MINISTRY OF EDUCATION AND TRAINING

FPT UNIVERSITY

Capstone Project Document

Data Structures and Algorithms Visualizer

Group 9	
Group members	Hà Lê Duy Khang – Team Leader – SE61886 Nguyễn Lương Triều Vỹ – Team Member – SE61811 (dropped out) Đặng Xuân Huy – Team Member – SE61318 (dropped out) Dư Đức Long - Team Member - SE62137 (dropped out)
Supervisor	Mr. Kiều Trọng Khánh
Ext. Supervisor	N/A
Capstone Project code	DSAV

- Ho Chi Minh City, 01/2018 –

TRƯỜNG ĐẠI HỌC FPT

CAPSTONE PROJECT REGISTER

Class: Duration time: from / /2018…. To / /2018…..

(*) Profession: <Software Engineer> Specialty: <ES> ☐ <IS> ☒

(*) Kinds of person make registers: Lecturer ☒ Students ☐

1. Register information for supervisor (if have)

	Full name	Phone	E-Mail	Title
Supervisor 1	Kiều Trọng Khánh		khanhkt@fpt.edu.vn	Mr.

2. Register information for students (if have)

	Full name	Student code	Phone	E-mail	Role in Group
Student 1	Hà Lê Duy Khang	SE61886			Team Leader
Student 2	Nguyễn Lương Triều Vỹ	SE61811			Member
Student 3	Đặng Xuân Huy	SE61318			Member
Student 4	Dư Đức Long	SE62137			Member

3. Register content of Capstone Project

(*) 3.1. Capstone Project name:

English: Data Structures and Algorithms Visualizer

Vietnamese: Học tốt cấu trúc dữ liệu và giải thuật

Abbreviation:

- DSAV

- Context:

+ How to get knowledge about data structures and algorithms in clearly?

+ How to verify your knowledge about this theory is right?

+ How to visual the steps of structures or algorithms to make clear and improve learner capacity?

- Building the system provides following services

+ The system can model some data structures and algorithms (at least 10 theories)

+ The system can visual the step by step of these theories to users by using example

+ The system allows the user to make input into formatted form that the application can visual with specify theory
+ Make the data abstraction for theory that can be imported to the system in dynamically
+ …

- Simulator

+ Visual selected theory depending on user's choice
+ Make the new data structure and algorithms that system is not, then import to the system and make the visualizer with sample for that theory

(*) 3.2. Main proposal content (including result and product)

a) Theory and practice (document):

- Student should apply the software development process and the UML
- Software artifacts include User Requirement, Software Requirement Specification, Architecture Design, Detail Design, System Implementation and Testing Document, Installation Guide, sources code, and deployable software packages
- 3 tiers should be applied
- Server side technique:
    - Database design, OOA, OOD, OOP, MVC, Java or .Net technology, …
- Client side technique
    - HTML5, CSS, JavaScript, JQuery, Ajax, Android**, ...**
- Communication technique
    - Exchange information and transfer data in effective in networks, communicating protocol between mobile devices, ...
- Research
    - Android/iOS mobile development
    - Data structure and Algorithms
    - JavaEE Specifications
    - Visualizer SDK or API
    - …

b) Program:

- Main functions
    - Web Application for supporting staff to import or maintain the new data structures or algorithms
    - Mobile Application for users to learn, find theory out, ..
    - Abstraction Form that configures dynamic describing the data structures and algorithms
    - …

c) Other products:

- All of management functions of the system must be implemented to support the operating system in best
- Papers

4. Other comment (propose all relative thing if have)

N/A

HCM city, date 14/12/2017

Supervisor (If have)	**On behalf of Registers**
(Sign and full name)	*(Sign and full name)*
Kiều Trọng Khánh	

A. Introduction

1. Project Information

- Project name: Data Structures and Algorithms Visualizer
- Project Code: DSAV
- Product Type: Web app & Mobile app
- Start Date: 08/01/2018
- End Date: 28/04/2018

2. Introduction

We introduce to an application that helps learners get clearly understanding and verify their knowledges about data structures and algorithms. In current situation, many learners do not understand deeply in data structures and algorithms.

We build a system, which helps users solve current problems. In the process of analysis, users can input their data based on forms and create new visualization models, so they can see the operating process of algorithms. We believe with visual graph and animations, users are capable to memorize and visualize data structures and algorithms more clearly. Generally, the system will visualize data structures and algorithms step by step by animations.

3. Current Situation

- When learning data structures and algorithms, learners or developers only have text documents, plain source codes about definitions or examples. With complicated algorithms, those algorithms cannot be demonstrated clearly and easily understanding by text definition. Learners often deal with the exercises by looking for source code on the internet rather than coding them.
- Currently, most of learners are studying by lecturers' instruction on class, written on board.
- Some websites support animated algorithms such as visualgo.net, it contains variety of structures and algorithms and it helps users to watch how the algorithm works step by step through animation.

4. Problem Definition

Advantages of current situation:

- Learners can easily understand the algorithm and structure through the animations.
- That application has many common algorithms and structures for learners to refer.
- When the algorithm and structures is implemented, the application can display pseudo code.

Disadvantages of current situation:

- Learners are only aware of basic data structures and algorithms.
- Cannot compare performance between algorithms having the same complexity in different cases.
- Learners lack of a platform to test their understanding and verify their knowledges.

Advantages of common algorithm visualizer websites:

- Implemented diverse of data structures and algorithms.
- Have multiple functions in each data structures.

- Have many algorithms and data structures for users to select.
- Clarify algorithms and data structures through animation.
- Show pseudo code when algorithms start running.

Disadvantages of common algorithm visualizer websites:

- Cannot tracks or visualize their plain source code.
- User cannot suggest or post their new algorithms by plain source code.

5. Proposed Solution

Our proposed solution is build a Web Application which can show users visualization model and transforms during process from beginning to the end through animations.

5.1. Feature functions

- The system can model some data structures and algorithms (at least 10 theories).
- The system can visualize the step by step of these theories to users using example.

5.2. Values and challenges

Values:

- Provides source code display function.
- Highlighting source code every steps of the function.
- The system analyzes the old models to support the best modeling for general structures and algorithms. If users can describe their data structures or algorithms based on the giving template, the system can be able to visualize their ideas.

Challenges:

- The system cannot provide all existing algorithms and data structures.
- System only understand one existing pseudo code format.

.

B. Software Project Management Plan

1. Problem Definition

1.1. Name of this Capstone Project

Data Structures and Algorithms Visualizer (DSAV).

1.2. Problem Abstract

A large body of research indicates that visual cues help us to better retrieve and remember information. The research outcomes on visual learning make complete sense when you consider that our brain is mainly an image processor (much of our sensory cortex is devoted to vision), not a word processor. In fact, the part of the brain used to process words is quite small in comparison to the part that processes visual images.

Data Structures and Algorithms is a difficult subject to imagine and understand clearly. We think that visualization may help us to deal with this subject. We want to build a reliable software about data structures and algorithms visualizer to help learners to better retrieve and remember knowledge about this subject.

1.3. Project Overview

1.3.1. Current Situation and Disadvantages

Project team are facing many challenges on this project:

- Learning new technologies (canvas, HTML, JavaScript, JQuery and d3js framework, ReactJS).
- Implementation (convert data structure into drawings, display method by animation, display algorithm's code progress).
- Design format for user to create their own algorithms.

1.3.2. The Proposed System

We have to study new technologies for some these main reasons:

- How to visualize a specify data structure into pictures, drawings.
- How to create animation when the system runs an algorithm.
- How to display algorithm's code to users.
- How to make form with format for user to input their new algorithms.

1.3.2.1. Web Application

- Staffs can import or maintain a data structure or algorithm.
- Users can browse all algorithms and data structures that our system support and select a specify one.
- Users can search for the data structure or algorithm that they want to learn.
- Users can view detail definitions of data structures and algorithms.
- Users can view and control the visualizer of data structure or algorithm.
- Users can import new data structure or algorithm to the system through flow chart.
- The system allows users to upgrade new algorithms to enhance the diversity.

1.3.3. Boundaries of the System

- The system should do:
    - Allow user to learn data structure that our system support.
    - Allow user to edit data structure's elements.
    - Allow user to watch the system runs algorithms step by step.

- Allow user to add new algorithms to some data structures that the system support.

- Our system will not:
  - Implement all existing data structures and algorithms.

1.3.4. Development Environment

1.3.4.1. Hardware requirements

For server

	Minimum requirements	Recommended
Internet Connection	Cable, Wi-Fi (8 Mbps)	Cable, Wi-Fi (20 Mbps)
Operating System	Window Server 2008	Window Server 2008
Computer Processor	Intel® Xeon ® 1.4GHz	Intel® Xeon ® Quad Core (12M Cache, 2.50 GHz)
Computer Memory	1GB RAM	2GB or more

For PC

PC	Minimum Requirements	Recommended
Internet Connection	Cable, Wi-Fi (4 Mbps)	Cable, Wi-Fi (8 Mbps)
Operating System	Window 7	Window 7 or more.
Computer Processor	Intel® Core i3 1.4GHz	Intel® Core i5 2.50GHz
Computer Memory	1GB RAM	2GB RAM or more
Web Browser	Firefox (v52 or higher), Chromes (v28 or higher)	Chrome latest stable version

1.3.4.2. Software requirements

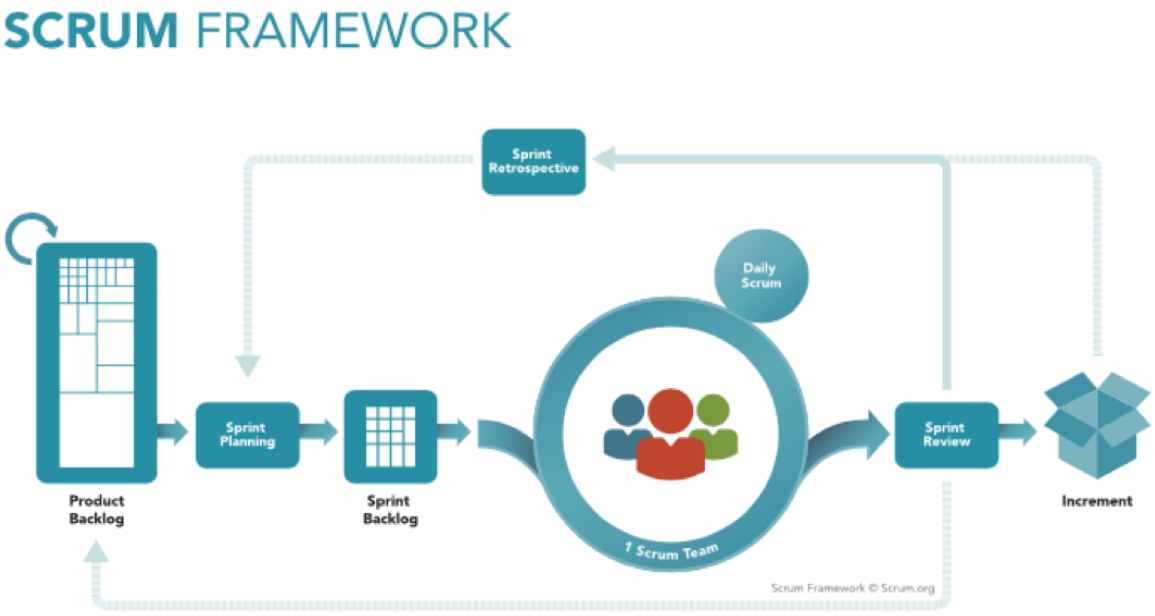
- MySQL: used to create and manage the database for system.
- Visual Studio Code: used to implement web application, web service.
- StarUML: used to create models and diagrams.
- Ninjamock: used to create UI mockup.

2. Project Organization

2.1. Software Process Model

This project is developed by using Scrum model – part of an agile framework for Software development project. Our team choose Scrum model because of the following reasons:

- Prototypes are delivered frequently for evaluation, usually weekly, rather than months due to our time is only 14 weeks.
- Take fewer risks when there is a change in requirement since we need to improve project more than the existing one.
- All members must work together in order to avoid misunderstanding or miscommunication.
- Able to study new skills or knowledge at the same time as developing.



https://www.scrum.org/

Artifact	**Feature**
Product backlog	The product backlog comprises an ordered list of requirements that a scrum team maintains for a product. It consists of features, bug fixes, non-functional requirements … - whatever must be done to successfully deliver a viable product.
Sprint backlog	The sprint backlog is the list of work the development team must address during the next sprint. The list is derived by the scrum team progressively selecting product backlog items in priority order from the top of the product backlog until they feel they have enough work to fill the sprint. The development team should keep in mind its past performance assessing its capacity for the new sprint, and use this as a guide line of how much 'effort' they can complete.
Product increment	The increment (or potentially shippable increment, PSI) is the sum of all the product backlog items completed during a sprint, integrated with the work of all previous sprints. At the end of a sprint, the increment must be complete, according to the scrum team's definition of done (DoD), fully functioning, and in a usable condition regardless of whether the product owner decides to actually release it.

2.2. Roles and responsibilities

No	Full name	Role	Responsibilities
1	Kiều Trọng Khánh	Project owner	- Specify scope and user requirement. - Supervise the development progress. - Provide professional techniques and business analysis support.
2	Hà Lê Duy Khang	Scrum master	- Managing process - Designing database - Clarifying requirements - Prepare documents - GUI design - Create test plan - Coding - Testing
3	Nguyễn Lương Triều Vỹ	Member	- Designing database - Clarifying requirements - Prepare documents - GUI design - Create test plan - Coding - Testing
4	Đặng Xuân Huy	Member	- Designing database - Clarifying requirements - Prepare documents - GUI Design - Create test plan - Coding - Testing
5	Dư Đức Long	Member	- Designing database - Clarifying requirements - Prepare documents - GUI Design - Create test plan - Coding - Testing

2.3. Tools and Techniques

	Tools	**Techniques**
Front-end	Visual Studio Code	- HTML5 - CSS3 - JavaScript - jQuery - ReactJS
Back-end	Visual Studio Code	- NodeJS
Database management system	MySQL	- SQL

3. Project Management Plan

3.1. Product Backlog

Story ID	Features	Task ID	Task description	Sprint
1	Create Product Backlog	1	Create Product Backlog	1
2	Write Introduction document	2	Write Introduction document	1
3	Write Project Management Plan	3.1	Problem definition	1
		3.2	Project organization	1
		3.3	Project management plan	1
		3.4	Coding convention	1
		3.5	Review document	1
4	Build System Structure	4.1	Backend Structure	2
		4.2	Web Structure	2
5	Write Software Requirements	5.1	User Requirement Specification	2
		5.2	External Interface Requirement	2
		5.3	Use case diagram	2
		5.4	Software System Attributes	2
6	Write Software Design Description	6.1	Design Overview	3
		6.2	System Architectural Design	3
		6.3	Component Diagram	3
		6.4	Detailed Description of Components	3
		6.5	Sequence Diagram	3
		6.6	User Interface Diagram	3
		6.7	Database Design	3
		6.8	Entity Diagram	3
		6.9	Class Diagram	3
7	Implementation	7		4
8	Create Software Test Documentation	8.1	Test Plan	5
		8.2	Test Cases	5
		8.3	Check lists	5
9	Quality Assurance	9.1	Quality Assurance for Backend	5

Story ID	Features	Task ID	Task description	Sprint
1	Create Product Backlog	1	Create Product Backlog	1
2	Write Introduction document	2	Write Introduction document	1
3	Write Project Management Plan	3.1	Problem definition	1
		3.2	Project organization	1
		9.2	Quality Assurance for Web	5
10	Software User's Manual	10.1	Installation Guide	6
		10.2	User's Guide	6

C. Software Requirement Specification

1. User Requirement Specification

1.1. Administrator Requirement

Administrator is a person who access to the system as Administrator role. Administrator can use following functions:

- Change user's role.

1.2. Staff Requirement

Staff is a person who access to the system as Staff role. Staff can use following functions:

- Manage course: create, update, remove course.
- Manage topic: create, update, remove topic.
- Manage lesson: create, update, remove lesson.
- Manage algorithm: create, update, remove algorithm.

1.3. Authenticated User Requirement

Authenticated User is a person who access to the system as Authenticated User role. Authenticated User can use following functions:

- Log out.
- View profile.
- Edit profile.
- Search course.
- View learned courses.
- View course.
- Enroll course.
- Resume last lesson.
- Learn lesson.
- Interact with the Visualizer: Create data, Control the Visualizer.

1.4. Unauthenticated User Requirement

Unauthenticated User is a person who does not access to the system. Unauthenticated User can use following functions:

- Sign up.
- Log in.

2. Software Requirement Specification

2.1. External Interface Requirement

2.1.1. User Interface

- The user interface uses English as main language for users and for Staff, Manager and Admin on Web application.

- The user interface displays best on 1024x768 and above screen size.

2.1.2. Hardware Interface

Desktop PC

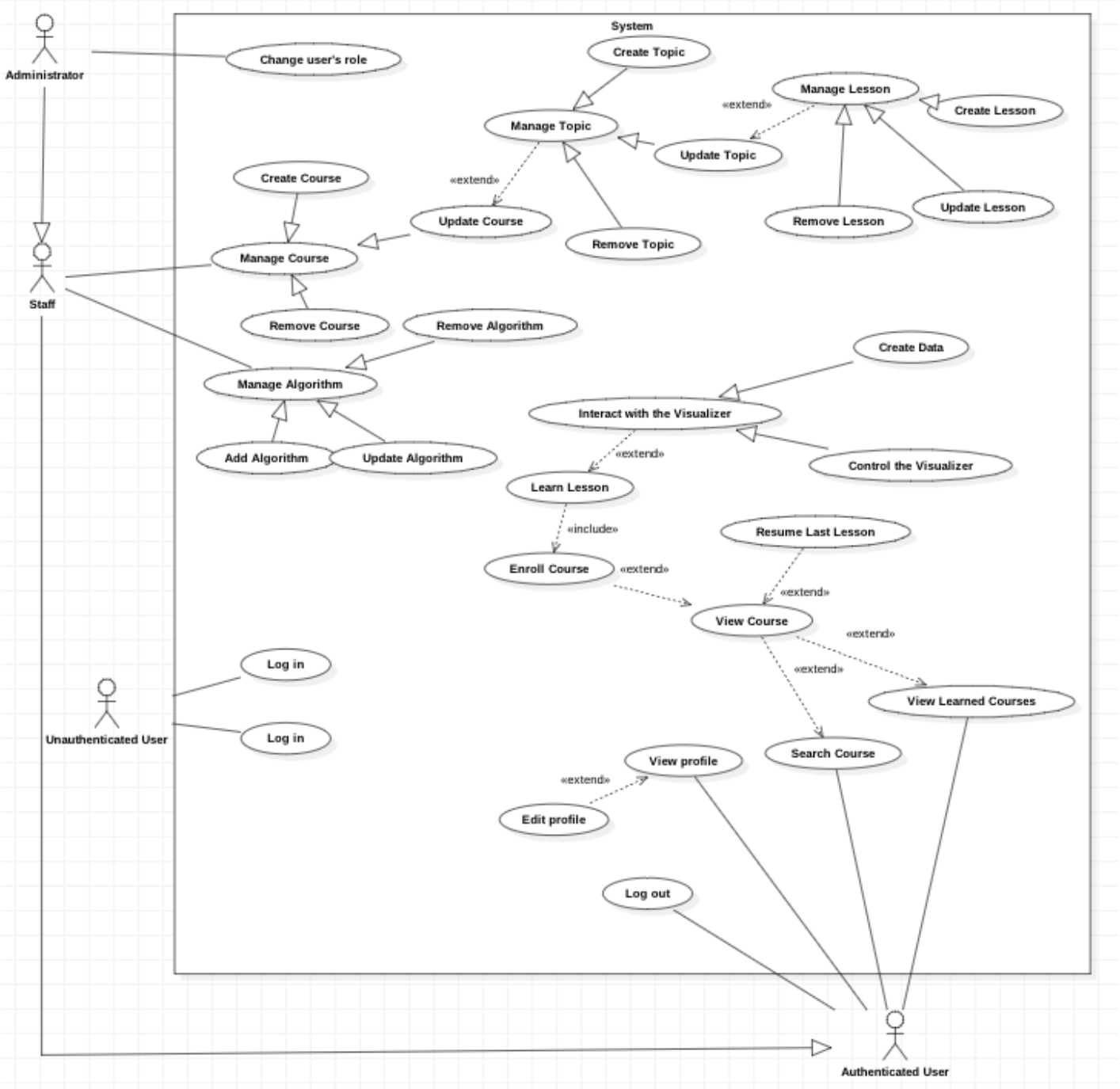
2.1.3. Software Interface

- Web application: work with Firefox (v30 or above), Chromes (v25 or above)

2.1.4. Communication Protocol

- Use HTTP protocol 1.1 for communication between the web browser and the web server.
- Use HTTP protocol 1.1 for communication between the server and the Microsoft service.

2.2. System Overview Use Case



3. Software Requirement Specification

3.1. Usability

- All the texts, labels, alerts and messages will be written in English.
- GUI for the web application is designed based on material design.
- The system usability is easy to use that users generally don't need to look at the document to use including admin and manager.
- Icons that indicate the actions should be easy to understand and users will not meet any troubles to recognize the feature of the page.

3.2. Reliability

- The data should be backed up every day.

3.3.Availability

- The web application must be available 24/7.

3.4. Security

- Each role of user has a specific permission to interact with the system.
- System always checks for authorization and authentication before doing anything.
- Only Administrator can grant permission to other roles.

3.5. Maintainability

- The system is divided into separated modules.
- The code is easy to maintain and upgrade.

3.6. Portability

- The software itself is a web application, therefore it can be used on any platform that has a web browser and can connect to the internet.

3.7. Performance

- System detects and returns results in 4 seconds or less under 4Mbps bandwidth.

D. Software Design Description

1. Design Overview

- This document describes the technical and user interface design of DSAV Web Application. It includes the architectural design, the detailed design of common functions and business functions and the design of database model.

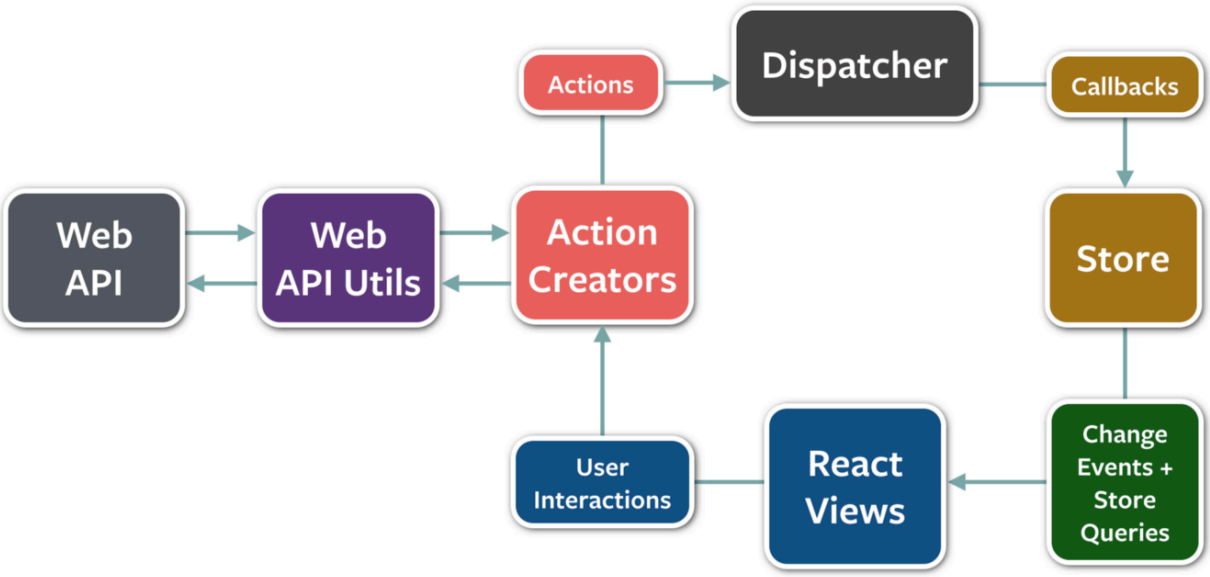
- The architectural design describes the overall architecture of the system and the architecture of each main component and subsystem.

- The detailed design describes static and dynamic structure for each component and functions. It includes class diagrams, class explanations and sequence diagrams for each use cases.

- The database design describes the relationships between entities and details of each entity.

2. System Architectural Design

2.1. System Architecture Overview



System architecture overview: Flux

Our application is developed mainly on Flux architecture.

Flux is an architecture that Facebook uses internally when working with React. It is *not* a framework or a library. It is simply a new kind of architecture that complements React and the concept of Unidirectional Data Flow.

Flux is probably better explained by explaining its individual components:

- Actions - Helper methods that facilitate passing data to the Dispatcher
- Dispatcher - Receives actions and broadcasts payloads to registered callbacks
- Stores - Containers for application state & logic that have callbacks registered to the dispatcher
- Controller Views - React Components that grab the state from Stores and pass it down via props to child components.
- Web API - Get data from database in server, then send to client via API call.

2.1.1. Dispatcher

The Dispatcher is basically the manager of this entire process. It is the central hub for your application. The dispatcher receives actions and dispatches the actions and data to registered callbacks.

The dispatcher broadcasts the payload to ALL of its registered callbacks, and includes functionality that allows you to invoke the callbacks in a specific order, even waiting for updates before proceeding. There is only ever one dispatcher, and it acts as the central hub within your application.

2.1.2. Stores

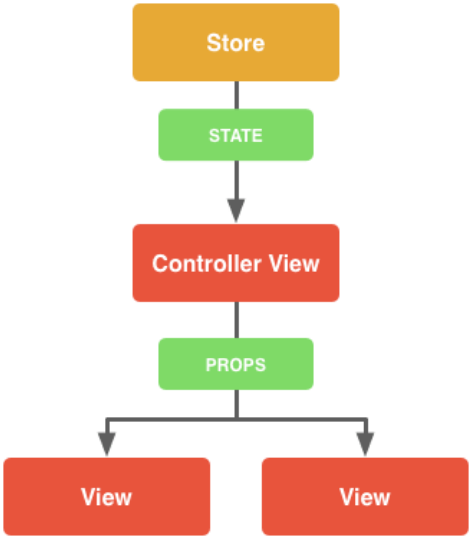
In Flux, Stores manage application state for a particular domain within application. From a high level, this basically means that per app section, stores manage the data, data retrieval methods and dispatcher callbacks.

2.1.3. Action Creators & Actions

Action Creators are collections of methods that are called within views (or anywhere else for that matter) to send actions to the Dispatcher. Actions are the actual payloads that are delivered via the dispatcher.

2.1.4. Controller Views

Controller views are really just React components that listen to change events and retrieve Application state from Stores. They then pass that data down to their child components via props.

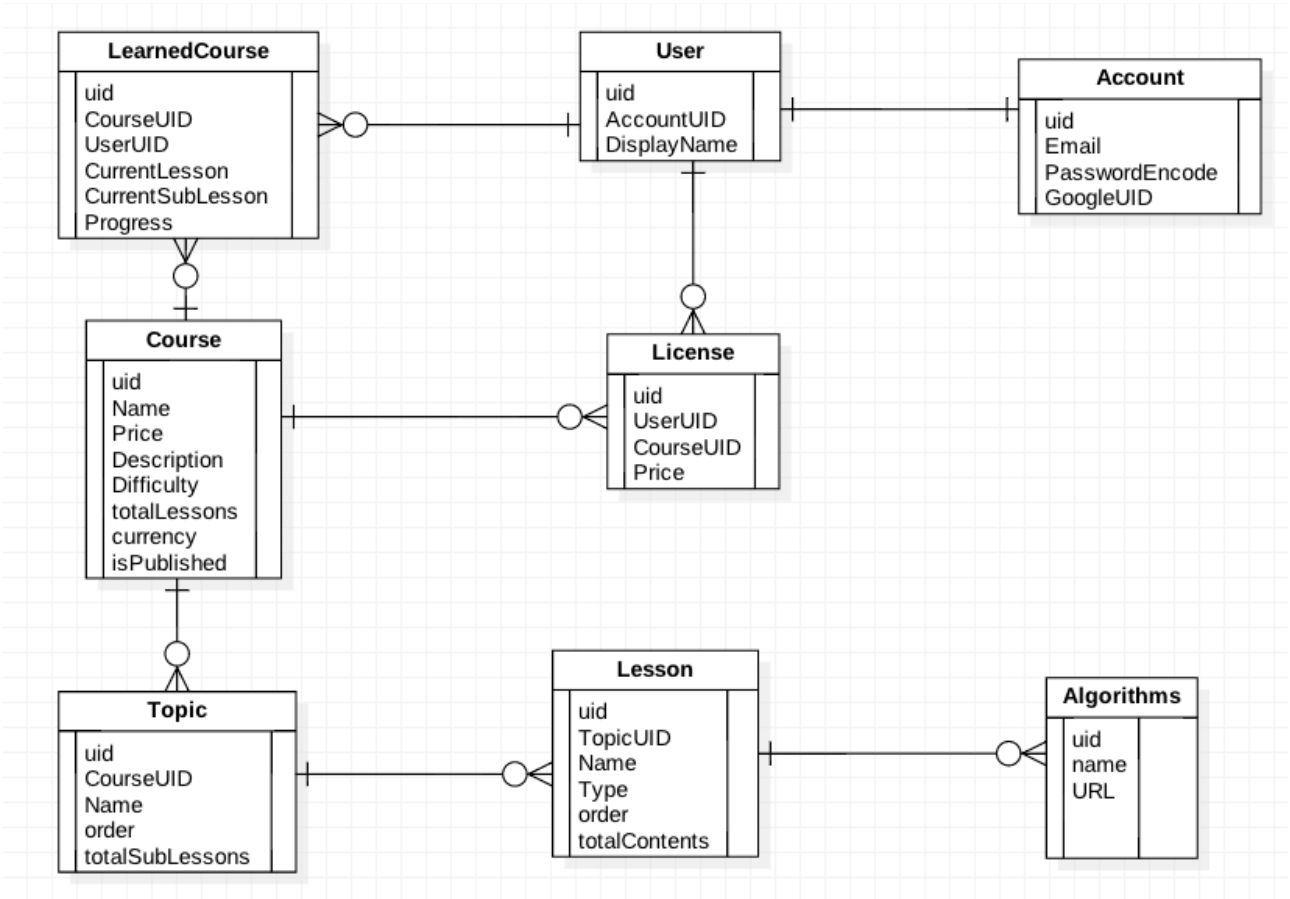


2.1.5. Web API

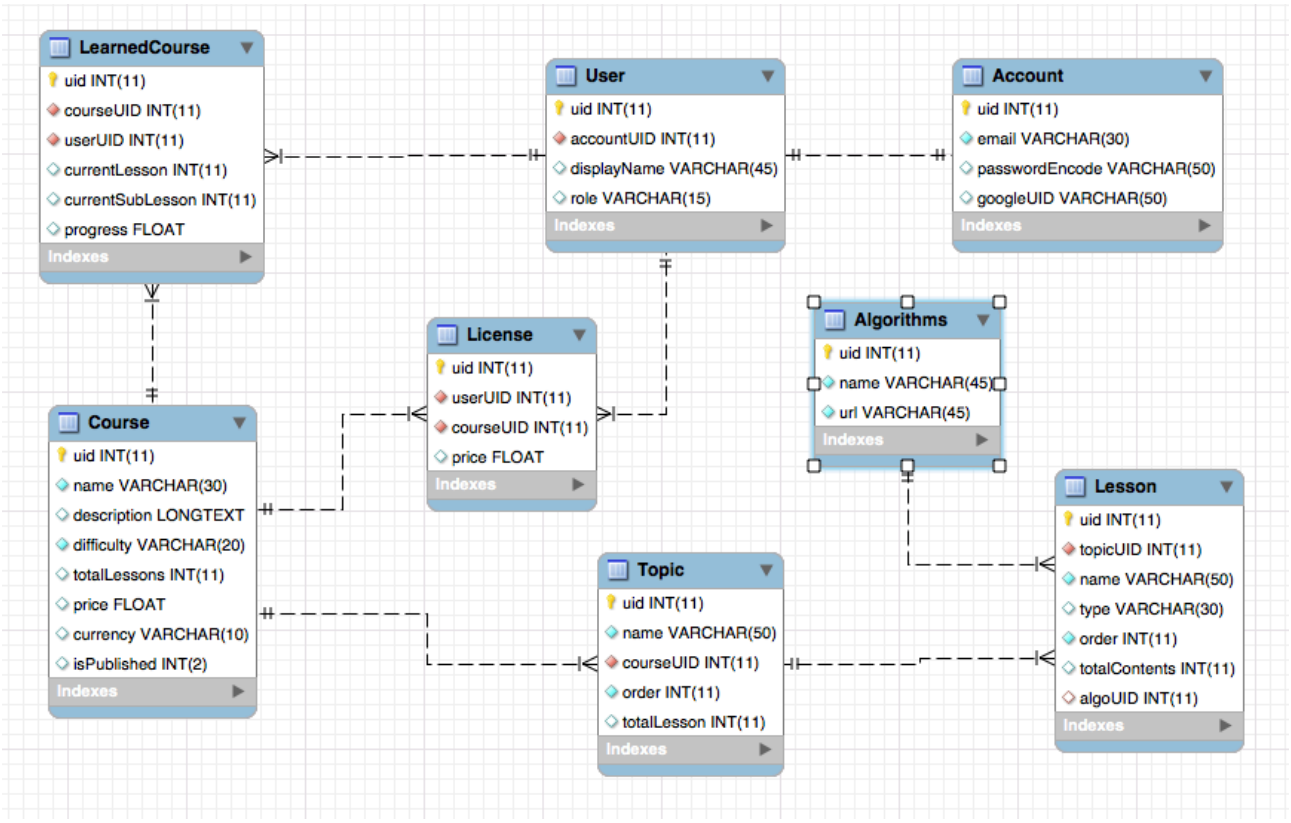
Web API is the component that be called by Actions. It goes to the server to get the data then send the data to Dispatcher.

We choose Flux with React because it's a new technology. And it can handle lots of requests, because all requests are called in clients. Then the requests call web API to get data. Server will be free from handling requests like a controller.

3. Entiry Relationship Diagram (ERD)



4. Database Relationship Diagram



E. Task Sheet

No.	Product Deliverables	Task	Khang HLD	Unit	Size
1	Report1 - Introduction	Project Information	O		1
		Introduction	O		1
		Current Situation	O		1
		Problem Definition	O		1
		Proposed Solution	O		1
		Functional Requirement	O		1
		Role and Responsibility	O		1
2	Report2- Software Project Management Plan	Problem Definition	O		1
		Project organization	O		1
		Project management plan	O		1
		Coding Convention	O		1
3	Report 3- Software Requirement Specification	User Requirement Specification	O		1
		System Requirement Specification	O		
		External Interface Requirements	O		1
		Functional Requirement	O		
		Admin function	O		1
		Staff functions (CRUD 12f)	O		12
		User functions (CRUD 13f)	O		13
		Animation models (4 models)	O		20
		Algorithms for animation (~ 15 algos)	O		75
		Non-functional Requirement			
		Reliability	O		1
		Availability	O		1
		Entity Relationship Diagram	O		1
4	Report 4- Software Design Description	System Architectural Design	O		4
		Class diagram	O		5
5	Report 5 - Software Implementation and Test Document	**Setup Environtment**	O		5
		DB diagrams	O		5
		Perform Testing	O		3
6	Report 6 - Software User's Manual	Other	O		1

F. Appendix

1. **MySQL [Online].** Available: https://dev.mysql.com/doc/
2. **ReactJS [Online].** Available: https://reactjs.org/
3. **D3JS [Online].** Available: https://d3js.org/
4. **NodeJS [Online]**. Available: https://nodejs.org/en/