Requirements

The goals of this document are as follows -

- To capture the complete, detailed description about how the AIA app is expected to perform.
- To illustrate the functional and non functional requirements.
- To list the capabilities proposed by stakeholders, as well as what is needed while defining high-level product features.
- To act as a reference point for the team during development.

Table of Contents

Project Overview

Purpose of the document

This document represents the different aspects of the project, what they stand for regarding the project, what the project is about, the history how it came to be, the present state of the project, as well as the desired system-to-be. The project overview serves as a document to introduce the project to a potential stakeholder.

Purpose of the AIA web app

The "Algorithms in Action" web app is to provide students an additional medium to learn and solidify their understanding of data structures and algorithms. The AIA project is predominantly an animation tool allowing students to modify various inputs and visualise the algorithms in different scenarios. The algorithms offered on the web app will have its own dedicated animation section along with a textual explanation and pseudocode.

The corresponding GitHub repository can be found here https://github.com/Sanj98/algorithms-in-action.github.io

About the clients

Name	Photo	Introduction
Harald Søndergaard		Harald Søndergaard is a Professor in the School of Computing and Information Systems at the University of Melbourne. His primary research areas are software reliability and security, program analysis, software verification, and declarative programming languages. Harald received an MSc in Computer Science in 1987 and a PhD in 1989, both from the University of Copenhagen. He has been a member of academic staff at the University of Melbourne since 1990, serving as Deputy Head of Department in 2002-2003, as Associate Dean (Learning) in the Engineering Faculty from November 2001 to June 2006, and as Assistant Dean (Teaching Quality) in the Melbourne School of Engineering from July 2007 to June 2009.
Linda Stern		Linda Stern is an Honorary Senior Fellow in the Department of Computing and Information Systems. Her research interests are bioinformatics and computer science education. Current and recent projects include: • Library Research Skills for Engineering and Information Technology – an online resource that supports research students learning to use the library effectively for their research. • Strategy for Inclusiveness in Undergraduate Computing – a report that focuses on improving the environment in computing for undergraduate students and increasing the proportion of female students in computing courses. • Algorithms in Action – client for a Software Engineering team that is redeveloping our previously successful application for learning computer science.
Lee Naish		Dr Lee Naish is a Senior Fellow in the School of Computing and Information Systems at the University of Melbourne. Lee's main research interests centre around correctness of computations. This includes significant work on declarative programming languages, semantics and debugging.

System

Existing AIA system

AIA was initially proposed by the three clients back in the 1990s. The project was re-established in 2020, and the current system is a web page developed using React framework.

It is a purely front-end system with no database and data transmission.

The current system contains five different interactive algorithms in four distinct areas. The users are able to follow the processes of each algorithm via intuitive animations. Additionally, the existing systems offers the following features -

- Background of the algorithms
- Explanation of the pseudocode
- Ability to increase the speed of the animation
- Go to previous steps after the algorithm has been implemented

Desired AIA system

There are still some bugs that need to be fixed in the current system and UI designs that can be polished to improve the user experiences.
Based on the current system design, the clients desire more algorithms to be implemented in the new system.

Scope

In Scope

The objective of Algorithm in Action is to provide a platform primarily for entry level IT students to visualise various programming algorithms using small sets of data and pseudocode in an intuitive way. A detailed explanation for the key concepts is shown in respect to the pseudocode. The sets of data should cover typical and corner cases for a broader understanding of the use cases and limitation of the algorithms. The visual of the algorithm should dynamically change according to the scope of the pseudocode and the user can customise the input for the algorithm for a better comprehension of the concepts.

Out of Scope

The are many potential areas that can be extended in the system. By simplifying the visualisation of the algorithm, larger datasets can also be demonstrated at a high level. Algorithms in Action visualises and explains the algorithms independently, but there is a possibility to extend the web app to compare similar algorithms to compare the efficiency and time complexity in different situations in the future.

Functional Requirements

This document lists the functional requirements suggested by the clients and to be fulfilled by the team.

Version 2.0

This version of the requirements has been updated to display new requirements received from clients for Sprint 2.

1. Interface

1.1 Label the speed slider so that the functionality of the slider is unambiguous to the user.

1.2 Replace the current progress bar with a simple progress bar to track the progress of code execution.

1.3 Enable click anywhere on the input box to insert/search/sort parameters so that it is easier for the user to add inputs.

1.4 The code blocks should have a feature to recursively close the code to reduce the efforts of manually closing each and every block of code.

2. Binary search tree

2.1 It should be specified explicitly that when the user clicks on the "Insert" and "Search" buttons, they enter the "Insert mode" and "Search mode" respectively.

2.2 There should be options for the execution of some basic cases such as the best-case and worst-case scenario so that it fulfils the application's teaching aim.

2.3 Map the 't' and 'p' arguments from the pseudocode with the graph, so that the user can identify these arguments clearly.

2.4 Highlight a found node in red and also, display a message "node found" to the user.

2.5 The graph must have distinguishable left and right subtrees.

2.6 The creation of a new node in the pseudocode must be collapsible to condense the execution and explanation of the algorithm for users.

2.7 Highlight the active tree and node in a different color so that the user can differentiate it from the inactive trees.

2.8 Add explanation for every line of code in the pseudocode of the algorithm.

3. Quicksort - Rightmost

3.1 Provide multiple ways of choosing a pivot element to the user.

3.2 Highlight the pivot element after it is chosen by the user.

3.3 Map the 'i' and 'j' arguments in the pseudocode with the elements in the array so that the user can keep track of the same while the algorithm is running.

3.4 Display the sorted array after the execution of the algorithm.

4. Quicksort - Median of Three

4.1 Add two checkboxes in the parameter panel for "Rightmost" and "Median of 3".

4.2 Add pseudocode for "Median of 3" to the right panel.

4.3 Add controller/animation for the "Median of 3" pseudocode to the middle panel.

4.4 Add on-click events for checkboxes - when "Rightmost" is checked, app must load the rightmost pseudocode and when "Median of 3 is checked", app must load median of 3 pseudocode.

5. Heapsort

5.1 Relabel array to 'array view' and heap to 'tree view'.

6. Graph Algorithms

6.1 Label the '+' and '-' signs as "GRAPH SIZE".

6.2 Relabel the "LOAD" button to "BUILD GAPH".

6.3 After the graph is built the "LOAD" button should change to "RESET".

7. Prim's Algorithm

7.1 Add priority queue to the algorithm in the middle panel.

8. Transitive Closure

8.1 Add a final step or result of the algorithm with respect to the "find all nodes reachable" code block in the pseudocode.

8.2 Highlight the 'i', 'j' and 'k' pointers in the graph.

8.3 Add a dynamic matrix to the app that changes values from 0 to 1 when a path between 'i' and 'j' is found.

9. Brute Force String Search

9.1 Add a new algorithm Brute Force String Search to the left panel under Searching.

9.2 Add background and pseudocode to right panel.

9.3 Add controller/animation to the pseudocode.

9.4 Add parameters in the parameter panel.

9.5 Add instructions in the middle panel.

Version 1.0

This version of the requirements highlights the initial requirements received from clients for Sprint 1.

1. Interface

1.1 Label the speed slider so that the functionality of the slider is unambiguous to the user.

1.2 Replace the current progress bar with a simple progress bar to track the progress of code execution.

1.3 Enable click anywhere on the input box to insert/search/sort parameters so that it is easier for the user to add inputs.

1.4 The code blocks should have a feature to recursively close the code to reduce the efforts of manually closing each and every block of code.

2. Binary search tree

2.1 It should be specified explicitly that when the user clicks on the "Insert" and "Search" buttons, they enter the "Insert mode" and "Search mode" respectively.

2.2 There should be options for the execution of some basic cases such as the best-case and worst-case scenario so that it fulfils the application's teaching aim.

2.3 Map the 't' and 'p' arguments from the pseudocode with the graph, so that the user can identify these arguments clearly.

2.4 Highlight a found node in red and also, display a message "node found" to the user.

2.5 The graph must have distinguishable left and right subtrees.

2.6 The creation of a new node in the pseudocode must be collapsible to condense the execution and explanation of the algorithm for users.

2.7 Highlight the active tree and node in a different color so that the user can differentiate it from the inactive trees.

2.8 Add explanation for every line of code in the pseudocode of the algorithm.

3. Quicksort

3.1 Provide multiple ways of choosing a pivot element to the user.

3.2 Highlight the pivot element after it is chosen by the user.

3.3 Map the 'i' and 'j' arguments in the pseudocode with the elements in the array so that the user can keep track of the same while the algorithm is running.

3.4 Display the sorted array aafter the execution of the algorithm.

4. Heapsort

4.1 Relabel array to 'array view' and heap to 'tree view'.

5. Graph Algorithms

5.1 Label the '+' and '-' signs as "GRAPH SIZE".

5.2 Relabel the "LOAD" button to "BUILD GAPH".

5.3 After the graph is built the "LOAD" button should change to "RESET".

6. Prim's Algorithm

6.1 Add priority queue to the algorithm in the middle panel.

7. Transitive Closure

7.1 Add a final step or result of the algorithm with respect to the "find all nodes reachable" code block in the pseudocode.

7.2 Highlight the 'i', 'j' and 'k' pointers in the matrix.

Non-Functional Requirements

1. Quality of Service (QoS)

1.1 Safety

Requirements of AIA that rule out software effect that might result in accidents, degradation, or losses in the environment.

- Product must be displayed correctly and unambiguously, having a clear view for users to understand the contents on the Product .

1.2 Security

Requirements in Product which prescribe the protection of system assets against undesirable environment behaviours.

1.2.1 Confidentiality

How the system prevents sensitive information from being disclosed to unauthorised parties.

- Users can only access the pages which shows the direct message to them, and admin or backend pages should be invisible to them.
- People who involved with this project should abide by the product regulations and don’t disclose product details.

1.2.2 Integrity

How the system ensures information may be modified only if correctly done with authorisation

- Code Walkthrough, Code Review should be contained during every sprint to ensure no error would affect the whole product.
- Different types users will have different permissions, which can be added or revoked.
- Client can access the project and give the guide of the project.

1.2.3 Availability

How the system ensures that some information or resource can be used at any point in time when it is needed and its usage is authorised.

- Team members are able to access their data at any time when they are connected to the Internet and logged into the system.
- A backup or cached version is available in situations where updates take time to resolve.
- `The carrying capacity of the system needs to be maintained to a level that can be modified by multiple people at the same time.`

1.3 Reliability

Requirements which constrain Product to operate as expected over long periods of time.

- The system shall maintain redundant backups of data for purposes of restoration in the event of data loss.
- People should be careful to merge two different branches into the main branch, system should have a dev branch.
- Test should be settled in the process to check any out-of-date date to be up-to-date over every sprint.

1.4 Performance

Quality requirements of AIA that constrain its operational conditions.

1.4.1 Time

How the system constrains the amount of time required by operations.

- Critical for response time of actions to perform within a time period of around 1 minute before an error message is displayed.
- Synchronisation within the back-end needs to happen within a small time period within the back-end, such as a mean time of 2 seconds.

1.4.2 Space

How the system constrains the amount of space required by operations.

- Databases should contain enough space to cater for the continuous introduction of data collected, and be able to scale.
- `People involved in the project should ensure that the dev branch does not have any conflicts before updating it to the main branch.`

1.4.3 Cost

How the system uses resources over a period of time

- The system maintenance should remain within the budget of the client.
- The initial system should not demand an expensive architecture to be set up.

1.4.4 Throughput

How the system constrains its overall throughput.

- Throughput of the system should match algorithm complexity,
  also the system should be able to solve the extreme situations cause the complexity of the algorithm to exceed expectations.

1.5 Interface

Quality requirements of AIA that constrain the phenomena shared by the software-to-be and the environment.

- The system interface should be easy to use.

1.5.1 User Interaction

How users of product act on the system and how the system itself acts on the user.

1.5.1.1 Usability

How product prescribes input/output formats and user dialogues to fit the abstractions, abilities, and expectations of the target users, to help with improving the usability, accessibility, and desirability provided in the interaction with the system.

1.5.1.1.1 Understandability

How the system enables the user to understand whether the software is suitable, and how it can be used for particular tasks and conditions of use.

- Software should display meaningful visualisations which are unobtrusive and understandable to its users.
- Users should be able to access the user manual from within the software if they want to dig into more complex usage of the system.
- Contents should be easy to understand for different kinds of users.

1.5.1.1.2 Accessibility

How the system provides an environment for improving ease-of-use or for people with disabilities.

- If the screen resolution changes suddenly (i.e. resizing of browser windows), the software needs to respond to the change and update the visuals without losing data.
- Software may have the option to provide display options for people with vision impairments such as colour-blindness.
- Product should be viewed in the same way on the different browsers(such as IE, Chrome, Firefox).

1.5.1.2 Convenience

Human interaction requirements which constraint software effects and describe how product ensures its users feel convenient in some system-specific sense.

- The software should have a minimal navigation structure which allows the user to perform actions with minimal effort.
- The user should be guided using tool-tips or tutorials for more complex functionality.
- If there are any animations, they should not prevent the user from navigating fast (i.e. slow page transitions).
- Avoid complex animations, should have a clear view to show the main functions.

1.5.2 Device Interaction

For interaction with other devices, how product represents its user interface, as well as other protocols which enable smooth interactions in those different environments.

- Software should conform to multiple resolutions and alter the user interface to maximise screen availability.
- Colours should be accurate, and possibly cater to people with visual disabilities such as colour-blindness.

1.5.3 Software/Service Interoperability

For interaction with existing software components, how product prescribes input/output formats and interaction protocols that enable effective cooperation with those environmental components.

- Different processes and modules in the system should communicate with each other using a standardised messaging system, regardless of the programming languages used, such as an API interface.
- The system should have a standardised interface for connecting and communicating with external APIs, so as to allow external services to communicate in the same way.
- Data exported from the platform should be in standardised formats to allow opening by other applications (i.e. Export as CSV to open in Excel).

1.6 Accuracy

How product constrains the state of information processed by the system to reflect the state of the corresponding physical information in the environment accurately.

- Unit of measurements for data should be standardised, and conversions adhere to a specific format.
- Software should handle missing data or idiosyncratic formats in a graceful manner.

2. Compliance Requirements (Standards)

How AIA prescribes software effects on the environment to conform to national laws, international regulations, social norms, cultural or political constraints, and standards.

- The activities and recommendations provided by the system should comply to human safety limits.
- Code that made by people should be created by them own or referenced in the documentation.

3. Architectural Constraint (SOA Principles)

3.1 Installation

Installation constraints in product to ensure that the software-to-be will run smoothly on the users own devices.

- The software should run without needing additional downloads on the client end.
- The system should not rely on subscription-based external services or services.
- System functionality should not rely on 3rd-party systems to function.

3.2 Distribution

Distribution constraints on software components to fit the geographically distributed structure of the software platform, the distribution of data to be processed, or the distribution of devices to be controlled.

- Source code should be in a state that can be automatically deployed on new instances with minimal effort.
- System architecture should be extensible to have multiple instances for performance improvements.

4. Development Constraint (Process)

Requirements which do not constrain the way CoachingMate should satisfy its functional requirements, but rather the way it should be developed.

4.1 Cost

How development costs are handled.

- The project development should not require funding, unless absolutely necessary.
- Usage of free development libraries is highly encouraged, unless there are no alternatives.

4.2 Deadline

Delivery schedules of artefacts.

- Requirements formulation should be done in the first three weeks of the project.
- Development of the platform should be done in the remaining weeks.
- The project should last 1 year, under the agreement with The University of Melbourne.
- The system should be fully operable after the final week of the project.

4.3 Variability

Variability of features in the system.

- The system should be welcome to changes in during the development phase, and scope should be flexible.
- Additional features may be suggested, and implemented if time and resources are sufficient.

4.4 Maintainability

The degree to which the AIA system is understood, repaired, or enhanced, by teams other than the development team.

4.4.1 Changeability

The capability of AIA to enable a specified modification to be implemented.

- The source code should have sufficient documentation to facilitate understanding of how to modify functionality.
- The product delivered should have sufficient settings to allow modification of fields and measurement units to cater for diverse users.

4.4.2 Analysability

How verbose the output generated by AIA is.

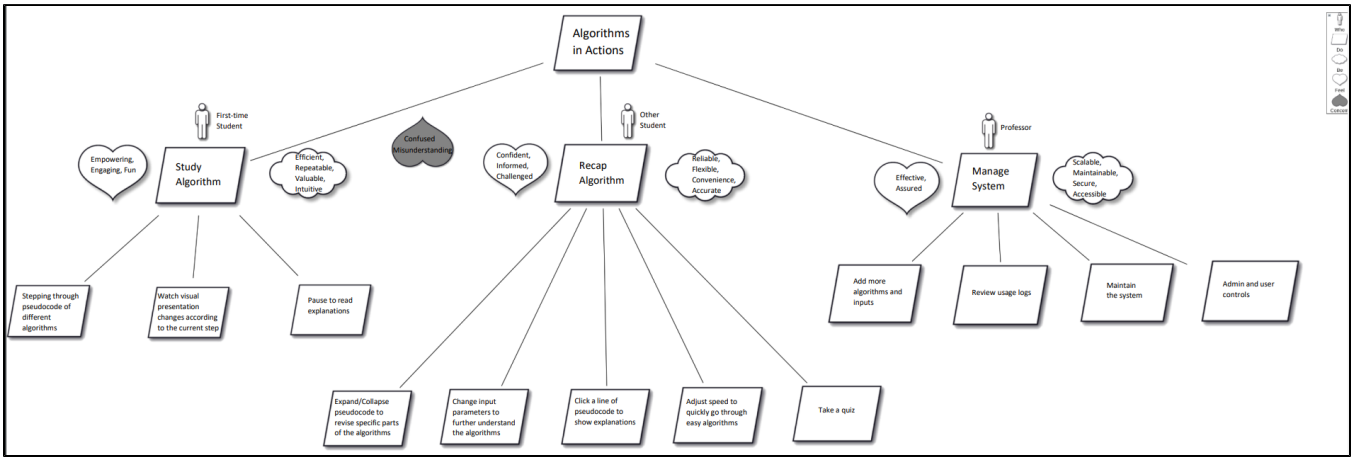
- The data stored in the system should be relational and connected with one-another.
- The system should be designed in a way to be extensible to develop recommendation capabilities or implement machine learning algorithms.
- The data collected by the system should be exportable in various different formats to suit the needs of users needing to perform further analysis.

Motivational Model

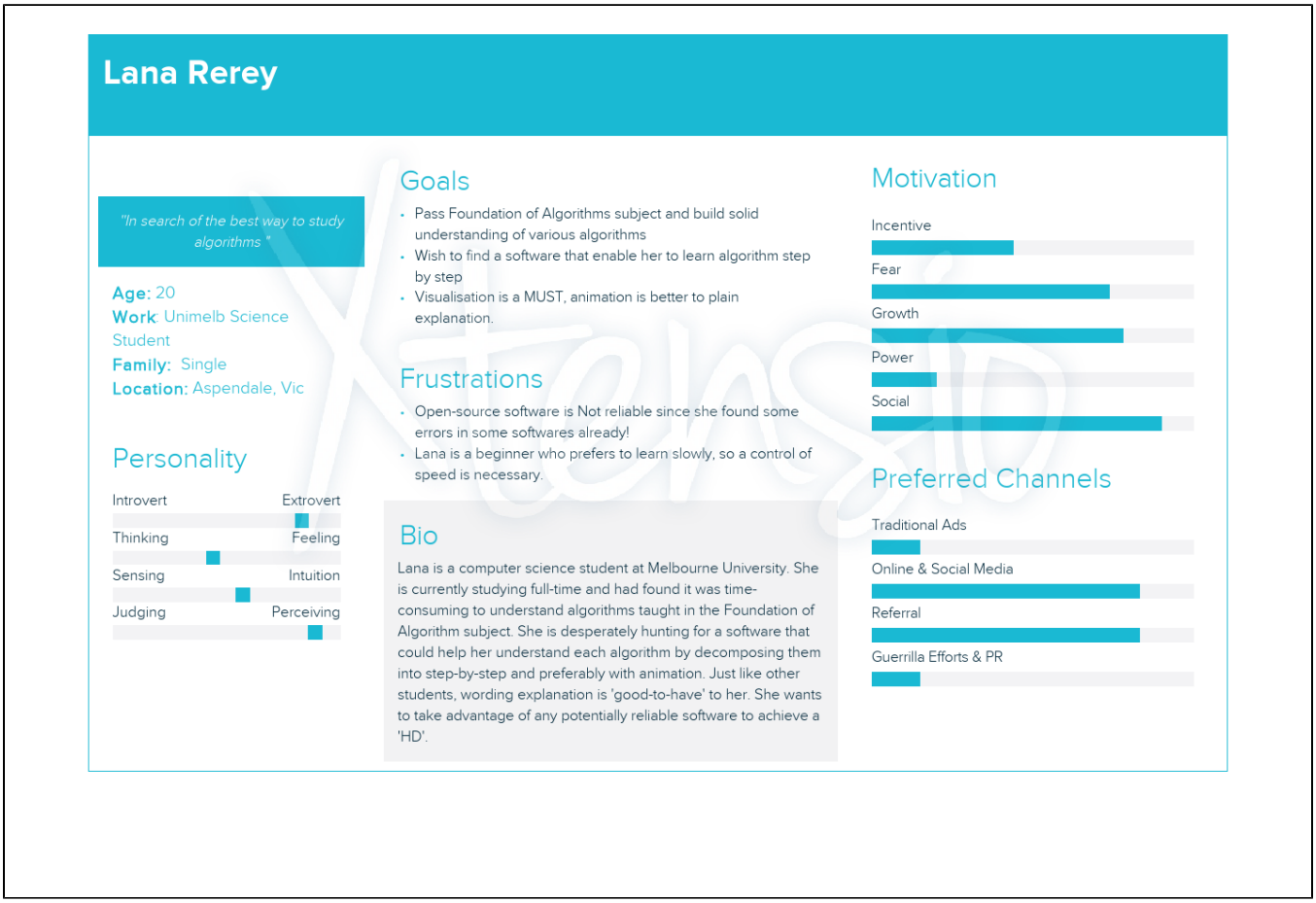
Do-Be-Feel List

Do	Be	Feel	Concern	Who
Stepping through pseudocode of different algorithms	Efficient	Fun	Confusing	First-time student
Watch visual presentation of changes according to the current step	Repeatable, Valuable	Empowering	Misunderstanding	First-time student
Pause to read explanations	Intuitive	Engaging		First-time student
Expand/Collapse pseudocode to revise specific parts of the algorithms	Flexible	Confident		Other student
Change input parameters	Convenience	Informed		Other student
Click a line of pseudocode to show explanations	Accurate	Informed		Other student
Adjust speed to quickly go through easy algorithms	Reliable	Informed		Other student
Take a quiz	Accurate	Challenged		Other student
Add more algorithms and inputs	Scalable	Effective		Professor
Review usage tags	Accessible	Assured		Professor
Maintain the system	Maintainable	Assured		Professor
Admin and user controls	Secure	Effective		Professor

Goal Model



Personas


Lana Rerey
"In search of the best way to study algorithms."
Age: 20
Work: Unimelb Science Student
Family: Single
Location: Aspendale, Vic
Personality
Introvert
Extrovert
Thinking
Feeling
Sensing
Intuition
Judging
Perceiving
Goals
Pass Foundation of Algorithms subject and build solid understanding of various algorithms
Wish to find a software that enable her to learn algorithm step by step
Visualisation is a MUST, animation is better to plain explanation.
Frustrations
Open-source software is Not reliable since she found some errors in some softwares already!
Lana is a beginner who prefers to learn slowly, so a control of speed is necessary.
Bio
Lana is a computer science student at Melbourne University. She is currently studying full-time and had found it was time-consuming to understand algorithms taught in the Foundation of Algorithm subject. She is desperately hunting for a software that could help her understand each algorithm by decomposing them into step-by-step and preferably with animation. Just like other students, wording explanation is 'good-to-have' to her. She wants to take advantage of any potentially reliable software to achieve a 'HD'.
Motivation
Incentive
Fear
Growth
Power
Social
Preferred Channels
Traditional Ads
Online & Social Media
Referral
Guerrilla Efforts & PR

Tria Lipa

"A quotation that captures this user's personality."

Age: 27
Work: Algorithm Lecturer

Family: Married, kids
Location: Melbourne, Vc
Character: Archetype

Personality

Goals

- I want to teach multiple algorithms
- I want to visualise algorithm for students.
- I want to show the pseudo-codes to student.

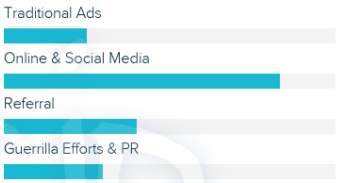
Frustrations

- No in-house algorithm visualisation software to use.
- Time-consuming and har to explain when explaining algorithm
- No tools available for students to learning algorithms after lectures

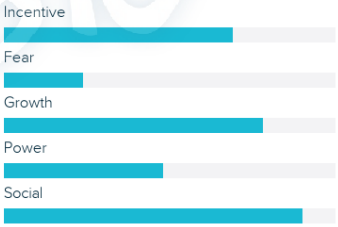
Bio

Tria is a lecturer working in Melbourne University and teaching Foundations of Algorithms. She is enthusiastic in teaching algorithmic concepts and always wanted to take the teaching quality to the next level. She tried many existing external software to visualise algorithms, but none of them is meet all her needs! She is in the hunt of a software that could let her clearly explain every steps of algorithm in class and let her student to revise anytime.

Preferred Channels



Motivation



User Stories

High Level User Stories

Story ID	Story
01	Be able to play the algorithms smoothly.
02	Be able to view the background, pseudocode and explanation of the same for each algorithm.
03	Be able to insert own input parameters.
04	Be able to increase/decrease the speed of the execution.
05	Have a user friendly, reliable and secure updated version of the web app.

Version 2.0

User stories of Sprint 2

*** The tag given to the user story is used as a branch name on GitHub and task card on Trello to ensure consistency.*

Story ID	Tag	User	Story	Priority
24A	US_24A_ADD_MATRIX_TC	Student	As a student user I should be able to view a matrix under the graph in Transitive Closure so that I can see the changes in the matrix distinctly.	Should have
24B	US_24B_ADD_MATRIX_ANIMATION_TC	Student	As a student user I should be able to view animation on the matrix so that I can see the values changing as the path is found in the pseudocode.	Should have
25	US_25_ADD_PSEUDOCODE_M3	Student	As a student user I should be able to view the pseudocode of "Median of Three" algorithm so that I can follow the animation with it.	Should have
26	US_26_ADD_ANIMATION_M3	Student	As a student user I should be able to play the animation of "Median of Three" algorithm so that I can visualize it and understand its working.	Should have
27	US_27_ADD_ONCLICK_EVENTS	Student	As a student user I should be able to click on checkboxes to choose the two ways of highlighting the pivot so that I am able to visualize them distinctly.	Should have
28	US_28_ADD_STRING_SEARCH_STRUCTURE	Student	As a student user I should be able to view a Brute Force String Search algorithm of the app so that I can visualize this algorithm and follow the pseudocode for my perusal.	Should have
29	US_29_ADD_STRING_SEARCH_LEFT_PANEL	Student	As a student user I should be able to select the Brute Force String Search algorithm at the left panel so that I can view the algorithm.	Should have
30	US_30_ADD_STRING_SEARCH_RIGHT_PANEL	Student	As a student user I should be able to view the background and pseudocode of the string search algorithm.	Should have
31	US_31_ADD_STRING_SEARCH_CONTROLLER	Student	As a student user I should be able to select inputs and control the animation of the algorithm.	Should have
32	US_32_ADD_STRING_SEARCH_INSTRUCTION	Student	As a student user I should be able to view the instructions before I run the algorithm.	Should have
33	US_33_FIX_ANIMATION_BUG	Student	As a student user I should be able to view the animation of the algorithm against the right line of the pseudocode so that it is unambiguous for me when I play the animation.	Should have
34	US_34_FIX_INDENTATION_BUG	Student	As a student user I should be able to understand the pseudocode with the right indentation so that I can comprehend its visualization correctly.	Should have

---

Version 1.0

User stories of Sprint 1.

*** The tag given to the user story is used as a branch name on GitHub and task card on Trello to ensure consistency.*

Story ID	Tag	User	Story	Priority

01	US_01_ADD_MODES	Student	As a student user I should be able to view Insert and Search as modes for the Binary Search Tree algorithm so that I am aware that the algorithm ideally offers two different types of functionalities and can differentiate between the two.	Should have
02	US_02_LABEL_SPEED_SLIDER	Student	As a student user I should be able to see the speed slider labelled as speed so that it is unambiguous for me in terms of its usability.	Should have
03	US_03_ADD_PROGRESS_BAR	Student	As a student user I should be able to see a simple progress bar on the interface so that I can view how long the algorithm is going to take to run.	Must have
04	US_04_CLICK_BOX_FOR_INPUT	Student	As a student user I should be able to click anywhere on the input box to insert/search/sort parameters so that it is easier to add my inputs to run the algorithm.	Could have
05	US_05_ADD_BASE_CASES	Student	As a student user I should be able to run some basic cases of the algorithms so that I have a complete understanding of the same and I can apply it to my coursework.	Should have
06	US_06_CLOSE_NESTED_BLOCKS	Student	As a student user I should be able to recursively close the nested blocks when I close the respective parent block so that I don't have to do it manually for every block of code.	Must have
07	US_07_ADD_POINTERS_BST	Student	As a student user I should be able to see which element in the tree is 't' and 'p' so that I can follow the pseudocode effectively.	Must have
08	US_08_HIGHLIGHT_FOUND_NODES	Student	As a student user I should be indicated when a node is found or not found after the algorithm is implemented so that I can relate to the algorithm clearly.	Should have
09	US_09_SPLIT_TREE	Student	As a student user I should be able to see the left and right trees separately in the Binary Search Tree algorithm so that I can distinguish between the two.	Must have
10	US_10_NEW_NODE_COLLAPSIBLE	Student	As a student user I should be able to expand and collapse the lines of code pertaining to the creation of a new node so that I can run the algorithm at a higher level.	Should have
11	US_11_HIGHLIGHT_ACTIVE_TREE	Student	As a student user I should be able to differentiate between the node and tree being investigated from the rest so that I can relate to the algorithm clearly.	Could have
12	US_12_ADD_EXPLANATION_BST	Student	As a student user I should be able to view an explanation for each line of pseudocode for Binary Search Tree algorithm so that I can understand what's happening with respect to every line in the code.	Could have
13	US_13_CHOOSE_PIVOT	Student	As a student user I should have options for different ways of choosing a pivot for the Quicksort algorithm so that I can select the pivot as per my way.	Should have
14	US_14_HIGHLIGHT_PIVOT	Student	As a student user I should be able to distinguish the pivot element from others after I have chosen it so that I can keep track of the pivot when the algorithm is running.	Must have
15	US_15_ADD_POINTERS_QS	Student	As a student user I should be able see which elements in the array are 'i' and 'j' so that I can spot them easily and understand the algorithm clearly.	Must have
16	US_16_DISPLAY_SORTED_ARRAY	Student	As a student user I should be able to view a sorted version of my input after the implementation of the algorithm so that I can compare it with the initial input given.	Must have
17	US_17_CHANGE_LABELS_FOR_VIEWS	Student	As a student user I should be able to see two different views distinctly, that is array view and tree view so that I can clearly distinguish between them.	Must have
18	US_18_LABEL_GRAPH_SIZE	Student	As a student user I should be able to interpret that the '+' and '-' buttons are for increasing and decreasing the size of the graph so that I do not mistake it for any other functionality and I can increase/decrease the graph size as I need.	Must have
19	US_19_CHANGE_LOAD_BUTTON	Student	As a student user I should be able to click on a "Build Graph" button to create my graph so that I can run the algorithm and view the result.	Must have
20	US_20_ADD_RESET_BUTTON	Student	As a student user I should be able to reset the graph after it has loaded by clicking on a button so that I do not have to change all the values manually incase I entered wrong values.	Must have
21	US_21_DISPLAY_PRIORITY_QUEUE	Student	As a student user I should be able to view a priority queue for Prim's algorithm so that I am aware of the changes taking place in the queue.	Must have
22	US_22_ADD_FINAL_RESULT	Student	As a student user I should be able to see the final result of Transitive Closure algorithm when the algorithm reaches the "find all nodes reachable" lines of code so that I can skip the sub-steps of this part of the algorithm.	Must have
23	US_23_ADD_POINTERS_TC	Student	As a student user I should be to locate which values are 'i', 'j' and 'k' in the graph so that I can keep up with the algorithm and avoid any unambiguity.	Should have

Product Backlog

Sprint 1

Sprint 1A - 22 Mar 2021 to 29 Mar 2021

Sprint 1B - 07 Apr 2021 to 16 Apr 2021

Sprint 1C - 17 Apr 2021 to 27 Apr 2021

Sprint 2

Sprint 2 - 04 May 2021 to 24 May 2021

Backlog

*** Story points are estimated in number of days*

FEATURE	STORY ID	USER	STORY	STORY ESTIMATE	PRIORITY	SPRINT 1A	SPRINT 1B	SPRINT 1C	SPRINT 2
INTERFACE	01	Student	As a student user I should be able to view Insert and Search as modes for the Binary Search Tree algorithm so that I am aware that the algorithm ideally offers two different types of functionalities and can differentiate between the two.	1	Should have	☑	☐	☐	
	02	Student	As a student user I should be able to see the speed slider labelled as speed so that it is unambiguous for me in terms of its usability.	1	Should have	☑	☐	☐	
	03	Student	As a student user I should be able to see a simple progress bar on the interface so that I can view how long the algorithm is going to take to run.	2	Must have	☑	☐	☐	
	04	Student	As a student user I should be able to click anywhere on the input box to insert/search/sort parameters so that it is easier to add my inputs to run the algorithm.	2	Could have	☐	☐	☐	☐
	05	Student	As a student user I should be able to run some basic cases of the algorithms so that I have a complete understanding of the same and I can apply it to my coursework.	4	Should have	☐	☐	☑	☐
PESUDOCODE	06	Student	As a student user I should be able to recursively close the nested blocks when I close the respective parent block so that I don't have to do it manually for every block of code.	2	Must have	☑	☐	☐	☐
BINARY SEARCH TREE	07	Student	As a student user I should be able to see which element in the tree is 't' and 'p' so that I can follow the pseudocode effectively.	3	Must have	☐	☑	☐	☐
	08	Student	As a student user I should be indicated when a node is found or not found after the algorithm is implemented so that I can relate to the algorithm clearly.	2	Should have	☐	☐	☑	☐
	09	Student	As a student user I should be able to see the left and right trees separately in the Binary Search Tree algorithm so that I can distinguish between the two.	2	Must have	☐	☑	☐	☐
	10	Student	As a student user I should be able to expand and collapse the lines of code pertaining to the creation of a new node so that I can run the algorithm at a higher level.	2	Should have	☐	☐	☑	☐
	11	Student	As a student user I should be able to differentiate between the node and tree being investigated from the rest so that I can relate to the algorithm clearly.	3	Could have	☐	☐	☑	☐
	12	Student	As a student user I should be able to view an explanation for each line of pseudocode for Binary Search Tree algorithm so that I can understand what's happening with respect to every line in the code.	3	Could have	☐	☑	☐	☐
QUICKSORT	13	Student	As a student user I should have options for different ways of choosing a pivot for the Quicksort algorithm so that I can select the pivot as per my way.	4	Should have	☐	☐	☑	☐

	14	Student	As a student user I should be able to distinguish the pivot element from others after I have chosen it so that I can keep track of the pivot when the algorithm is running.	2	Must have	☐	☑	☐	☐
	15	Student	As a student user I should be able see which elements in the array are 'i' and 'j' so that I can spot them easily and understand the algorithm clearly.	2	Must have	☐	☑	☐	☐
	16	Student	As a student user I should be able to view a sorted version of my input after the implementation of the algorithm so that I can compare it with the initial input given.	3	Must have	☐	☑	☐	☐
HEAPSORT	17	Student	As a student user I should be able to see two different views distinctly, that is array view and tree view so that I can clearly distinguish between them.	1	Must have	☑	☐	☐	☐
GRAPH ALGORITHMS	18	Student	As a student user I should be able to interpret that the '+' and '-' buttons are for increasing and decreasing the size of the graph so that I do not mistake it for any other functionality and I can increase/decrease the graph size as I need.	1	Must have	☑	☐	☐	☐
	19	Student	As a student user I should be able to click on a "Build Graph" button to create my graph so that I can run the algorithm and view the result.	1	Must have	☑	☐	☐	☐
	20	Student	As a student user I should be able to reset the graph after it has loaded by clicking on a button so that I do not have to change all the values manually incase I entered wrong values.	1	Must have	☑	☐	☐	☐
PRIM'S ALGORITHM	21	Student	As a student user I should be able to view a priority queue for Prim's algorithm so that I am aware of the changes taking place in the queue.	3	Must have	☐	☑	☐	☐
TRANSITIVE CLOSURE	22	Student	As a student user I should be able to see the final result of Transitive Closure algorithm when the algorithm reaches the "find all nodes reachable" lines of code so that I can skip the sub-steps of this part of the algorithm.	4	Must have	☐	☑	☐	☐
	23	Student	As a student user I should be to locate which values are 'i', 'j' and 'k' in the graph so that I can keep up with the algorithm and avoid any unambiguity.	3	Should have	☐	☐	☑	☐
	24A	Student	As a student user I should be able to view a dynamic matrix under the graph in Transitive Closure so that I can see the values changing as the path is found in the pseudocode.	2	Should have	☐	☐	☐	☑
	24B	Student	As a student user I should be able to view animation on the matrix so that I can see the values changing as the path is found in the pseudocode.	2	Should have	☐	☐	☐	☑
QUICKSORT-MEDIAN OF THREE	25	Student	As a student user I should be able to view the pseudocode of "Median of Three" algorithm so that I can follow the animation with it.	3	Should have	☐	☐	☐	☑
	26	Student	As a student user I should be able to play the animation of "Median of Three" algorithm so that I can visualize it and understand its working.	3	Should have	☐	☐	☐	☑
	27	Student	As a student user I should be able to click on checkboxes to choose the two ways of highlighting the pivot so that I am able to visualize them distinctly.	1	Should have	☐	☐	☐	☑
BRUTE FORCE STRING SEARCH	28	Student	As a student user I should be able to view a Brute Force String Search algorithm of the app so that I can visualize this algorithm and follow the pseudocode for my perusal.	4	Should have	☐	☐	☐	☑
	29	Student	As a student user I should be able to select the Brute Force String Search algorithm at the left panel so that I can view the algorithm.	1	Should have	☐	☐	☐	☑
	30	Student	As a student user I should be able to view the background and pseudocode of the string search algorithm.	3	Should have	☐	☐	☐	☑
	31	Student	As a student user I should be able to select inputs and control the animation of the algorithm.	2	Should have	☐	☐	☐	☑
	32	Student	As a student user I should be able to view the instructions before I run the algorithm.	1	Should have	☐	☐	☐	☑
GENERAL BUGS	33	Student	As a student user I should be able to view the animation of the algorithm against the right line of the pseudocode so that it is unambiguous for me when I play the animation.	3	Should have	☐	☐	☐	☑

	34	Student	As a student user I should be able to understand the pseudocode with the right indentation so that I can comprehend its visualization correctly.	3	Should have	☐	☐	☐	☑

Total story points = 76