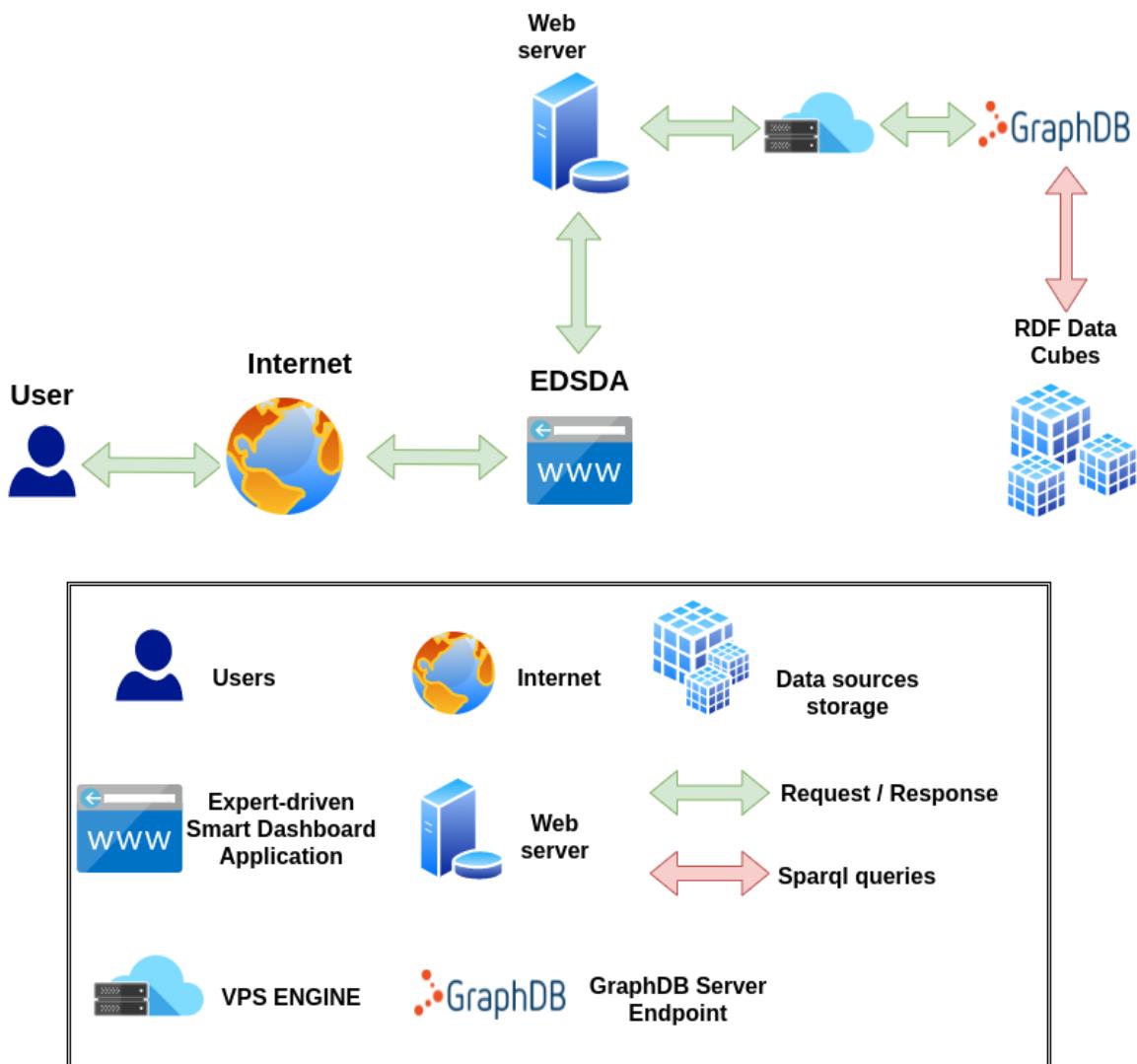


Figure 5: ESDA Allocation View

#### Prose:

The user can access our web app via the internet. When they use it, the web app will connect to the GCloud server to get the data via the internet.

Role & Responsibility	Description
Data sources storage	RDF Data Cubes is stored on GraphDB
User	User that interact with ESDA
Web server	HTTP Server with NodeJS
Expert-driven Smart Dashboard Application	Our application.
VPS Engine	Store and Execute ESDA's server

Role & Responsibility	Description
GraphDB Server Endpoint	Stage to received and run SPARQL for interacting with the data cubes
Request / Response	Get request metadata from client and response the data to client
SPARQL queries	Query language for execute the method of RDF Data Cubes

## 6. LOW-LEVEL ARCHITECTURE

### 6.1. Database Design

#### 6.1.1. Data warehouse architecture

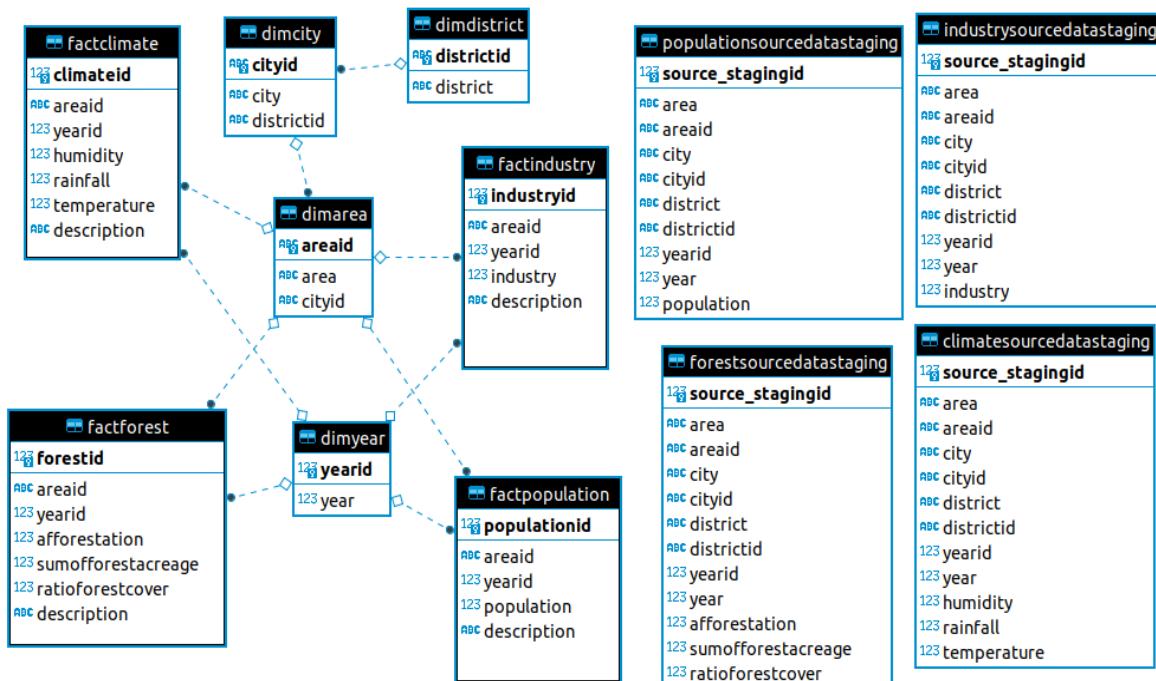


Figure 6: Data warehouse Schema

A data warehouse is subject-oriented as it offers information regarding a theme instead of companies' ongoing operations. These subjects can be sales, marketing, distributions, etc.

A data warehouse never focuses on the ongoing operations. Instead, it put emphasis on modeling and analysis of data for decision making. It also provides a simple and concise

view around the specific subject by excluding data which is not helpful to support the decision process.

### **6.1.2. Star schema, Galaxy schema, Fact Tables, and Dimension Tables**

The star schema architecture is the simplest data warehouse schema. It is called a star schema because the diagram resembles a star, with points radiating from a center. The center of the star consists of a fact table and the points of the star are the dimension tables.

From the star schema data warehouse architecture, relating more dimension tables. It becomes the Constellation Schema (Galaxy) with dimensions in this schema separated into separate dimensions based on the various levels of hierarchy. Moreover, it is possible to build this type of schema by splitting the one-star schema into more Star schemes.

A fact table typically has two types of columns: foreign keys to dimension tables id and measures those that contain numeric facts. A fact table can contain fact's data on a detailed or aggregated level.

A dimension is a structure usually composed of one or more hierarchies that categorize data. If a dimension hasn't got hierarchies and levels it is called flat dimension or list. The primary keys of each of the dimension tables are part of the composite primary key of the fact table. Dimensional attributes help to describe the dimensional value. They are normally descriptive, textual values. Dimension tables are generally smaller in size than fact tables.

Typical fact tables store data about sales while dimension tables data about the geographic region (markets, cities), clients, products, times, channels.

The EDSDA database system is designed based on the data warehouse platform. In EDSDA data warehouse, there are three main concepts (table types), those are staging area table, dimensional table, and fact table with star schema:

- Staging area tables: Climate Staging area, Forest Staging area, Industry Staging area, Population Staging area.
- Dimensional tables: dimyear, dimarea, dimcity, dimdistrict.
- Fact tables: factclimate, factforest, factindustry, factpopulation.

## **6.2. Data lake**

Data lake is a centralized repository that allows storing structured and unstructured data at any scale. Data is collected from multiple sources, and moved into the data lake in its original format. This process allows us to scale to data of any size, while saving time of defining data structures, schema, and transformations.

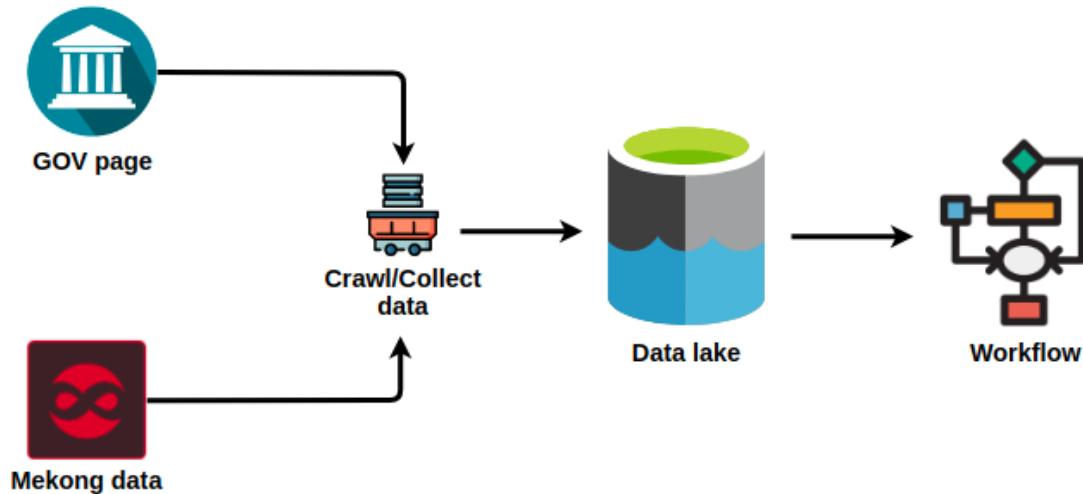


Figure 7: Data lake storing data.

Collecting and crawling data from the government page, Mekong site and starting the loading process to load data into the data warehouse with the design workflows.

### 6.3. ETL(Extract, Transform, Load) workflows

#### 6.3.1. Base workflow

The first workflow created by the available data sources found.

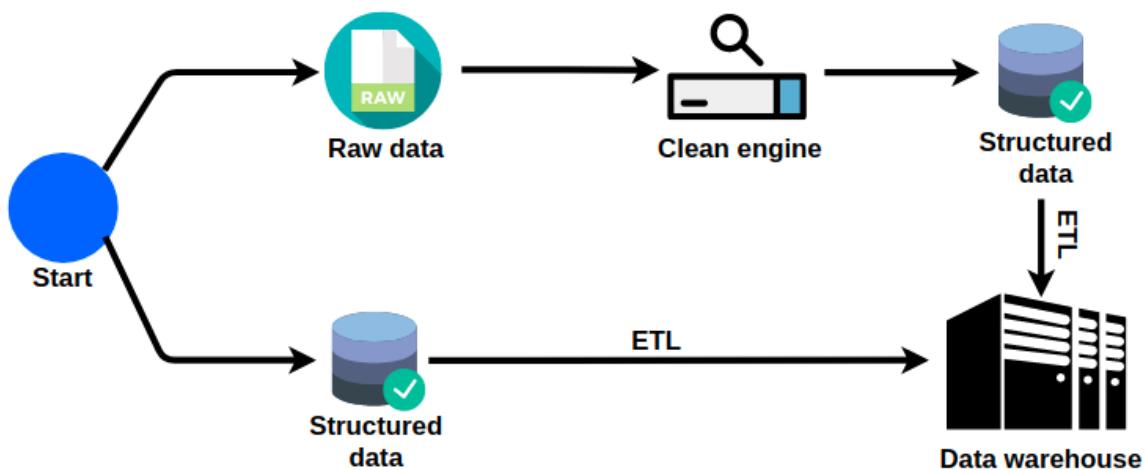


Figure 8: Workflow for the base database.

The purpose of this workflow is to create a base data warehouse, some of the raw data is really messy and can not be used for loading data to the Data warehouse so that having a process makes things easier.

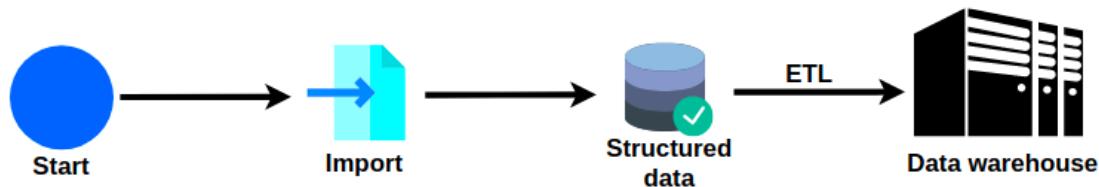
- The collected raw data will get through the clean engine, for each raw data source will have a specific clean engine to handle with a specific raw data. After

getting through the clean engine the raw data becomes structured ready to load into the Data warehouse.

- Clean engine contains some of the ETL works, removing the unnecessary parts of the raw data.
- Nothing much to do with structured data. It is ready to load data into the data warehouse.
- Loading structured data into the data warehouse, before this process happens we can perform our business constraints, removing unnecessary parts, adding logic, and etc ... before we load it into the data warehouse.

### 6.3.2. Import workflow

The data imported from the user interface gets through the process to ETL into the data warehouse.



*Figure 9: Workflow for importing data.*

The data imported must be structured data, constraining the input values really necessary because the input data are a variety of types.

Handling the imported input, letting the user select their own dimensions and measures. After the process, using HTTP requests to get data from the client and ready to import the data into the data warehouse.

### 6.3.3. Data pipeline workflow

Automating perform tasks, scheduling in the specific time to perform the data pipeline.

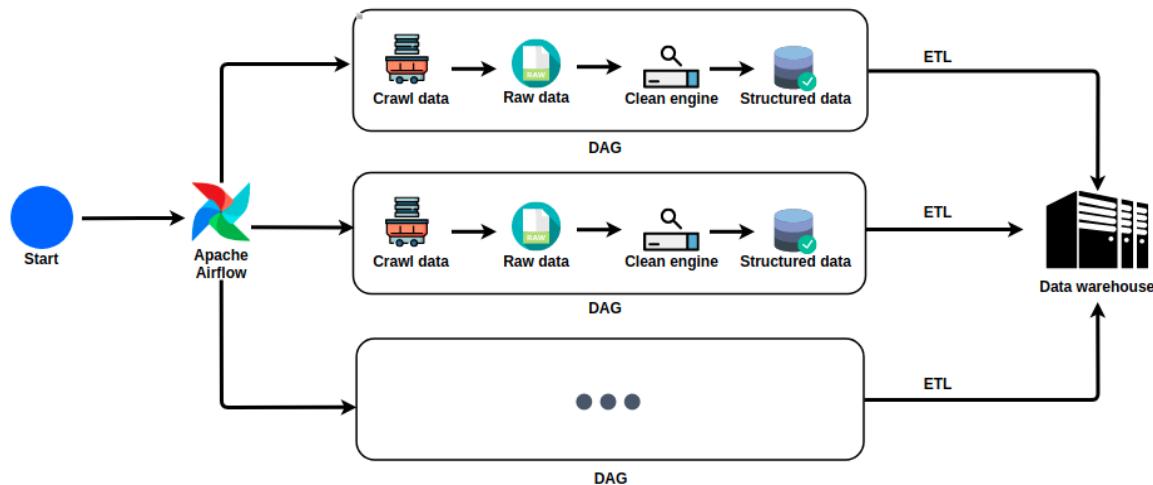


Figure 10: Data pipeline workflow with Apache Airflow managing DAGS.

Apache airflow allows us to schedule, automate performing tasks by using DAG.

DAG(Directed Acyclic Graph) shows us the data pipeline we are running, using dag easily manages the workflow by showing the error, exact phase in the flow is wrong, and logs ...etc.

Each DAG has its own page to crawl and a clean engine to clean. Basically, Each page has its own content and each content has its own structure because of that we have to design a whole new crawl script and clean script to perform with the new DAG.

Scheduling the time to perform specific DAG to get the workflow automatically.

## 6.4. RDF DATA CUBES

### 6.4.1. RDF DATA CUBES DESIGN

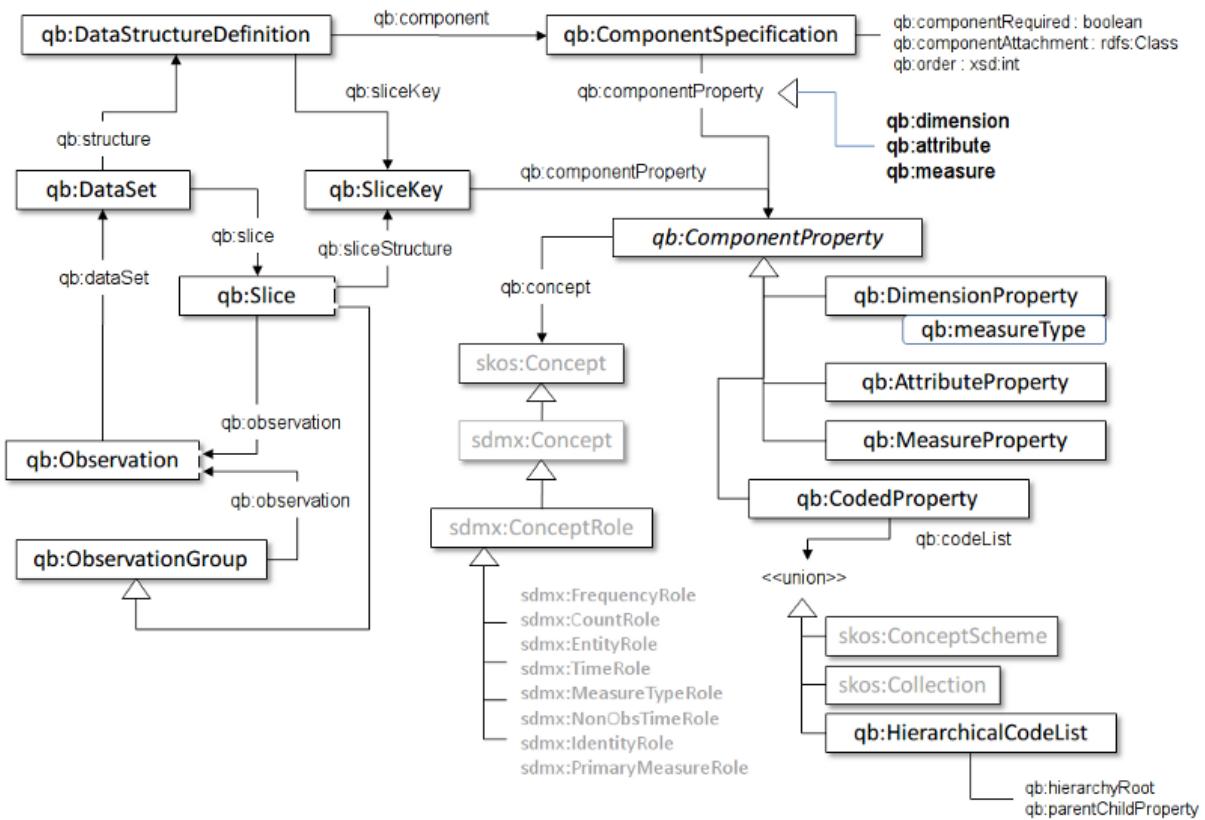


Figure 11: Pictorial summary of key terms and their relationship.

### 6.4.2. RDF Data cubes Architecture

The vocabulary of the RDF Data Cube was recently proposed as a W3C Recommendation for the destruction of various mathematical data using the RDF format. The model under this vocabulary is based on the SDMX model, the ISO standard for representing statistical data that can be shared between organizations. . The data cube, called the dataset, is an example of class qb:Dataset. Cube data cells, called observation, are the conditions of phase qb:Observation. Each look can have one or more attributes, sizes, and dimensions of a structure. Features defining qb type steps: MeasureProperty. This positive relationship correlates with recognition and measurement values (e.g., value of humidity), which have a range of numbers. Qualities and sizes of species qb:AttributeProperty and qb:DimensionProperty, respectively. The size represents the viewing context (e.g., the city in which the humidity is calculated), while the attributes provide additional information about the measure, such as the unit of measure.

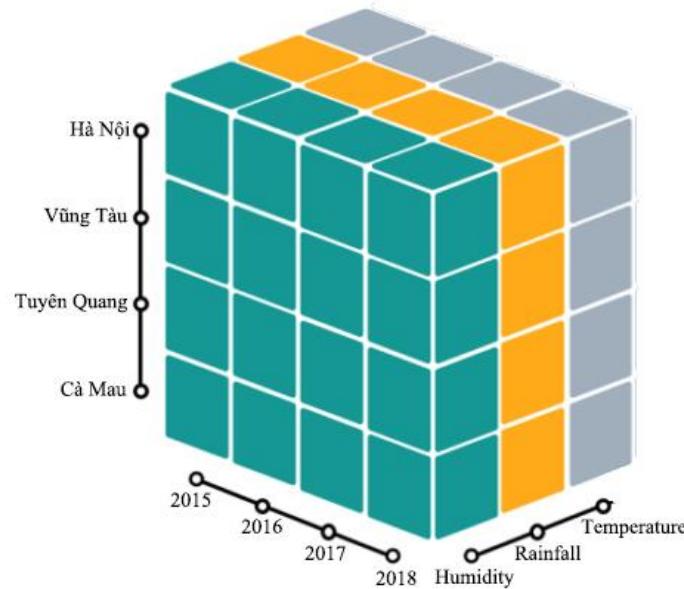


Figure 12: Climate Data Cube in 3D

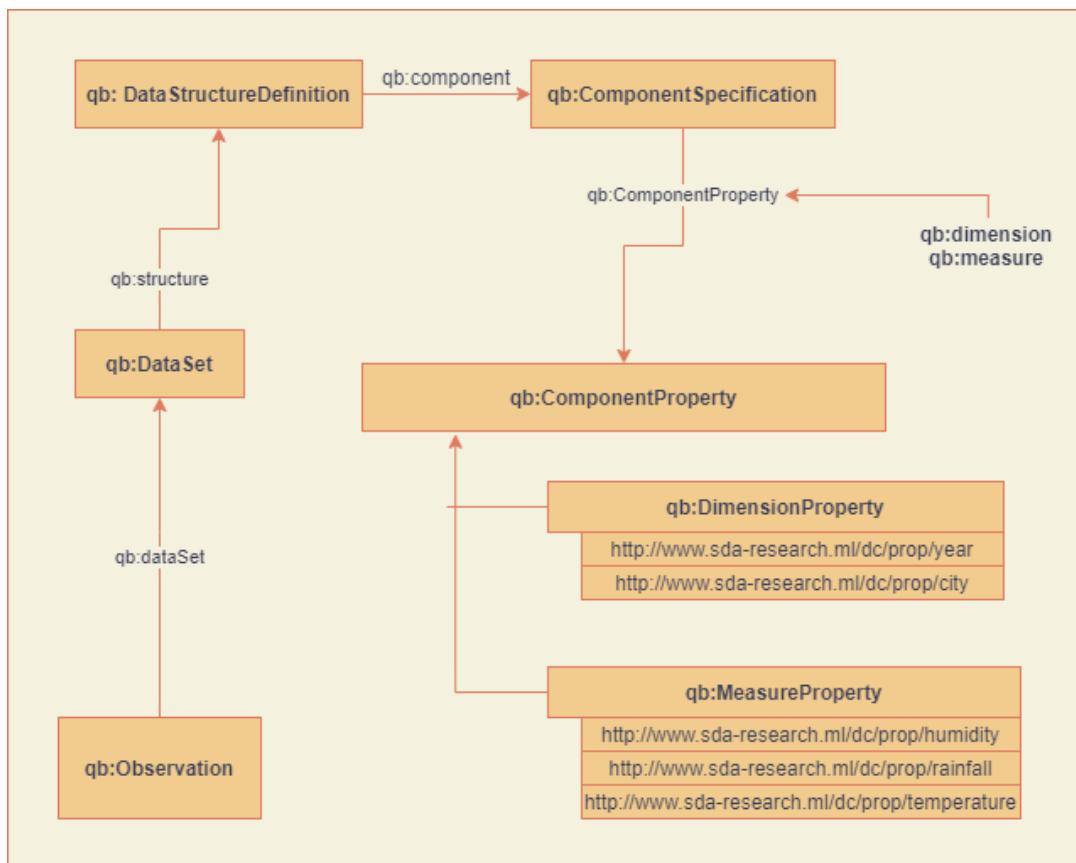


Figure 13: Climate Data Cube Structure

As the schema above, the Data Structure Definition has a child node ComponentSpecification, where qb:ComponentProperty is configured. The qb:ComponentProperty includes two main concepts, qb:DimensionProperty and

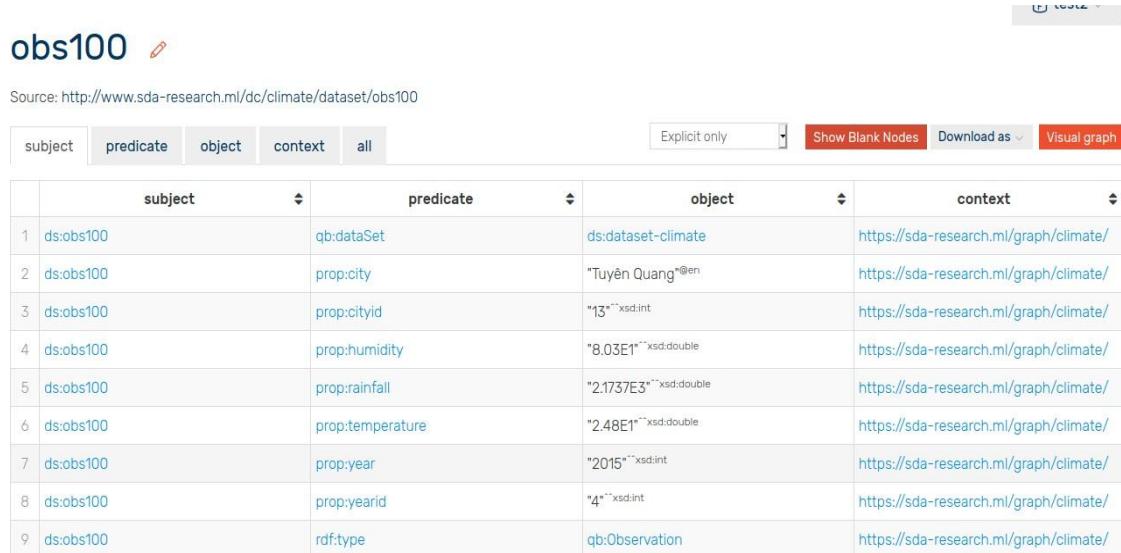
qb:MeasureProperty - the DSD definitions for each dimension and measure in the data warehouse. Each class has an specific URI that is regulated for querying with SPARQL Endpoint and makes the Linked Data Structure for the Data Cube.

For instance, The Climate Data Cube structure has:

- Dimensions:
  - City : Hà Nội, Vũng Tàu, Tuyêñ Quang, Cà Mau,etc...
  - Year : 2015, 2016, 2017, 2018,etc...
- Measures:
  - Humidity
  - Rainfall
  - Temperature
- Observations:
  - Humidity value of Hà Nội in 2015, 2016.
  - Rainfall value of Cà Mau in 2015, 2016.
  - Temperature of Tuyêñ Quang in 2017, 2018.
  - ...etc...

When a user need to figure out a set of statistical data that depends on multiple categories, such as humidity of an area/city in a specific time period (year), the data cube will refer to the statistics value (observation) of the measure (humidity) that match the dimension (city, year).

For instance, this is the observation of the rainfall value of Tuyen Quang in 2015.



The screenshot shows a SPARQL query results table titled "obs100". The table has columns: subject, predicate, object, context, and all. The "all" column contains a detailed view of the data. The data is as follows:

subject	predicate	object	context	all
1 ds:obs100	qb:dataSet	ds:dataset-climate	https://sda-research.ml/graph/climate/	
2 ds:obs100	prop:city	"Tuyêñ Quang"@en	https://sda-research.ml/graph/climate/	
3 ds:obs100	prop:cityid	"13"--xsd:int	https://sda-research.ml/graph/climate/	
4 ds:obs100	prop:humidity	"8.03E1"--xsd:double	https://sda-research.ml/graph/climate/	
5 ds:obs100	prop:rainfall	"2.1737E3"--xsd:double	https://sda-research.ml/graph/climate/	
6 ds:obs100	prop:temperature	"2.48E1"--xsd:double	https://sda-research.ml/graph/climate/	
7 ds:obs100	prop:year	"2015"--xsd:int	https://sda-research.ml/graph/climate/	
8 ds:obs100	prop:yearid	"4"--xsd:int	https://sda-research.ml/graph/climate/	
9 ds:obs100	rdf:type	qb:Observation	https://sda-research.ml/graph/climate/	

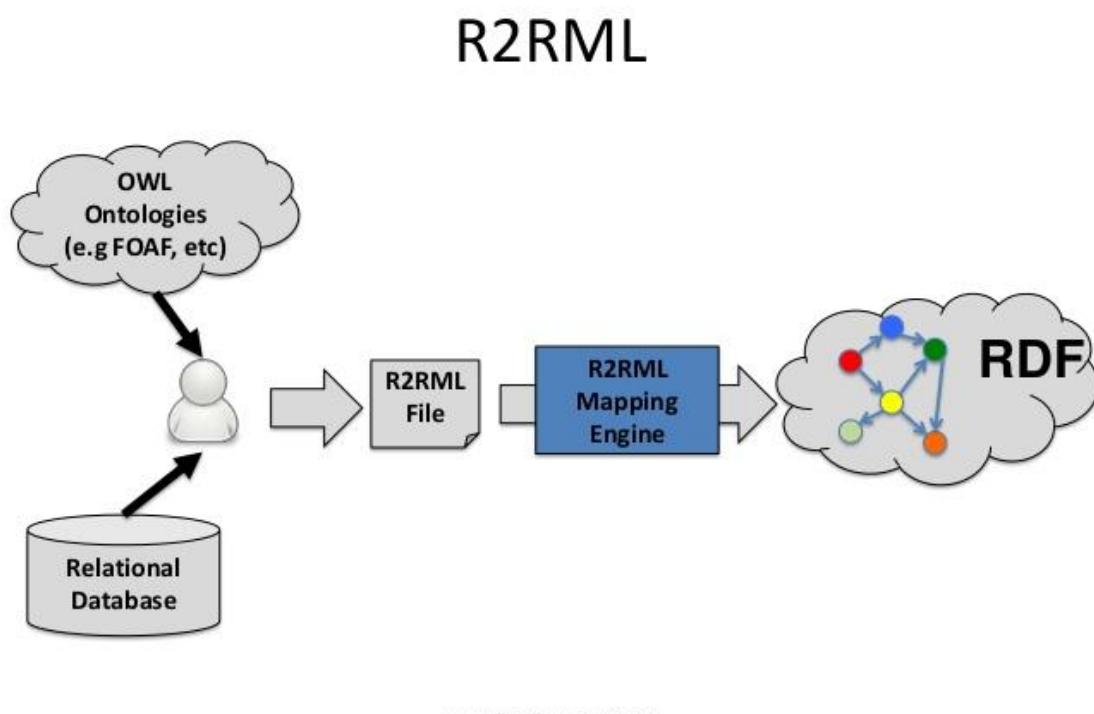
Figure 14: Observation 100th.

## 6.5. RDF DATA CUBES AUTO GENERATOR

### 6.5.1. Traditional R2RML Methods

At the present time, although there are many tools to convert data from tabular to linked data, there is still a lack of self-synchronization to make conversion flows fully automated without needing to rely on manual work each time.

In manual conversion methods, for example with classic R2RML engines like figure 6.1.4, we can see that the conversion relies on mapping files with the R2RML language. These mapping files are required to be manually defined by default, which leads to a lot of limitations in expanding the linked data stores.



[www.rdb2rdf.org - ISWC2013](http://www.rdb2rdf.org - ISWC2013)

Figure 15: Data flow of Manual Relational Database to RDF process.

Let's suppose, we have a system where data is crawled and processed continuously, not just a static database. With a traditional R2RML engine, this cannot be executed automatically, because as we know, each data source is specified by a mapping file, and it must be defined manually with the R2RML vocabulary. If there is a new data source, we must write an additional mapping file and restart the r2rml engine. This is completely inappropriate for systems where data pipelines are run and fed into a persistent data warehouse.

### 6.5.2. Automatic RDF Data Cubes Generator

For EDSDA, a system where data is scraped continuously and allows users to upload their primary data source to the system, it is not appropriate to use traditional R2RML methods. Therefore, we have researched and implemented an automatic conversion method from Relational Database and Tabular Data to RDF Data Cubes (Figure 16).

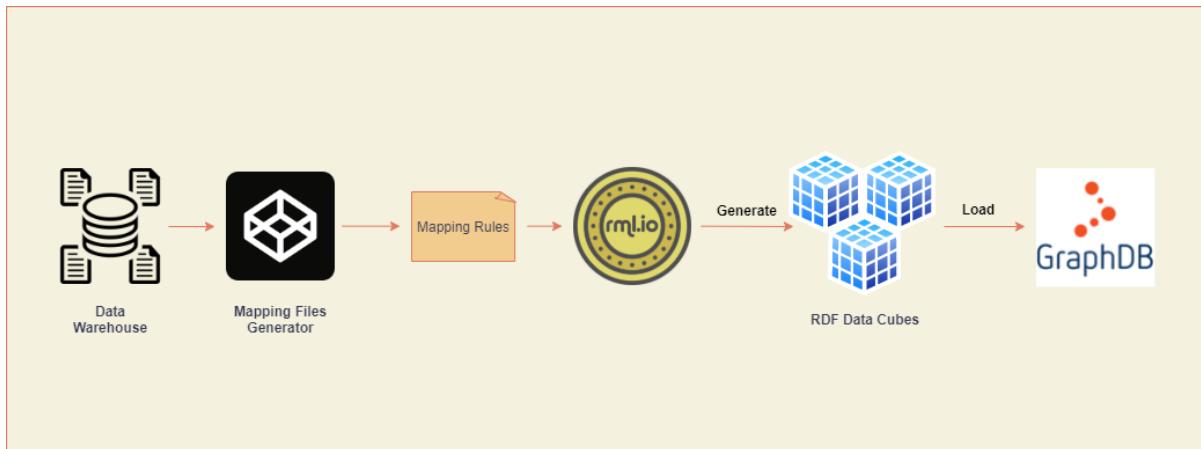


Figure 16: Data flow of Automatic RDF Data Cubes Generator.

As illustrated for the process above, the basic EDSDA RDF Data Cubes Generator will have 6 main parts:

- Data Warehouse: At first, data will be loaded manually from the data warehouse. Then, every time users upload new data sources, the system will check if the data warehouse is updated and start getting the new data source from it.
- Mapping Files Generator: After a new data source is obtained, the Mapping Files Generator will be launched. In the first version, it will handle the cases where the fact table has 1,2 or 3 measures and dimensions. Names of the fact table, measures, and dimensions will be assigned as variables in the R2RML mapping language.
- Mapping Rules: Mapping Files generated from Mapping Files Generator.
- RML.io: To execute the process of generating RDF Data Cubes from Mapping Rules and data sources, we use the RMLMapper and RMLStreamer frameworks provided by the RML.io organization. They are platforms that help users produce high quality linked data and knowledge graphs that are fast and easy to install.
- RDF Data Cubes that are generated from the previous parts.

- OntoText GraphDB: RDF data blocks will be uploaded to the Ontotext GraphDB datastore through the API to respond to data requests from the server and publicize open linked data to end users.

## 6.6. APPLICATION APIs

For interacting between client and server, we choose to deploy ESDA onto a Linux Virtual Private Server. Using Linux VPS, it is very easy and simple to develop, maintain and especially run the server in the most stable way.

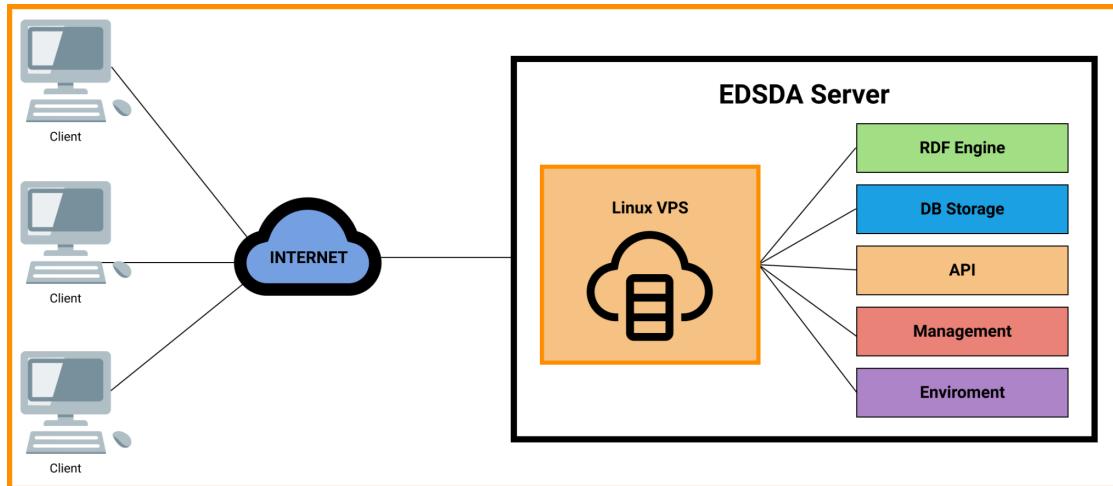


Figure 17: Deployment Diagram

We write the APIs of ESDA system with enapso-graphdb library, a Javascript library that give developers the most flexible way to configure, connect and interact with OntoText GraphDB platform, and the help of Express.JS, the most famous also strongest NodeJS library for building REST API server.

## 7. REFERENCES

- Server Endpoint: <http://server.sda-research.ml>
- Figma: <https://figma.com>
- ETL (Extract, Transform, and Load) Process: <https://www.guru99.com/etl-extract-load-process.html>
- Extract, transform, load: [https://en.wikipedia.org/wiki/Extract,\\_transform,\\_load](https://en.wikipedia.org/wiki/Extract,_transform,_load)
- What is Extract, Transform, Load? Definition, Process, and Tools: <https://www.talend.com/resources/what-is-etl/>
- ETL - Extract, Transform, Load: <https://www.webopedia.com/TERM/E/ETL.html>
- What is OLAP? Cube, Operations & Types in Data Warehouse: <https://www.guru99.com/online-analytical-processing.html>
- What is OLAP: <https://www.ibm.com/cloud/learn/olap>
- OLAP cube: [https://en.wikipedia.org/wiki/OLAP\\_cube](https://en.wikipedia.org/wiki/OLAP_cube)
- OLAP (Online analytical Processing): <https://techterms.com/definition/olap>
- The RDF Data Cube Vocabulary: <https://www.w3.org/TR/vocab-data-cube/>

- Optimizing RDF Data Cubes: <http://ceur-ws.org/Vol-1426/paper-02.pdf>
- CubeQA—Question Answering on RDF Data Cubes:  
[https://www.researchgate.net/publication/313073658\\_CubeQA-Question\\_Answering\\_on\\_RDF\\_Data\\_Cubes](https://www.researchgate.net/publication/313073658_CubeQA-Question_Answering_on_RDF_Data_Cubes)
- RDF Data Cube - graphical representation :  
[https://www.researchgate.net/figure/RDF-Data-Cube-graphical-representation\\_fig3\\_265690180](https://www.researchgate.net/figure/RDF-Data-Cube-graphical-representation_fig3_265690180)
- Linked clinical data cube architecture aligned with the RDF data cube:  
[https://www.researchgate.net/figure/Linked-clinical-data-cube-architecture-aligned-with-the-RDF-data-cube-Depicts-the\\_fig2\\_274700091](https://www.researchgate.net/figure/Linked-clinical-data-cube-architecture-aligned-with-the-RDF-data-cube-Depicts-the_fig2_274700091)
- The RDF Data Cube: <https://www.w3.org/TR/eo-qb/#Datacube>
- Data Warehouse Architecture: Types, Components, & Concepts:  
<https://www.astera.com/type/blog/data-warehouse-architecture/>
- Data Warehousing - Architecture:  
[https://www.tutorialspoint.com/dwh/dwh\\_architecture.htm](https://www.tutorialspoint.com/dwh/dwh_architecture.htm)
- Data Warehouse Architecture: Traditional vs. Cloud:  
<https://panoply.io/data-warehouse-guide/data-warehouse-architecture-traditional-vs-cloud/>
- RDB2RDF Tutorial (R2RML and Direct Mapping) at ISWC 2013:  
<https://www.slideshare.net/juansequeda/rdb2-rdf-tutorial-iswc2013>
- RML.io: <https://github.com/RMLio>
- Generating Executable Mappings from RDF Data Cube Data Structure Definitions:  
[https://link.springer.com/chapter/10.1007/978-3-030-02671-4\\_21](https://link.springer.com/chapter/10.1007/978-3-030-02671-4_21)
- Apache airflow documentation:  
<https://airflow.apache.org/docs/apache-airflow/stable/>



## CAPSTONE PROJECT 2

CMU-SE-451 / CMU-IS-451 / CMU-CS-451

### INTERFACE DESIGN

Version 2.2

Date: 1 - Mar - 2021

## EXPERT-DRIVEN SMART DASHBOARD APPLICATION

#### Submitted by

Vo Van Hoa  
Pham Van Tin  
Ky Huu Dong  
Tran Thi Thanh Kieu

#### Approved by

##### Capstone Project 2 - Mentor:

Name

Signature

Date

Binh, Thanh Nguyen \_\_\_\_\_  \_\_\_\_\_ 26 - May - 2020

PROJECT INFORMATION			
<b>Project Acronym</b>	EDSDA		
<b>Project Title</b>	Expert-Driven Smart Dashboard Application		
<b>Project Web URL</b>	<a href="https://sda-research.ml/">https://sda-research.ml/</a>		
<b>Start Date</b>	01 - Mar - 2021		
<b>End Date:</b>	02 - Jun - 2021		
<b>Lead Institution</b>	International School, Duy Tan University		
<b>Project Mentor</b>	Ph.D Binh, Thanh Nguyen		
<b>Scrum Master</b>	Hoa, Vo	hoavo.dng@gmail.com	0935.193.182
<b>Team Members</b>	Tin, Pham Van	tinphamvan123@gmail.com	0932.535.175
	Dong, Ky Huu	kyhuudong@gmail.com	0898.246.980
	Kieu, Tran Thi Thanh	thanhkieutran391@gmail.com	0358.583.251

DOCUMENT INFORMATION			
<b>Document Title</b>	Interface Design		
<b>Author(s)</b>	Team C2SE.06		
<b>Role</b>	[EDSDA] Interface Design v2.2		
<b>Date</b>	01 - Mar - 2021	Filename	[EDSDA] 005 Interface Design
<b>URL</b>	<a href="https://github.com/sdateamdtu2020/SDA-v2.0">https://github.com/sdateamdtu2020/SDA-v2.0</a>		
<b>Access</b>	Project and CMU Program		

## REVISION HISTORY

Version	Person(s)	Date	Description	Approval
Draft	Tin, Pham	01 - Mar - 2021	Initiate document	x
2.0	Tin, Pham	14 - Mar - 2021	Finish content of interface design	x
2.1	Tin, Pham	01 - Apr - 2021	Update new UI	x
2.2	Tin, Pham	20 - May - 2021	Fix typography	x

## TABLE OF CONTENTS

PROJECT INFORMATION	1
DOCUMENT INFORMATION	1
REVISION HISTORY	2
<b>TABLE OF CONTENTS</b>	<b>3</b>
<b>1. INTRODUCTION</b>	<b>4</b>
1.1. PURPOSE	4
1.2. INTENDED AUDIENCE	4
1.3. DOCUMENT OVERVIEW	4
<b>2. BRAND IDENTITY</b>	<b>5</b>
2.1. LOGO	5
2.2. TYPOGRAPHY	5
2.3. COLOR	6
<b>3. DETAILED INTERFACE DESIGN</b>	<b>7</b>
3.1. DEFAULT	7
3.2. INTRODUCTION	8
3.3. EXAMPLE	9
3.4. TOOLBOX	11
3.5. ABOUT	13
3.6. WIDGET WITH PORT OUT	14
3.7. WIDGET WITH PORT IN	14
3.8. WIDGET WITH TWO-PORT	15
<b>4. REFERENCES</b>	<b>15</b>

# 1. INTRODUCTION

## 1.1. PURPOSE OF DOCUMENT

This IDD of the SDA system has the following purpose:

- Document the design of user interface of functions of SDA
- Be the primary document driving the implementation of the user interfaces.
- Introduce interface design for the reader. Including the background, content information display, command buttons.

## 1.2. INTENDED AUDIENCES

This IDD is intended for the following audiences:

- Technical reviewers including:
- Scrum Master
- Product owner
- Implementers, including:
  - Developers.
  - Testers.

## 1.3. DOCUMENT OVERVIEW

This IDD is organized into the following sections:

- This document helps to better understand the interface of the Smart Dashboard Application System, detailed specification components and function of the Application.
- With this document, the development team can understand the structure and composition to create a consistent and complete system.
- Overview of the stages and modules of the software and create a complete interface.

## 2. BRAND IDENTITY

### 2.1. LOGO



### 2.2. TYPOGRAPHY

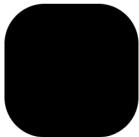
Font Family		
No	Name	Font Family
1	Primary	IBM Plex Sans
2	Secondary	Source Sans Pro
3	Another	sans-serif

Typography				
No	Name	Font size (px)	Line-Height (px)	Weight
1	Title	12	16	Bold, uppercase
2	Button	14	18	Bold
3	Headline	16	21	Bold
4	Body	14	18	Regular
5	Caption	12	16	Regular

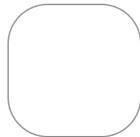


## 2.3. COLOR

### Basic



#000



#FFF

### Primary



#524F88

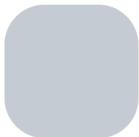


#FF0072



#1A192B

### Secondary



#C5CBD2



#F8F8F2



#221F38

### Button



#FF0072



#FFFFFF



#FFFFFF



#FFFFFF

transparent

Ghost btn



CTA btn

CTA :hover



#FF0072

transparent

Ghost :hover

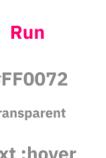


Run

#F8F8F2

transparent

Text btn



Run

#FF0072

transparent

Text :hover



## 3. DETAILED INTERFACE DESIGN

To view the all the screen and UI Design, connect to figma link:

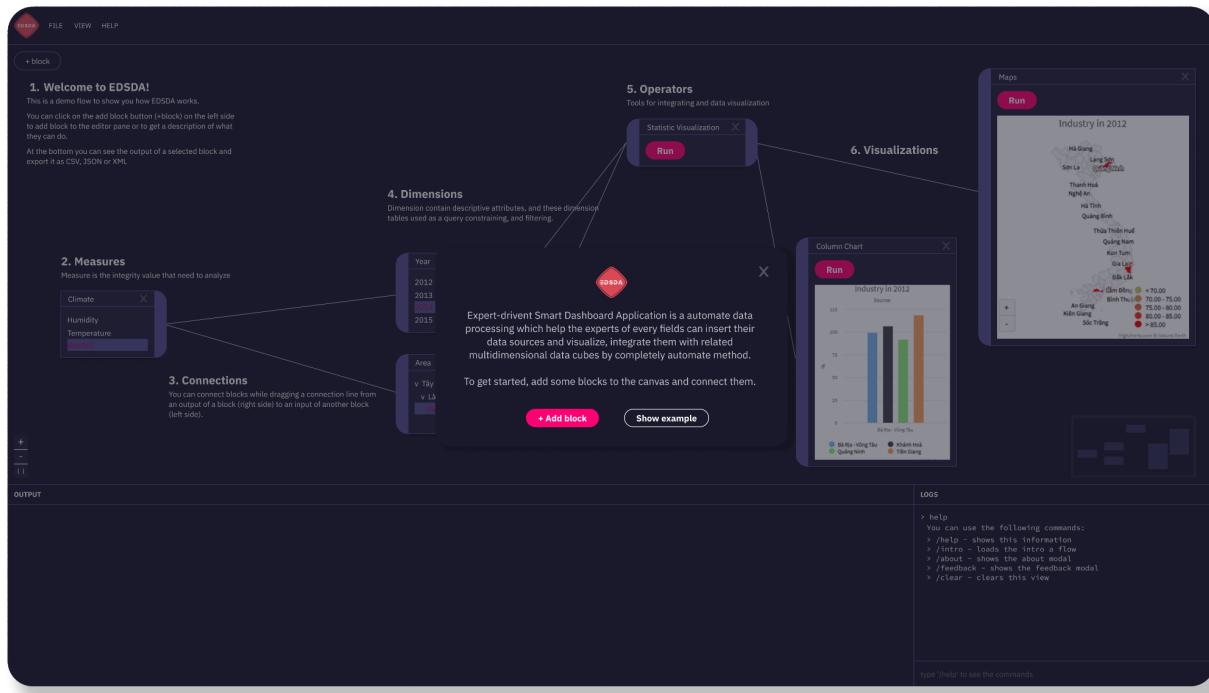
<https://www.figma.com/file/zTQJXv6HdxFLHDVznar2ms/EDSDA-Design?node-id=501%3A56>

### 3.1. DEFAULT



Default screen

## 3.2. INTRODUCTION



### Content:

Expert-driven Smart Dashboard Application is an automated data processing tool that helps the experts of every field insert their data sources and visualize, integrate them with related multidimensional data cubes by a completely automated method.

To get started, add some blocks to the canvas and connect them.

### Button onClick:

- Add block: clear all widgets, and open the “Add widget” popup
- Show example: in the 3.3 Show Example

### 3.3. EXAMPLE

The screenshot illustrates a data analysis workflow within the EDSDA interface:

- Welcome to EDSDA!**: A welcome message with instructions on how to add blocks.
- Measures**: A block listing measures like Climate, Humidity, Temperature, and Rainfall.
- Connections**: A block showing connections between different data sources.
- Dimensions**: A block listing dimensions like Year (2012, 2013, 2014, 2015) and Area (v. Tây Bắc Bộ, v. Lào Cai, Sapa).
- Operators**: Tools for integrating and data visualization.
- Visualizations**: A column chart titled "Industry in 2012" showing revenue for three categories: Bắc Giang (70.00), Lào Cai (75.00), and Sapa (85.00).
- Maps**: A map titled "Industry in 2012" showing the locations of various provinces in Vietnam, each marked with a colored dot corresponding to its industry value.

**OUTPUT** (Table):

country	population	life expectancy	Income
Australia	24800000	82.9	45800
Central African Republic	4740000	51.6	689
El Salvador	6410000	75.8	8290
Equatorial Guinea	1310000	66.1	20500
Guatemala	17200000	73.2	7530
Italy	59300000	82.6	35200
Malawi	19200000	61	1130
Malaysia	32000000	75.9	27200
Maldives	444000	80.1	13000
Mali	19100000	62.9	2020
Vanuatu	430000	61.6	5200

**LOGS**

```
> /help
You can use the following commands:
> /help - shows this information
> /intro - loads the intro a flow
> /about - shows the about modal
> /feedback - show the feedback modal
> /clear - closure this view
Type '/help' to see the commands
```

#### Content:

##### 1. Welcome

Content:

Welcome to EDSDA!

This is a demo flow to show you how EDSDA works.

You can click on the add block button (+block) on the left side to add block to the editor pane or to get a description of what they can do.

At the bottom, you can see the output of a selected block and export it as CSV, JSON, or XML

##### 2. Measures

Content:

Measure is the integrity value that needs to analyze

Show widget Climate, and select Rainfall

### 3. Connections

Content:

You can connect blocks while dragging a connection line from an output of a block (right side) to an input of another block (left side).

### 4. Dimensions

Content:

Dimensions contain descriptive attributes, and these dimension tables used as query constraining, and filtering.

Show widgets: Year (2014), Area (Sapa)

### 5. Operator

Content:

Tools for integrating and data visualization

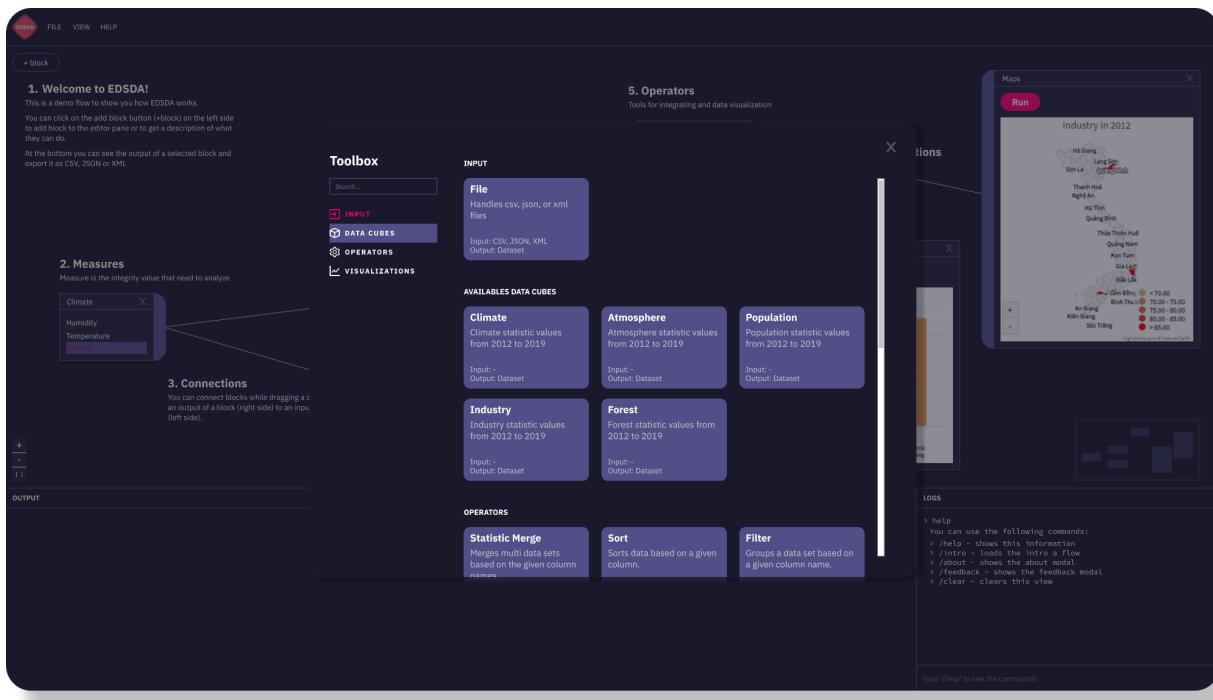
Show widgets: Statistic Visualization

### 6. Visualizations

Show widgets: Column Chart & Maps



## 3.4. TOOLBOX



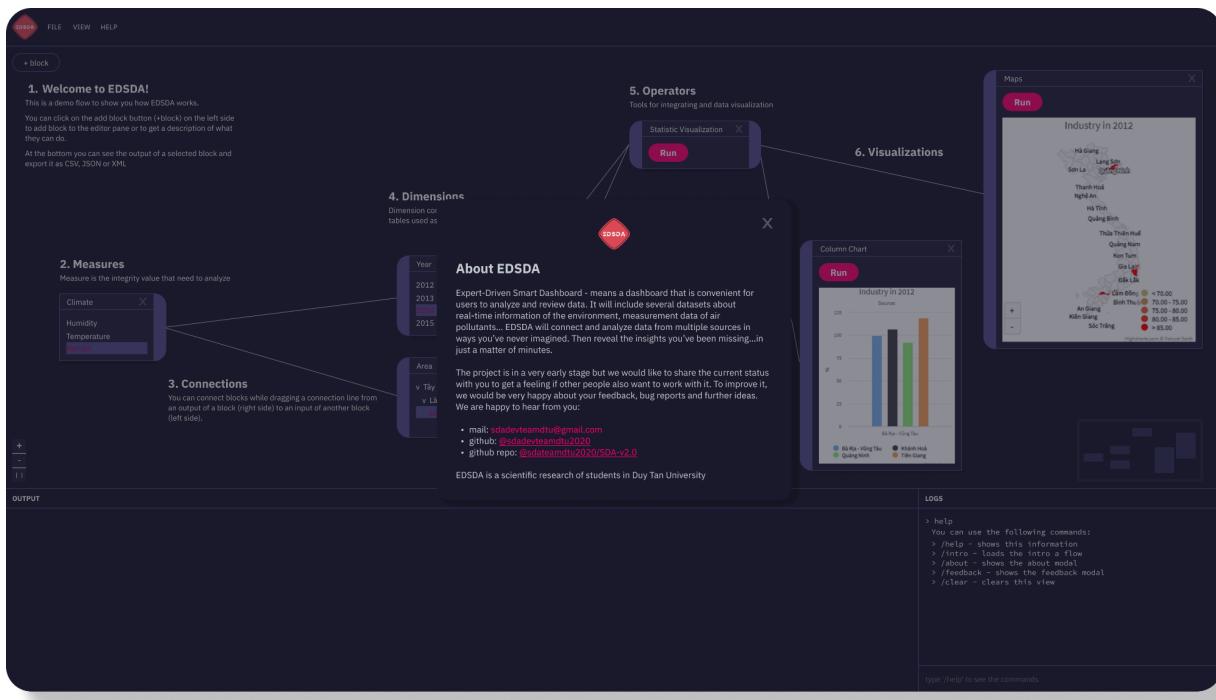
The Toolbox includes 4 types:

- Input
- Data Cubes
- Operators
- Visualizations

Title	Description	Input	Output
<b>Input</b>			
File	Handles csv, json, or xml files	CSV, JSON, XML	Dataset
<b>Data Cubes</b>			
Climate	Climate statistic values from 2012 to 2019	-	Dataset
Atmosphere	Atmosphere statistic values from 2012 to 2019	-	Dataset

Population	Population statistic values from 2012 to 2019	-	Dataset
Industry	Industry statistic values from 2012 to 2019	-	Dataset
Forest	Forest statistic values from 2012 to 2019	-	Dataset
<b>Operators</b>			
Statistic Merge	Merges multi data sets based on the given column names.	Dataset	Dataset
Sort	Sorts data based on a given column.	Dataset	Dataset
Filter	Groups a data set based on a given column name.	Dataset	Dataset
<b>Visualizations</b>			
Table	Merges multi data sets based on the given column names.	Dataset	Table
Column Chart	Displays a column chart of given x and y column names.	Dataset	Chart
Line Chart	Displays a line chart of given x and y column names.	Dataset	Chart
Multi Y-Axis Chart	Displays a multi Y-axis chart of given x and multi y column names.	Dataset	Chart
Pie Chart	Displays a pie chart of given x and y column names.	Dataset	Chart
Maps	Displays a maps of given data maps	Dataset	Maps

## 3.5. ABOUT



### Content:

#### About EDSA

Expert-Driven Smart Dashboard - means a dashboard that is convenient for users to analyze and review data. It will include several datasets about real-time information of the environment, measurement data of air pollutants... EDSA will connect and analyze data from multiple sources in ways you've never imagined. Then reveal the insights you've been missing...in just a matter of minutes.

The project is in a very early stage but we would like to share the current status with you to get a feeling if other people also want to work with it. To improve it, we would be very happy about your feedback, bug reports and further ideas.

We are happy to hear from you:

mail: [sdadateamdtu@gmail.com](mailto:sdadateamdtu@gmail.com)

github: [@sdadateamdtu2020](https://github.com/sdateamdtu2020) (<https://github.com/sdateamdtu2020>)

github repo: [@sdateamdtu2020/SDA-v2.0](https://github.com/sdateamdtu2020/SDA-v2.0) (<https://github.com/sdateamdtu2020/SDA-v2.0>)

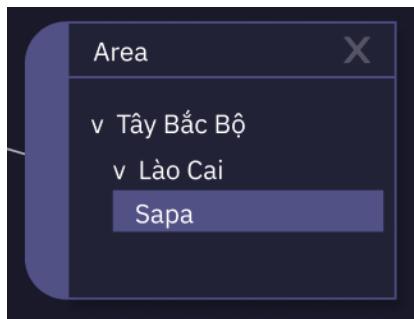
EDSDA is a scientific research of students in Duy Tan University

### 3.6. WIDGET WITH PORT OUT



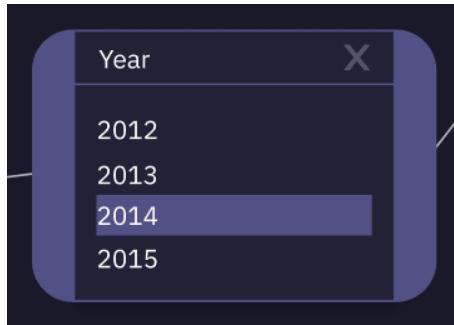
Name	Font size	Color	Line height	Another
Title	12	#FFF	16	
Tree Item	12	#FFF	16	
Tree Item hover	12	#FFF	16	Background: #524F88

### 3.7. WIDGET WITH PORT IN



Name	Font size	Color	Line height	Another
Title	12	#FFF	16	
Tree Item	12	#FFF	16	
Tree Item hover	12	#FFF	16	Background: #524F88

### 3.8. WIDGET WITH TWO PORT



Name	Font size	Color	Line height	Another
Title	12	#FFF	16	
Tree Item	12	#FFF	16	
Tree Item hover	12	#FFF	16	Background: #524F88

## 4. REFERENCES

- Figma link:  
<https://www.figma.com/file/zTQJXv6HdxFLHDVznar2ms/EDSA-Design?node-id=0%3A1>
- 



## CAPSTONE PROJECT 2

CMU-SE-451 / CMU-IS-451 / CMU-CS-451

### DATABASE DESIGN

Version 2.0

Date: 1 - Mar - 2021

## EXPERT-DRIVEN SMART DASHBOARD APPLICATION

### Submitted by

Vo Van Hoa  
Pham Van Tin  
Ky Huu Dong  
Tran Thi Thanh Kieu

### Approved by

#### Capstone Project 2 - Mentor:

Name

Signature

Date

Binh, Thanh Nguyen \_\_\_\_\_

A handwritten signature in blue ink, appearing to read 'Binh, Thanh Nguyen'.

\_\_\_\_\_ 26 - May - 2020

PROJECT INFORMATION			
<b>Project Acronym</b>	EDSDA		
<b>Project Title</b>	Expert-Driven Smart Dashboard Application		
<b>Project Web URL</b>	<a href="https://sda-research.ml/">https://sda-research.ml/</a>		
<b>Start Date</b>	01 - Mar - 2021		
<b>End Date:</b>	02 - Jun - 2021		
<b>Lead Institution</b>	International School, Duy Tan University		
<b>Project Mentor</b>	Ph.D Binh, Thanh Nguyen		
<b>Scrum Master</b>	Hoa, Vo	hoavo.dng@gmail.com	0935.193.182
<b>Team Members</b>	Tin, Pham Van	tinphamvan123@gmail.com	0932.535.175
	Dong, Ky Huu	kyhuudong@gmail.com	0898.246.980
	Kieu, Tran Thi Thanh	thanhkieutran391@gmail.com	0358.583.251

DOCUMENT INFORMATION			
<b>Document Title</b>	Database Design		
<b>Author(s)</b>	Team C2SE.06		
<b>Role</b>	[EDSDA] DatabaseDesign_v2.0		
<b>Date</b>	01 - Mar - 2021	Filename	[EDSDA] 006 Database Design
<b>URL</b>	<a href="https://github.com/sdateamdtu2020/SDA-v2.0">https://github.com/sdateamdtu2020/SDA-v2.0</a>		
<b>Access</b>	Project and CMU Program		

## REVISION HISTORY

# TABLE OF CONTENTS

PROJECT INFORMATION	1
DOCUMENT INFORMATION	1
REVISION HISTORY	2
<b>TABLE OF CONTENTS</b>	<b>3</b>
<b>1. INTRODUCTION</b>	<b>5</b>
1.1. PURPOSE	5
1.2. DOCUMENT OBJECTIVES	5
1.3. INTENDED AUDIENCE	5
1.4. SCOPE, APPROACH AND METHODS	6
1.5. SYSTEM OVERVIEW	6
1.6. ACRONYMS AND ABBREVIATIONS	6
1.7. KEY PERSONNEL	6
<b>2. ASSUMPTIONS, CONSTRAINTS AND DEPENDENCIES</b>	<b>7</b>
2.1. ASSUMPTIONS	7
2.2. CONSTRAINTS	7
<b>3. SYSTEM OVERVIEW</b>	<b>7</b>
3.1. DATABASE MANAGEMENT SYSTEM CONFIGURATION	7
3.2. DATABASE SOFTWARE UTILITIES	7
3.3. SUPPORT SOFTWARE	8
<b>4. ARCHITECTURE</b>	<b>8</b>
4.1. HARDWARE & SOFTWARE ARCHITECTURE	8
4.2. DATASTORES	8
<b>5. DATABASE-WIDE DESIGN DECISIONS</b>	<b>9</b>
5.1. KEY FACTORS INFLUENCING DESIGN	9
5.2. PERFORMANCE AND AVAILABILITY DECISIONS	9
<b>6. DATABASE ADMINISTRATIVE FUNCTIONS</b>	<b>9</b>
6.1. RESPONSIBILITY	9
6.2. APPLICATIONS/SYSTEMS USING THE DATABASE	10
6.3. PHYSICAL DESIGN	10
6.3.1. FACT-INDUSTRY	10

6.3.2. FACT-POPULATION	11
6.3.3. FACT-CLIMATE	11
6.3.4. FACT-FOREST	12
6.3.5. DIMAREA	12
6.3.6. DIMCITY	13
6.3.7. DIMDISTRICT	13
6.3.8. DIMYEAR	13
6.3.9. FOREST-SOURCE-DATA-STAGING	14
6.3.10. CLIMATE-SOURCE-DATA-STAGING	15
6.3.11. POPULATION-SOURCE-DATA-STAGING	16
6.3.12. INDUSTRY-SOURCE-DATA-STAGING	16
6.4. RDF DATA CUBES DESIGN	17
6.4.1. DIMENSION	17
6.4.2. MEASURE	17
6.4.3. STRUCTURE, PATTERNS AND LOCAL PREFIXES	18
6.4.4. EXTERNAL VOCABULARIES	21
6.4.5. CLASS HIERARCHY	22
6.4.6. CLASS RELATIONSHIPS	23
6.5. ENTITY MAPPING	24
6.5.1. ENTITY MAPPING DIAGRAM	24
6.5.2. INDUSTRY	24
6.5.3. CLIMATE	24
6.5.4. FOREST	25
6.5.5. POPULATION	25
<b>7. REFERENCES</b>	<b>25</b>

# 1. INTRODUCTION

The Database Design maps the logical data model to the target database management system with consideration to the system's performance requirements. The Database Design converts logical or conceptual data constructs to physical data constructs (e.g tables,...) of the target Database Management System.

## 1.1. PURPOSE

The purpose of the Database Design is to ensure that every database transaction meets or exceeds its performance requirements. This document takes into account data and transaction volume to produce a schema and environment that will meet necessary performance

## 1.2. DOCUMENT OBJECTIVES

The Database Design Document has the following objectives:

- To describe the design of a database, that is, a collection of related data stored in one or more computerized files that can be accessed by users or developers via a DBMS
- To serve as a basis for implementing the database and related software units. It provides the acquirer visibility into the design and provides information necessary for software development.

## 1.3. INTENDED AUDIENCE

This document is intended for the following audiences:

Technical developers, who must evaluate the quality of this document

Developer including:

- Architects, whose overall architecture design must meet the requirements specified in this document.
  - Designers, whose design must meet the requirements specified in this document.
  - Developers, whose software must implement the requirements specified in this document.
  - Quality Assurance personnel, whose test cases must validate the requirements specified in this document.
- 

## 1.4. SCOPE, APPROACH AND METHODS

The Database Design for the SDA is composed of definitions for database objects derived by mapping entities to tables, attributes to columns, unique identifiers to unique keys and relationships to foreign keys.

## 1.5. SYSTEM OVERVIEW

System Overview		Details
<b>System Name</b>		SMART DASHBOARD APPLICATION
<b>System type</b>		Web application
<b>Operational status</b>		In development

## 1.6. ACRONYMS AND ABBREVIATIONS

Acronym/Abbreviation	Meaning
SDA	SMART DASHBOARD APPLICATION
PG	Postgres
RDMS	Relational Database Management System
DBMS	Database Management System
RDC	RDF Data Cube

## 1.7. KEY PERSONNEL

Full Name	Email	Phone number	Role
Hoa, Vo	hoavo.dng@gmail.com	0935.193.182	Scrum master
Tin, Pham Van	tinphamvan123@gmail.com	0932.535.175	Team member
Dong, Ky Huu	kyhuudong@gmail.com	0898.246.980	Team member
Kieu, Tran Thanh Thi	thanhkieutran391@gmail.com	0358.583.251	Team member

## 2. ASSUMPTIONS, CONSTRAINTS AND DEPENDENCIES

### 2.1. ASSUMPTIONS

- Users can drag the widgets to generate charts, maps, or any available solution on GUI.
- Users can link nodes to the others nodes to generate charts based on their needs.

### 2.2. CONSTRAINTS

- Users can only use all the functions available on GUI.

## 3. SYSTEM OVERVIEW

### 3.1. DATABASE MANAGEMENT SYSTEM CONFIGURATION

- **System:** GraphDB
- **Vendor:** Ontotext
- **Services enabled:**
  - RDF Data storage.
- **System:** Linux VM
- **Vendor:** Google
- **Services enabled:**
  - Linux Virtual Machine

### 3.2. DATABASE SOFTWARE UTILITIES

Vendor	Product	Version	Comments
pgAdmin	PGAdmin	4.0.0	This application enables the ability to manage the cloud PG from local machine

### 3.3. SUPPORT SOFTWARE

Product	Version	Purpose
DBeaver	Offline software	Help showing the table itself and its relations with other tables inside the schema on PostgreSQL platform

## 4. ARCHITECTURE

### 4.1. HARDWARE & SOFTWARE ARCHITECTURE

Handled by GraphDB

### 4.2. DATASTORES

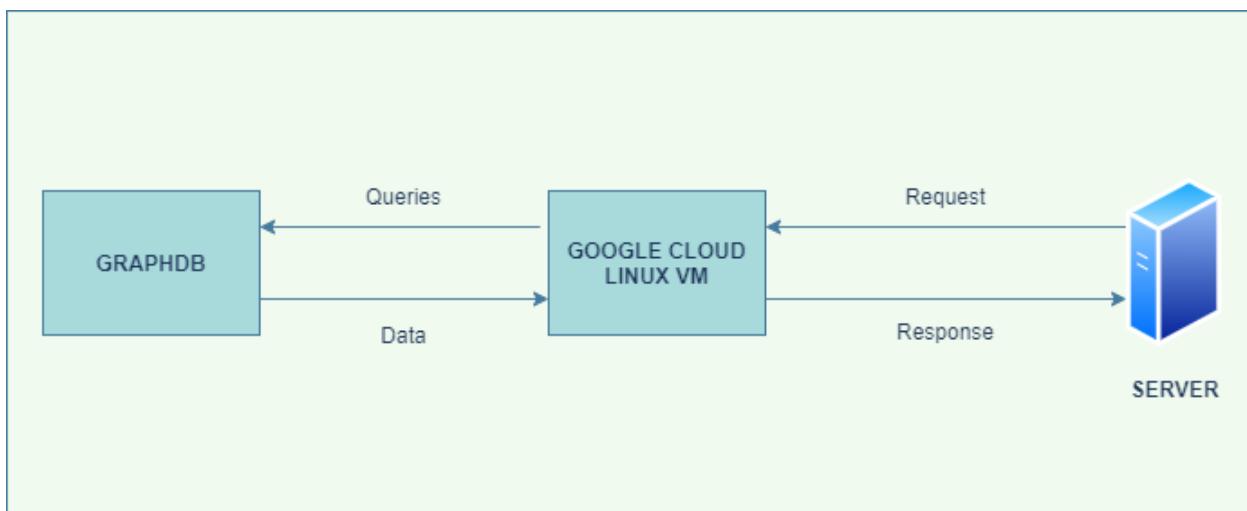


Figure 1. Datastores

GraphDB is a family of well-functioning, robust and awesome RDF databases. It reduces the load and use of connected cloud databases, as well as the RDF resources. We use GraphDB on a Linux Virtual Machine that is runned on Google Cloud platform for better querying, high performance processing and faster response.

## 5. DATABASE-WIDE DESIGN DECISIONS

### 5.1. KEY FACTORS INFLUENCING DESIGN

- The database should be designed independent when the frontend and backend are still being developed.
- The database should be designed to meet the data warehouse principles.

### 5.2. PERFORMANCE AND AVAILABILITY DECISIONS

- We use common dimensional tables for all the fact tables for better performance.
- The data warehouse should be designed with the star schema for speed-up querying and processing time.
- The data warehouse should be redundant for reducing response time and meet up with the backend design pattern of the system.
- Data warehouse should be stored with PostgreSQL.
- RDF Data Cubes should be fully defined.
- RDF Data Cubes should be deployed on GraphDB.
- Class hierarchy and relationships should be done before extracting the data.
- RDF Data Cubes should be validated correctly before writing RestAPI.

## 6. DATABASE ADMINISTRATIVE FUNCTIONS

### 6.1. RESPONSIBILITY

Role	Name	Responsibility	Email Address
Database Administrator	Dong, Ky Huu		kyhuudong@gmail.com
System Administrator	Hoa, Vo		hoavo.dng@gmail.com
Security Administrator	Hoa, Vo		hoavo.dng@gmail.com

## 6.2. APPLICATIONS/SYSTEMS USING THE DATABASE

System ID	Model	Version	System Code
EDSDA-Web	NA	In Development	NA

## 6.3. PHYSICAL DESIGN

### 6.3.1. FACT-INDUSTRY

factindustry				
Field	Type	Constraint	Nullable	Description
Industryid	int(auto increment)	PK	No	Industry id is primary key to specific unique row
areaid	VARCHAR	FK	No	area id is the foreign key and is the primary key of the area dimension
yearid	int	FK	No	Year id is the foreign key and is the primary key of the city dimension
industry	double		Yes	Industrial data
description	text		Yes	Description for each unique row id

### 6.3.2. FACT-POPULATION

factpopulation				
Field	Type	Constraint	Nullable	Description
populationid	int(auto increment)	PK	No	Population id is primary key to specific unique row
areaid	VARCHAR	FK	No	area id is the foreign key and is the primary key of the area dimension
yearid	int	FK	No	Year id is the foreign key and is the primary key of the city dimension
population	double		Yes	Population data
description	text		Yes	Description for each unique row id

### 6.3.3. FACT-CLIMATE

factclimate				
Field	Type	Constraint	Nullable	Description
climateid	int(auto increment)	PK	No	Climate id is primary key to specific unique row
areaid	VARCHAR	FK	No	area id is the foreign key and is the primary key of the area dimension
yearid	int	FK	No	Year id is the foreign key and is the primary key of the city dimension
humidity	double		Yes	Humidity data
rainfall	double		Yes	Rainfall data
temperature	double		Yes	Temperature data
description	text		Yes	Description for each unique row id

### 6.3.4. FACT-FOREST

factforest				
Field	Type	Constraint	Nullable	Description
forestid	int(auto increment)	PK	No	Forest id is primary key to specific unique row
areaid	VARCHAR	FK	No	area id is the foreign key and is the primary key of the area dimension
yearid	int	FK	No	Year id is the foreign key and is the primary key of the city dimension
afforestation	double		Yes	Afforestation data
sumofforestacrea ge	double		Yes	Forest acreage data
ratioforestcover	double		Yes	The cover ratio of forest data
description	text		Yes	Description for each unique row id

### 6.3.5. DIMAREA

dimcity				
Field	Type	Constraint	Nullable	Description
areaid	VARCHAR	PK	No	Area id is primary key to specific unique row
area	VARCHAR		No	Area name data
cityid	VARCHAR		No	City id is the foreign key and is the primary key of the city dimension

### 6.3.6. DIMCITY

dimcity				
Field	Type	Constraint	Nullable	Description
cityid	VARCHAR	PK	No	City id is primary key to specific unique row
city	VARCHAR		No	City name data
districtid	VARCHAR		NO	District id is the foreign key and is the primary key of the district dimension

### 6.3.7. DIMDISTRICT

dimcity				
Field	Type	Constraint	Nullable	Description
districtid	VARCHAR	PK	No	District id is primary key to specific unique row
district	VARCHAR		No	District name data

### 6.3.8. DIMYEAR

dimyear				
Field	Type	Constraint	Nullable	Description
yearid	int(auto increment)	PK	No	Year id is primary key to specific unique row
year	int		No	Year data

### 6.3.9. FOREST-SOURCE-DATA-STAGING

forestsourcedatastaging				
Field	Type	Constraint	Nullable	Description
Source_stagingid	int(auto increment)	PK	No	Source_staging id is primary key to specific unique row
area	VARCHAR		Yes	Area name data
areaid	VARCHAR		Yes	Area id from dimarea
city	VARCHAR		Yes	City name data
cityid	VARCHAR		Yes	City id from dimcity
district	VARCHAR		Yes	District name data
districtid	VARCHAR		Yes	District id from dimdistrict
year	int		Yes	Year data
yearid	int		Yes	Year id from dimyear
afforestation	double		Yes	Afforestation data
sumofforestacreage	double		Yes	The forest acreage data
ratioforestcover	double		Yes	The cover ratio of forest data

### 6.3.10. CLIMATE-SOURCE-DATA-STAGING

climatesourcedatastaging				
Field	Type	Constraint	Nullable	Description
source_stagingid	int(auto increment)	PK	No	Source_staging id is primary key to specific unique row
area	VARCHAR		Yes	Area name data
areaid	VARCHAR		Yes	Area id from dimarea
city	VARCHAR		Yes	City name data
cityid	VARCHAR		Yes	City id from dimcity
district	VARCHAR		Yes	District name data
districtid	VARCHAR		Yes	District id from dimdistrict
year	int		Yes	Year data
yearid	int		Yes	Year id from dim year
humidity	double		Yes	Humidity data
rainfall	double		Yes	Rainfall data
temperature	double		Yes	Temperature data

### 6.3.11. POPULATION-SOURCE-DATA-STAGING

populationsourcedatastaging				
Field	Type	Constraint	Nullable	Description
source_stagingid	int(auto increment)	PK	No	Source_staging id is primary key to specific unique row
area	VARCHAR		Yes	Area name data
areaid	VARCHAR		Yes	Area id from dimarea
city	VARCHAR		Yes	City name data
cityid	VARCHAR		Yes	City id from dimcity
district	VARCHAR		Yes	District name data
districtid	VARCHAR		Yes	District id from dimdistrict
year	int		Yes	Year data
yearid	int		Yes	Year id from dimyear
population	double		Yes	Population data

### 6.3.12. INDUSTRY-SOURCE-DATA-STAGING

industryourcedatastaging				
Field	Type	Constraint	Nullable	Description
source_stagingid	int(auto increment)	PK	No	Source_staging id is primary key to specific unique row
area	VARCHAR		Yes	Area name data
areaid	VARCHAR		Yes	Area id from dimarea

Field	Type	Constraint	Nullable	Description
city	VARCHAR		Yes	City name data
cityid	VARCHAR		Yes	City id from dimcity
district	VARCHAR		Yes	District name data
districtid	VARCHAR		Yes	District id from dimdistrict
year	int		Yes	Year data
yearid	int		Yes	Year id from dimyear
industry	double		Yes	Industrial data

## 6.4. RDF Data Cubes Design

### 6.4.1. Dimension

Column	Description
city	The area belongs to dataset
cityid	ID name transformed from city data
year	The time period belongs to dataset

### 6.4.2. Measure

Column	Description
humidity	Observed humidity value
rainfall	Observed rainfall value
temperature	Observed temperature value
forestarea	Observed forest area value
forestcover	Observed forest cover value

Column	Description
deforestation	Observed deforestation value
naturalforestarea	Observed natural forest area value
industry	Observed industrial value
population	Observed population value

### 6.4.3. Structure, Patterns, and Local Prefixes

#### 6.4.3.1. Climate Data Cube

Table 1. Structure, Patterns, Prefixes

Item [prefix] {pattern} Description	Value for Project
Cube Name (Dataset name )	climate
BaseURI	http://sda-research.ml/
Data Cube {BaseURI}dc/{cube name}	http://sda-research.ml/dc/climate
<b>DataSet</b> [ds]  {BaseURI}dc/{cube name}/dataset  Includes the qb:DataSet, the qb:DataStructureDefinition and the qb:Observation.  The values of each dimension (specified as the value of the cube property in each dimension as part of qb:Observation) are also placed here because they are values that are a part of the cube. This would change if codelists are used. Slices [qb:Slice, qb:SliceKey] would also be included here, if used.	http://sda-research.ml/dc/climate/dataset

<b>Properties [prop]</b> {BaseURI}dc/{cube name}/prop/  Properties of the Data Cube. a) <b>qb:ComponentProperty</b> , <b>qb:DimensionProperty</b> , <b>qb:MeasureProperty</b> , <b>qb:AttributeProperty</b> , qb:CodedProperty b) <b>qb:component</b> defined under each Data	http://sda-research.ml/dc/climate/prop/
<b>Cube Component Specifications [dccs]</b> {BaseURI}dc/{cube name}/dccs/  Cube Component specifications. <b>qb:ComponentSpecification</b>	http://sda-research.ml/dc/climate/dccs/

**Table 2. URI's for Dimensions and Measures**

Component Pattern	Value for Project
dimension	1. <a href="http://sda-research.ml/dc/climate/prop/">http://sda-research.ml/dc/climate/prop/</a> city 2. http://sda-research.ml/dc/climate/prop/cityid 3. http://sda-research.ml/dc/climate/prop/year
measure	1. <a href="http://sda-research.ml/dc/climate/prop/">http://sda-research.ml/dc/climate/prop/</a> humidity 2. http://sda-research.ml/dc/climate/prop/rainfall 3. http://sda-research.ml/dc/climate/prop/temperature

#### 6.4.3.2. Industry Data Cube

**Table 1. Structure, Patterns, Prefixes**

Item [prefix] {pattern} Description	Value for Project
<b>Cube Name</b> (Dataset name )	Industry
<b>BaseURI</b>	http://sda-research.ml/
<b>Data Cube</b>	http://sda-research.ml/dc/industry

{BaseURI}dc/{cube name}	
<p><b>DataSet</b> [ds]</p> <p>{BaseURI}dc/{cube name}/dataset</p> <p>Includes the qb:DataSet, the qb:DataStructureDefinition and the qb:Observation.</p> <p>The values of each dimension (specified as the value of the cube property in each dimension as part of qb:Observation) are also placed here because they are values that are a part of the cube. This would change if codelists are used. Slices [qb:Slice, qb:SliceKey] would also be included here, if used.</p>	<a href="http://sda-research.ml/dc/industry/dataset">http://sda-research.ml/dc/industry/dataset</a>
<p><b>Properties</b> [prop]</p> <p>{BaseURI}dc/{cube name}/prop/</p> <p>Properties of the Data Cube.</p> <p>a) qb:ComponentProperty, qb:DimensionProperty, qb:MeasureProperty, qb:AttributeProperty, qb:CodedProperty b) qb:component defined under each Data</p>	<a href="http://sda-research.ml/dc/industry/prop/">http://sda-research.ml/dc/industry/prop/</a>
<p><b>Cube Component Specifications</b> [dccs]</p> <p>{BaseURI}dc/{cube name}/dccs/</p> <p>Cube Component specifications.</p> <p><b>qb:ComponentSpecification</b></p>	<a href="http://sda-research.ml/dc/industry/dccs/">http://sda-research.ml/dc/industry/dccs/</a>

**Table 2. URI's for Dimensions and Measures**

Component Pattern	Value for Project
dimension	1. <a href="http://sda-research.ml/dc/industry/prop/city">http://sda-research.ml/dc/industry/prop/city</a> 2. http://sda-research.ml/dc/industry/prop/cityid 3. http://sda-research.ml/dc/industry/prop/year
measure	1. <a href="http://sda-research.ml/dc/industry/prop/">http://sda-research.ml/dc/industry/prop/</a> humidity

#### 6.4.4. External vocabularies

Prefix	URI	Comment
qb	<a href="http://purl.org/linked-data/cube#">http://purl.org/linked-data/cube#</a>	Cube spec.
rdfs	<a href="http://www.w3.org/2000/01/rdf-schema#">http://www.w3.org/2000/01/rdf-schema#</a>	Labels, comments
xsd	<a href="http://www.w3.org/2001/XMLSchema#">http://www.w3.org/2001/XMLSchema#</a>	Data types
dcat	<a href="http://www.w3.org/ns/dcat#">http://www.w3.org/ns/dcat#</a>	Distribution information
dct	<a href="http://purl.org/dc/terms/">http://purl.org/dc/terms/</a>	Creator, issued date, title, description...
prov	<a href="http://www.w3.org/ns/prov#">http://www.w3.org/ns/prov#</a>	Provenance

#### 6.4.5. Class hierarchy

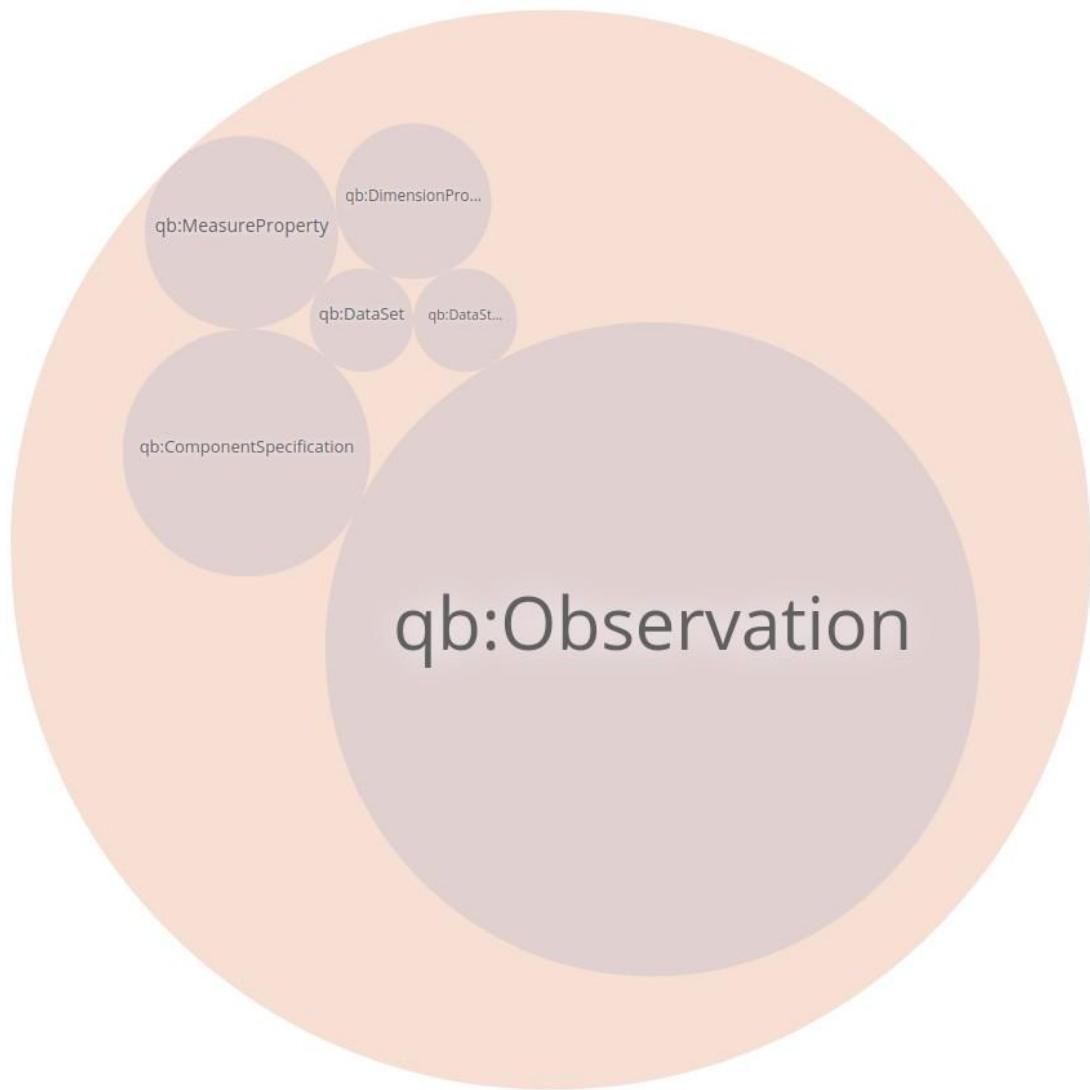


Figure 2. Class Hierarchy

#### 6.4.6. Class relationships

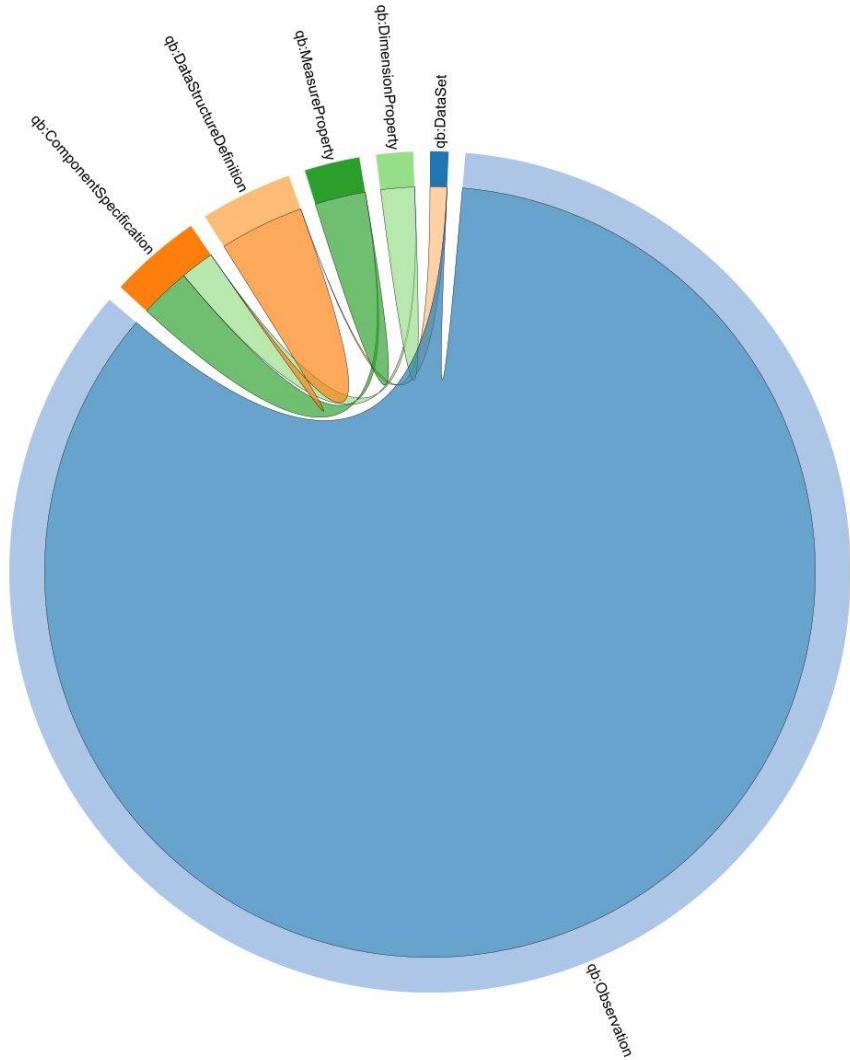


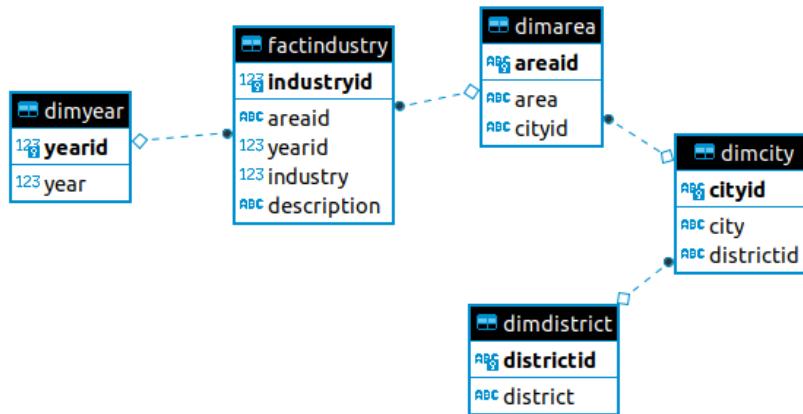
Figure 3. Class Relationships

## 6.5. ENTITY MAPPING

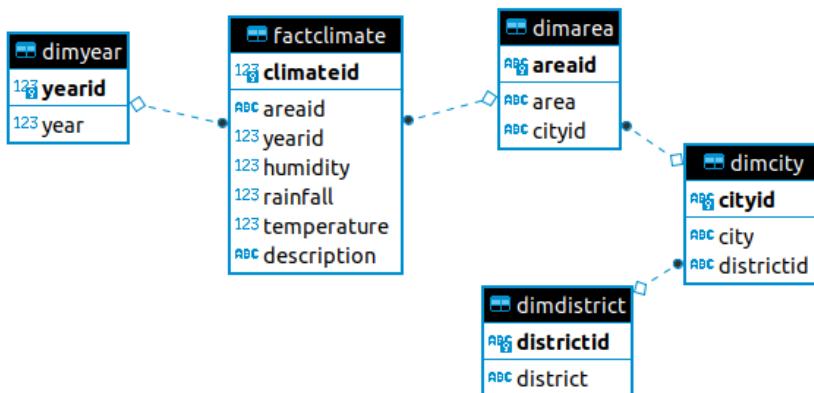
### 6.5.1. ENTITY MAPPING DIAGRAM

Please see the attached image for more details.

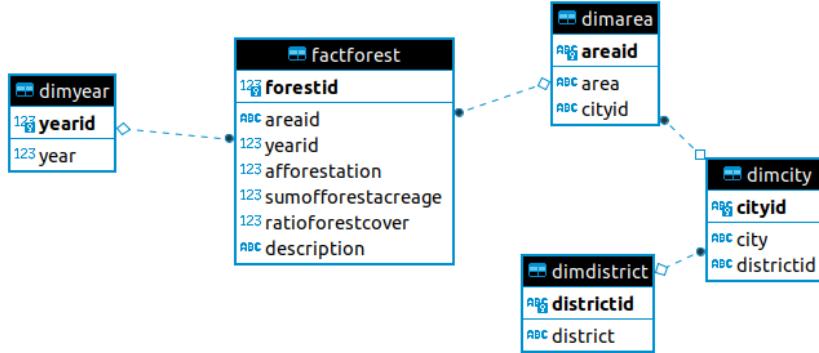
### 6.5.2. INDUSTRY



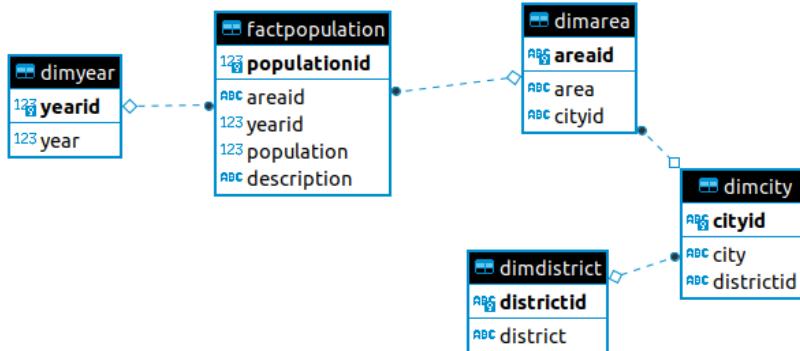
### 6.5.3. CLIMATE



#### 6.5.4. FOREST



#### 6.5.5. POPULATION



## 7. REFERENCES

- Technologies Stack Document
- dbdiagram.io : <https://dbdiagram.io>
- The RDF Data Cube: <https://www.w3.org/TR/eo-qb/#Datacube>
- Entity Relationship Mapping: [https://docs.oracle.com/cd/A97688\\_16/generic.903/a97677/ormap.htm](https://docs.oracle.com/cd/A97688_16/generic.903/a97677/ormap.htm)
- Entity Mapping Diagram For modeling ETL processes: [https://www.researchgate.net/profile/Ali\\_El-Bastawissy/publication/236030320\\_Entity\\_Mapping\\_Diagram/links/00463515e17cb192b4000000/Entity-Mapping-Diagram.pdf](https://www.researchgate.net/profile/Ali_El-Bastawissy/publication/236030320_Entity_Mapping_Diagram/links/00463515e17cb192b4000000/Entity-Mapping-Diagram.pdf)



## CAPSTONE PROJECT 2

CMU-SE-451 / CMU-IS-451 / CMU-CS-451

### TEST PLAN DOCUMENT

Version 2.1

Date: 1 - Mar - 2021

### EXPERT-DRIVEN SMART DASHBOARD APPLICATION

#### Submitted by

Vo Van Hoa  
Pham Van Tin  
Ky Huu Dong  
Tran Thi Thanh Kieu

#### Approved by

##### Capstone Project 2 - Mentor:

Name

Signature

Date

Binh, Thanh Nguyen \_\_\_\_\_  \_\_\_\_\_ 26 - May - 2021

PROJECT INFORMATION			
<b>Project Acronym</b>	EDSDA		
<b>Project Title</b>	Expert-Driven Smart Dashboard Application		
<b>Project Web URL</b>	<a href="https://sda-research.ml/">https://sda-research.ml/</a>		
<b>Start Date</b>	01 - Mar - 2021		
<b>End Date:</b>	02 - Jun - 2021		
<b>Lead Institution</b>	International School, Duy Tan University		
<b>Project Mentor</b>	Ph.D Binh, Thanh Nguyen		
<b>Scrum Master</b>	Hoa, Vo	hoavo.dng@gmail.com	0935.193.182
<b>Team Members</b>	Tin, Pham Van	tinphamvan123ail.co@gmm	0932.535.175
	Dong, Ky Huu	kyhuudong@gmail.com	0898.246.980
	Kieu, Tran Thi Thanh	thanhkieutran391@gmail.com	0358.583.251

DOCUMENT INFORMATION			
<b>Document Title</b>	Test Plan		
<b>Author(s)</b>	Team C2SE.06		
<b>Role</b>	[EDSDA] Test Plan v2.2		
<b>Date</b>	01 - Mar - 2021	Filename	[EDSDA] 007 Test Plan
<b>URL</b>	<a href="https://github.com/sdateamdtu2020/SDA-v2.0">https://github.com/sdateamdtu2020/SDA-v2.0</a>		
<b>Access</b>	Project and CMU Program		

## REVISION HISTORY

## TABLE OF CONTENTS

PROJECT INFORMATION	1
DOCUMENT INFORMATION	1
REVISION HISTORY	2
<b>TABLE OF CONTENTS</b>	<b>3</b>
<b>1. QUALITY OBJECTIVES</b>	<b>4</b>
<b>2. SCOPE OF TEST</b>	<b>4</b>
2.1. FUNCTIONS	4
2.2. USER INTERFACE	5
2.3. DATABASES	5
2.4. RDF DATA CUBES	6
<b>3. TEST STRATEGY</b>	<b>6</b>
<b>4. TEST CRITERIA</b>	<b>6</b>
<b>5. TEST MANAGEMENT</b>	<b>7</b>
5.1. TEAM	7
5.2. COMMUNICATION TOOLS	7
5.3. TEST MANAGEMENT TOOLS	7
<b>6. RISKS &amp; ASSUMPTIONS</b>	<b>8</b>
6.1. RISKS	8
6.2. ASSUMPTIONS	8
<b>7. TEST SCHEDULE</b>	<b>9</b>
<b>8. REFERENCES</b>	<b>11</b>

## 1. QUALITY OBJECTIVES

Planning for the project Smart Dashboard Application testing, to ensure that the testing is done according to plan, implement fully the necessary requirements, high work efficiency, and give the best product.

## 2. SCOPE OF TEST

### 2.1. FUNCTIONS

- **Drag & Drop**
  - Drag Widgets from the toolbar.
  - Drop Widgets to the mashup content dashboard.
- **Data Widgets**
  - Filter by year.
  - Filter by city.
  - Filter by multiple cities
  - Filter by multiple years.
- **Visualization Widgets**
  - Column Chart content.
  - Line Chart content.
  - Pie Chart content.
  - Map content.
- **Connect Data between Connected Widgets**
  - Fetch data from Data Widgets to Visualization Widget.
  - Fetch data from Data Widgets to Operator Widget and Visualization Widget.
- **Operators**
  - Merge data from multiple Data Widgets.
- **Connector**
  - Draw connector from a point of widget.
  - Draw connector that connects multiple widgets together.
- **Example**
  - Example content.

## 2.2. USER INTERFACE

- **Dashboard scene**
  - Navigation bar scene.
    - Menu Button
      - File
      - View
      - Help
    - File Button
      - Save
      - Clear
      - Open recent
    - View Button
      - Out to layout nodes
      - Fit view
      - Zoom 100%
      - Zoom in
      - Zoom out
      - .
    - Logo
  - **Average Humidity scene**
    - Visualization widget in tree view.
    - Filter by city.
    - Visualization data by Column chart, Line chart.
  - **Industry scene**
    - Visualization widget in tree view
    - Filter by city.
    - Filter by year.
    - Visualization data by Maps.

## 2.3. Databases

- **Staging Area Tables**
  - Extract data from CSV
  - Transform the data
  - Load data to the Staging Area Tables
    - Load data into Forest Source Data Staging
    - Load data into Population Source Data Staging
    - Load data into Industry Source Data Staging

- Load data into Climate Source Data Staging
- **Fact Tables**
  - Load data into Forest Source Data Staging to Fact Forest
  - Load data into Forest Source Data Staging to Fact Population
  - Load data into Forest Source Data Staging to Fact Industry
  - Load data into Forest Source Data Staging to Fact Climate
- **Dimension**
  - Dim Year
    - Queries available values years in Fact tables but not duplicate
    - Load the values into Dimension Year
  - Dim City
    - Queries available values City in Fact tables but not duplicate
    - Load the values into Dimension City

## 2.4. RDF Data Cubes

- **Dimension**
  - Dimension must have a label.
- **Measure**
  - Measure must have a label.
- **Dataset**
  - Dataset must have dimensions.
  - Dataset must have at least one measure.
- **Data Structure Definition**
  - Data Structure Definition must have dimensions.
  - Data Structure Definition must have at least one measure.
- **Observation**
  - Observation must have values.

## 3. TEST STRATEGY

We are using Black box testing due to fast lifecycle of project:

- Manual Test (Exploratory Testing).
- Acceptance Test.
- Functional Test (Function, UI).

## 4. TEST CRITERIA

- The Testing process finishes when 90% test cases pass status.
- Testing all test cases.
- The document will be delivered to the customer when finished each sprint.

## 5. TEST MANAGEMENT

### 5.1. TEAM

Full Name	Email	Phone number	Role
Hoa, Vo	hoavo.dng@gmail.com	0935.193.182	Scrum master
Tin, Pham Van	tinphamvan123@gmail.com	0932.535.175	Team member
Dong, Ky Huu	kyhuudong@gmail.com	0898.246.980	Team member
Kieu, Tran Thanh	thanhkieutran391@gmail.com	0358.583.251	Team member

### 5.2. COMMUNICATION TOOLS

- **Slack:** Report bugs, notify updates,...
- **Trello:** Tracking issues.
- **Discord:** discuss online.
- **Skype:** Contact, discuss with mentor Binh.

### 5.3. TEST MANAGEMENT TOOLS

Purpose	Tool	Vendor/In-house	Version
Excel Sheet to track Test Plan and Test Case	Google Sheet	Google	Latest

## 6. RISKS & ASSUMPTIONS

### 6.1. RISKS

Risk	Definition	Probability	Severity	Mitigation Strategy
Scope Risk	A high numbers of modules	H	H	All team members join to test system Testing in each plan
Scheduling Risk	Testing projects are not efficiently or completely	M	M	Move the not finish part of module to the next sprint
Time management	Most of the time is for development, not for testing.	M	H	Overtime
Operation Risk	Ineffective processing, system failures, or unanticipated circumstance define operational risk	M	M	Estimate more time to testing and other issues

Probability		Severity	
L	Rarely happens.	L	Low damaged
M	Sometime happened	M	Medium damaged
H	Usually happened	H	Serious damaged

### 6.2. ASSUMPTIONS

Assumption to be proven	Impact of Assumption being incorrect	Owners
Network Available	Dropout network, unstable network	Network Providers

## 7. TEST SCHEDULE

No	Task Name	Duration (Hours)	Start	Finish	Resources
1	<b>Test Sprint 1</b>	35	Mar 1, 2021	Mar 14, 2021	
1.1	Test Plan document	4			Kieu
1.2	Write Test Case document	31			Kieu
2	<b>Test Sprint 2</b>	75	Mar 15, 2020	Mar 28, 2021	
2.1	Write Test case for all features in Base UI	35			Kieu
2.2	Test the feature in Base UI	40			Kieu
3	<b>Test Sprint 3</b>	65	Mar 29, 2021	Apr 11, 2021	
3.1	Testing the data pipeline for raw data sources	15			Kieu
3.2	Write test cases for new UI	50			Kieu
4	<b>Test Sprint 4</b>	90	Apr 19, 2021	May 2, 2021	
4.1	Test RML process	10			Hoa
4.2	Test DW ETL process	10			Kieu
4.3	Test workflow for DW	10			Kieu
4.4	Test default Screen	5			Kieu
4.5	Test introduction screen	5			Kieu



4.6	Test about screen	5			Kieu
4.7	Test Toolbox screen	10			Kieu
4.8	Test Show Example screen	35			Kieu
<b>5</b>	<b>Test Sprint 5</b>	<b>100</b>	<b>May 3, 2021</b>	<b>May 16, 2021</b>	
5.1	Test RDF generator & ETL pipeline	5			Hoa
5.2	Test dim & measure detector	5			Kieu
5.3	Test automatic processing	5			Hoa
5.4	Test the import widget	5			Kieu
5.5	Test the logic of import function	10			Kieu
5.6	Test the Tree View of EDSDA data source on widget	5			Kieu
5.7	Test the data in Information component	20			Kieu
5.8	Test Toolbox UI	15			Kieu
5.9	Write test for visualization on Charts	5			Kieu
5.10	Continue Test the Show Example screen	25			Kieu
<b>6</b>	<b>Test Sprint 6</b>	<b>109</b>	<b>May 17, 2021</b>	<b>May 30, 2021</b>	
6.1	Test data of visualization on Charts	12			Kieu
6.2	Test UI of Charts	5			Kieu





6.3	Test the Tree View of data source on widget	5			Kieu
6.4	Test Connector and port of widgets	5			Kieu
6.5	Write Test for visualization on Maps & Table	10			Kieu
6.6	Write test for Import function	5			Kieu
6.7	Test function of import	7			Kieu
6.8	Test the UI on domain: sda-research.surge.sh	40			Kieu
6.9	Import data from UI	2			Kieu
6.10	Test RDF format	2			Kieu
6.11	Test workflows, data pipelines	4			Hoa
6.12	Test consistancy of RDF and DW	4			Kieu
6.13	Public User Acceptance Test	8			All member

## 8. REFERENCES

- Trello issues tracking: <https://trello.com/b/4TXGKBgu/edsda-issues>
- Sprint backlog & burndown chart:  
[https://docs.google.com/spreadsheets/d/1a1GqHsPL9oKlyFvqAPi3B2CEYL89deCXzCUGTDB\\_kcw/edit#gid=283855912](https://docs.google.com/spreadsheets/d/1a1GqHsPL9oKlyFvqAPi3B2CEYL89deCXzCUGTDB_kcw/edit#gid=283855912)



## CAPSTONE PROJECT 2

CMU-SE-451 / CMU-IS-451 / CMU-CS-451

### TEST CASE DOCUMENT

Version 2.0

Date: 1 - Mar - 2021

### EXPERT-DRIVEN SMART DASHBOARD APPLICATION

#### Submitted by

Vo Van Hoa  
Pham Van Tin  
Ky Huu Dong  
Tran Thi Thanh Kieu

#### Approved by

##### Capstone Project 2 - Mentor:

Name

Signature

Date

Binh, Thanh Nguyen \_\_\_\_\_  \_\_\_\_\_ 26 - May - 2020

PROJECT INFORMATION			
<b>Project Acronym</b>	EDSDA		
<b>Project Title</b>	Expert-Driven Smart Dashboard Application		
<b>Project Web URL</b>	<a href="https://sda-research.ml/">https://sda-research.ml/</a>		
<b>Start Date</b>	01 - Mar - 2021		
<b>End Date:</b>	02 - Jun - 2021		
<b>Lead Institution</b>	International School, Duy Tan University		
<b>Project Mentor</b>	Ph.D Binh, Thanh Nguyen		
<b>Scrum Master</b>	Hoa, Vo	hoavo.dng@gmail.com	0935.193.182
<b>Team Members</b>	Tin, Pham Van	tinphamvan123@gmail.com	0932.535.175
	Dong, Ky Huu	kyhuudong@gmail.com	0898.246.980
	Kieu, Tran Thi Thanh	thanhkieutran391@gmail.com	0358.583.251

DOCUMENT INFORMATION			
<b>Document Title</b>	Test Case		
<b>Author(s)</b>	Team C2SE.06		
<b>Role</b>	[EDSDA] Test Case v2.1		
<b>Date</b>	01 - Mar - 2021	Filename	[EDSDA] 008 Test Case
<b>URL</b>	<a href="https://github.com/sdateamdtu2020/SDA-v2.0">https://github.com/sdateamdtu2020/SDA-v2.0</a>		
<b>Access</b>	Project and CMU Program		

# REVISION HISTORY

Test Cases: NavBar									
ID	Items	Sub-items	Steps to Execute	Expected output	Test Data / Parameters	PC		Bug	
Front-end						IE	FF 86	Chrome 89	
TC-Nav_01	Navbar		1. Open Datablocks 2. Go to "Navbar" section 3. Check this section to make sure it match the design.	3. This section matched the design		Pass	Pass	Pass	
TC-Nav_02	Navbar		1. Open Dashboard 2. Go to "Navbar" section 3. Check backgroundColor	3. The background color is #1a192b		Pass	Pass	Pass	
TC-Nav_03	Navbar		1. Open Dashboard 2. Go to "Navbar" section 3. Check borderbottom	3.the border is solid has height is 1px and color is #333154		Pass	Pass	Pass	
TC-Nav_04	Navbar		1. Open Dashboard 2. Go to "Navbar" section 3. Check the align items	3. The align items is center		Pass	Pass	Pass	
TC-Nav_05	Logo		1. Open Dashboard 2. Go to "Navbar" section 3. Check the Logo	3. This section matched the design		Pass	Pass	Pass	
TC-Nav_06	Button		1. Open Dashboard 2. Go to "Navbar" section 3. Check the button background color	3. The background color is #1a192b		Pass	Pass	Pass	
TC-Nav_07	Button		1. Open Dashboard 2. Go to "Navbar" section 3. Check the button font family	3. The font family is IBM Plex Sans		Pass	Pass	Pass	
TC-Nav_08	Button		1. Open Dashboard 2. Go to "Navbar" section 3. Check the button font size	3. The font size is 11px		Pass	Pass	Pass	

Test Cases: NavBar								
ID	Items	Sub-items	Steps to Execute	Expected output	Test Data / Parameters	PC		Bug
TC-Nav_09	Button		1. Open Dashboard 2. Go to "Navbar" section 3. Check the text color	3. The color is white		Pass	Pass	
TC-Nav_10	Button		1. Open Dashboard 2. Go to "Navbar" section 3. Check the text	3. The text is uppercase		Pass	Pass	
TC-Nav_11	Button		1. Open Dashboard 2. Go to "Navbar" section 3. Hover in button 4. Check the text color	4. The color is #ff0072		Pass	Pass	
TC-Nav_12	Button		1. Open Dashboard 2. Go to "Navbar" section 3. Hover in button 4. Check the cursor	4. The cursor is point		Pass	Pass	
TC-Nav_13	Button	Menu	1. Open Dashboard 2. Go to "Navbar" section 3. Click "File, View and Help" button 4. Check the list menu of file	4. Show the 3 buttons are Save, Clear and Open Recent		Pass	Pass	
TC-Nav_14	Button	Menu	1. Open Dashboard 2. Go to "Navbar" section 3. Click "File, View and Help" button 4. Check the background color of list menu	4. The background color is #222138		Pass	Pass	
TC-Nav_15	Button	Menu	1. Open Dashboard 2. Go to "Navbar" section 3. Click "File, View and Help" button 4. Check the border of list menu	4. The border is solid has height is 1px and color is #333154		Pass	Pass	
			1. Open Dashboard 2. Go to "Navbar" section 3. Click "File, View and Help" button	4. 20px from the top				

Test Cases: NavBar								
ID	Items	Sub-items	Steps to Execute	Expected output	Test Data / Parameters	PC		Bug
TC-Nav_16	Button	Menu	4. Check the position of list menu			Pass	Pass	Pass
TC-Nav_17	Button	Items	1. Open Dashboard 2. Go to "Navbar" section 3. Click "File, View and Help" button 4. Check items	4. Padding the top, bottom is 4px and the left, right is 16px		Pass	Pass	
TC-Nav_18	Button		1. Open Dashboard 2. Go to "Navbar" section 3. Click "File, View and Help" button 4. Check text items					
TC-Nav_19	Button		1. Open Dashboard 2. Go to "Navbar" section 3. Click "File, View and Help" button 4. Hover in items 5. Check background color items		4. the text transform is capitalize 5. The background color is #757575	Pass	Pass	Pass
TC-Nav_20	Button		1. Open Dashboard 2. Go to "Navbar" section 3. Click "File, View and Help" button 4. Hover in items 5. Check the curson					
TC-Nav_21	Button	Save Item	1. Open Dashboard 2. Go to "Navbar" section 3. Click "File" button 4. Click on "Save" items	4. Show the message on the Logs of Info section . The message is: 'Successfully saved project locally.'		Pass	Pass	Pass
			1. Open Dashboard 2. Go to "Navbar" section					

Test Cases: NavBar									
ID	Items	Sub-items	Steps to Execute	Expected output	Test Data / Parameters	PC			Bug
TC-Nav_22	Button	Clear Item	3. Click "File" button 4. Click on "Clear" items	display. The message is: 'Are you sure?. And Menu still show, Clear items appear the background color when hover		Pass	Pass	Pass	
TC-Nav_23	Button	Clear Item	1. Open Dashboard 2. Go to "Navbar" section 3. Click "File" button 4. Click on "Clear" items 5. Click "OK"	5. Clear all widgets		Pass	Pass	Pass	
TC-Nav_24	Button	Clear Item	1. Open Dashboard 2. Go to "Navbar" section 3. Click "File" button 4. Click on "Clear" items 5. Click "Cancel"	5. Close the Alert and close the Menu		Pass	Pass	Pass	
TC-Nav_25	Button	Zoom 100% Item	1. Open Dashboard 2. Go to "Navbar" section 3. Click "View" button 4. Click on "Zoom 100%" items	4. Zoom Main section to 100%		Pass	Pass	Pass	
TC-Nav_26	Button	Zoom In Item	1. Open Dashboard 2. Go to "Navbar" section 3. Click "View" button 4. Click on "Zoom In" items	4. Zoom In Main section to 12.5%		Pass	Pass	Pass	
TC-Nav_27	Button	Zoom Out Item	1. Open Dashboard 2. Go to "Navbar" section 3. Click "View" button 4. Click on "Zoom Out" items	4. Zoom out Main section 12.5%		Pass	Pass	Pass	
			1. Open Dashboard 2. Go to "Navbar" section	4. Show Introduction					

Test Cases: NavBar										
ID	Items	Sub-items	Steps to Execute	Expected output	Test Data / Parameters	PC			Bug	
TC-Nav_28	Button	Items	3. Click "Help" button 4. Click on "Help" items	4. Show introduction Flow, About, Feedback Item on display		Pass	Pass	Pass		
TC-Nav_29	Button	About Item	1. Open Dashboard 2. Go to "Navbar" section 3. Click "Help" button 4. Click on "About" items	4. Show the infomation		Pass	Pass	Pass		

Test Cases: Output&Logs				Expected output	Test Data / Parameters	PC			Bug
ID	Items	Sub-items	Steps to Execute			IE	FF 86	Chrome 89	
Front-end									
TC-OAL_01	Output and logs		1. Open Datablocks 2. Go to "Output and Logs" section 3. Check this section to make sure it match the design.	3. This section matched the design		Pass	Pass	Pass	
TC-OAL_02	Output and logs		1. Open Datablocks 2. Go to "Output and Logs" section 3. Check the background color	3. The background color is #1a192b		Pass	Pass	Pass	
TC-OAL_03	Output and logs		1. Open Dashboard 2. Go to "Output and Logs" section 3. Check border bottom	3. The border is solid has height is 1px and color is #333154		Pass	Pass	Pass	
TC-OAL_04	Output		1. Open Dashboard 2. Go to "Output" section 3. Check Output section	3. Width is 75%		Pass	Pass	Pass	
TC-OAL_05	Output	Header	1. Open Dashboard 2. Go to "Output" section 3. Check Header	3. Height is 30px		Pass	Pass	Pass	
TC-OAL_06	Output	Header	1. Open Dashboard 2. Go to "Output" section 3. Check the title	3. Align is center		Pass	Pass	Pass	
TC-OAL_07	Output	Header	1. Open Dashboard 2. Go to "Output" section 3. Check the text	3. Font size is 11px and bold, text is uppercase		Pass	Pass	Pass	
TC-OAL_08	Logs		1. Open Dashboard 2. Go to "Logs" section 3. Check Logs section	3. Width is 25%		Pass	Pass	Pass	
			1. Open Dashboard 2. Go to "Logs" section						

Test Cases: Output&Logs				Expected output	Test Data / Parameters	PC			Bug
ID	Items	Sub-items	Steps to Execute						
TC-OAL_09	Logs	Header	3.Check Header	3. Height is 30px		Pass	Pass	Pass	
TC-OAL_10	Logs	Header	1. Open Dashboard 2. Go to "Logs" section 3.Check the title	3. Align is center		Pass	Pass	Pass	
TC-OAL_11	Logs	Header	1. Open Dashboard 2. Go to "Logs" section 3.Check the text	3. Font size is 11px and bold, text is uppercase		Pass	Pass	Pass	
TC-OAL_12	Logs	Input form	1. Open Dashboard 2. Go to "Logs" section 3. Check Input form	"help" to see the commands"		Pass	Pass	Pass	
TC-OAL_13	Logs	Input form	1. Open Dashboard 2. Go to "Logs" section 3. Check text	3. Font size is 12px, color is inherit		Pass	Pass	Pass	
TC-OAL_14	Logs	Input form	1. Open Dashboard 2. Go to "Logs" section 3. Input to form	3. Display console.log on Logs main		Pass	Pass	Pass	

<b>Test Cases: Main</b>				<b>Expected output</b>	<b>Test Data / Parameters</b>	<b>PC</b>			<b>Bug</b>
<b>ID</b>	<b>Items</b>	<b>Sub-items</b>	<b>Steps to Execute</b>			<b>IE</b>	<b>FF 86</b>	<b>Chro me 89</b>	
<b>Front-end</b>									
TC-Main_01	Main	Message	1. Open Datablocks 2. Go to "Main" section 3. Check message	3. Message should be displayed with logo, description about project, closed icon and 2 buttons , and background color made dim ,boder radius is 20px .		Pass	Pass	Pass	
TC-Main_02	Main	Message	1. Open Datablocks 2. Go to "Main" section 3. Check message background color	3. Background color is #1A192B		Pass	Pass	Pass	
TC-Main_03	Main	Message	1. Open Datablocks 2. Go to "Main" section 3. Check Logo	3. Logo on top and center		Pass	Pass	Pass	
TC-Main_04	Main	Message	1. Open Datablocks 2. Go to "Main" section 3. Check text	3. Font size is 16px , text align is center and the color is #C5CBD2		Pass	Pass	Pass	
TC-Main_05	Main	Message	1. Open Datablocks 2. Go to "Main" section 3. Check Add Block button	3. The background color is #FF0072, boder radius : 20px , color of Text #FFFFFF		Pass	Pass	Pass	
TC-Main_06	Main	Message	1. Open Datablocks 2. Go to "Main" section 3. Click Add Block 4. Check button	4 . Show toolbox		Pass	Pass	Pass	
TC-Main_07	Main	Message	1. Open Datablocks 2. Go to "Main" section 3. Check Show Example	3. Border radius 20px , border color and text color is #FFFFFF, background color is #1A192B		Pass	Pass	Pass	
			1. Open Datablocks						

Test Cases: Main				Expected output	Test Data / Parameters	PC			Bug
ID	Items	Sub-items	Steps to Execute						
TC-Main_08	Main	Message	2. Go to "Main" section	4 . Show example on main		Pass	Pass	Pass	
			3. Click Show Example						
			4. Check button						
TC-Main_09	Main		1. Open Datablocks	4. Background color is #1A192B, Mini map , group zoom button , block button		Pass	Pass	Pass	
			2. Go to "Main" section						
			3. Check Main section						
TC-Main_10	Main	Block button	1. Open Datablocks	3. Background color is #1A192B, border radius 20px , border color is #524F88		Pass	Pass	Pass	
			2. Go to "Main" section						
			3. Check Block button						
TC-Main_11	Main	Block button	1. Open Datablocks	3. Background color #524F88 , cursor is pointer		Pass	Pass	Pass	
			2. Go to "Main" section						
			3. Hover over on Block button						
TC-Main_12	Main	Block button	1. Open Datablocks	3. Show toolbox		Pass	Pass	Pass	
			2. Go to "Main" section						
			3. Click Block button						
TC-Main_13	Main	Toolbox	1. Open Datablocks	4. Background color #1A192B, box shadow , border radius 20px		Pass	Pass	Pass	
			2. Go to "Main" section						
			3. Click Block button						
			4. Check Toolbox						
TC-Main_14	Main	Toolbox	1. Open Datablocks	4. Search bar, menu list , board, scroll,close icons ,Background color #1A192B, box shadow , border radius 20px ,		Pass	Pass	Pass	
			2. Go to "Main" section						
			3. Click Block button						
			4. Check Toolbox						
TC-Main_15	Main	Toolbox	1. Open Datablocks	4. Show search, Input , data cubes, operator , visualization in the left message		Pass	Pass	Pass	
			2. Go to "Main" section						
			3. Click Block button						
			4. Check Menu list						
TC-Main_16	Main	Toolbox	1. Open Datablocks	4. The text and icon has color is		Pass	Pass	Pass	
			2. Go to "Main" section						

Test Cases: Main				Expected output	Test Data / Parameters	PC			Bug
ID	Items	Sub-items	Steps to Execute			Pass	Pass	Pass	
TC-Main_16	Main	Toolbox	3. Click Block button	#FF0072		Pass	Pass	Pass	
			4. Click items in menu list						
TC-Main_17	Main	Toolbox	1. Open Datablocks	4. The background color is #524F88;		Pass	Pass	Pass	
			2. Go to "Main" section						
			3. Click Block button						
			4. Hover items in menu list						
TC-Main_18	Main	Toolbox	1. Open Datablocks	4. Show item on board		Pass	Pass	Pass	
			2. Go to "Main" section						
			3. Click Block button						
			4. Input item in Search bar						
TC-Main_19	Main	Toolbox	1. Open Datablocks	4. input section , Availables data cube section , operaotors section , visualizations section		Pass	Pass	Pass	
			2. Go to "Main" section						
			3. Click Block button						
			4. Check the board						
TC-Main_20	Main	Toolbox	1. Open Datablocks	4. Show File cell to handle file, input: csv, json, xml and output is dataset		Pass	Pass	Pass	
			2. Go to "Main" section						
			3. Click Block button						
			4. Check the input section						
TC-Main_21	Main	Toolbox	1. Open Datablocks	4. About Climate, Atmosphere, Population, Industry, Forest and descript values from 2012 to 2019		Pass	Pass	Pass	
			2. Go to "Main" section						
			3. Click Block button						
			4. Check the available data cubes section						
TC-Main_22	Main	Toolbox	1. Open Datablocks	4. About Statistic merge, Sort, Filter and descript function		Pass	Pass	Pass	
			2. Go to "Main" section						
			3. Click Block button						
			4. Check the operators section						
TC-Main_23	Main	Toolbox	1. Open Datablocks	4. About Table, Column chart, Line Chart, Multi Y-Axis Chart, Pie chart, Maps and descript		Pass	Pass	Pass	
			2. Go to "Main" section						
			3. Click Block button						

Test Cases: Main				Expected output	Test Data / Parameters	PC		Bug
ID	Items	Sub-items	Steps to Execute					
			4. Check the visualizations section	function				
TC-Main_24	Main	Toolbox	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Hover over items in board	4. The cursor is pointer		Pass	Pass	Pass
TC-Main_25	Main	Toolbox	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click Close icon	4. Close Toolbox		Pass	Pass	Pass
TC-Main_26	Main	Toolbox	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click outside Toolbox	4. Close Toolbox		Pass	Pass	Pass
TC-Main_27	Main	Widgets	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click widget in the data cubes on board	4. Show widget in main: widget name, close icon, dimension and years , areas	Widgets: Climate, Atmosphere, Population, Industry,Forest	Pass	Pass	Pass
TC-Main_28	Main	Widgets	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click widget in the board 5. Click close icon	5. Close widget	Widget	Pass	Pass	Pass
TC-Main_29	Main	Widgets	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click widget in the board 5. Check the widget head	5. Background color is #222138, the text font size is 12px , color is #FFFFFF	Widget	Pass	Pass	Pass
			1. Open Datablocks	Background color is #222138,				

Test Cases: Main				Expected output	Test Data / Parameters	PC			Bug
ID	Items	Sub-items	Steps to Execute						
TC-Main_30	Main	Widgets	2. Go to "Main" section	the text font size is 12px , color is #FFFFFF	Widget	Pass	Pass	Pass	
			3. Click Block button						
			4. Click widget in the board						
			5. Check the widget body						
TC-Main_31	Main	Widgets	1. Open Datablocks	5. Background color is #524F88	Widget	Pass	Pass	Pass	
			2. Go to "Main" section						
			3. Click Block button						
			4. Click widget in the board						
			5. Hover over the dimension in widget body						
TC-Main_32	Main	Operators	1. Open Datablocks	4. Show Statistic visualization : name , close icon, run button	Statistic visualization	Pass	Pass	Pass	
			2. Go to "Main" section						
			3. Click Block button						
			4. Click Statistic visualization in the board						
TC-Main_33	Main	Operators	1. Open Datablocks	5. Close Statistic visualization	Statistic visualization	Pass	Pass	Pass	
			2. Go to "Main" section						
			3. Click Block button						
			4. Click Statistic visualization in the board						
			5. Click close icon						
TC-Main_34	Main	Operators	1. Open Datablocks	5. Background color is #222138, the text font size is 12px , color is #FFFFFF	Statistic visualization	Pass	Pass	Pass	
			2. Go to "Main" section						
			3. Click Block button						
			4. Click Statistic visualization in the board						
			5. Check the head						
TC-Main_35	Main	Operators	1. Open Datablocks	5. Background color is #222138	Statistic visualization	Pass	Pass	Pass	
			2. Go to "Main" section						
			3. Click Block button						

Test Cases: Main				Expected output	Test Data / Parameters	PC			Bug
ID	Items	Sub-items	Steps to Execute			Pass	Pass	Pass	
TC-Main_35	Main	Operators	4. Click Statistic visualization in the board 5. Check the body						
TC-Main_36	Main	Operators	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click Statistic visualization in the board 5. Check the Run button	5. Background color is #FF0072; border-radius is 20px , the text : font size 14px, color is #FFFFFF	Statistic visualization	Pass	Pass	Pass	
TC-Main_37	Main	Operators	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click Statistic visualization in the board 5. Check the Run button	5. Background color is #FF0072; border-radius is 20px , the text : font size 14px, color is #FFFFFF	Statistic visualization	Pass	Pass	Pass	
TC-Main_38	Main	Visualizations - Charts	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click chart in the board	4. Show colum chart : name , close icon, run button, chart	Colum chart, Line chart	Pass	Pass	Pass	
TC-Main_39	Main	Visualizations - Charts	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click chart in the board 5. Click close icon	5. Close chart	Colum chart, Line chart	Pass	Pass	Pass	
TC-Main_40	Main	Visualizations - Charts	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click chart in the board 5. Check the head	5. Background color is #222138, the text font size is 12px , color is #FFFFFF	Colum chart, Line chart	Pass	Pass	Pass	
			1. Open Datablocks	5. Background color is #222138					

Test Cases: Main				Expected output	Test Data / Parameters	PC			Bug
ID	Items	Sub-items	Steps to Execute						
TC-Main_41	Main	Visualizations - Charts	2. Go to "Main" section 3. Click Block button 4. Click chart in the board 5. Check the body	5. Background color is #222138	Colum chart, Line chart	Pass	Pass	Pass	
TC-Main_42	Main	Visualizations - Charts	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click chart in the board 5. Check Run button	5. Background color is #FF0072, border-radius is 20px , the text : font size 14px, color is #FFFFFF	Colum chart, Line chart	Pass	Pass	Pass	
TC-Main_43	Main	Visualizations - Charts	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click chart in the board 5. Check the Column chart	5. About name , year, gloss, percent unit	Colum chart, Line chart	Pass	Pass	Pass	
TC-Main_44	Main	Visualizations - Maps	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click Maps in the board	4. Show : name , close icon, run button, map	Maps	Pass	Pass	Pass	
TC-Main_45	Main	Visualizations - Maps	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click Maps in the board 5. Click close icon	4. Close Maps	Maps	Pass	Pass	Pass	
TC-Main_46	Main	Visualizations - Maps	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click Maps in the board 5. Click the head	5. Background color is #222138, the text font size is 12px , color is #FFFFFF	Maps	Pass	Pass	Pass	
		Visualizations	1. Open Datablocks 2. Go to "Main" section	5. Background color is #222138					

Test Cases: Main				Expected output	Test Data / Parameters	PC			Bug	
ID	Items	Sub-items	Steps to Execute							
TC-Main_47	Main	Visualizations - Maps	3. Click Block button		Maps	Pass	Pass	Pass		
			4. Click Maps in the board							
			5. Click the body							
TC-Main_48	Main	Visualizations - Maps	1. Open Datablocks	5. Background color is #FF0072, border-radius is 20px , the text : font size 14px, color is #FFFFFF	Maps	Pass	Pass	Pass		
			2. Go to "Main" section							
TC-Main_49	Main		3. Click Block button			Pass	Pass	Pass		
			4. Click Maps in the board							
TC-Main_50	Main	Widgets	5. Click map		Maps	Pass	Pass	Pass		
			1. Open Datablocks	5. Show name , value , close icon						
TC-Main_51	Main		2. Go to "Main" section	Widgets: Climate, Atmosphere, Population, Industry,Forest	Pass	Pass	Pass			
			3. Click Block button							
TC-Main_52	Main	Widgets	4. Click widget in the data cubes on board	Widgets: Climate, Atmosphere, Population, Industry,Forest	Pass	Pass	Pass			
			5. Check Year in main							
			6. Click close icon							
			1. Open Datablocks	5. Background color is #222138, the text font size is 12px . color	Widgets: Climate, Atmosphere. Pooulation.	Pass	Pass	Pass		
			2. Go to "Main" section							
			3. Click Block button							

Test Cases: Main				Expected output	Test Data / Parameters	PC			Bug
ID	Items	Sub-items	Steps to Execute			Pass	Pass	Pass	
TC-Main_52	Main	Widgets	4. Click widget in the data cubes on board 5. Check the head of Year	the text font size is 12px , color is #FFFFFF	Atmosphere, Population, Industry, Forest	Pass	Pass	Pass	
TC-Main_53	Main	Widgets	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click widget in the data cubes on board 5. Check the body of Year	5. Background color is #222138, the value from 2012 to 2019	Widgets: Climate, Atmosphere, Population, Industry, Forest	Pass	Pass	Pass	
TC-Main_54	Main	Widgets	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click widget in the data cubes on board 5. Hover over the body of Year	5. Background color is #524F88, cursor is pointer	Widgets: Climate, Atmosphere, Population, Industry, Forest	Pass	Pass	Pass	
TC-Main_55	Main	Widgets	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click widget in the data cubes on board 5. Check the Area in Main	5. Show name , value , close icon	Widgets: Climate, Atmosphere, Population, Industry, Forest	Pass	Pass	Pass	
TC-Main_56	Main	Widgets	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click widget in the data cubes on board 5. Check Area in main 6. Click close icon	6. Close Area	Widgets: Climate, Atmosphere, Population, Industry, Forest	Pass	Pass	Pass	
TC-Main_57	Main	Widgets	1. Open Datablocks 2. Go to "Main" section 3. Click Block button	5. Background color is #222138, the text font size is 12px . color	Widgets: Climate, Atmosphere. Poulation.	Pass	Pass	Pass	

Test Cases: Main				Expected output	Test Data / Parameters	PC			Bug
ID	Items	Sub-items	Steps to Execute			Pass	Pass	Pass	
TC-Main_57	Main	Widgets	4. Click widget in the data cubes on board 5. Check the head of Area	the text font size is 12px, color is #FFFFFF	Atmosphere, Population, Industry,Forest	Pass	Pass	Pass	
TC-Main_58	Main	Widgets	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click widget in the data cubes on board 5. Check the body of Area	5. Background color is #222138	Widgets: Climate, Atmosphere, Population, Industry,Forest	Pass	Pass	Pass	
TC-Main_59	Main	Widgets	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click widget in the data cubes on board 5. Check the value in Area	5. The value	Widgets: Climate, Atmosphere, Population, Industry,Forest	Pass	Pass	Pass	
TC-Main_60	Main	Process	1. Open Datablocks 2. Go to "Main" section 3. Click Block button 4. Click widget in the data cubes on board 5. Click Statistic visualization in Operators on board 6. Click Chart in visualizations on board 7. Click Maps in visualizations on board 8. Choose dimension in widget 9. Choose year 10. Choose area 11. Drag a connection from an output of a block (right side) to an input of another block (left side).	12. The results are displayed in the map, the chart in the main and the table in the output	Widgets: Climate, Atmosphere, Population, Industry,Forest, Year, Area; Statistic visualization, Chart ,Maps	Pass	Pass	Pass	

<b>Test Cases: Main</b>				<b>Expected output</b>	<b>Test Data / Parameters</b>	<b>PC</b>		<b>Bug</b>
<b>ID</b>	<b>Items</b>	<b>Sub-items</b>	<b>Steps to Execute</b>					
			12. Click Run					
TC-Main_61	Main	Process	1. Open Datablocks	13. Output data on screen	Widgets: Climate, Atmosphere, Population, Industry,Forest, Year, Area; Statistic visualization, Chart ,Maps	Pass	Pass	Pass
			2. Go to "Main" section					
			3. Click Block button					
			4. Click widget in the data cubes on board					
			5. Click Statistic visualization in Operators on board					
			6. Click Chart in visualizations on board					
			7. Click Maps in visualizations on board					
			8. Choose dimension in widget					
			9. Choose year					
			10. Choose area					
			11. Drag a connection from an output of a block (right side) to an input of another block (left side).					
			12. Click Run					
			13. Click Export in Output					
TC-Main_62	Main	Connector	1. Open Datablocks	13. Connect between 2 node	Widgets: Climate, Atmosphere, Population, Industry,Forest, Year, Area; Statistic visualization, Chart ,Maps	Pass	Pass	Pass
			2. Go to "Main" section					
			3. Click Block button					
			4. Click widget in the data cubes on board					
			5. Click Statistic visualization in Operators on board					
			6. Click Chart in visualizations on board					
			7. Click Maps in visualizations on board					
			8. Choose dimension in widget					

<b>Test Cases: Main</b>				<b>Expected output</b>	<b>Test Data / Parameters</b>	<b>PC</b>		<b>Bug</b>
<b>ID</b>	<b>Items</b>	<b>Sub-items</b>	<b>Steps to Execute</b>					
			9. Choose year 10. Choose area 11. Drag a connection from an output of a block (right side) to an input of another block (left side). 12. Click Run 13. Check connector					
TC-Main_63	Main	Recommend	1. Open Datablocks 2. Go to "Main" section 3. Check recommended in main section	3. Show suggested steps		Pass	Pass	Pass



## CAPSTONE PROJECT 2

CMU-SE-451 / CMU-IS-451 / CMU-CS-451

### SPRINT BACKLOG & BURN DOWN CHART DOCUMENT

Version 2.0

Date: 1 - Mar - 2021

### EXPERT-DRIVEN SMART DASHBOARD APPLICATION

#### Submitted by

Vo Van Hoa  
Pham Van Tin  
Ky Huu Dong  
Tran Thi Thanh Kieu

#### Approved by

##### Capstone Project 2 - Mentor:

Name

Signature

Date

Binh, Thanh Nguyen \_\_\_\_\_  \_\_\_\_\_ 26 - May - 2020

PROJECT INFORMATION			
<b>Project Acronym</b>	EDSDA		
<b>Project Title</b>	Expert-Driven Smart Dashboard Application		
<b>Project Web URL</b>	<a href="https://sda-research.ml/">https://sda-research.ml/</a>		
<b>Start Date</b>	01 - Mar - 2021		
<b>End Date:</b>	02 - Jun - 2021		
<b>Lead Institution</b>	International School, Duy Tan University		
<b>Project Mentor</b>	Ph.D Binh, Thanh Nguyen		
<b>Scrum Master</b>	Hoa, Vo	hoavo.dng@gmail.com	0935.193.182
<b>Team Members</b>	Tin, Pham Van	tinphamvan123@gmail.com	0932.535.175
	Dong, Ky Huu	kyhuudong@gmail.com	0898.246.980
	Kieu, Tran Thi Thanh	thanhkieutran391@gmail.com	0358.583.251

DOCUMENT INFORMATION			
<b>Document Title</b>	Sprint Backlog & Burn Down Chart		
<b>Author(s)</b>	Team C2SE.06		
<b>Role</b>	[EDSDA] Sprint Backlog & Burn Down Chart v2.1		
<b>Date</b>	01 - Mar - 2021	Filename	[EDSDA] 009 Sprint Backlog & Burn Down Chart
<b>URL</b>	<a href="https://github.com/sdateamdtu2020/SDA-v2.0">https://github.com/sdateamdtu2020/SDA-v2.0</a>		
<b>Access</b>	Project and CMU Program		

## REVISION HISTORY

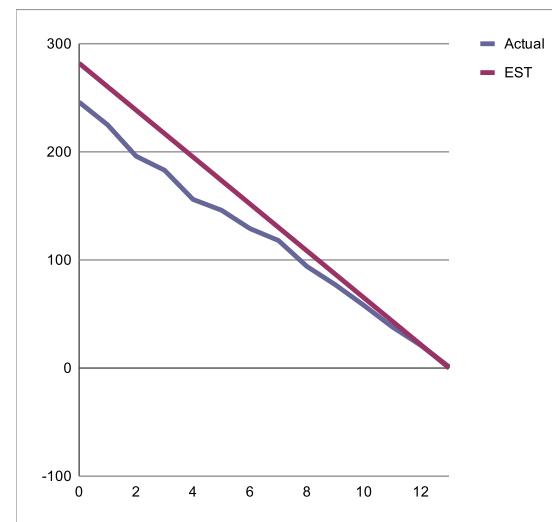
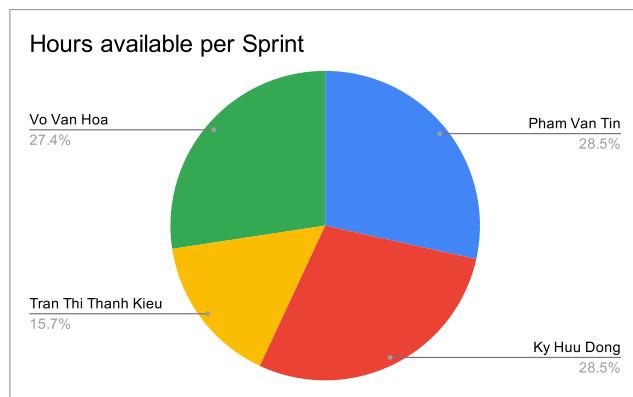
Version	Person(s)	Date	Description	Approval
Draft	Hoa, Vo	01 - Mar - 2021	Initiate document	Draft
2.0	All members	14 - Mar- 2021	Finish content of the document	2.0
2.1	All members	25 - May - 2021	Update content, fix typo	2.1

Sprint 1																																																				
C2SE-06 Team																																																				
#	Hours available per Sprint	Member Name																																																		
1	80	Pham Van Tin		Start date		Mar 1, 2021																																														
2	80	Ky Huu Dong		End date		Mar 14, 2021																																														
3	44	Tran Thi Thanh Kieu																																																		
4	77	Vo Van Hoa																																																		
No.	BACKLOG ID	Task Description			Responsible	Status	Initial Est.	Comp.	Day 14 01/03/2022	Day 13 02/03/2022	Day 12 03/03/2022	Day 11 04/03/2022	Day 10 05/03/2022	Day 9 06/03/2022	Day 8 07/03/2022	Day 7 08/03/2022	Day 6 09/03/2022	Day 5 10/03/2022	Day 4 11/03/2022	Day 3 12/03/2022	Day 2 13/03/2022	Day 1 14/03/2022	Tin	Hoa	Dong	Kieu																										
1	Sprint Detail Plan				All member	Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1																					
2	Sprint Backlog				All member	Completed	4	0	4	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1																					
3	Pre-sprint	Planning meeting		All member	Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1																					
4		Proposal Document		All member	Completed	4	0	4	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1																					
5		Project Plan Document		All member	Completed	4	0	4	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1																					
6		Product Backlog Document		All member	Completed	4	0	4	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1																					
7		Design Wireframe UI		All member	Completed	16	0	16	16	12	8	4	4	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	4																						
8	Data Modeling	Initiate RML process		Hoa Vo	Completed	24	0	24	21	17	14	11	6	3	0	0	0	0	0	0	0	0	0	0	0	x				24																						
9		Initiate MIRROR		Hoa Vo	Completed	23	0	23	23	23	23	23	23	23	22	17	15	9	4	0	0	0	x					23																								
10		Initiate GraphDB Database		Hoa Vo	Completed	20	0	15	13	10	5	0	0	0	0	0	0	0	0	0	0	0	0	0	x				20																							
11		Overview Database Design		Dong Ky	Completed	5	0	4	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x			5																							
12	Data Warehouse Designing	Turn information items into columns		Dong Ky	Completed	5	0	5	5	4	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	x			5																							
13		Specify primary, foreign keys		Dong Ky	Completed	5	0	5	5	5	4	4	0	0	0	0	0	0	0	0	0	0	0	0	0	x			5																							
14		Specify dimensions, measure, staging area tables		Dong Ky	Completed	10	0	5	5	5	5	4	4	0	0	0	0	0	0	0	0	0	0	0	0	x			10																							
15		Relation all the tables		Dong Ky	Completed	5	0	5	5	5	5	4	4	0	0	0	0	0	0	0	0	0	0	0	0	x			5																							
16		Design database OOP for generating database		Dong Ky	Completed	10	0	5	5	5	5	5	4	4	0	0	0	0	0	0	0	0	0	0	0	x			10																							
17	Set up database	Create python script for OOP database		Dong Ky	Completed	15	0	5	5	5	5	5	5	4	4	2	2	0	0	0	0	0	0	0	x			15																								
18		Database dimensions, facts, staging area generator query		Dong Ky	Completed	15	0	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	0	x			15																								
19		Brand Identity		Tin Pham	Completed	20	0	20	15	15	15	5	5	0	0	0	0	0	0	0	0	0	0	0	x			20																								
20	Design	Default screen		Tin Pham	Completed	5	0	5	5	5	5	5	5	5	5	0	0	0	0	0	0	0	0	0	0	x			5																							
21		Introduction screen		Tin Pham	Completed	5	0	5	5	5	5	5	5	5	5	0	0	0	0	0	0	0	0	0	0	x			5																							
22		Example screen		Tin Pham	Completed	20	0	20	20	20	20	20	20	20	20	17	14	6	0	0	0	0	0	0	x			20																								
23		Toolbox screen		Tin Pham	Completed	15	0	15	15	15	15	15	15	15	15	15	15	15	15	15	15	13	5	0	x			15																								

Sprint 1																												
C2SE-06 Team																												
#	Hours available per Sprint	Member Name	Start date	Mar 1, 2021	End date	Mar 14, 2021	Responsible	Status	Initial Est.	Comp.	Day 14 01/03/2021	Day 13 02/03/2021	Day 12 03/03/2021	Day 11 04/03/2021	Day 10 05/03/2021	Day 9 06/03/2021	Day 8 07/03/2021	Day 7 08/03/2021	Day 6 09/03/2021	Day 5 10/03/2021	Day 4 11/03/2021	Day 3 12/03/2021	Day 2 13/03/2021	Day 1 14/03/2021	Tin Hoa	Tin Hoa	Dong Kieu	Kieu
1	80	Pham Van Tin																										
2	80	Ky Huu Dong																										
3	44	Tran Thi Thanh Kieu																										
4	77	Vo Van Hoa																										
24		About modal	Tin Pham	Completed	5	0	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	x		5			
25	Testing	Test Plan document	Kieu Tran	Completed	4	0	4	4	4	4	4	4	4	4	2	0	0	0	0	0	0	0	x		4			
26		Write Test Case document	Kieu Tran	In Progress	31	1	31	31	31	31	31	31	31	31	31	26	21	16	11	6	1		x		30			

Working hours per day												
21.7												

Day	0	1	2	3	4	5	6	7	8	9	10	11	12	13
EST	282	260	239	217	195	174	152	130	108	87	65	43	22	0
Actual	246	225	196	183	156	146	129	118	94	77	58	38	21	1

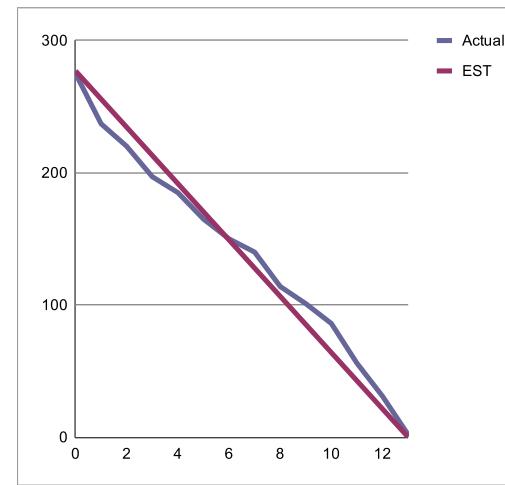
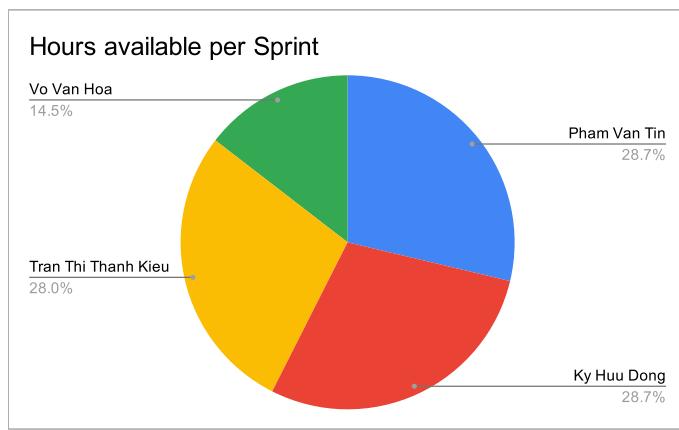


Sprint 2		EXPERT-DRIVEN SMART DASHBOARD APPLICATION																								
C2SE-06 Team																										
#	Hours available per Sprint	Member Name																								
1	79	Pham Van Tin		Start date		Mar 15, 2021																				
2	79	Ky Huu Dong		End date		Mar 28, 2021																				
3	77	Tran Thi Thanh Kieu																								
4	40	Vo Van Hoa																								
No.	BACKLOG ID	Task Description			Responsible	Status	Initial Est.	Comp.	Day 14 15/03/202	Day 15 16/03/202	Day 16 17/03/202	Day 17 18/03/202	Day 18 19/03/202	Day 19 20/03/202	Day 20 21/03/202	Day 21 22/03/202	Day 22 23/03/202	Day 23 24/03/202	Day 24 25/03/202	Day 25 26/03/202	Day 26 27/03/202	Day 27 28/03/202	Ho	Tin	Dong	Kieu
1	Sprint Detail Plan				All member	Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1		
2	Sprint Backlog				All member	Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1		
3	Pre-sprint	Planning meeting			All member	Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1		
4		Sprint Retrospective			All member	Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1		
5	Data processing	Define common mapping for RML & MIRROR			Hoa Vo	Completed	16	0	14	14	14	11	9	9	9	9	3	0	0	0	0	0	x	16		
6		Implement API for importing data, getting data from GraphDB			Hoa Vo	Completed	20	0	20	18	16	16	16	16	16	16	16	16	16	10	5	0	x	20		
7		Design edsda-ctl folder structure			Dong Ky	Completed	5	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	x	5		
8		Design clean engine for raw data source 1 with bonobo			Dong Ky	Completed	10	0	10	5	5	0	0	0	0	0	0	0	0	0	0	0	x	10		
9		Create clean engine, and data pipeline for raw data source 1 with bonobo			Dong Ky	Completed	15	0	15	15	10	5	5	0	0	0	0	0	0	0	0	0	x	15		
10		Perform clean engine for raw data source 1			Dong Ky	Completed	15	0	15	15	15	15	10	5	5	0	0	0	0	0	0	0	x	15		
11		Design clean engine for raw data source 2 with bonobo			Dong Ky	Completed	10	0	10	10	10	10	10	10	10	5	5	0	0	0	0	0	x	10		
12		Create clean engine, and data pipeline for raw data source 2 with bonobo			Dong Ky	Completed	10	0	10	10	10	10	10	10	10	10	10	10	10	2	0	0	x	10		
13		Perform clean engine for raw data source 2			Dong Ky	Completed	10	0	10	10	10	10	10	10	10	10	10	10	10	10	8	0	x	10		
14	Build UI	Initial React Project			Tin Pham	Completed	5	0	5	0	0	0	0	0	0	0	0	0	0	0	0	x	5			
15		Setup environment for Dev & Test			Tin Pham	Completed	10	0	10	10	5	0	0	0	0	0	0	0	0	0	0	x	10			
16		Implement common components			Tin Pham	Completed	20	0	20	20	20	20	15	10	5	0	0	0	0	0	0	x	20			
17		Navigation bar component			Tin Pham	Completed	10	0	10	10	10	10	10	10	10	10	5	0	0	0	0	x	10			
18		Setup basic Information component			Tin Pham	Completed	20	0	20	20	20	20	20	20	20	20	20	20	15	10	5	0	x	20		
19		Setup basic Editor Paner component			Tin Pham	Completed	10	0	10	10	10	10	10	10	10	10	10	10	10	7	4	0	x	10		
20	Testing	Write Test case for all features in Base UI			Kieu Tran	Completed	35	0	35	30	25	20	15	10	5	0	0	0	0	0	0	x	35			
21		Test the feature in Base UI			Kieu Tran	In Progress	40	2	40	40	40	40	40	40	40	40	35	30	25	17	9	2	x	38		

Sprint 2			EXPERT-DRIVEN SMART DASHBOARD APPLICATION																							
C2SE-06 Team																										
#	Hours available per Sprint	Member Name					Initial Est.	Comp.	Day 14 15/03/2022	Day 13 16/03/2022	Day 12 17/03/2022	Day 11 18/03/2022	Day 10 19/03/2022	Day 9 20/03/2022	Day 8 21/03/2022	Day 7 22/03/2022	Day 6 23/03/2022	Day 5 24/03/2022	Day 4 25/03/2022	Day 3 26/03/2022	Day 2 27/03/2022	Day 1 28/03/2022	1 Hoa	Tin	Dong	Kieu
1	79	Pham Van Tin	Start date	Mar 15, 2021																						
2	79	Ky Huu Dong	End date	Mar 28, 2021																						
3	77	Tran Thi Thanh Kieu																								
4	40	Vo Van Hoa																								
No.	BACKLOG ID	Task Description		Responsible	Status																					

Working hours per day													
21.3													

Day	0	1	2	3	4	5	6	7	8	9	10	11	12	13
EST	277	256	234	213	192	171	149	128	107	85	64	43	21	0
Actual	275	237	220	197	185	165	150	140	114	101	86	56	31	2

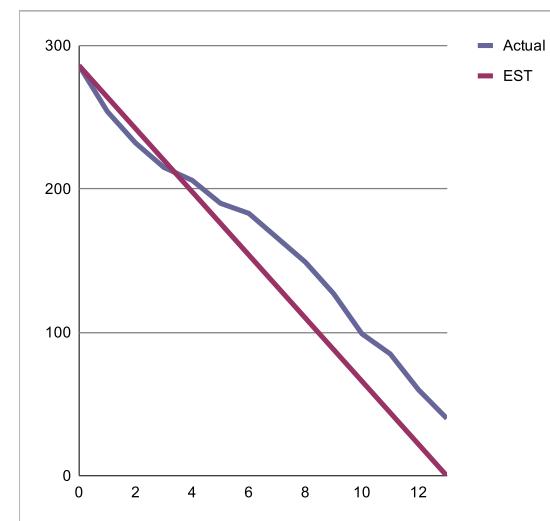
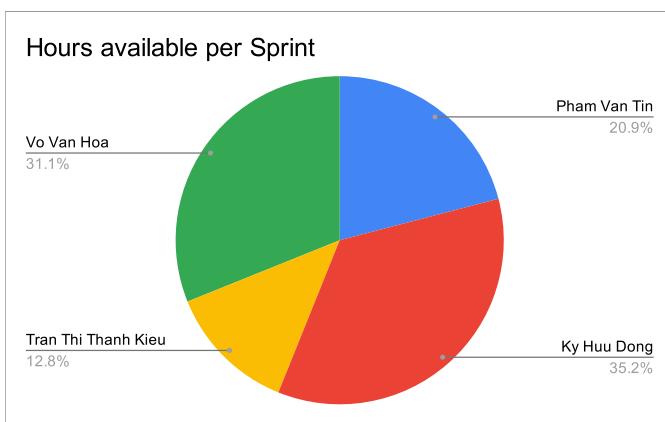


Sprint 3			EXPERT-DRIVEN SMART DASHBOARD APPLICATION																					
C2SE-06 Team																								
#	Hours available per Sprint	Member Name																						
1	51.5	Pham Van Tin	Start date		Mar 29, 2021																			
2	86.5	Ky Huu Dong	End date		Apr 11, 2021																			
3	31.5	Tran Thi Thanh Kieu																						
4	76.5	Vo Van Hoa																						
No.	BACKLOG ID	Task Description (User Story/Use Case Broken Down)	Responsible	Status(Not Started In Progress Completed)	Initial Est.	Comp.	Day 14 29/03/2021	Day 13 30/03/2021	Day 12 31/03/2021	Day 11 01/04/2021	Day 10 02/04/2021	Day 9 03/04/2021	Day 8 04/04/2021	Day 7 05/04/2021	Day 6 06/04/2021	Day 5 07/04/2021	Day 4 08/04/2021	Day 3 09/04/2021	Day 2 10/04/2021	Day 1 11/04/2021	Hoia	Tin	Dong	Kieu
					286	40	286	254	232	215	206	190	183	166	149	127	99	85	60	40				
1	Sprint Detail Plan	All member	Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1
2	Sprint Backlog	All member	Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1
3	Pre-sprint	Planning meeting	All member	Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1
4		Sprint Retrospective	All member	Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1
5	Data processing	Describe common schema for RDF Data Cubes	Hoa Vo	Completed	30	0	30	27	23	20	16	13	13	10	6	2	0	0	0	x				30
6		Run RML & MIRROR with sample data	Hoa Vo	Completed	35	0	35	35	35	35	35	35	35	30	24	20	14	10	5	0	x			35
7		Create clean engine, and data pipeline for raw data source 3 with bonobo	Dong Ky	Completed	10	0	10	5	0	0	0	0	0	0	0	0	0	0	0	x				10
8		Perform clean engine for raw data source 3	Dong Ky	Completed	10	0	10	10	5	0	0	0	0	0	0	0	0	0	0	x				10
9		Design clean engine for raw data source 3 with bonobo	Dong Ky	Completed	10	0	10	10	10	5	5	0	0	0	0	0	0	0	0	x				10
10		Create clean engine, and data pipeline for raw data source 3 with bonobo	Dong Ky	Completed	15	0	15	15	15	15	15	12	10	6	4	0	0	0	0	x				15
11		Perform clean engine for raw data source 3	Dong Ky	Completed	15	0	15	15	15	15	15	15	15	15	15	10	5	5	0	x				15
12	Dashboard UI	Build function drag & drop item from sidebar to main	Tin Pham	Completed	20	0	20	12	4	0	0	0	0	0	0	0	0	0	0	x				20
13		Restructure UI	Tin Pham	Completed	20	0	20	20	20	20	15	10	5	0	0	0	0	0	0	x				20
14	Desgin	Design new format of UI	All member	In Progress	40	10	40	40	40	40	40	40	40	40	40	35	30	25	20	15	X	x	X	7.5
15	Testing	Testing the data pipeline for raw data sources	Dong Ky	Completed	15	0	15	15	15	15	15	15	15	15	15	15	10	10	5	0	x			15
16		Write test cases for new UI	Kieu Tran	In Progress	50	30	50	50	50	50	50	50	50	50	50	50	50	45	40	35	30		X	20

Sprint 3			EXPERT-DRIVEN SMART DASHBOARD APPLICATION																			
C2SE-06 Team																						
#	Hours available per Sprint	Member Name			Start date Mar 29, 2021	End date Apr 11, 2021																
1	51.5	Pham Van Tin	Responsible	Status(Not Started In Progress Completed)			Initial Est.	Comp.	Day 14 29/03/2021	Day 13 30/03/2021	Day 12 31/03/2021	Day 11 01/04/2021	Day 10 02/04/2021	Day 9 03/04/2021	Day 8 04/04/2021	Day 7 05/04/2021	Day 6 06/04/2021	Day 5 07/04/2021	Day 4 08/04/2021	Day 3 09/04/2021	Day 2 10/04/2021	Day 1 11/04/2021
2	86.5	Ky Huu Dong																				
3	31.5	Tran Thi Thanh Kieu																				
4	76.5	Vo Van Hoa																				
No.	BACKLOG ID	Task Description (User Story/Use Case Broken Down)																				

Working hours per day													
22													

Day	0	1	2	3	4	5	6	7	8	9	10	11	12	13
EST	286	264	242	220	198	176	154	132	110	88	66	44	22	0
Actual	286	254	232	215	206	190	183	166	149	127	99	85	60	40

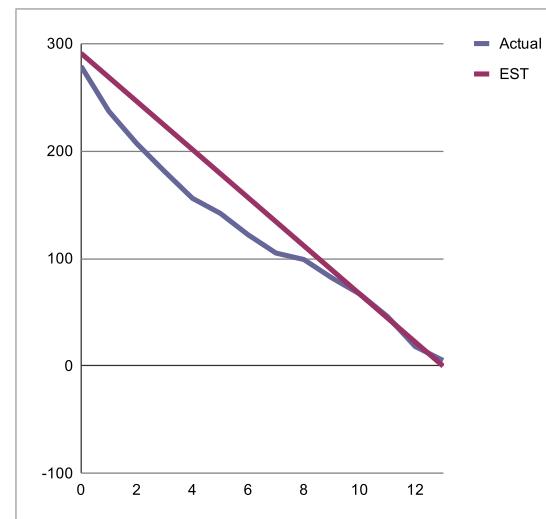
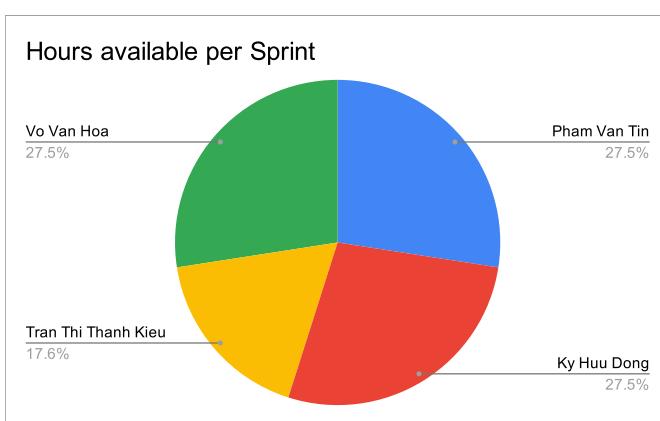


Sprint 4		EXPERT-DRIVEN SMART DASHBOARD APPLICATION																																					
C2SE-06 Team																																							
#	Hours available per Sprint	Member Name																																					
1	84	Pham Van Tin		Start date		Apr 19, 2021																																	
2	84	Ky Huu Dong		End date		May 2, 2021																																	
3	54	Tran Thi Thanh Kieu																																					
4	84	Vo Van Hoa																																					
No.	BACKLOG ID	Task Description (User Story/Use Case Broken Down)				Responsible	Status(Not Started In Progress Completed)	Initial Est.	Comp.	Day 14 19/04/2021	Day 13 20/04/2021	Day 12 21/04/2021	Day 11 22/04/2021	Day 10 23/04/2021	Day 9 24/04/2021	Day 8 25/04/2021	Day 7 26/04/2021	Day 6 27/04/2021	Day 5 28/04/2021	Day 4 29/04/2021	Day 3 30/04/2021	Day 2 01/05/2021	Day 1 02/05/2021	Hoa	Tin	Dong	Kieu												
1	Sprint Detail Plan					All member	Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1												
2	Sprint Backlog					All member	Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1												
3	Pre-sprint	Planning meeting				All member	Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1												
4		Sprint Retrospective				All member	Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1												
5	Data processing	Customize RML & MIRROR				Hoa Vo	Completed	40	0	38	32	27	21	16	12	12	7	7	7	7	7	3	0	x			40												
6		Implement pipeline for automating RDF Data Cube generation				Hoa Vo	Completed	30	0	20	20	20	20	20	20	20	16	12	8	8	8	2	0	x			30												
7		Implement insert, update method				Dong Ky	Completed	10	0	10	5	0	0	0	0	0	0	0	0	0	0	0	x				10												
8		Implement ETL to perform data to Data warehouse				Dong Ky	Completed	15	0	15	10	5	0	0	0	0	0	0	0	0	0	0	x				15												
9		Design base workflow for Data warehouse				Dong Ky	Completed	10	0	10	10	10	5	0	0	0	0	0	0	0	0	0	x				10												
10		Perform base workflow for Data warehouse				Dong Ky	Completed	15	0	15	15	15	15	10	5	5	0	0	0	0	0	0	x				15												
11	ETL Process Validating	Perform ETL with bonobo				Dong Ky	In Progress	15	0	15	15	15	15	15	15	15	12	10	5	3	0	0	0	x			15												
12		Implement enrich data engine				Dong Ky	In Progress	15	0	15	15	15	15	15	15	15	15	15	10	5	0	0	x			15													
13	Dashboard UI	Setup the basic common component on new UI				Tin Pham	Completed	5	0	5	0	0	0	0	0	0	0	0	0	0	0	0	x				5												
14		Create Default Screen				Tin Pham	Completed	10	0	10	10	5	0	0	0	0	0	0	0	0	0	0	x				10												
15		Create introduction screen				Tin Pham	Completed	10	0	10	10	10	10	5	0	0	0	0	0	0	0	x				10													
16		create about screen				Tin Pham	Completed	5	0	5	5	5	5	5	5	0	0	0	0	0	0	0	x				5												
17		create Toolbox screen				Tin Pham	In Progress	15	5	15	15	15	15	15	15	5	5	5	5	5	5	x				10													
18		create Show Example screen				Tin Pham	Completed	40	0	40	40	40	40	40	40	40	40	40	32	24	16	8	0	x			40												
19	Test	Test RML process				Hoa Vo	Completed	10	0	10	10	10	10	10	10	10	10	10	10	10	5	0	0	x			10												
20		Test DW ETL process				Kieu Tran	Completed	10	0	10	5	0	0	0	0	0	0	0	0	0	0	0		x			10												
21		Test workflow for DW				Kieu Tran	Completed	10	0	10	10	5	0	0	0	0	0	0	0	0	0	0		x			10												

Sprint 4			EXPERT-DRIVEN SMART DASHBOARD APPLICATION																					
C2SE-06 Team																								
#	Hours available per Sprint	Member Name																						
1	84	Pham Van Tin	Start date	Apr 19, 2021																				
2	84	Ky Huu Dong	End date	May 2, 2021																				
3	54	Tran Thi Thanh Kieu																						
4	84	Vo Van Hoa																						
No.	BACKLOG ID	Task Description (User Story/Use Case Broken Down)	Responsible	Status(Not Started In Progress Completed)	Initial Est.	Comp.	Day 14 19/04/2021	Day 13 20/04/2021	Day 12 21/04/2021	Day 11 22/04/2021	Day 10 23/04/2021	Day 9 24/04/2021	Day 8 25/04/2021	Day 7 26/04/2021	Day 6 27/04/2021	Day 5 28/04/2021	Day 4 29/04/2021	Day 3 30/04/2021	Day 2 01/05/2021	Day 1 02/05/2021	Hoa	Tin	Dong	Kieu
22	Testing	Test default Screen	Kieu Tran	Completed	5	0	5	5	5	5	0	0	0	0	0	0	0	x	5					
23		Test introduction screen	Kieu Tran	Completed	5	0	5	5	5	5	5	0	0	0	0	0	0	x	5					
24		Test about screen	Kieu Tran	Completed	5	0	5	5	5	5	5	5	0	0	0	0	0	x	5					
25		Test Toolbox screen	Kieu Tran	In Progress	10	5	10	10	10	10	10	10	10	5	5	5	5	x	5					
26		Test Show Example screen: widget on Editor Pane component	Kieu Tran	In Progress	35	25	35	35	35	35	35	35	35	35	35	35	30	25	25	x	10			

Working hours per day													
22.4													

Day	0	1	2	3	4	5	6	7	8	9	10	11	12	13
EST	291	269	246	224	201	179	157	134	112	89	67	45	22	0
Actual	279	237	207	181	156	142	122	105	99	82	67	46	18	5

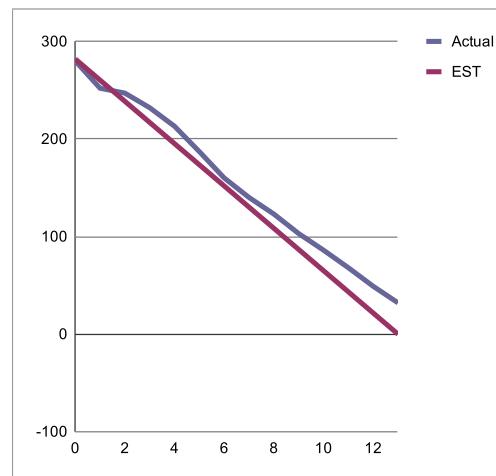
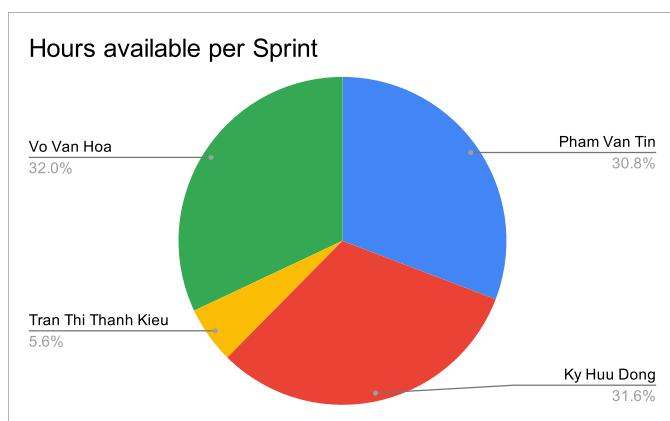


Sprint 5			EXPERT-DRIVEN SMART DASHBOARD APPLICATION																					
C2SE-06 Team																								
#	Hours available per Sprint	Member Name																						
1	77	Pham Van Tin	Start date	May 3, 2021																				
2	79	Ky Huu Dong	End date	May 16, 2021																				
3	14	Tran Thi Thanh Kieu																						
4	80	Vo Van Hoa																						
No.	BACKLOG ID	Task Description (User Story/Use Case Broken Down)	Responsible	Status(Not Started In Progress Completed)	Initial Est.	Comp.	Day 14 03/05/2021	Day 13 04/05/2021	Day 12 05/05/2021	Day 11 06/05/2021	Day 10 07/05/2021	Day 9 08/05/2021	Day 8 09/05/2021	Day 7 10/05/2021	Day 6 11/05/2021	Day 5 12/05/2021	Day 4 13/05/2021	Day 3 14/05/2021	Day 2 15/05/2021	Day 1 16/05/2021	Hoa	Tin	Dong	Kieu
					282	32	280	252	247	232	213	187	160	140	123	103	86	68	49	32				
1	Sprint Detail Plan			Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x
2	Sprint Backlog			Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x
3	Pre-sprint	Planning meeting		Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x
4		Sprint Retrospective		Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x
5	Data processing	Integrate automate ETL & RDF generation	Hoa Vo	Completed	34	0	32	30	30	25	21	16	12	10	10	10	8	4	4	0	x			34
6		Automating detect dimension & measure	Hoa Vo	Completed	32	0	32	32	32	32	26	21	16	12	10	8	8	4	0	x				32
7	Import data source	Build basic UI for import interface	Dong Ky	Completed	10	0	10	10	10	5	0	0	0	0	0	0	0	0	0	0	x			10
8		Implement logic for import UI	Dong Ky	Completed	15	0	15	15	15	15	10	5	0	0	0	0	0	0	0	0	x			15
9		Implement server for handling import data	Dong Ky	In Progress	20	0	20	20	20	20	20	15	10	5	0	0	0	0	0	0	x			20
10		Implement import engine	Dong Ky	In Progress	20	0	20	20	20	20	20	20	20	20	20	15	10	5	0	0	x			20
11	ETL Process Validating	Perform ETL with bonobo	Dong Ky	Completed	5	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	x			5
12		Implement enrich data engine	Dong Ky	Completed	5	0	5	5	0	0	0	0	0	0	0	0	0	0	0	0	x			5
13	Dashboard UI	continue create Toolbox	Tin Pham	Completed	5	0	5	5	5	0	0	0	0	0	0	0	0	0	0	0	x			5
14		create import widget	Tin Pham	Completed	5	0	5	5	5	5	0	0	0	0	0	0	0	0	0	0	x			5
15		create import function	Tin Pham	In Progress	35	22	35	35	35	35	30	22	22	22	22	22	22	22	22	22	x			13
16		create function to view the EDSDA's data source on widget	Tin Pham	Completed	25	0	25	25	25	25	25	25	17	9	1	0	0	0	0	0	x			25
17		create function to merge multi widget	Tin Pham	Completed	10	0	10	10	10	10	10	10	10	10	10	10	3	0	0	0	x			10
18		visualization data on Charts	Tin Pham	Completed	15	0	15	15	15	15	15	15	15	15	15	15	9	4	0	x				15
19		Test RDF generator & ETL pipeline	Hoa Vo	Completed	5	0	5	5	5	5	5	5	5	5	5	5	5	5	5	0	x			5
20		Test dim & measure detector	Kieu Tran	Completed	5	0	5	0	0	0	0	0	0	0	0	0	0	0	0	0	x			5
21		Test automatic processing	Hoa Vo	Completed	5	0	5	5	5	5	5	5	5	5	5	5	5	5	5	0	x			5

Sprint 5		EXPERT-DRIVEN SMART DASHBOARD APPLICATION																								
C2SE-06 Team																										
#	Hours available per Sprint	Member Name																								
1	77	Pham Van Tin		Start date May 3, 2021																						
2	79	Ky Huu Dong		End date May 16, 2021																						
3	14	Tran Thi Thanh Kieu																								
4	80	Vo Van Hoa																								
No.	BACKLOG ID	Task Description (User Story/Use Case Broken Down)			Responsible	Status(Not Started In Progress Completed)	Initial Est.	Comp.	Day 14 03/05/2021	Day 13 04/05/2021	Day 12 05/05/2021	Day 11 06/05/2021	Day 10 07/05/2021	Day 9 08/05/2021	Day 8 09/05/2021	Day 7 10/05/2021	Day 6 11/05/2021	Day 5 12/05/2021	Day 4 13/05/2021	Day 3 14/05/2021	Day 2 15/05/2021	Day 1 16/05/2021	Hoa	Tin	Dong	Kieu
22	Testing	Test the import widget			Kieu Tran	Completed	5	0	5	5	5	5	5	0	0	0	0	0	0	x	5					
23		Test the logic of import function			Kieu Tran	Completed	10	10	10	10	10	10	10	10	10	10	10	10	10	x	0					
24		Test the Tree View of EDSDA data source on widget			Kieu Tran	Completed	5	0	5	5	5	5	5	5	5	5	5	5	0	0	5					
25		Test the data in Information component			Kieu Tran	Completed	20	0	20	20	20	20	20	20	17	10	7	5	2	0	0	0				
26		Test Toolbox UI			Kieu Tran	Completed	15	0	15	15	15	15	15	15	15	15	15	12	8	3	0					
27		Write test for visualization on Charts			Kieu Tran	Completed	5	0	5	5	5	5	5	5	5	5	5	5	5	0	x	5				
28		Continue Test the Show Example screen			Kieu Tran	Completed	25	0	25	25	20	15	10	5	0	0	0	0	0	0	x	25				

Working hours per day												
21.7												

Day	0	1	2	3	4	5	6	7	8	9	10	11	12	13
EST	282	260	239	217	195	174	152	130	108	87	65	43	22	0
Actual	280	252	247	232	213	187	160	140	123	103	86	68	49	32



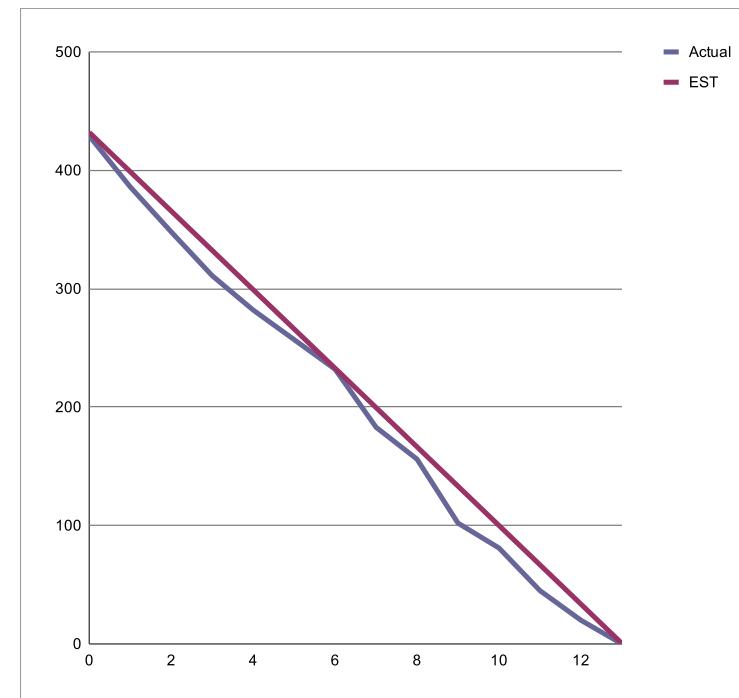
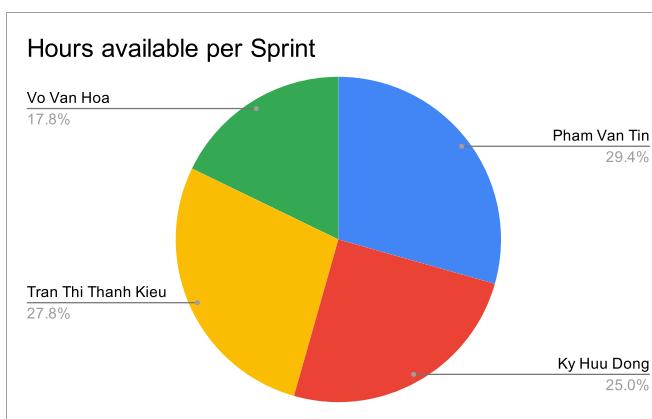
Sprint 6			EXPERT-DRIVEN SMART DASHBOARD APPLICATION																				
C2SE-06 Team																							
#	Hours available per Sprint	Member Name																					
1	127	Pham Van Tin																					
2	108	Ky Huu Dong																					
3	120	Tran Thi Thanh Kieu																					
4	77	Vo Van Hoa																					
No.	BACKLOG ID	Task Description (User Story/Use Case Broken Down)	Responsible	Status(Not Started In Progress Completed)	Initial Est.	Comp.	Day 14 17/05/2021	Day 13 18/05/2021	Day 12 19/05/2021	Day 11 20/05/2021	Day 10 21/05/2021	Day 9 22/05/2021	Day 8 23/05/2021	Day 7 24/05/2021	Day 6 25/05/2021	Day 5 26/05/2021	Day 3 28/05/2021	Day 2 29/05/2021	Day 1 30/05/2021	Hoa	Tin	Dong	Kieu
1	Sprint Detail Plan	All member	Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1
2	Sprint Backlog	All member	Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1
3	Pre-sprint	Planning meeting	All member	Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1
4		Sprint Retrospective	All member	Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	0	x	x	x	x	1
5	Documents	Reworking Proposal	All member	Completed	4	0	4	4	4	4	4	4	4	4	0	0	0	0	0	x	x	x	1
6		Reworking Project Plan	All member	Completed	4	0	4	4	4	4	4	4	4	4	0	0	0	0	0	x	x	x	1
7		Architecture Document	All member	Completed	4	0	4	4	4	4	4	4	4	4	0	0	0	0	0	x	x	x	1
8		Interface Design	All member	Completed	4	0	4	4	4	4	4	4	4	4	0	0	0	0	0	x	x	x	1
9		Database Design	All member	Completed	4	0	4	4	4	4	4	4	4	4	0	0	0	0	0	x	x	x	1
10		Test Plan	All member	Completed	4	0	4	4	4	4	4	4	4	4	0	0	0	0	0	x	x	x	1
11		Test Case	All member	Completed	4	0	4	4	4	4	4	4	4	4	4	0	0	0	0	x	x	x	1
12		Acceptance Criteria	All member	Completed	4	0	4	4	4	4	4	4	4	4	4	0	0	0	0	x	x	x	1
13		Technologies Stack	All member	Completed	4	0	4	4	4	4	4	4	4	4	4	0	0	0	0	x	x	x	1
14		Team Reflection	All member	Completed	4	0	4	4	4	4	4	4	4	4	4	0	0	0	0	x	x	x	1
15		Team meeting document	All member	Completed	4	0	4	4	4	4	4	4	4	4	4	0	0	0	0	x	x	x	1
16		Mentor meeting document	All member	Completed	4	0	4	4	4	4	4	4	4	4	4	0	0	0	0	x	x	x	1
17	Server Setup	Set up VPS for deploying	Hoa Vo	Completed	26	0	26	22	16	12	6	4	2	0	0	0	0	0	x			26	
18		Migrate GraphDB to VPS	Hoa Vo	Completed	24	0	24	20	15	10	10	10	10	10	5	0	0	0	0	x			24
19	Import data source	Implement server for handling import data	Dong Ky	Completed	10	0	10	5	0	0	0	0	0	0	0	0	0	0		x		10	
20		Implement import engine	Dong Ky	Completed	10	0	10	10	5	0	0	0	0	0	0	0	0	0		x		10	
21		Running Apache Airflow web server and scheduler	Dong Ky	Completed	10	0	10	10	10	5	0	0	0	0	0	0	0	0		x		10	

Sprint 6										EXPERT-DRIVEN SMART DASHBOARD APPLICATION																						
C2SE-06 Team																																
#	Hours available per Sprint	Member Name																														
1	127	Pham Van Tin			Start date		May 17, 2021																									
2	108	Ky Huu Dong			End date		May 30, 2021																									
3	120	Tran Thi Thanh Kieu																														
4	77	Vo Van Hoa																														
No.	BACKLOG ID	Task Description (User Story/Use Case Broken Down)				Responsible	Status(Not Started	In Progress	Completed	Initial Est.	Comp.	Day 14 17/05/2021	Day 15 18/05/2021	Day 13 19/05/2021	Day 12 20/05/2021	Day 11 21/05/2021	Day 10 22/05/2021	Day 9 23/05/2021	Day 8 24/05/2021	Day 7 25/05/2021	Day 6 26/05/2021	Day 5 27/05/2021	Day 4 28/05/2021	Day 3 29/05/2021	Day 2 30/05/2021	Day 1 31/05/2021	Hoa	Tin	Dong	Kieu		
22	Create workflow for automation ETL	Create base workflow with Apache airflow				Dong Ky	Completed	15	0	15	15	15	15	15	10	5	0	0	0	0	0	0	0	0	0	X		15				
23		Create work flow for scheduling craw data source				Dong Ky	Completed	20	0	20	20	20	20	20	20	20	15	10	5	0	0	0	0	0	0	X		20				
24		Create work flow for scheduling perform ETL data to data warehouse				Dong Ky	Completed	20	0	20	20	20	20	20	20	20	20	20	20	15	10	5	0	0	0	X		20				
25	Dashboard UI	Continue create import function				Tin Pham	Completed	22	0	22	18	10	2	0	0	0	0	0	0	0	0	0	0	0	0	X		22				
26		Visualization data on Maps & Table				Tin Pham	Completed	25	0	25	25	25	25	19	11	3	0	0	0	0	0	0	0	0	0	X		25				
27		Implement import function with server API				Tin Pham	Completed	40	0	40	40	40	40	40	40	35	27	19	10	1	0	0	0	0	0	X		40				
28		Deployment				Tin Pham	Completed	15	0	15	15	15	15	15	15	15	15	15	15	15	15	8	0	0	0	X		15				
29	Testing	Test data of visualization on Charts				Kieu Tran	Completed	12	0	12	8	4	0	0	0	0	0	0	0	0	0	0	0	0	0	X		12				
30		Test UI of Charts				Kieu Tran	Completed	5	0	5	5	4	0	0	0	0	0	0	0	0	0	0	0	0	0	X		5				
31		Test the Tree View of data source on widget				Kieu Tran	Completed	5	0	5	5	5	5	0	0	0	0	0	0	0	0	0	0	0	0	X		5				
32		Test Connector and port of widgets				Kieu Tran	Completed	5	0	5	5	5	5	5	5	0	0	0	0	0	0	0	0	0	0	X		5				
33		Write Test for visualization on Maps & Table				Kieu Tran	Completed	10	0	10	10	10	10	10	10	5	0	0	0	0	0	0	0	0	0	X		10				
34		Write test for Import function				Kieu Tran	Completed	5	0	5	5	5	5	5	5	5	0	0	0	0	0	0	0	0	0	X		5				
35		Test function of import				Kieu Tran	Completed	7	0	7	7	7	7	7	2	2	2	0	0	0	0	0	0	0	0	X		7				
36		Test the UI on domain: sda-research.surge.sh				Kieu Tran	Completed	40	0	40	40	40	40	40	40	40	40	33	26	19	12	5	0	0	0	X		40				
37		Import data from UI				Kieu Tran	Completed	2	0	2	2	2	0	0	0	0	0	0	0	0	0	0	0	0	0	X		2				
38		Test RDF format				Kieu Tran	Completed	2	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	X		2				
39		Test workflows, data pipelines				Hoa Vo	Completed	4	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	X		4				
40		Test consistancy of RDF and DW				Kieu Tran	Completed	4	0	4	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	X		4				
41		Public User Acceptance Test				All member	Completed	8	0	8	8	8	8	8	8	8	8	8	8	8	8	8	4	0	X	X	X	2				
42	Presentation				All member		16	0	16	16	16	16	16	16	16	16	16	16	16	16	16	8	8	0	X	X	X	4				
43	Print Document				Tin Pham	Completed	2	0	2	2	2	2	2	2	2	2	2	2	2	2	2	2	0	0	X			2				

Sprint 6			EXPERT-DRIVEN SMART DASHBOARD APPLICATION																						
C2SE-06 Team																									
#	Hours available per Sprint	Member Name																							
1	127	Pham Van Tin	Start date	May 17, 2021																					
2	108	Ky Huu Dong	End date	May 30, 2021																					
3	120	Tran Thi Thanh Kieu																							
4	77	Vo Van Hoa																							
No.	BACKLOG ID	Task Description (User Story/Use Case Broken Down)	Responsible	Status(Not Started In Progress Completed)	Initial Est.	Comp.	Day 14 17/05/2021	Day 13 18/05/2021	Day 12 19/05/2021	Day 11 20/05/2021	Day 10 21/05/2021	Day 9 22/05/2021	Day 8 23/05/2021	Day 7 24/05/2021	Day 6 25/05/2021	Day 5 26/05/2021	Day 4 27/05/2021	Day 3 28/05/2021	Day 2 29/05/2021	Day 1 30/05/2021	Hoa	Tin	Dong	Kieu	
44	Deployment & Release	All member	Completed	4	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	X	X	X	X	1	

Working hours per day												
33.2												

Day	0	1	2	3	4	5	6	7	8	9	10	11	12	13
EST	432	399	366	332	299	266	233	200	166	133	100	67	34	0
Actual	429	386	348	311	282	257	232	183	156	102	81	45	20	0





## CAPSTONE PROJECT 2

CMU-SE-451 / CMU-IS-451 / CMU-CS-451

### CODE STANDARD

Version 2.0

Date: 1 - Mar - 2021

## EXPERT-DRIVEN SMART DASHBOARD APPLICATION

### Submitted by

Vo Van Hoa  
Pham Van Tin  
Ky Huu Dong  
Tran Thi Thanh Kieu

### Approved by

#### Capstone Project 2 - Mentor:

Name

Signature

Date

Binh, Thanh Nguyen \_\_\_\_\_  \_\_\_\_\_ 26 - May - 2020

PROJECT INFORMATION			
<b>Project Acronym</b>	EDSDA		
<b>Project Title</b>	Expert-Driven Smart Dashboard Application		
<b>Project Web URL</b>	<a href="https://sda-research.ml/">https://sda-research.ml/</a>		
<b>Start Date</b>	01 - Mar - 2021		
<b>End Date:</b>	02 - Jun - 2021		
<b>Lead Institution</b>	International School, Duy Tan University		
<b>Project Mentor</b>	Ph.D Binh, Thanh Nguyen		
<b>Scrum Master</b>	Hoa, Vo	hoavo.dng@gmail.com	0935.193.182
<b>Team Members</b>	Tin, Pham Van	tinphamvan123@gmail.com	0932.535.175
	Dong, Ky Huu	kyhuudong@gmail.com	0898.246.980
	Kieu, Tran Thi Thanh	thanhkieutran391@gmail.com	0358.583.251

DOCUMENT INFORMATION			
<b>Document Title</b>	Code Standard		
<b>Author(s)</b>	Team C2SE.06		
<b>Role</b>	[EDSDA] Code Standard v2.2		
<b>Date</b>	01 - Mar - 2021	Filename	[EDSDA] 010 Code Standard
<b>URL</b>	<a href="https://github.com/sdateamdtu2020/SDA-v2.0">https://github.com/sdateamdtu2020/SDA-v2.0</a>		
<b>Access</b>	Project and CMU Program		

## REVISION HISTORY

# Table of Contents

- [Python Style Guide](#)
  - [Table of Contents](#)
  - [Types](#)
  - [References](#)
  - [Dictionaries](#)
  - [Lists](#)
  - [Destructuring](#)
  - [Strings](#)
  - [Functions](#)
  - [Classes & Constructors](#)
  - [Modules](#)
  - [Iterators and Generators](#)
  - [Variables](#)
  - [Comparison Operators & Equality](#)
  - [Comments](#)
  - [Whitespace](#)
  - [Commas](#)
  - [Naming Conventions](#)
- [React/JSX Style Guide](#)
  - [Basic Rules](#)
  - [Class vs React.createClass vs stateless](#)
  - [Mixins](#)
  - [Naming](#)
  - [Declaration](#)
  - [Alignment](#)
  - [Quotes](#)
  - [Spacing](#)
  - [Props](#)
  - [Refs](#)
  - [Parentheses](#)
  - [Tags](#)
  - [Methods](#)
  - [Ordering](#)
  - [isMounted](#)

## Types

- **1.1 Primitives:** When you access a primitive type you work directly on its value.
  - [string](#)
  - [number](#)
  - [boolean](#)
  - [None](#)

```
foo = 1
bar = foo

bar = 9

print(foo, bar) # => 1, 9
```

- **1.2 Complex:** When you access a complex type you work on a reference to its value.

- `dict`
- `list`
- `function`

```
foo = [1, 2]
bar = foo

bar[0] = 9

print(foo[0], bar[0]) # => 9, 9
```

[↑ back to top](#)

## References

- **2.1** Use `CONST` for all of your references; avoid using `var`. Python does not have `constant` type, so you need to observe the convention using UPPERCASE for constants and never modify them.

Why? This ensures that you can't reassign your references, which can lead to bugs and difficult to comprehend code.

```
# bad
foo = 1
bar = 2

# good
FOO = 1
BAR = 2
```

[↑ back to top](#)

## Dictionaries

- **3.1** Use the literal syntax for dictionary creation.

```
# bad
item = dict()

# good
item = {}
```

- 3.2 Use computed key names when creating dictionaries with dynamic key names.

Why? They allow you to define all the key of a dictionary in one place.

```
def get_key(k):
    return f'a key named {k}'

# bad
obj = {
    'id': 5,
    'name': 'San Francisco',
}
obj[get_key('enabled')] = True

# good
obj = {
    'id': 5,
    'name': 'San Francisco',
    get_key('enabled'): True,
}
```

- 3.3 Prefer the dictionary spread operator over `copy()` to shallow-copy and extend dictionaries.

```
# bad
original = {'a': 1, 'b': 2}
clone = original.copy()
clone.update({'c': 3})

# good
original = {'a': 1, 'b': 2}
clone = {**original, 'c': 3}

# good
original = {'a': 1, 'b': 2}
original_2 = {'c': 3, 'd': 4}
long_clone = {**original, **original_2, 'e': 5}
```

- 3.4 Use line breaks after open and before close dictionary braces only if a dictionary has multiple lines.

```
# bad - single item will not exceed one line
single_map = {
    'a': 1,
}

# bad - single line
item_map = {
    'a': 1, 'b': 2, 'c': 3,
}

# good
single_map = {'a': 1}

item_map = {
    'a': 1,
    'b': 2,
    'c': 3,
}
```

- 3.5 Use `dict.get(key)` to get properties.

Why? Getting via `dict[key]` will break on missing key, and requires bloated code to guard against.

```
item_map = {
    'a': 1,
    'b': 2,
}

# bad - throws error
item_map['c']

# bad - bloated code
try:
    item_map['c']
except KeyError:
    item_map['c'] = 3
    return item_map['c']

# good
item_map.get('c')

# good
item_map['c'] = item_map.get('c') or 3
```

[↑ back to top](#)

## Lists

- 4.1 Use the literal syntax for list creation.

```
# bad
items = list()

# good
items = []
```

- 4.2 Use list spreads \* to copy and extend lists.

```
# bad
items = ['a', 'b']
clone = items.copy() + ['c']

# good
items = ['a', 'b']
clone = [*items, 'c']

# good
items = ['a', 'b']
items_2 = ['c', 'd']
clone = [*items, *items2, 'e']
```

- 4.3 Use line breaks after open and before close list brackets only if a list has multiple lines.

```
# bad - single line
items = [
    [0, 1], [2, 3], [4, 5],
]

# bad - no line break after bracket
dict_list = [{{
    'id': 1
}, {
    'id': 2
}]

number_list = [
    1, 2,
]

# good
items = [[0, 1], [2, 3], [4, 5]]

dict_list = [
    {'id': 1},
    {'id': 2},
]
```

```
number_list = [
    1,
    2,
]
```

[↑ back to top](#)

## Destructuring

- 5.1 Use list destructuring.

```
items = [1, 2, 3, 4, 5]

# bad
first = items[0]
second = items[1]

# good
first, second, *tail = items
first, second, *rest, last = items
```

[↑ back to top](#)

## Strings

- 6.1 Use single quotes ' ' for strings.

Why? Less escaping for double quote " ", less bloat, and makes code more searchable.

```
# bad
name = "Capt. Janeway"

# bad
json_string = "{\"a\": 1}"

# bad - f string should contain interpolation or newlines
name = f'Capt. Janeway'

# good
name = 'Capt. Janeway'

# good
json_string = '{"a": 1}'
```

- 6.2 Strings that cause the line to go over 79 characters should not be written across multiple lines using string concatenation.

Why? Broken strings are painful to work with and make code less searchable.

```
# bad
error_msg = 'This is a super long error that was thrown because \
of Batman. When you stop to think about how Batman had anything to do \
\
with this, you would get nowhere \
fast.'

# bad
error_msg = 'This is a super long error that was thrown because ' + \
    'of Batman. When you stop to think about how Batman had anything \
to do ' + \
    'with this, you would get nowhere fast.'

# good
error_msg = 'This is a super long error that was thrown because of \
Batman. When you stop to think about how Batman had anything to do \
with this, you would get nowhere fast.'
```

- 6.3 When programmatically building up strings, use template strings instead of concatenation.

Why? Template strings give you a readable, concise syntax with proper newlines and string interpolation features.

```
# bad
def say_hi(name):
    return 'How are you, ' + name + '?'

# bad
def say_hi(name):
    return ''.join(['How are you, ', name, '?'])

# bad
def say_hi(name):
    return f'How are you, {name}?'

# good
def say_hi(name):
    return f'How are you, {name}?'
```

For python < 3.6, convert f-string to template string by `format()`:

```
a = 1
b = 2
c = 3

# python >=3.6
```

```
f'a: {a} b: {b} c: {c}'

# python <3.6
'a: {a} b: {b} c: {c}'.format(a=a, b=b, c=c)

param = {'a': a, 'b': b, 'c': c}
'a: {a} b: {b} c: {c}'.format(**param)

'a: {} b: {} c: {}'.format(a, b, c)
```

- 6.4 Never use `eval()` on a string, it opens too many vulnerabilities.
- 6.5 Do not unnecessarily escape characters in strings.

**Why?** Backslashes harm readability, thus they should only be present when necessary.

```
# bad
foo = '\'this\' \is \"quoted\"'

# good
foo = '\'this\' is "quoted"'
foo = f'my name is "{name}"'
```

[↑ back to top](#)

## Functions

- 7.1 Use default parameter syntax rather than mutating function arguments.

```
# really bad
def do_something(opt):
    # No! We shouldn't mutate function arguments.
    # Double bad: if opt is falsy it'll be set to an object which may
    # be what you want but it can introduce subtle bugs.
    opt = opt or 'foo'
    #

# still bad
def do_something(opt):
    if (opt is None):
        opt = 'foo'
    #

# good
def do_something(opt='foo'):
    # ...
```

- 7.2 Do not use complex data type as default parameter.

Why? Variable to a complex type is a reference, and so the single instance will be modified.

```
# bad
def init_list(value, new_list=[]):
    new_list.append(value)
    return new_list

init_list(1)
# => [1]
init_list(2)
# => [1, 2], instead of the new init [2]

# good
def init_list(value, new_list=None):
    if new_list is None:
        new_list = []
    new_list.append(value)
    return new_list

init_list(1)
# => [1]
init_list(2)
# => [2]
```

- 7.3 No spacing in a function signature.

Why? Consistency is good, and eases code search.

```
# bad
def foo(a): print(a)
def bar (b): print(b)

# good
def foo(a): print(a)
def bar(b): print(b)
```

- 7.4 Never reassign parameters.

Why? Reassigning parameters can lead to unexpected behavior.

```
# bad
def fn_1(a):
    a = 1
    # ...

def fn_2(a):
    if (!a): a = 1
    # ...
```

```
# good
def fn_3(a):
    b = a or 1
    # ...

def fn_4(a=1):
    # ...
```

- 7.5 Functions with multiline signatures, or invocations, should be indented just like every other multiline list in this guide: with each item on a line by itself, with a trailing comma on the last item.

```
# bad
def some_fn(foo,
            bar,
            baz):
    # ...

# good
def some_fn(
    foo,
    bar,
    baz,
):
    # ...

# bad
some_fn(foo,
        bar,
        baz)

# good
some_fn(
    foo,
    bar,
    baz,
)
```

- 7.6 Call function with parameters by specifying their names.

**Why?** Clarity of parameters and future-proofing. When updating source code function parameters, it can be done reliably with minimal propagation.

```
def move(x, y, roll=False):
    # ...

# bad - unclear what the params mean
move(1, 0, True)
```

```
# good
move(x=1, y=0, roll=True)

# later when updating method, no need to propagate function calls
# since they will auto-assume z=0 reliably
def move(x, y, z=0, roll=False):
    # ...
```

- 7.7 Break down code logic into digestible chunks, and refactor a lot.

**Why?** Other programmers and your future self will thank you for writing understandable code.

```
# bad - short but extremely confusing
def long_logic():
    return [a for a in small_list for small_list in large_matrix if
len(small_list) else ['replacement'] if len(a) > 2]

# good - longer but very clear
def long_logic():
    result = []
    for small_list in large_matrix:
        if len(small_list) > 0:
            used_list = small_list
        else:
            used_list = ['replacement']
        for a in used_list:
            if len(a) > 2:
                result.append(a)
    return result
```

[↑ back to top](#)

## Classes & Constructors

- 8.1 Avoid duplicate class members.

**Why?** Duplicate class member declarations will silently prefer the last one - having duplicates is almost certainly a bug.

```
# bad
class Foo():
    def bar(): return 1
    def bar(): return 2

# good
class Foo():
    def bar(): return 1
```

[⬆ back to top](#)

## Modules

- 9.1 Do not use wildcard imports.

Why? To prevent namespace pollution and conflicts, and to know which modules your variables or functions come from.

```
# bad
from common.util import *

# good
from common import util
```

- 9.2 Do not import unused modules.

Why? Performance, reliability, containment. If a module breaks, your code that should be isolated from the module will break too. This causes more errors and makes it harder to debug.

- 9.3 Only import from a path in one place.

Why? Having multiple lines that import from the same path can make code harder to maintain.

```
# bad
from foo import bar
# ... some other imports
from foo import baz, qux

# good
from foo import bar, baz, qux

# good
from foo import (
    bar,
    baz,
    qux,
)
```

- 9.4 Put all `import`s above non-import statements.

Why? Since `import`s are hoisted, keeping them all at the top prevents surprising behavior.

```
# bad
import a_module
from b_module import foo
foo.init()
```

```
from c_module import bar

# good
import a_module
from b_module import foo
from c_module import bar

foo.init();
```

- 9.5 Sort the imports by `import` then `from`, and sort alphabetically.

Why? `import` are often more generic than `from`; sort to ease manual inspection and for maintainability.

```
# bad
from a_module import foo
import e_module
import b_module
from c_module import c_fn, b_fn

# good
import b_module
import e_module
from a_module import foo
from c_module import b_fn, c_fn
```

- 9.6 Multiline imports should be indented just like multiline list and dictionary literals.

Why? The parentheses follow the same indentation rules as every other bracket or brace block in the style guide, as do the trailing commas.

```
# bad
from a_module import long_name_a, long_name_b, long_name_c,
long_name_d

# good
from a_module import (
    long_name_a,
    long_name_b,
    long_name_c,
    long_name_d,
)
```

[↑ back to top](#)

## Iterators and Generators

(Pending)

[↑ back to top](#)

## Variables

- 11.1 Use UPPERCASE to declare constants, and observe the convention - do not modify them in the program. Python has no **constant** type, so it must be observed manually.

```
# bad
a_constant = 1
os.environ['py_env'] = 'development'

# good
A_CONSTANT = 1
os.environ['PY_ENV'] = 'development'
```

- 11.2 Declare one constant per line.

Why? For clarity, and it's easier to add/remove declarations this way, and with minimal git-diffs. You can also step through each declaration with the debugger, instead of jumping through all of them at once.

```
# bad
FOO, BAR, BAZ = 1, 2, 3

# good
FOO = 1
BAR = 2
BAZ = 3
```

- 11.3 Group all your **CONSTs** and then group all your **vars**.

Why? For clarity and ease of reference. This is also helpful when later on you might need to assign a variable depending on one of the previous assigned variables.

```
# bad
FOO = 1
counter = 0
BAR = 2
length = counter

# good
FOO = 1
BAR = 2

counter = 0
length = counter
```

- 11.4 Assign variables with the minimally sufficient scope at where you need them, but place them in a reasonable place.

#### Why? Prevent variable scope-leak and conflicts

```
# bad - leak to sibling
counter = 0 # mean to count group_a only
for list_a in group_a:
    counter += len(list_a)

for list_b in group_b:
    counter += len(list_b)

# bad - leak into smaller scope
counter = 0 # mean to count within groups
for group in super_group:
    counter += len(group)
    for list in group:
        counter += len(list)

# good
counter_a = 0
for list_a in group_a:
    counter_a += len(list_a)

counter_b = 0
for list_b in group_b:
    counter_b += len(list_b)

# good
group_counter = 0 # mean to count within groups
for group in super_group:
    group_counter += len(group)
    list_counter = 0 # mean to count within lists
    for list in group:
        list_counter += len(list)
```

- 11.5 Prepend underscore `_` when naming variables that are unused. Also a part of PEP8.

#### Why? To be aware of data usage and side effects.

```
# bad
first, unused, last = [1, 2, 3]

# bad - finder is not used though expected to be
for finder, replacer in some_map.items():
    do_something_without_key(replacer)

# bad - lose track of what the first key is
```

```

for _, replacer in some_map.items():
    do_something_without_key(replacer)

# good
first, _unused, last = [1, 2, 3]

# good - we know what the variable is, and it is unused
for _finder, replacer in some_map.items():
    do_something_without_key(replacer)

```

[↑ back to top](#)

## Comparison Operators & Equality

- [12.1](#) Use concise boolean conditionals, refactor long compound statements.

**Why?** Long boolean statements are hard to read and understand.

```

# bad
if (can_move_x() and can_move_y() or is_light() or is_dry() or
has_high_drag()):
    execute_operation_tumbleweed()
else:
    execute_operation_cactus()

# good
can_move = can_move_x() and can_move_y()
movable = is_light() or is_dry() or has_high_drag()
if (can_move or movable):
    execute_operation_tumbleweed()
else:
    execute_operation_cactus()

```

- [12.2](#) Be direct with booleans, avoid unnecessary negations.

**Why?** Negations are harder to understand and longer to write.

```

# bad
if not a_is_legal():
    do_b()
else:
    do_a()

# good
if a_is_legal():
    do_a()
else:
    do_b()

```

- 12.3 Use shortcuts for booleans, but explicit comparisons for strings and numbers.

```

# bad
if is_valid == true:
    #

# good
if is_valid:
    #

# bad
if name:
    #

# good
if name != '':
    #

# bad
if len(a_list):
    #

# good
if len(a_list) > 0:
    #

```

- 12.4 Ternaries should not be nested and generally be single line expressions.

```

# bad
foo = 'bar' if maybe_1 > maybe_2 else 'baz' if value_1 > value_2 else
None

# best
maybe_none = 'baz' if value1 > value2 else None
foo = 'bar' if maybe1 > maybe2 else maybe_none

```

- 12.5 Avoid unneeded ternary statements.

```

# bad
foo = a if a else b
bar = True if c else False
baz = False if c else True

# good
foo = a or b
bar = c
baz = not c

```

[↑ back to top](#)

## Comments

- 13.1 Use `''' ... '''` for multi-line comments.

```
# bad
def make(tag):
    # make() returns a new element
    # based on the passed in tag name
    #
    # @param {string} tag
    # @return {Element} element

    # ...
    return element

# good
def make(tag):
    """
    make() returns a new element
    based on the passed in tag name

    @param {string} tag
    @return {Element} element
    """

    # ...
    return element
```

- 13.2 Use `#` for single line comments. Place single line comments on a newline above the subject of the comment. Put an empty line before the comment unless it's on the first line of a block.

```
# bad
active = True # is current tab

# good
# is current tab
active = True

# bad
def get_type():
    print('fetching type...')
    # set the default type to 'no type'
    type = self.type or 'no type'

    return type

# good
def get_type():
```

```

print('fetching type...')

# set the default type to 'no type'
type = self.type or 'no type'

return type

# also good
def get_type():
    # set the default type to 'no type'
    type = self.type or 'no type'

    return type

```

- 13.3 Start all comments with a space to make it easier to read.

```

# bad
#is current tab
active = True

# good
# is current tab
active = True

```

- 13.4 Prefixing your comments with **TODO** helps yourself and other developers be aware of items to revisit or implement. It keeps the issues visible and easy to find. These are different than regular comments because they are actionable.

```

def complex_calculator():
    compute_basic()

    # TODO figure out improvements to the logic
    return compute_core_logic()

```

[↑ back to top](#)

## Whitespace

- 14.1 Use soft tabs (space character) set to 4 spaces as per PEP8.

```

# bad
def foo():
..return bar

# bad
def foo():
·return bar

```

```
# good
def foo():
    ...return bar
```

- 14.2 Set off operators with spaces.

```
# bad
x=y+5

# good
x = y + 5
```

- 14.3 End files with a single newline character.

```
# bad
import util
# ...
def foo():
    return bar
```

```
# bad
import util
# ...
def foo():
    return bar
↵
```

```
# bad
import util
# ...
def foo():
    return bar
↵
```

- 14.4 Leave a blank line after blocks and before the next statement.

```
# bad
if foo:
    return bar
return baz

# good
if foo:
```

```
    return bar

    return baz

# bad
results = [
    fn_1(),
    fn_2(),
]
return results

# good
results = [
    fn_1(),
    fn_2(),
]
return results
```

- 14.5 Do not pad your blocks with blank lines.

```
# bad
def bar():

    print(foo)

# bad
if (baz):

    print(qux)
else:
    print(foo)

# good
def bar():
    print(foo)

# good
if (baz):
    print(qux)
else:
    print(foo)
```

- 14.6 Do not add spaces inside parentheses.

```
# bad
def bar( foo ):
    return foo
```

```
# good
def bar(foo):
    return foo

# bad
if ( foo and fux ):
    print(foo)

# good
if (foo and fux):
    print(foo)
```

- 14.7 Do not add spaces inside brackets or braces.

```
# bad
foo = [ 1, 2, 3 ]
print(foo[ 0 ])
bar = { 'a': 1 }

# good
foo = [1, 2, 3]
print(foo[0])
bar = {'a': 1}
```

- 14.8 Avoid having lines of code that are longer than 79 characters (including whitespace) as per PEP8.  
Note: per [above](#), long strings are exempt from this rule, and should not be broken up.

**Why?** This ensures readability and maintainability.

```
# bad
foo = nested_object and nested_object.foo and nested_object.foo.bar
and nested_object.foo.bar.baz and nested_object.foo.bar.baz.quux and
nested_object.foo.bar.baz.quux.xyzzy

# bad
http_call({'method': 'POST', 'url': 'https://airbnb.com/', 'data':
{name: 'John', 'age': 20}})

# good
foo = (nested_object and
       nested_object.foo and
       nested_object.foo.bar and
       nested_object.foo.bar.baz and
       nested_object.foo.bar.baz.quux and
       nested_object.foo.bar.baz.quux.xyzzy)

# good
http_call({
    'method': 'POST',
```

```
'url': 'https://airbnb.com/',
'data': {name: 'John', 'age': 20},
})
```

[⬆ back to top](#)

## Commas

- 15.1 Leading commas: **Nope.**

```
# bad
story = [
    once
, upon
, a_time
]

# good
story = [
    once,
    upon,
    a_time,
]

# bad
hero = {
    'first_name': 'Ada'
, 'last_name': 'Lovelace'
, 'birth_year': 1815
, 'super_power': 'computers'
}

# good
hero = {
    'first_name': 'Ada',
    'last_name': 'Lovelace',
    'birth_year': 1815,
    'super_power': 'computers',
}
```

- 15.2 Additional trailing comma: **Yup.**

Why? This leads to cleaner git diffs during code change.

```
# bad - git diff without trailing comma
hero = {
    'first_name': 'Ada',
-     'last_name': 'Lovelace'
+     'last_name': 'Lovelace',
```

```
+     'super_power': 'computers',
}

# good - git diff with trailing comma
hero = {
    'first_name': 'Ada',
    'last_name': 'Lovelace',
+     'super_power': 'computers',
}
```

```
# bad
hero = {
    'first_name': 'Ada',
    'last_name': 'Lovelace'
}

heroes = [
    'Batman',
    'Superman'
]

# good
hero = {
    'first_name': 'Ada',
    'last_name': 'Lovelace',
}

heroes = [
    'Batman',
    'Superman',
]

# bad
def create_hero(
    first_name,
    last_name,
    superpower
):
    # ...

# good
def create_hero(
    first_name,
    last_name,
    superpower,
):
    # ...

# good - note that a comma must not appear after a "spread" element
def create_hero(
    first_name,
```

```
last_name,  
superpower,  
**kwargs  
):  
# ...  
  
# bad  
create_hero(  
    first_name,  
    last_name,  
    superpower  
)  
  
# good  
create_hero(  
    first_name,  
    last_name,  
    superpower,  
)  
  
# good - note that a comma must not appear after a "spread" element  
create_hero(  
    first_name,  
    last_name,  
    superpower,  
    **kwargs  
)
```

[↑ back to top](#)

## Naming Conventions

- **16.1** Use `snake_case` when naming variables, functions, and instances. Use it for file names too as they will be used in imports.

```
# bad  
import myModule  
OBJEcttsssss = {}  
thisIsMyObject = {}  
def thisIsMyFunction():  
  
# good  
import my_module  
objects = {}  
this_is_my_object = {}  
def this_is_my_function():
```

- **16.2** Use `PascalCase` only when naming classes.

```
# bad
class prioritizedMemoryReplay():
    # ...

memory = prioritizedMemoryReplay()

# good
class PrioritizedMemoryReplay():
    # ...

memory = PrioritizedMemoryReplay()
```

- 16.3 Avoid single letter names. Use descriptive and meaningful names - tell what the function does, or what data type an object is. Use `description_object` instead of `object_description`.

```
# bad
def q():
    # ...

# good
def query():
    # ...

# bad - no convention to know what data type it is
df_raw_data = pd.DataFrame(some_data)
id_map_num = {'a': 1, 'b': 2}

# good - convention to tell data type by the last term
raw_data_df = pd.DataFrame(some_data)
id_num_map = {'a': 1, 'b': 2}

# bad - meaningless names, lost context
LIST_1 = ['Jack', 'Alice', 'Emily']
# ... many lines of code later
for item in LIST_1:
    register_human(item)

# good
NAME_LIST = ['Jack', 'Alice', 'Emily']
# ... many lines of code later
for name in NAME_LIST:
    register_human(name)
```

- 16.4 Avoid using close naming to prevent typo and confusion.

```
# bad
objects = ['rock', 'paper', 'scissors']
for object in objects:
```

```

    register_item(object)
close_box(objects)

# okay
objects = ['rock', 'paper', 'scissors']
for obj in objects:
    register_item(obj)
close_box(objects)

# best
object_list = ['rock', 'paper', 'scissors']
for object in object_list:
    register_item(object)
close_box(object_list)

```

- **16.5** Use singular or base words in naming; avoid using plural and instead append singular with the data type.

**Why?** To prevent inconsistencies and second-guesses when using variables. Also, plurals are 1 letter away from a typo, are hard to read, and are ambiguous on the data type.

```

# bad
def moves_object(x, y):
    # ...

# good
def move_object(x, y):
    # ...

# bad - inconsistent naming for same data type and usage
teacher = ['Michael']
students = ['Jack', 'Alice', 'Emily']
books = pd.DataFrame({'title': ['lorem', 'ipsum']})

for t in teacher:
    register_human(t)

for student in students:
    register_human(student)

for book in books:
    register_item(book) # wrong; iterate column name instead of book

# good
teacher_list = ['Michael']
student_list = ['Jack', 'Alice', 'Emily']
book_df = pd.DataFrame({'title': ['lorem', 'ipsum']})

for teacher in teacher_list:
    register_human(teacher)

```

```

for student in student_list:
    register_human(student)

# naming as df suggests it shall be treated as a dataframe
for _idx, book in book_df.iterrow():
    register_item(book)

```

- 16.6 Use singular naming for modules and source files.

```

# bad
from commons import utils

utils.read_string()

# good
from common import util

util.read_string()

```

- 16.7 Use abbreviations if they are clear and make for more readable and writable code.

**Why?** Names are for humans, so always make code readable and easy to spell.

```

# bad
flight_prerequisites_checklist = ['landing gear', 'engine', 'flaps']
initialize_flight(flight_prerequisites_checklist)

# good
flight_prereq_checklist = ['landing gear', 'engine', 'flaps']
init_flight(flight_prereq_checklist)

```

- 16.8 Use simple, concise names over long, explicit ones.

**Why?** Names are for humans to read, and should make the code clean.

```

# bad - explicit Java-style naming clutters code and harms readability
poscode_to_city_name_map = {11223: 'brooklyn'}
poscode_to_city_name_to_state_name_map =
map_city_to_state(poscode_to_city_name_map)
poscode_to_city_name_to_state_name_to_country_map = {}

# good - understandable and fast to read
poscode_city_map = {11223: 'brooklyn'}
poscode_state_map = map_city_to_state(poscode_city_map)
poscode_country_map = {}

```

[↑ back to top](#)

# React/JSX Style Guide

---

## Basic Rules

- Always use JSX syntax.
- Do not use `React.createElement` unless you're initializing the app from a file that is not JSX.

## Class vs `React.createClass` vs stateless

- If you have internal state and/or refs, prefer `class extends React.Component` over `React.createClass`.  
eslint: `react/prefer-es6-class` `react/prefer-stateless-function`

```
// bad
const Listing = React.createClass({
  // ...
  render() {
    return <div>{this.state.hello}</div>;
  },
});

// good
class Listing extends React.Component {
  // ...
  render() {
    return <div>{this.state.hello}</div>;
  }
}
```

And if you don't have state or refs, prefer normal functions (not arrow functions) over classes:

```
// bad
class Listing extends React.Component {
  render() {
    return <div>{this.props.hello}</div>;
  }
}

// bad (relying on function name inference is discouraged)
const Listing = ({ hello }) => <div>{hello}</div>;

// good
function Listing({ hello }) {
  return <div>{hello}</div>;
}
```

## Mixins

- **Do not use mixins.**

Why? Mixins introduce implicit dependencies, cause name clashes, and cause snowballing complexity. Most use cases for mixins can be accomplished in better ways via components, higher-order components, or utility modules.

## Naming

- **Extensions:** Use `.jsx` extension for React components. eslint: `react/jsx-filename-extension`
- **Filename:** Use PascalCase for filenames. E.g., `ReservationCard.jsx`.
- **Reference Naming:** Use PascalCase for React components and camelCase for their instances. eslint: `react/jsx-pascal-case`

```
// bad
import reservationCard from "./ReservationCard";

// good
import ReservationCard from "./ReservationCard";

// bad
const ReservationItem = <ReservationCard />

// good
const reservationItem = <ReservationCard />;
```

- **Component Naming:** Use the filename as the component name. For example, `ReservationCard.jsx` should have a reference name of `ReservationCard`. However, for root components of a directory, use `index.jsx` as the filename and use the directory name as the component name:

```
// bad
import Footer from "./Footer/Footer";

// bad
import Footer from "./Footer/index";

// good
import Footer from "./Footer";
```

- **Higher-order Component Naming:** Use a composite of the higher-order component's name and the passed-in component's name as the `displayName` on the generated component. For example, the higher-order component `withFoo()`, when passed a component `Bar` should produce a component with a `displayName` of `withFoo(Bar)`.

Why? A component's `displayName` may be used by developer tools or in error messages, and having a value that clearly expresses this relationship helps people understand what is happening.

```
// bad
export default function withFoo(WrappedComponent) {
  return function WithFoo(props) {
    return <WrappedComponent {...props} foo />;
  }
}

// good
export default function withFoo(WrappedComponent) {
  function WithFoo(props) {
    return <WrappedComponent {...props} foo />;
  }

  const wrappedComponentName = WrappedComponent.displayName
    || WrappedComponent.name
    || 'Component';

  WithFoo.displayName = `withFoo(${wrappedComponentName})`;
  return WithFoo;
}
```

- **Props Naming:** Avoid using DOM component prop names for different purposes.

Why? People expect props like `style` and `className` to mean one specific thing. Varying this API for a subset of your app makes the code less readable and less maintainable, and may cause bugs.

```
// bad
<MyComponent style="fancy" />

// bad
<MyComponent className="fancy" />

// good
<MyComponent variant="fancy" />
```

## Declaration

- Do not use `displayName` for naming components. Instead, name the component by reference.

```
// bad
export default React.createClass({
  displayName: 'ReservationCard',
```

```
// stuff goes here
});

// good
export default class ReservationCard extends React.Component {
```

## Alignment

- Follow these alignment styles for JSX syntax. eslint: react/jsx-closing-bracket-location  
react/jsx-closing-tag-location

```
// bad
<Foo superLongParam="bar"
      anotherSuperLongParam="baz" />

// good
<Foo
  superLongParam="bar"
  anotherSuperLongParam="baz"
/>

// if props fit in one line then keep it on the same line
<Foo bar="bar" />

// children get indented normally
<Foo
  superLongParam="bar"
  anotherSuperLongParam="baz"
>
  <Quux />
</Foo>

// bad
{showButton &&
  <Button />
}

// bad
{
  showButton &&
  <Button />
}

// good
{showButton && (
  <Button />
)}

// good
{showButton && <Button />}
```

```
// good
{someReallyLongConditional
  && anotherLongConditional
  && (
    <Foo
      superLongParam="bar"
      anotherSuperLongParam="baz"
    />
  )
}

// good
{someConditional ? (
  <Foo />
) : (
  <Foo
    superLongParam="bar"
    anotherSuperLongParam="baz"
  />
)}
```

## Quotes

- Always use double quotes ("") for JSX attributes, but single quotes ('') for all other JS. eslint: `jsx-quotes`

Why? Regular HTML attributes also typically use double quotes instead of single, so JSX attributes mirror this convention.

```
// bad
<Foo bar='bar' />

// good
<Foo bar="bar" />

// bad
<Foo style={{ left: "20px" }} />

// good
<Foo style={{ left: '20px' }} />
```

## Spacing

- Always include a single space in your self-closing tag. eslint: `no-multi-spaces, react/jsx-tag-spacing`

```
// bad
<Foo/>

// very bad
<Foo           />

// bad
<Foo
  />

// good
<Foo />
```

- Do not pad JSX curly braces with spaces. eslint: `react/jsx-curly-spacing`

```
// bad
<Foo bar={ baz } />

// good
<Foo bar={baz} />
```

## Props

- Always use camelCase for prop names, or PascalCase if the prop value is a React component.

```
// bad
<Foo
  UserName="hello"
  phone_number={12345678}
/>

// good
<Foo
  userName="hello"
  phoneNumber={12345678}
  Component={SomeComponent}
/>
```

- Omit the value of the prop when it is explicitly `true`. eslint: `react/jsx-boolean-value`

```
// bad
<Foo
  hidden={true}
/>

// good
```

```

<Foo
  hidden
/>

// good
<Foo hidden />

```

- Always include an `alt` prop on `<img>` tags. If the image is presentational, `alt` can be an empty string or the `<img>` must have `role="presentation"`. eslint: `jsx-a11y/alt-text`

```

// bad


// good


// good


// good


```

- Do not use words like "image", "photo", or "picture" in `<img>` `alt` props. eslint: `jsx-a11y/img-redundant-alt`

**Why?** Screenreaders already announce `img` elements as images, so there is no need to include this information in the alt text.

```

// bad


// good


```

- Use only valid, non-abstract ARIA roles. eslint: `jsx-a11y/aria-role`

```

// bad - not an ARIA role
<div role="datepicker" />

// bad - abstract ARIA role
<div role="range" />

// good
<div role="button" />

```

- Do not use `accessKey` on elements. eslint: `jsx-a11y/no-access-key`

Why? Inconsistencies between keyboard shortcuts and keyboard commands used by people using screenreaders and keyboards complicate accessibility.

```
// bad
<div accessKey="h" />

// good
<div />
```

- Avoid using an array index as `key` prop, prefer a stable ID. eslint: `react/no-array-index-key`

Why? Not using a stable ID is an anti-pattern because it can negatively impact performance and cause issues with component state.

We don't recommend using indexes for keys if the order of items may change.

```
// bad
{
  todos.map((todo, index) => <Todo {...todo} key={index} />);
}

// good
{
  todos.map((todo) => <Todo {...todo} key={todo.id} />);
}
```

- Always define explicit defaultProps for all non-required props.

Why? propTypes are a form of documentation, and providing defaultProps means the reader of your code doesn't have to assume as much. In addition, it can mean that your code can omit certain type checks.

```
// bad
function SFC({ foo, bar, children }) {
  return (
    <div>
      {foo}
      {bar}
      {children}
    </div>
  );
}

SFC.propTypes = {
  foo: PropTypes.number.isRequired,
  bar: PropTypes.string,
  children: PropTypes.node,
};
```

```
// good
function SFC({ foo, bar, children }) {
  return (
    <div>
      {foo}
      {bar}
      {children}
    </div>
  );
}
SFC.propTypes = {
  foo: PropTypes.number.isRequired,
  bar: PropTypes.string,
  children: PropTypes.node,
};
SFC.defaultProps = {
  bar: '',
  children: null,
};
```

- Use spread props sparingly.

**Why?** Otherwise you're more likely to pass unnecessary props down to components. And for React v15.6.1 and older, you could [pass invalid HTML attributes to the DOM](#).

Exceptions:

- HOCs that proxy down props and hoist propTypes

```
function HOC(WrappedComponent) {
  return class Proxy extends React.Component {
    Proxy.propTypes = {
      text: PropTypes.string,
      isLoading: PropTypes.bool
    };

    render() {
      return <WrappedComponent {...this.props} />
    }
  }
}
```

- Spreading objects with known, explicit props. This can be particularly useful when testing React components with Mocha's beforeEach construct.

```
export default function Foo {
  const props = {
    text: '',
    isPublished: false
```

```

    }

    return (<div {...props} />);
}

```

Notes for use: Filter out unnecessary props when possible. Also, use [prop-types-exact](#) to help prevent bugs.

```

// bad
render() {
  const { irrelevantProp, ...relevantProps } = this.props;
  return <wrappedComponent {...this.props} />
}

// good
render() {
  const { irrelevantProp, ...relevantProps } = this.props;
  return <wrappedComponent {...relevantProps} />
}

```

## Refs

- Always use ref callbacks. eslint: [react/no-string-refs](#)

```

// bad
<Foo
  ref="myRef"
/>

// good
<Foo
  ref={(ref) => { this.myRef = ref; }}
/>

```

## Parentheses

- Wrap JSX tags in parentheses when they span more than one line. eslint: [react/jsx-wrap-multilines](#)

```

// bad
render() {
  return <MyComponent variant="long body" foo="bar">
    <MyChild />
  </MyComponent>;
}

// good
render() {

```

```

        return (
          <MyComponent variant="long body" foo="bar">
            <MyChild />
          </MyComponent>
        );
      }

      // good, when single line
      render() {
        const body = <div>hello</div>;
        return <MyComponent>{body}</MyComponent>;
      }
    
```

## Tags

- Always self-close tags that have no children. eslint: `react/self-closing-comp`

```

// bad
<Foo variant="stuff"></Foo>

// good
<Foo variant="stuff" />

```

- If your component has multiline properties, close its tag on a new line. eslint: `react/jsx-closing-bracket-location`

```

// bad
<Foo
  bar="bar"
  baz="baz" />

// good
<Foo
  bar="bar"
  baz="baz"
/>

```

## Methods

- Use arrow functions to close over local variables. It is handy when you need to pass additional data to an event handler. Although, make sure they **do not massively hurt performance**, in particular when passed to custom components that might be PureComponents, because they will trigger a possibly needless rerender every time.

```

function ItemList(props) {
  return (

```

```
<ul>
  {props.items.map((item, index) => (
    <Item
      key={item.key}
      onClick={(event) => {
        doSomethingWith(event, item.name, index);
      }}
    />
  ))}
</ul>
);
}
```

- Bind event handlers for the render method in the constructor. eslint: react/jsx-no-bind

Why? A bind call in the render path creates a brand new function on every single render. Do not use arrow functions in class fields, because it makes them challenging to test and debug, and can negatively impact performance, and because conceptually, class fields are for data, not logic.

```
// bad
class extends React.Component {
  onClickDiv() {
    // do stuff
  }

  render() {
    return <div onClick={this.onClickDiv.bind(this)} />;
  }
}

// very bad
class extends React.Component {
  onClickDiv = () => {
    // do stuff
  }

  render() {
    return <div onClick={this.onClickDiv} />
  }
}

// good
class extends React.Component {
  constructor(props) {
    super(props);

    this.onClickDiv = this.onClickDiv.bind(this);
  }

  onClickDiv() {
```

```
// do stuff
}

render() {
  return <div onClick={this.onClickDiv} />;
}
}
```

- Do not use underscore prefix for internal methods of a React component.

Why? Underscore prefixes are sometimes used as a convention in other languages to denote privacy. But, unlike those languages, there is no native support for privacy in JavaScript, everything is public. Regardless of your intentions, adding underscore prefixes to your properties does not actually make them private, and any property (underscore-prefixed or not) should be treated as being public. See issues #1024, and #490 for a more in-depth discussion.

```
// bad
React.createClass({
  _onClickSubmit() {
    // do stuff
  },

  // other stuff
});

// good
class extends React.Component {
  onClickSubmit() {
    // do stuff
  }

  // other stuff
}
```

- Be sure to return a value in your `render` methods. eslint: `react/require-render-return`

```
// bad
render() {
  (<div />);
}

// good
render() {
  return (<div />);
}
```

# Ordering

- Ordering for `class extends React.Component`:
  1. optional `static` methods
  2. `constructor`
  3. `getChildContext`
  4. `componentWillMount`
  5. `componentDidMount`
  6. `componentWillReceiveProps`
  7. `shouldComponentUpdate`
  8. `componentWillUpdate`
  9. `componentDidUpdate`
  10. `componentWillUnmount`
  11. event handlers starting with 'handle' like `handleSubmit()` or `handleChangeDescription()`
  12. event handlers starting with 'on' like `onClickSubmit()` or `onChangeDescription()`
  13. getter methods for `render` like `getSelectReason()` or `getFooterContent()`
  14. optional render methods like `renderNavigation()` or `renderProfilePicture()`
  15. `render`
- How to define `propTypes`, `defaultProps`, `contextTypes`, etc...

```
import React from "react";
import PropTypes from "prop-types";

const propTypes = {
  id: PropTypes.number.isRequired,
  url: PropTypes.string.isRequired,
  text: PropTypes.string,
};

const defaultProps = {
  text: "Hello World",
};

class Link extends React.Component {
  static methodsAreOk() {
    return true;
  }

  render() {
    return (
      <a href={this.props.url} data-id={this.props.id}>
        {this.props.text}
      </a>
    );
  }
}

Link.propTypes = propTypes;
```

```
Link.defaultProps = defaultProps;

export default Link;
```

- Ordering for `React.createClass`: eslint: `react/sort-comp`

1. `displayName`
2. `propTypes`
3. `contextTypes`
4. `childContextTypes`
5. `mixins`
6. `statics`
7. `defaultProps`
8. `get defaultProps`
9. `getInitialState`
10. `getChildContext`
11. `componentWillMount`
12. `componentDidMount`
13. `componentWillReceiveProps`
14. `shouldComponentUpdate`
15. `componentWillUpdate`
16. `componentDidUpdate`
17. `componentWillUnmount`
18. *clickHandlers or eventHandlers* like `onClickSubmit()` or `onChangeDescription()`
19. *getter methods for render* like `getSelectReason()` or `getFooterContent()`
20. *optional render methods* like `renderNavigation()` or `renderProfilePicture()`
21. `render`

## isMounted

- Do not use `isMounted`. eslint: `react/no-is-mounted`

Why? `isMounted` is an anti-pattern, is not available when using ES6 classes, and is on its way to being officially deprecated.

[↑ back to top](#)



## CAPSTONE PROJECT 2

CMU-SE-451 / CMU-IS-451 / CMU-CS-451

### TECHNOLOGIES STACK DOCUMENT

Version 2.0

Date: 1 - Mar - 2021

### EXPERT-DRIVEN SMART DASHBOARD APPLICATION

#### Submitted by

Vo Van Hoa  
Pham Van Tin  
Ky Huu Dong  
Tran Thi Thanh Kieu

#### Approved by

##### Capstone Project 2 - Mentor:

Name

Signature

Date

Binh, Thanh Nguyen \_\_\_\_\_

A handwritten signature in blue ink, appearing to read 'Binh, Thanh Nguyen'.

\_\_\_\_\_ 26 - May - 2020

PROJECT INFORMATION			
<b>Project Acronym</b>	EDSDA		
<b>Project Title</b>	Expert-Driven Smart Dashboard Application		
<b>Project Web URL</b>	<a href="https://sda-research.ml/">https://sda-research.ml/</a>		
<b>Start Date</b>	01 - Mar - 2021		
<b>End Date:</b>	02 - Jun - 2021		
<b>Lead Institution</b>	International School, Duy Tan University		
<b>Project Mentor</b>	Ph.D Binh, Thanh Nguyen		
<b>Scrum Master</b>	Hoa, Vo	hoavo.dng@gmail.com	0935.193.182
<b>Team Members</b>	Tin, Pham Van	tinphamvan123@gmail.com	0932.535.175
	Dong, Ky Huu	kyhuudong@gmail.com	0898.246.980
	Kieu, Tran Thi Thanh	thanhkieutran391@gmail.com	0358.583.251

DOCUMENT INFORMATION			
<b>Document Title</b>	Technologies Stack		
<b>Author(s)</b>	Team C2SE.06		
<b>Role</b>	[EDSDA] TechonologiesStack_v2.0		
<b>Date</b>	01 - Mar - 2021	Filename	[EDSDA] 012 Technologies Stack
<b>URL</b>	<a href="https://github.com/sdateamdtu2020/SDA-v2.0">https://github.com/sdateamdtu2020/SDA-v2.0</a>		
<b>Access</b>	Project and CMU Program		

## REVISION HISTORY

# TABLE OF CONTENTS

PROJECT INFORMATION	1
DOCUMENT INFORMATION	1
REVISION HISTORY	2
<b>TABLE OF CONTENTS</b>	<b>3</b>
<b>1. INTRODUCTION</b>	<b>5</b>
1.1. PURPOSE	5
1.2. DOCUMENT OBJECTIVES	5
1.3. INTENDED AUDIENCE	5
1.4. ACRONYMS AND ABBREVIATIONS	5
<b>2. MAIN DECISIONS TO CHOOSE TECHNOLOGIES</b>	<b>5</b>
<b>3. PROGRAMMING LANGUAGE</b>	<b>6</b>
3.1. JAVASCRIPT	6
3.2. PYTHON	6
3.3. HTML/CSS	6
3.4. SQL	6
<b>4. TECHNOLOGIES OF WEB APPLICATIONS</b>	<b>6</b>
4.1. REACT	6
4.2. REDUX	7
4.3. RESTful API	7
4.4. ExpressJS	7
<b>5. Database / Data Cubes....</b>	<b>8</b>
5.1. PostgreSQL	8
5.2. PgAdmin4	8
5.3. RML.io	8
5.4. GraphDB	8
<b>6. DEVELOPMENT STACK</b>	<b>8</b>
6.1. VISUAL STUDIO CODE	8
6.2. ESLINT	9
6.3. PRETTIER	9
6.4. REACT DEVTOOLS AND REDUX DEVTOOLS	9
<b>7. VERSION CONTROL</b>	<b>9</b>

7.1. Bitbucket	9
<b>8. DEPLOYMENT</b>	<b>9</b>
8.1. FIREBASE	9
8.2. VIRTUAL MACHINE	10
8.3. Apache Airflow	10

## 1. INTRODUCTION

### 1.1. PURPOSE OF DOCUMENT

The purpose of the Technology Stack Document is to provide a brief, yet clear information about all the technologies that make the project.

### 1.2. DOCUMENT OBJECTIVES

The Technology Stack has the following objectives:

- To describe the technologies that we are using to build the project.

### 1.3. INTENDED AUDIENCE

This document is intended for the following audiences.

### 1.4. ACRONYMS AND ABBREVIATIONS

Acronym/Abbreviation	Meaning
<b>EDSDA</b>	<b>Expert-Driven Smart Dashboard Application</b>

## 2. MAIN DECISIONS TO CHOOSE TECHNOLOGIES

We have some key decisions to choose our technologies stack:

- Open source technologies: We would like to choose all the frameworks, tech stacks that are licensed under MIT license so that we can have more support from the community
- New but proven technologies: We use React, Redux as well as NodeJS for our project because they are new, so that we can really learn new things by building our project. Also, they've been used in production in many projects by the community so that we know that they are safe.
- Document rich: We want to have a framework with a good and well organised document so that we save our time reading the docs.

## 3. PROGRAMMING LANGUAGE

### 3.1. JAVASCRIPT

JavaScript often abbreviated as JS, is a high-level, dynamic, weakly typed, prototype-based, multi-paradigm, and interpreted programming language. It's the most popular language nowadays and can be used everywhere. Javascript is so popular that we can easily find a solution for our problem on Google and Stackoverflow.

### 3.2. PYTHON

Python is an elegant, versatile language with an ecosystem of powerful modules and code libraries. There are over a hundred tools that act as a framework, libraries, or software for ETL. These tools become your go-to source once you start dealing with complex schemas and massive amounts of data. Therefore, we use Python to make the ETL process become easier to program and more effective.

### 3.3. HTML/CSS

HTML (the Hypertext Markup Language) and CSS (Cascading Style Sheets) are two of the core technologies for building Web pages. HTML provides the structure of the page, CSS the (visual and aural) layout, for a variety of devices. Along with graphics and scripting, HTML and CSS are the basis of building Web pages and Web Applications.

### 3.4. SQL

HTML (the Hypertext Markup Language) and CSS (Cascading Style Sheets) are two of the core technologies for building Web pages. HTML provides the structure of the page, CSS the (visual and aural) layout, for a variety of devices. Along with graphics and scripting, HTML and CSS are the basis of building Web pages and Web Applications.

## 4. TECHNOLOGIES OF WEB APPLICATIONS

### 4.1. REACT

Home page: <https://reactjs.org/>

React is the current industry standard that offers a lot of out of the box benefits. It is fast, efficient, and scalable. Due to the large community, finding solutions to potential problems



and reference material is much easier, even for a potential dev without a lot of experience who would like to contribute to the Main Course.

### Pros of React

#### The community

Like with most online developer communities, the React one is growing and offers a great network of experienced developers.

#### Faster development

Speed is often the name of the game where development is concerned. Anything that can speed up the development of an application is much appreciated!

With React, the development time is considerably shorter.

## 4.2. REDUX

Home page: <https://redux.js.org/>

Redux is a predictable state container for JavaScript apps.

It helps you write applications that behave consistently, run in different environments (client, server, and native), and are easy to test. On top of that, it provides a great developer experience, such as live code editing combined with a time traveling debugger.

You can use Redux together with React, or with any other view library.

## 4.3. RESTful API

RESTful API is an architectural style for an application program interface (API) that uses HTTP requests to access and use data. We build APIs based on RESTful principles for easy data accessing and interacting between different levels of the system.

## 4.4. ExpressJS

A prebuilt NodeJS framework that makes creating server side applications simple, fast, and flexible. NodeJS is powered by Google's V8 Engine which means it's powerful and can handle a large number of requests without lapsing in dependability. Also, this means that this is a highly scalable choice when you consider the Event Loop which manages all asynchronous operations allowing the program to continue to run as expected without stops.



## 5. Database / Data Cubes

### 5.1. PostgreSQL

PostgreSQL is one of the most well developed and mature relational database management systems, with its development, ongoing for more than 20 years now, from supporting small websites on single machines to data warehouses and large scale concurrent setups.

### 5.2. PgAdmin 4

pgAdmin is the most popular and feature rich Open Source administration and development platform for PostgreSQL, the most advanced Open Source database in the world.

### 5.3. RML.io

To execute the process of generating RDF Data Cubes from Mapping Rules and data sources, we use the RMLMapper and RMLStreamer frameworks provided by the RML.io organization. They are platforms that help users produce high quality linked data and knowledge graphs that are fast and easy to install.

### 5.4. GraphDB

Enterprise RDF and graph database with efficient reasoning, cluster and external index synchronization support.

## 6. DEVELOPMENT STACK

### 6.1. VISUAL STUDIO CODE

Home page: <http://code.visualstudio.com/>

Visual Studio Code (Vscode) is a great open source code editor developed by Microsoft, it provides a rich and easy to use user interface as well as thousands of extensions that we can install to suit our needs.



## 6.2. ESLINT

Eslint is a must have tool to lint our project, so that we can provide a consistent code style to work in our team.

## 6.3. PRETTIER

Prettier, used with Eslint, helps us to auto format our code on save.

## 6.4. REACT DEVTOOLS AND REDUX DEVTOOLS

React and Redux are being used heavily in our project. That's why the devtools will help us debug our code easier.

# 7. VERSION CONTROL

## 7.1. Bit Bucket

Home page: <https://bitbucket.org>

Bitbucket is a repository hosting service. Think of it as the "cloud" for code.

The main benefit of Bitbucket is storing the code in one and secure place, leaving it open for all teammates, so they can without any obstruction get access to the code or parts of it, or pull out the code based on their work. Also you can easily track all changes and rollback if needed. Actually, you can store not only code but also the scripts and even the server configuration. Due to the simple and straight-forward web interface, you can navigate and search for what you need without any effort.

# 8. DEPLOYMENT

## 8.1. FIREBASE

Home page: <http://firebase.google.com/>

We use Firebase to deploy hosting

Firebase, offered by Google, is a suite of cloud computing services that runs on the same infrastructure that Google uses internally for its end-user products, such as Google Search and YouTube. Alongside a set of management tools, it provides a series of modular cloud services including computing, data storage, data analytics and machine learning.

Firebase provides a rich set of tools, configs and tutorials that can help us quickly deploy our application in just a matter of minutes.

## 8.2. Virtual Machine

Linux VPS is a cost methodology in which there is a physical server, although several completely independent virtual servers run from each other. We use Linux VPS to configure server configurations in the cloud, as well as a cloud environment for deploying and launching APIs, thereby making it easier for users to interact with the system and database.

## 8.3. Apache Airflow

Apache Airflow is an open-source workflow management platform. The solution is to manage the project's increasingly complex workflows. Creating Airflow allowed programmatically author and schedule the workflows and monitor works via the built-in Airflow user interface.

Airflow is written in Python, and workflows are created via Python scripts. Airflow is designed under the principle of "configuration as code". While other "configuration as code" workflow platforms exist using markup languages like XML, using Python allows developers to import libraries and classes to help them create their workflows.





International School

## CAPSTONE PROJECT 2

CMU-SE-451 / CMU-IS-451 / CMU-CS-451

### TEAM MEETING

Version 2.0

Date: 1 - Mar - 2021

## EXPERT-DRIVEN SMART DASHBOARD APPLICATION

### Submitted by

Vo Van Hoa  
Pham Van Tin  
Ky Huu Dong  
Tran Thi Thanh Kieu

### Approved by

#### Capstone Project 2 - Mentor:

Name

Signature

Date

Binh, Thanh Nguyen \_\_\_\_\_

A handwritten blue ink signature of the name "Binh, Thanh Nguyen".

\_\_\_\_\_ 26 - May - 2020

PROJECT INFORMATION			
<b>Project Acronym</b>	EDSDA		
<b>Project Title</b>	Expert-Driven Smart Dashboard Application		
<b>Project Web URL</b>	<a href="https://sda-research.ml/">https://sda-research.ml/</a>		
<b>Start Date</b>	01 - Mar - 2021		
<b>End Date:</b>	02 - Jun - 2021		
<b>Lead Institution</b>	International School, Duy Tan University		
<b>Project Mentor</b>	Ph.D Binh, Thanh Nguyen		
<b>Scrum Master</b>	Hoa, Vo	hoavo.dng@gmail.com	0935.193.182
<b>Team Members</b>	Tin, Pham Van	tinphamvan123@gmail.com	0932.535.175
	Dong, Ky Huu	kyhuudong@gmail.com	0898.246.980
	Kieu, Tran Thi Thanh	thanhkietran391@gmail.com	0358.583.251

DOCUMENT INFORMATION			
<b>Document Title</b>	Team Meeting		
<b>Author(s)</b>	Team C2SE.06		
<b>Role</b>	[EDSDA] Team Meeting v2.0		
<b>Date</b>	01 - Mar - 2021	Filename	[EDSDA] 012 Team Meeting
<b>URL</b>	<a href="https://github.com/sdateamdtu2020/SDA-v2.0">https://github.com/sdateamdtu2020/SDA-v2.0</a>		
<b>Access</b>	Project and CMU Program		

# Integration 1 (24 - 31/Jan/21)

---

## Agenda:

- Review knowledge of each member
- Write Scientific Research Report
- Planning Architecture
- Planning Proposal

## Meeting Content:

- Define UI
- Define Adding Description content method
- Define data warehouse overview
- Define Import UI
- Discussing data source & crawling method
- Discussing metadata

## Post Meeting Action Items

### Hoa

#### This week:

- Design Data Warehouse
- Continue researching crawling & metadata
  - Sync up with Dong
- Continue researching data cube definition
- Research Automatic Detecting Dim & Mea
  - Sync up with Dong
- Write Scientific Research Report

### Dong

#### This week:

- Write Scientific Research Report
- Continue researching crawling & metadata
  - Sync up with Hoa
- Research Automatic Detecting Dim & Mea
  - Sync up with Hoa
- Design Data Warehouse

### Tin

#### This week:

- Write Scientific Research Report

- Design UI
- Instruct Kieu to learn ReactJs
  - Sync up with Kieu

## Kieu

This week:

- Write Scientific Research Report
- Continue learning ReactJS
  - Sync up with Tin

# Integration 2 (31/Jan - 7/Feb/21)

---

Agenda:

- Review task of each member on the meeting before
- Update content of Scientific Research Report after feedback from Mr.Binh

Meeting Content:

- add research field: automation ETL, automation RDF Data cubes generation.
- review caps1 architecture.
- add data source.
- Modify DW Design.
- Modify UI Design, prototype
- Review what Kieu has learned last week.

## Post Meeting Action Items

### Hoa

Last week:

- Design Data Warehouse
- Continue researching crawling & metadata
- Continue researching data cube definition
- Research Automatic Detecting Dim & Mea
- Write Scientific Research Report

This week:

- Continue researching crawling & metadata
  - Sync up with Dong
- Continue researching data cube definition
- Research automation RDF Data Cube Generation
- Update content Scientific Research Report

### Dong

Last week:

- Write Scientific Research Report
- Continue researching crawling & metadata
- Research Automatic Detecting Dim & Mea
- Design Data Warehouse

This week:

- Design Data Warehouse
- Continue researching crawling & metadata
  - Sync up with Hoa
- Research automation ETL

## Tin

Last week:

- Write Scientific Research Report
- Design UI
- Instruct Kieu to learn ReactJS

This week:

- Continue Design UI, Prototype
- Continue Instruct Kieu to learn ReactJS
  - Sync up with Kieu

## Kieu

Last week:

- Write Scientific Research Report
- Continue learning ReactJS

This week:

- Continue learning ReactJS

# Integration 3 (6/Feb - 21/Feb/21)

---

Agenda:

- Review task of each member on the meeting before

Meeting Content:

- Discuss on Module View Server
- Discuss on User Case
- Discuss on Context Diagram

## Post Meeting Action Items

### Hoa

Last week:

- Continue researching crawling & metadata
  - Sync up with Dong
- Continue researching data cube definition
- Research automation RDF Data Cube Generation
- Update content Scientific Research Report

This week:

- Write User Case
  - Sync up with Dong
- Continue researching data cube definition
- Research automation RDF Data Cube Generation
- Continue researching crawling & metadata
  - Sync up with Dong

### Dong

Last week:

- Design Data Warehouse
- Continue researching crawling & metadata
  - Sync up with Hoa
- Research automation ETL

This week:

- Write User Case
  - Sync up with Hoa
- Design Data Warehouse
- Continue researching crawling & metadata
  - Sync up with Hoa
- Research automation ETL

### Tin

Last week:

- Continue Design UI, Prototype
- Continue Instruct Kieu to learn ReactJS
  - Sync up with Kieu

This week:

- Draw Module View Server

**Kieu**

Last week:

- Continue learning ReactJS
- 8 / 23 PROJECTS REACT

This week:

- Continue learning ReactJS
  - FINISH 23 projects
  - FINISH 2 PROJECT COVID-19

## Integration 4 (22/Feb - 08/Mar/21)

---

Agenda:

- Review task of each member on the meeting before

Meeting Content:

- Discuss on Module View Server
- Discuss on User Case
- Discuss on auto ETL process

Post Meeting Action Items

**Hoa**

Last week:

- Write User Case
  - Sync up with Dong
- Continue researching data cube definition
- Research automation RDF Data Cube Generation
- Continue researching crawling & metadata
  - Sync up with Dong

This week:

- Research Data Lake
  - Sync up with Dong
- Research customize [RML.io](#) library
- Continue researching data cube definition
- Continue researching crawling & metadata
  - Sync up with Dong

**Dong**

Last week:

- Write User Case
  - Sync up with Hoa
- Design Data Warehouse
- Continue researching crawling & metadata
  - Sync up with Hoa
- Research automation ETL

This week:

- Choose auto ETL platform
- Research Data Lake
  - Sync up with Hoa

## Tin

Last week:

- Draw Module View Server
- Continue Instruct Kieu to learn ReactJS
  - Sync up with Kieu

This week:

- Continue Instruct Kieu to learn ReactJS
  - Sync up with Kieu
- Write User Case document

## Kieu

Last week:

- Continue learning ReactJS
  - FINISH 15/23 projects
  - FINISH 1 PROJECT COVID-19

This week:

- Continue learning ReactJS

# Integration 5 (01-Mar - 07/Mar/21)

---

Agenda:

- Review task of each member on the meeting before
- Planning meeting
- Sprint Retrospective
- Discuss on Proposal document

## Meeting Content:

- Review Proposal document
- Review project timeline
- Review UI Design

## Post Meeting Action Items

### Hoa

Last week:

- Research Data Lake
  - Sync up with Dong
- Research customize [RML.io](#) library
- Continue researching data cube definition
- Continue researching crawling & metadata
  - Sync up with Dong

This week:

- Adapt new requirement for UI
- Modify proposal document

### Dong

Last week:

- Choose auto ETL platform
- Research Data Lake
  - Sync up with Hoa

This week:

- Write Proposal
- Choose auto ETL platform
- Research Data Lake
  - Sync up with Hoa

### Tin

Last week:

- Continue Instruct Kieu to learn ReactJS
  - Sync up with Kieu
- Write User Case document

This week:

- Write Proposal
- Design UI

## Kieu

Last week:

- Continue learning ReactJS

This week:

- Write Proposal
- Continue learning ReactJS

# Integration 6 (08-Mar - 14/Mar/21)

---

Agenda:

- Review task of each member on the meeting before
- Modify proposal document
- Modify requirements product document

Meeting Content:

- Modify proposal document
- Modify requirements product document
- Prepare for mentor meeting on Mon, 08 - Mar - 21

Post Meeting Action Items

## Hoa

Last week:

- Adapt new requirement for UI
- Modify proposal document

This week:

- Initiate RML process
- Initiate MIRROR
- Initiate GraphDB Database

## Dong

Last week:

- Write Proposal
- Choose auto ETL platform
- Research Data Lake
  - Sync up with Hoa

This week:

- Research Data Lake
  - Sync up with Hoa
- Crawl data
- Clean data

## Tin

Last week:

- Write Proposal
- Design UI

This week:

- Design UI
- Initial & config for React project

## Kieu

Last week:

- Write Proposal
- Continue learning ReactJS

This week:

- Continue learning ReactJS

# Integration 7 (15-Mar - 21/Mar/21)

---

Agenda:

- Review task of each member on the meeting before
- Sprint Retrospective

Meeting Content:

- Sprint Retrospective
- Prepare for Scientific Research Report on Sat (20-Mar)
- Planning for Sprint 2

## Sprint Retrospective (Sprint 1)

**MAD - What made us mad?** 😡

- aaaaa

**SAD - What made us sad?** 😢

- aaaa

## GLAD - What made us glad? 😎

- aaaa

## Post Meeting Action Items

### Hoa

Last week:

- Initiate RML process
- Initiate MIRROR
- Initiate GraphDB Database

This week:

- Prepare Presentation for Scientific Research Report
- Research Architecture of Data Lake
  - Sync up with Dong
- Setup server temporary

### Dong

Last week:

- Write Proposal
- Choose auto ETL platform
- Research Data Lake
  - Sync up with Hoa

This week:

- Prepare Presentation for Scientific Research Report
- Choose auto ETL platform
- Research Architecture of Data Lake
  - Sync up with Hoa

### Tin

Last week:

- Write Proposal
- Design UI

This week:

- Design UI
- Prepare Presentation for Scientific Research Report

### Kieu

Last week:

- Write Proposal
- Continue learning ReactJS
- Practice with first UI

This week:

- Continue learning ReactJS
- Prepare for Scientific Research Report

## Integration 8 (22-Mar - 28/Mar/21)

---

Agenda:

- Review task of each member on the meeting before

Meeting Content:

- Planing for Integration 8

Post Meeting Action Items

### Hoa

Last week:

- Prepare Presentation for Scientific Research Report
- Research Architecture of Data Lake
  - Sync up with Dong
- Setup server temporary

This week:

- Implement API for GraphDB Admin
- Implement API for importing data, getting data from GraphDB
- Initial Backend
- Build Server API with Mock data

### Dong

Last week:

- Prepare Presentation for Scientific Research Report
- Choose auto ETL platform
- Research Architecture of Data Lake
  - Sync up with Hoa

This week:

- Create DW

- Cleaning Process for data sources

## Tin

Last week:

- Design UI
- Prepare Presentation for Scientific Research Report

This week:

- build base UI
- design Modal when onClick in Navbar
- design UI when merge 2 dimension
- setup DEV & Test environment

## Kieu

Last week:

- Continue learning ReactJS
- Prepare for Scientific Research Report

This week:

- Learn Bug Reporting
- Learn to write Test Cases
- Write Test Cases for all features in base UI

# Integration 1 (29/Mar - 05/Apr/21)

---

Agenda:

- Review task of each member on the meeting before
- Sprint Retrospective

Meeting Content:

- Sprint Retrospective
- Planing for Integration 9
- Integration base UI with server

Sprint Retrospective (Sprint 1 & 2)

**MAD - What made us mad?** 😡

- We need to write too much methods without the help of any libraries/frameworks.
- Handling all cases of raw data is something impossible.

**SAD - What made us sad?** 😢

- Timeline is delayed.
- We couldn't follow the sprint.

### GLAD - What made us glad? 😎

- Completely designed the architecture for fully-automatic process.
- Basically implemented the user-interface and data-cleaning process.

## Post Meeting Action Items

### Hoa

Last week:

- Implement API for GraphDB Admin
- Implement API for importing data, getting data from GraphDB
- Initial Backend
- Build Server API with Mock data

This week:

- Implement API for calling RDF Engine that built in Scala
- Continue building common APIs

### Dong

Last week:

- Create DW
- Cleaning Process for data sources

This week:

- Fill up DW, and ETL with Bonobo

### Tin

Last week:

- build base UI
- design Modal when onClick in Navbar
- design UI when merge 2 dimension
- setup DEV & Test environment

This week:

- build function dnd item from sidebar to main

### Kieu

Last week:

- Learn Bug Reporting
- Learn to write Test Cases
- Write Test Cases for all features in base UI

This week:

- Write Test Cases for all features in base UI

## Integration 10 (05/Apr - 12/Apr/21)

---

Agenda:

- Review task of each member on the meeting before

Meeting Content:

- Sprint Retrospective
- Planing for Integration 9
- Integration base UI with server

Post Meeting Action Items

**Hoa**

Last week:

- Implement API for calling RDF Engine that built in Scala
- Continue building common APIs

This week:

- Continue building APIs and RDF Automation Generator
- Restructure UI

**Dong**

Last week:

- Fill up DW, and ETL with Bonobo

This week:

- Continue Fill up DW, and ETL with Bonobo
- Restructure UI

**Tin**

Last week:

- build function dnd item from sidebar to main

This week:

- Restructure UI

### Kieu

Last week:

- Write Test Cases for all features in base UI

This week:

- Write test cases for new UI
- Restructure UI

## Integration 11 (26/Apr - 02/May)

---

Agenda:

- Review task of each member on the meeting before
- Sprint Retrospective

Meeting Content:

- New UI
- Change server implement method
- Change flow of APIs

Post Meeting Action Items

### Hoa

Last week:

- Continue building APIs and RDF Automation Generator
- Restructure UI

This week:

- Change APIs structure for dynamic server

### Dong

Last week:

- Continue Fill up DW, and ETL with Bonobo
- Restructure UI

This week:

- Finishing up fill up DW, and ETL with Bonobo

- Modify structure of DW

### Tin

Last week:

- Restructure UI

This week:

- build basic new UI
- create default screen
- create introduction screen
- create about screen

### Kieu

Last week:

- Write test cases for new UI
- Restructure UI

This week:

- Test case for new UI

## Integration 12 (03/May - 09/May)

---

Agenda:

- Review task of each member on the meeting before
- Sprint Retrospective

Post Meeting Action Items

### Hoa

Last week:

- Change APIs structure for dynamic server

This week:

- Custom source code and flow of RMLStreamer
- Change mapping file structure from RML to R2RML

### Dong

Last week:

- Finishing up fill up DW, and ETL with Bonobo

- Modify structure of DW

This week:

- Define base UI and Logic
- Define backend server for importing

### Tin

Last week:

- build basic new UI
- create default screen
- create introduction screen
- create about screen

This week:

- Create function for Show Example

### Kieu

Last week:

- Test case for new UI

This week:

- Write test case for show Example function

## Integration 13 (10/May - 16/May)

---

Agenda:

- Review task of each member on the meeting before
- Sprint Retrospective

Sprint Retrospective (Sprint 3 & 4)

**MAD - What made us mad?** 😡

- It's hard to find a way for auto-processing a messy file
- Automatic processing everything is too hard

**SAD - What made us sad?** 😢

- Timeline is delayed.
- We couldn't follow the sprint.

**GLAD - What made us glad?** 😎

- The team spirit is awesome

- Server runs smoothly

## Post Meeting Action Items

### Hoa

Last week:

- Custom source code and flow of RMLStreamer
- Change mapping file structure from RML to R2RML

This week:

- Find methods to integrate auto ETL with RDF generator
- Research automating detect dimension & measure method

### Dong

Last week:

- Define base UI and Logic
- Define backend server for importing

This week:

- Define base UI and Logic
- Define backend server for importing

### Tin

Last week:

- Create function for Show Example

This week:

- create Toolbox
- create import widget
- create import function

### Kieu

Last week:

- Write test case for show Example function

This week:

- Test Navigation bar component
- Test widget in Editor Pane component
- Test data in Information component

# Integration 14 (17/May - 23/May)

---

## Agenda:

- Review task of each member on the meeting before
- Sprint Retrospective

## Post Meeting Action Items

### Hoa

Last week:

- Find methods to intergrate auto ETL with RDF generator
- Research automating detect dmision & measure method

This week:

- Implement bash flow for running ETL and RDF generator script
- Implement automating detection method

### Dong

Last week:

- Define base UI and Logic
- Define backend server for importing

This week:

- Find solution for automation ETL
- Create workflows, and data pipeline with Apache airflow

### Tin

Last week:

- create Toolbox
- create import widget
- create import function

This week:

- create function to view the EDSDA's data source on widget
- create function to merge multi widget
- visualization data on Charts

### Kieu

Last week:

- Test Navigation bar component
- Test widget in Editor Pane component
- Test data in Information component

This week:

- test Toolbox UI
- test Toolbox function
- test the UI of import widget
- write test for visualization on Charts

## Integration 15 (17/May - 23/May)

---

Agenda:

- Review task of each member on the meeting before
- Sprint Retrospective

Sprint Retrospective (Sprint 1 & 2)

**MAD - What made us mad?** 😡

- Covid-19 pandemic comes again

**SAD - What made us sad?** 😢

- Timeline is delayed.
- We couldn't follow the sprint
- It's hard to integrate server and data processing and UI

**GLAD - What made us glad?** 😊

- Keep the sprint on time

Post Meeting Action Items

**Hoa**

Last week:

- Implement bash flow for running ETL and RDF generator script
- Implement automating detection method

This week:

- Buy Google Cloud for deploying
- Setup Google Cloud and server dependencies

**Dong**

Last week:

- Find solution for automation ETL
- Create workflows, and data pipeline with Apache airflow

This week:

- Find solution for automation ETL
- Create workflows, and data pipeline with Apache airflow
- Define base workflow for Data warehouse

## Tin

Last week:

- create function to view the EDSDA's data source on widget
- create function to merge multi widget
- visualization data on Charts

This week:

- visualization data on Maps & Table
- implement import function with server API

## Kieu

Last week:

- test Toolbox UI
- test Toolbox function
- test the UI of import widget
- write test for visualization on Charts

This week:

- Test data of visualization on Charts
- Test UI of Charts
- Test the Tree View of data source on widget
- Test Connector and port of widgets
- Write Test for visualization on Maps & Table
- Write test for Import function

# Integration 16 (24/May - 30/May)

---

Agenda:

- Review task of each member on the meeting before
- Sprint Retrospective

Post Meeting Action Items

## Hoa

Last week:

- Buy Google Cloud for deploying
- Setup Google Cloud and server dependencies

This week:

- Migrate source code and dependencies to Google Cloud
- Setup GraphDB in Google Cloud
- Public IP

## Dong

Last week:

- Find solution for automation ETL
- Create workflows, and data pipeline with Apache airflow
- Define base workflow for Data warehouse

This week:

- Workflows for crawl data, ETL automation

## Tin

Last week:

- visualization data on Maps & Table
- implement import function with server API

This week:

- implement import function with server API
- Deployment

## Kieu

Last week:

- Test data of visualization on Charts
- Test UI of Charts
- Test the Tree View of data source on widget
- Test Connector and port of widgets
- Write Test for visualization on Maps & Table
- Write test for Import function

This week:

- Test function of import
- Test the UI on domain: sda-research.surge.sh



## CAPSTONE PROJECT 2

CMU-SE-451 / CMU-IS-451 / CMU-CS-451

### MENTOR MEETING

Version 2.0

Date: 1 - Mar - 2021

## EXPERT-DRIVEN SMART DASHBOARD APPLICATION

#### Submitted by

Vo Van Hoa  
Pham Van Tin  
Ky Huu Dong  
Tran Thi Thanh Kieu

#### Approved by

##### Capstone Project 2 - Mentor:

Name

Signature

Date

Binh, Thanh Nguyen \_\_\_\_\_

A handwritten signature in blue ink, appearing to read 'Binh, Thanh Nguyen'.

\_\_\_\_\_ 26 - May - 2020

PROJECT INFORMATION			
<b>Project Acronym</b>	EDSDA		
<b>Project Title</b>	Expert-Driven Smart Dashboard Application		
<b>Project Web URL</b>	<a href="https://sda-research.ml/">https://sda-research.ml/</a>		
<b>Start Date</b>	01 - Mar - 2021		
<b>End Date:</b>	02 - Jun - 2021		
<b>Lead Institution</b>	International School, Duy Tan University		
<b>Project Mentor</b>	Ph.D Binh, Thanh Nguyen		
<b>Scrum Master</b>	Hoa, Vo	hoavo.dng@gmail.com	0935.193.182
<b>Team Members</b>	Tin, Pham Van	tinphamvan123@gmail.com	0932.535.175
	Dong, Ky Huu	kyhuudong@gmail.com	0898.246.980
	Kieu, Tran Thi Thanh	thanhkieutran391@gmail.com	0358.583.251

DOCUMENT INFORMATION			
<b>Document Title</b>	Mentor Meeting		
<b>Author(s)</b>	Team C2SE.06		
<b>Role</b>	[EDSDA] Mentor Meeting v2.0		
<b>Date</b>	01 - Mar - 2021	Filename	[EDSDA] 013 Mentor Meeting
<b>URL</b>	<a href="https://github.com/sdateamdtu2020/SDA-v2.0">https://github.com/sdateamdtu2020/SDA-v2.0</a>		
<b>Access</b>	Project and CMU Program		

# Meeting 01 (18-Jan-21)

---

Agenda:

- Discuss development strategy of capstone project 2

Meeting Content:

- Discuss development strategy
- Mr.Binh give some Feature Request

Post Meeting Action Items

**All members**

This week:

- Planning Proposal, Architecture document

# Meeting 01 (28-Jan-21)

---

Agenda:

- Review Scientific Research Report
- Discuss development strategy of capstone project 2

Meeting Content:

- Discuss development strategy
- Mr.Binh give some Feature Request
- Mr.Binh give feedback for Scientific Research Report

Post Meeting Action Items

**All members**

Last week:

- Planning Proposal, Architecture document

This week:

- Planning Proposal, Architecture document
- Update content for Scientific Research Report

# Meeting 03 (8-May-21)

---

**Agenda:**

- Discuss development strategy of capstone project 2

**Meeting Content:**

- Discuss development strategy
- Mr.Binh give some Feature Request

**Post Meeting Action Items****All members**

This week:

## Meeting 04 (19-May-21)

---

**Agenda:**

- Check the documents of project

**Meeting Content:**

- Check the proposal document
- Check the Architecture document
- Check the Product Backlog document

**Post Meeting Action Items****All members**

This week:

- []



## CAPSTONE PROJECT 2

CMU-SE-451 / CMU-IS-451 / CMU-CS-451

### TEAM REFLECTION

Version 2.2

Date: 1 - Mar - 2021

### EXPERT-DRIVEN SMART DASHBOARD APPLICATION

#### Submitted by

Vo Van Hoa  
Pham Van Tin  
Ky Huu Dong  
Tran Thi Thanh Kieu

#### Approved by

##### Capstone Project 2 - Mentor:

Name

Signature

Date

Binh, Thanh Nguyen \_\_\_\_\_  \_\_\_\_\_ 26 - May - 2020

PROJECT INFORMATION			
<b>Project Acronym</b>	EDSDA		
<b>Project Title</b>	Expert-Driven Smart Dashboard Application		
<b>Project Web URL</b>	<a href="https://sda-research.ml/">https://sda-research.ml/</a>		
<b>Start Date</b>	01 - Mar - 2021		
<b>End Date:</b>	02 - Jun - 2021		
<b>Lead Institution</b>	International School, Duy Tan University		
<b>Project Mentor</b>	Ph.D Binh, Thanh Nguyen		
<b>Scrum Master</b>	Hoa, Vo	hoavo.dng@gmail.com	0935.193.182
<b>Team Members</b>	Tin, Pham Van	tinphamvan123@gmail.com	0932.535.175
	Dong, Ky Huu	kyhuudong@gmail.com	0898.246.980
	Kieu, Tran Thi Thanh	thanhkieutran391@gmail.com	0358.583.251

DOCUMENT INFORMATION			
<b>Document Title</b>	Team Reflection		
<b>Author(s)</b>	Team C2SE.06		
<b>Role</b>	[EDSDA] Team Reflection v2.0		
<b>Date</b>	01 - Mar - 2021	Filename	[EDSDA] 014 Team Reflection
<b>URL</b>	<a href="https://github.com/sdateamdtu2020/SDA-v2.0">https://github.com/sdateamdtu2020/SDA-v2.0</a>		
<b>Access</b>	Project and CMU Program		

## REVISION HISTORY

Version	Person(s)	Date	Description	Approval
Draft	Hoa, Vo	01 - Mar - 2021	Initiate document	x
2.0	All members	26 - May - 2020	Finish content of document	x

## TABLE OF CONTENTS

PROJECT INFORMATION	1
REVISION HISTORY	2
<b>TABLE OF CONTENTS</b>	<b>3</b>
<b>1. INTRODUCTION</b>	<b>4</b>
<b>2. REFLECTION</b>	<b>4</b>

# 1. INTRODUCTION

In this document, every team member will write a reflection about themself that including:

- What I have done
- What I have learned
- Experiences
- Pros & Cons
- Improvement
- Plan in the future

# 2. REFLECTION

Name	Role	Reflection
Hoa Vo	Scrum master	<p>Working on the project as both a scrum master and a backend developer, I guided the members of the system design team, participated in RDF data cube and data warehouse training. Access to and training in new data technologies not only helps me improve my thinking ability and knowledge but also improves my ability to read and understand English.</p> <p>More specifically, as a scrum master, besides building the project process and designing the work to be done according to the scrum model, as well as breaking down the project implementation steps, I am also the person involved. Communicate directly between mentors and team members to communicate ideas and suggested solutions to the project.</p> <p>As a backend developer, I do all the creation of RDF data blocks, building an express server to provide the system API, and setting up the server on the virtual machine. Building a server by myself has helped me improve my programming thinking ability to a new level, as well as understand how the backend behind a system works.</p> <p>However, due to some objective reasons from my family and personal affairs, I cannot organize some annual offline meetings, instead are online meetings. In addition, I still cannot come up with a machine learning solution for automatically linking blocks of data as the original project goal because of the time limitation. I plan to try to find solutions to these problems in the near future to help the project achieve its original goal, as well as improve the technical knowledge for myself.</p>

Tin Pham	Team member	<p>After finishing the first step of our final project for graduation, I have learned a lot of skills including soft skills and hard skills. We received a very interesting topic from our mentor. I have learned how to organize a team, how to arrange my schedules to fit with the team, how to communicate, how to resolve conflicts between team members, and how to deal with our mentor.</p> <p>Also, I have learned a lot of new technologies in the front end. Making a pure Javascript Stack for every codebase is fun and I love it. I also know how to build a dashboard for visualization data. I have learned about using Redux in React which is a new modern way to state management and applied them to our application. I also have learned about deploying a website, I hosted a website in Firebase and added a custom domain for free. The big point is I have learned about React,Redux, which gives me a new vision of modern software development. Hoa Vo helped us a lot while we were in the learning phase.</p> <p>After 4 sprints of hard-working, we have done a long way with our project. However, I am not satisfied with our achievement at this moment, because this project is not as simple as I thought. There are still many things that we need to deal with in the future. I will try to do my best to make my team become the strongest team &amp; finish our project before our graduation.</p>
Dong Ky	Team member	<p>In the project, I do not only work individually but also have face-to-face meetings to interact and communicate the ideas and progress of the project. I also learn about many expertise and technology in the implementation of the project, such as how to cooperate and collaborate with persons who have different knowledge backgrounds, how to present the result of the project. How to realize the design of the application, and so on. Those knowledge and information are significant for me since they are useful and helpful in my following learning and working.</p> <p>With interesting research topics, I learned more about the new technologies, new concepts, and know more about works behind the scene of many large operations.</p> <p>Working with scrum processes we have the chance to practice in scum how to organize and follow the steps that are planned.</p> <p>Working with new people learns new things about partners. Asking for help when something is on the road, and always ready for help from others.</p> <p>Learning new things about team members: coding skills, and new technologies from them.</p> <p>Improving English skills by writing documents in English, searching in English.</p>

Kieu Tran	Team member, Secretary	<p>After completing the project, I improved a lot of soft skills as well as hard skills that I have the benefit of repairing for my future career. Through this interesting topic, I learned more about advanced HTML / CSS, javascript, react and new technologies, and how to test for errors. Thereby, coordinating with the members to create a complete website. It's hard skills, soft skills, I learn more about how to work with my team, improve English documentation, how to organize my schedule to suit my needs and it's important. more, how to follow the Scrum process. Finally, I learned a good presentation and received feedback from the teachers.</p>
-----------	------------------------	--



## CAPSTONE PROJECT 2

CMU-SE-451 / CMU-IS-451 / CMU-CS-451

### TEAM EVALUATE

Version 2.2

Date: 1 - Mar - 2021

## EXPERT-DRIVEN SMART DASHBOARD APPLICATION

### Submitted by

Vo Van Hoa  
Pham Van Tin  
Ky Huu Dong  
Tran Thi Thanh Kieu

### Approved by

#### Capstone Project 2 - Mentor:

Name

Signature

Date

Binh, Thanh Nguyen \_\_\_\_\_  \_\_\_\_\_ 26 - May - 2020

PROJECT INFORMATION			
<b>Project Acronym</b>	EDSDA		
<b>Project Title</b>	Expert-Driven Smart Dashboard Application		
<b>Project Web URL</b>	<a href="https://sda-research.ml/">https://sda-research.ml/</a>		
<b>Start Date</b>	01 - Mar - 2021		
<b>End Date:</b>	02 - Jun - 2021		
<b>Lead Institution</b>	International School, Duy Tan University		
<b>Project Mentor</b>	Ph.D Binh, Thanh Nguyen		
<b>Scrum Master</b>	Hoa, Vo	hoavo.dng@gmail.com	0935.193.182
<b>Team Members</b>	Tin, Pham Van	tinphamvan123@gmail.com	0932.535.175
	Dong, Ky Huu	kyhuudong@gmail.com	0898.246.980
	Kieu, Tran Thi Thanh	thanhkieutran391@gmail.com	0358.583.251

DOCUMENT INFORMATION			
<b>Document Title</b>	Team Evaluate		
<b>Author(s)</b>	Team C2SE.06		
<b>Role</b>	[EDSDA] Team Evaluate v2.0		
<b>Date</b>	01 - Mar - 2021	Filename	[EDSDA] 014 Team Evaluate
<b>URL</b>	<a href="https://github.com/sdateamdtu2020/SDA-v2.0">https://github.com/sdateamdtu2020/SDA-v2.0</a>		
<b>Access</b>	Project and CMU Program		

# 1. EVALUATE

Name	Role	Tasks	Points
Hoa Vo Van	Scrum master, RDF Data Cubes Developer, Backend Developer, Document Editor	<p>Scrum Master: Estimate time, Manage project &amp; process quality, Connect between members in team and mentors, Give the tasks to each member and ensure these tasks, Analyze Project requirements, Research solution for the project objective.</p> <p>RDF Data Cubes Developer: Design &amp; implement RDF Data Cubes, Ensure Data in RDF Data Cubes, Validate RDF Data Cubes, Create &amp; Manage GraphDB server.</p> <p>Backend Developer: Design system architecture, Implement &amp; Validate the server &amp; API, Deploy the server to VPS, Manage the health of VPS server.</p> <p>Documents Editor: Write documents content, Manage the contents of documents.</p>	
Tin Pham Van	Frontend Developer, Tester, Document Editor	<p>Frontend Developer: Design Module View client, Components &amp; Connectors. Fetch API from server and visualise in Charts, Maps. Visualize Datacube in UI that users can easily select. Hosting, and domain manager,</p> <p>Tester: Create UI Test Plan, create test cases in Dashboard. Ensure all city &amp; year dimensions that display on the widget are working.</p> <p>Documents Editor: Create documents format, Draw User Interface Design, Manage the quality attribute of documents, Write documents content, Create slides of the presentation.</p> <p>Ensure all source code of each member is up to date on the Github repository,</p>	

Dong Ky Huu	Database Developer, Tester, Document Editor	<p>Database Developer: Research for Data solution, Define Data requirements, Research &amp; Define Data sources, Design &amp; Implement ETL process, Design &amp; Building Data warehouse, Maintain Data warehouse, Improve Data quality.</p> <p>Document Editor: Write documents content, Design diagrams.</p> <p>Tester: Test the ETL process, Validate Data in Data warehouse, Test the performance of Data warehouse.</p>	
Kieu Tran Thi Thanh	Frontend Developer, Tester Secretary, Document Editor	<p>Frontend Web Developer: Build a homepage UI that displays information about the application.</p> <p>Secretary: Take note and record feedback of the meetings, Make meetings reports, Make &amp; Ensure the Hard copy of Documents.</p> <p>Document Editor: Write documents, Ensure the document contents.</p> <p>Tester: Create Test Case of Homepage. Ensure as many test cases of the homepage as possible</p>	