

Design and implementation of a web application for practising internet reading skills

UNIVERSITY OF TURKU
Department of Computing
Master of Science in Technology Thesis
Software Engineering
May 2022
Kasper Rautio

Supervisors:
Tuomas Mäkilä
Seppo Virtanen

The originality of this thesis has been checked in accordance with the University of Turku quality assurance system using the Turnitin OriginalityCheck service.

UNIVERSITY OF TURKU
Department of Computing

Kasper Rautio

Design and implementation of a web application for practising internet reading skills

Master of Science in Technology Thesis, 68 p.
Software Engineering
May 2022

Being able to critically evaluate the reliability of sources is an important skill, and a part of digital literacy competences and internet reading. Conducting research on the matter and teaching the skills related to internet reading in a digital environment require discrete software that is designed for said purposes. This thesis is associated with a project where one of the objectives is to design and implement a new web application, StudentNet, where pupils can practise and improve their skills in internet reading by completing assignments. Internet reading is goal-directed activity where the reader a) searches for information to set questions from the Internet, b) evaluates the reliability of the found information, and c) composes a synthesis based on several online sources.

The basis of StudentNet is a web application called Neurone, which main purpose is research and measurement instead of education, and it provides a linear user flow. In contrast, StudentNet should provide a nonlinear model regarding the application's user flow. In this thesis it is studied how this nonlinear model works in practice especially when the users are children, and what specific things should be considered when developing software for educational purposes.

Several practical experiments were organised during the project. In these experiments, each participant completed an assignment in StudentNet and answered a set of statements in an end questionnaire. In addition, log data of users' interactions in the application was recorded. The data collected from the questionnaire was analysed and utilised to get some information on what could be improved, and ideas for the development of new features as well as improving existing ones. In addition, the gathered feedback was used to answer the research question considering the application's nonlinear model. It was found out that the model works very well for most pupils but for some individuals there should be additional support in the application in order to ensure a great user experience. The results also indicate that StudentNet was perceived useful, and it has good potential for further development and to be used as a complementary learning tool.

Keywords: educational software, web application, internet reading, digital literacy

Contents

1	Introduction.....	1
1.1	Motivation and scope.....	2
1.1.1	Requirements for the application	2
1.1.2	Collecting and analysing data	3
1.2	Research questions	4
1.3	Research methods	5
1.4	Thesis structure.....	6
2	Web applications.....	7
2.1	Essential web concepts	8
2.2	Advantages and disadvantages	8
2.3	Three-tier architecture.....	9
2.4	Architecture types	11
2.4.1	Multi-Page Application (MPA).....	11
2.4.2	Single-Page Application (SPA).....	11
2.4.3	Progressive Web Application (PWA).....	13
2.4.4	Isomorphic Application	14
2.5	Design patterns.....	14
2.5.1	Model-view-controller (MVC).....	14
2.5.2	Hierarchical model-view-controller (HMVC).....	16
2.5.3	Model-view-adapter (MVA).....	16
2.5.4	Model-view-presenter (MVP).....	17
2.6	Frameworks and libraries	17
2.6.1	React.....	18
2.6.2	Angular	23
2.6.3	Vue.....	24
3	Educational software	26
3.1	Types	27
3.1.1	Tutorials.....	27
3.1.2	Courseware.....	27
3.1.3	Edutainment.....	28
3.1.4	Reference software.....	28
3.2	Benefits of educational software.....	28
3.3	Design and development.....	31
4	Neurone.....	33
4.1	Default stages	34
4.1.1	Search and Selection	34
4.1.2	Collection.....	34
4.1.3	Critical Evaluation	34
4.1.4	Synthesis.....	35
4.2	Technologies.....	35
4.2.1	MongoDB	35
4.2.2	Apache Solr.....	36
4.2.3	Meteor.....	36

5	StudentNet	38
5.1	Technologies.....	38
5.1.1	Node.js	39
5.1.2	Npm	41
5.1.3	Express	41
5.1.4	Bootstrap	42
5.1.5	PostgreSQL.....	42
5.1.6	Elasticsearch	43
5.1.7	React Query	44
5.1.8	React Table	46
5.2	Modules	47
5.2.1	Search.....	47
5.2.2	Bookmarks.....	49
5.2.3	Snippets.....	49
5.2.4	Evaluation	50
5.2.5	Synthesis.....	51
6	Experiments.....	53
6.1	Pilot experiment.....	53
6.2	Experiment at Rauma Teacher Training School	55
6.2.1	Structure.....	56
6.2.2	Assignment.....	57
6.2.3	End questionnaire.....	57
6.2.4	Analysis	58
7	Discussion	61
7.1	Evaluation.....	64
7.2	Further development.....	65
8	Conclusion	67

1 Introduction

In the 21st century the Internet has become generally speaking the single most used resource for finding information and learning. As of January 2021, there were 4.6 billion people actively connected to the Internet. That is nearly 60 percent of the global population [1]. With an ever-growing amount of information available online, it cannot be denied that the Internet is an effective and convenient tool for information searching. However, its proper and efficient utilisation requires some skills and knowledge that can be learned and improved with practise.

One particularly important skill is being able to critically evaluate the reliability of sources, which is a part of internet literacy competences. However, it seems that there is room for improvement regarding internet reading skills of primary school pupils. Conducting research on the matter and teaching skills related to online learning require discrete software that is designed for said purposes.

Internet reading (also called online reading or digital reading) is the central topic of the application that this thesis discusses. Internet reading is goal-directed reading where the reader:

- Searches for information to set questions from the Internet
- Evaluates the reliability of the found information
- Composes a synthesis based on several online sources

Internet reading usually demands multiliteracy, since the Internet is a diverse environment with different types of content. A multiliteracy-competent pupil possesses the skills to acquire and edit information, as well as evaluate, produce, and, present information in different forms and with different tools. Practicing internet reading also supports the development of multiliteracy skills. [2]

1.1 Motivation and scope

There is a need for research on the matter of internet reading skills of students. The application that has been used for this kind of research, called 'Neurone', uses some obsolete technologies, has limitations and its user flows do not match real world situations well. The main purpose of Neurone is to facilitate research and measurement for the subject matter, education being a secondary intent.

The goal of this thesis is to design and implement a web application that provides the same functionalities than Neurone but with improvements, and allows for an experience that is closer to a real online learning situation. It should also be more suitable for education purposes. The new application is built from the ground up, starting with software technology considerations.

1.1.1 Requirements for the application

The focus of this thesis is on the technical implementation of a web application that offers a closed environment for teaching and learning internet reading. The new application should include the same core functionalities than the current application but should provide a more flexible model where the user can transition freely between different stages. The application is supposed to cover four digital literacy competence dimensions with corresponding main modules:

1. Searching for information and selecting relevant sources
2. Finding the most important parts in the selected sources
3. Evaluating the reliability and relevance of the selected sources
4. Synthesising information across the selected sources

Searching for information is done with a search engine in a restricted environment. The search engine should look like an internet search engine, but it only searches from documents located on a private database. Available documents can be determined by a researcher or a teacher according to the given assignment.

Selecting the relevant sources is implemented with a Bookmarks module. When viewing a document page after opening it from the search results, the user is given the possibility to save the page as a bookmark. Bookmarks are saved in the application's database.

Identifying the main ideas from each source can be accomplished with the Snippets-module. It should provide the possibility to select the most important parts of a bookmarked source and save them as text snippets that can be utilised when writing the synthesis.

Evaluation of the reliability and relevance of selected sources can be done on the source page. An intuitive rating component should be provided. Each document has a pre-determined objective rating for reliability and relevance. The pre-determined relevance rating is given in relation to each assignment.

The synthesis module should include a simple but sufficient text editor for composing a synthesis based on the selected sources. Ideally, the saved snippets should be visible on the synthesis page, as well as the whole source texts easily accessible. For example, the whole source text can be shown as an overlay or in a new window when a snippet is mouse-overed or clicked.

1.1.2 Collecting and analysing data

In addition to discussing and describing the development of the web application and related theory, this thesis' scope includes collecting data from experiments that address testing of both Neurone and StudentNet. The collected data is analysed, and the conclusions can be utilised to assist in the application development.

Quantitative data is collected by logging users' actions when they are using the software. This data can be used to compare Neurone and StudentNet, and to find things that could be improved. Log data can also show if there are some differences between the two different applications considering the behaviour of users.

Qualitative data is collected with inquiries and by interviewing persons who have completed test assignments in both Neurone and StudentNet. This qualitative data is analysed and composed into possible conclusions that can support the development of the application and give answers to the research questions.

1.2 Research questions

There are two research question this thesis tries to answer:

1. What specific things should be considered when developing an application for educational purposes?
2. How does a nonlinear model work in an educational application, regarding the user experience of children?

The first question asks if there are any specific needs or requirements for a software application that has pedagogical objectives. This question is supposed to be answered during the development of the application by searching and studying literature on the topic, and when designing features for the application with people whose background is in educational sciences.

The Neurone application is restricted so that the user must go through the different phases in a determined order. In contrast, StudentNet is supposed to offer a more open and flexible environment that simulates a real-world situation better. The answer to the second question can be found from the conclusions that are made from the analysed data, as well as studying literature on the matter.

1.3 Research methods

There are several data collection techniques suitable for field studies but in this project the techniques are limited to questionnaires and interviews. The main advantage of questionnaires is that they are time and cost effective [3]. They are quite effortless to make, especially with web-based tools. A disadvantage is that since there is no interviewer, questions must be very well worded and thought out so they are not ambiguous. Interviews, on the other hand, are time and cost inefficient, but an efficient means of collecting the same data from a large number of respondents [3].

The principal research method in this project is empirical research, that is put into practice with controlled experiments. In these experiments, quantitative data is collected by logging each participants actions while they are using the applications during an experiment, whereas qualitative data is collected via questionnaires and interviews. The collected data is analysed, and the learnings are used to improve and to help with the further development of the new application. The experiments are conducted for both Neurone and StudentNet in the same manner, so they can also be compared based on the results.

Controlled experiments allow for conducting well-defined studies with the possibility to focus on specific variables and measures. This gives the potential to provide statistically significant results. This kind of a study provides good insights for the reasons of relationships and results and forces to analyse the threats to validity. [4]

1.4 Thesis structure

After this introduction there are seven chapters:

Chapter 2 is about web applications in general. It provides information about essential web concepts, web application architecture types and technologies as well as the advantages and disadvantages of web applications.

Chapter 3 discusses educational software. The different types of educational software are presented, along with the benefits of using educational software. Finally, the most important things to consider when designing educational software are addressed.

Chapter 4 introduces the Neurone application which was used as the basis of StudentNet. First, an overview of Neurone is provided, including its purpose and structure. Then, the different modules and main technologies are described.

Chapter 5 is about the application at the centre of this thesis, StudentNet. The technologies that were chosen are covered, along with some rationale for the choices. The modules of StudentNet as well as their implementations are addressed.

Chapter 6 provides information about the experiments that were utilised as a source of data for this thesis. The arrangements and structures of the experiments are presented, as well as analysis of the results.

Chapter 7 consists of discussion about the results of the experiments that were described in chapter 6, with evaluation. In addition, improvements and features that are planned to be implemented in the future are discussed.

Chapter 8 is the conclusion chapter where the research questions are answered in a short and clear manner, and contains general discussion about the whole project.

2 Web applications

People use web applications (commonly “web app”) every day. Email and social media services, online office software and online shops are some examples of commonly used web apps. With the development of technologies, web apps have become exceedingly sophisticated and feature-rich, and therefore increasingly complex applications can be implemented as web apps. In addition to the most used and relatively simple types of apps, there are web apps for purposes such as graphic design, video editing, project management, and more.

A web application is a client-server software program that is accessed with a web browser. The logic is distributed among the client which initiate users’ requests, and the server or servers, which are responsible for the business logic and the database. [5] The client requests data from the server usually through an application programming interface (API). Web APIs can be public or private; private APIs require an access token.

In terms of technical implementation, a web application can be either static or dynamic. ‘Static’, essentially meaning a traditional website, can be considered an architecture type on its own, while ‘dynamic’ comprises multiple subtypes that are discussed later. The contents of a static website are generated by a server and offer virtually no interactivity at all, whereas a dynamic web app’s content changes depending on the user’s actions. [6] In addition to interactivity, a web app allows for authentication and extra functionality compared to a static website [5].

2.1 Essential web concepts

The World Wide Web (WWW, “web”) has grown and evolved enormously since its early years but the web of today still incorporates the same core elements [7]:

- Uniform Resource Locator (URL) to uniquely identify and locate a resource on the web.
- Hypertext Transfer Protocol (HTTP) for transmitting hypermedia documents, such as HTML.
- A software program (web server) that can respond to HTTP requests.
- Hypertext Markup Language (HTML) to define the meaning and structure of web content that a browser can render.
- Cascading Style Sheets (CSS) for creating good-looking interfaces.
- JavaScript (conforms to the ECMAScript specification) for scripting the behaviour of websites and web apps.
- JavaScript Object Notation (JSON) for representing structured data based on JavaScript object syntax.
- Document Object Model (DOM) for representing the objects that comprise the structure and content of a document on the web.
- Application Programming Interface (API) that services and products use to communicate with each other.
- A program (web browser) that can make HTTP requests from URLs and render the HTML content it receives.

2.2 Advantages and disadvantages

There are a few prominent differences between web applications and desktop applications. For example, the location of data storage is different, and web applications have limited access to operating system features. Web applications do not require installation; they can be accessed by merely navigating to the desired URL with a supported browser. Up-to-date versions of the most popular browsers - Chrome, Safari, Firefox, and Edge – support HTML5 form features. Updates are

installed automatically, which is most convenient for users. Web apps are also platform-independent which means that most web applications can run on virtually any system, including mobile devices [8].

Web applications also have some disadvantages compared to desktop applications. Since the data is stored on a remote server, or in the cloud, the question of security arises. End users cannot necessarily be sure that the data controller has proper safety measures implemented. The dependence on internet access could be considered a disadvantage, but at the present it is rarely a problem since the internet can be accessed virtually everywhere, and modern software have safeguards against disconnections. While the independence on hardware was mentioned as an advantage above, it can also be considered a disadvantage. Since web applications have less processing power available, very complicated programs can perform slower in comparison to desktop equivalents. Many web applications, if not most, require a user account in order to be used, which may also be considered a disadvantage. [8]

2.3 Three-tier architecture

Modern web applications use the three-tier architecture concept, where the application is separated into three computing tiers that each have a specific purpose. The tiers are called presentation, application, and data. Three-tier architecture is the predominant software architecture for traditional client-server applications. This chapter discussing three-tier architecture is based on source [9].

In software that conforms to three-tier architecture each tier runs on its own infrastructure. Consequently, each of the tiers can be developed at the same time by different development teams without impacting the other tiers. This can be considered a significant benefit of the three-tier architecture.

Presentation tier

This top-level tier provides the UI and the possibility for users to interact with the application. The presentation tier is responsible for displaying information to and collecting information from the user. The presentation tier can run, for instance, on a web browser, a mobile application, or as desktop application. Presentation tiers in web apps, also known as web servers, are typically developed using HTML, CSS, and JavaScript or TypeScript. In contrast, desktop applications can be written in almost any programming language, depending on the platform.

Application tier

The application tier is the core of the whole application, where all dynamic content and the interactions between the presentation tier and the data tier is processed. This tier contains the business logic, sometimes called domain logic. The business logic is a specific set of business rules that define how data can be processed and stored. The application tier communicates with the data tier in the form of API calls and can mutate the data located in the data tier. All communication between the presentation tier and the data tier goes through the application tier. They cannot communicate directly with each other. In a web application this tier usually runs an environment or framework such as Node.JS, Django, Laravel or ASP.NET.

Data tier

The managing of information processed by the application happens in the data tier and is the tier where the information is stored in a database. The selected database solution can be a relational database management system (RDBMS) such as MySQL, Oracle, PostgreSQL, MariaDB, Microsoft SQL, or a NoSQL Database such as Apache Cassandra, Redis, or the currently most popular choice MongoDB.

2.4 Architecture types

Application architecture refers to the fundamental, high-level structure that determines how a software product or business will operate and scale. It functions as a blueprint of the system and comprises software elements, relations between them, and properties of those relations and elements. In this section the different architecture types of web applications are presented.

2.4.1 Multi-Page Application (MPA)

Multi-page applications are the traditional approach to web app development. MPAs consist of a collection of pages that are entirely reloaded every time when the data required by a page changes. MPAs are typically implemented with server-side rendering. In an application or website that is server-side rendered, the content is retrieved from the back end and transferred to the client's browser to be displayed on the screen [5]. Whenever a website is visited, the browser makes a request to the server that holds the contents of the website. The request usually does not take more than a few milliseconds, but depends on factors such as internet connection speed, the server's location, and the number of concurrent users.

While SPAs – discussed in the next chapter – are conquering the world of web development, there are still use cases for traditional multi-page applications. Examples of MPAs are large ecommerce sites such as Amazon and eBay, which feature numerous categories and pages with both static and dynamic elements.

2.4.2 Single-Page Application (SPA)

Most modern websites and web applications are implemented as single page applications. In SPAs, all the content such as CSS styling, images, and JavaScript are loaded at one time when the page initially loads. After that, the content and components are loaded dynamically according to the user's interactions. When the initial page loading has been done, every subsequent request is handled more

quickly because only the particular section of the page is refreshed, instead of reloading the entire page or application [10]. Single-page applications are all about serving a great user experience by minimising reloads and thus waiting time. This is made possible by utilising a programming concept called AJAX, short for "Asynchronous JavaScript and XML". AJAX enables the fetching of content to web pages using JavaScript included within the HTML, without the need to re-render the entire page. AJAX was new and revolutionary in the early 2000's web technology. The term itself has somewhat faded away but the approach is today so commonplace that it is taken for granted. [11] The lifecycles of a multi-page app and a single page app are shown in Figures 1 and 2.

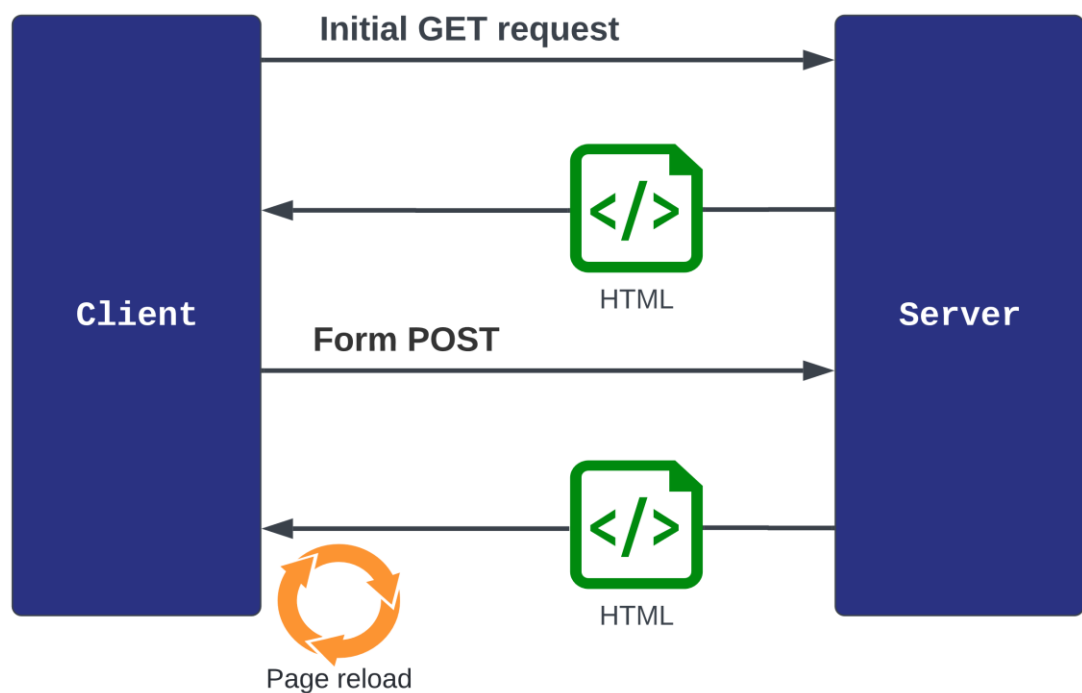


Figure 1. The lifecycle of a multi-page app. Data changes or data transfers to the server cause the associated page to be reloaded in the browser.

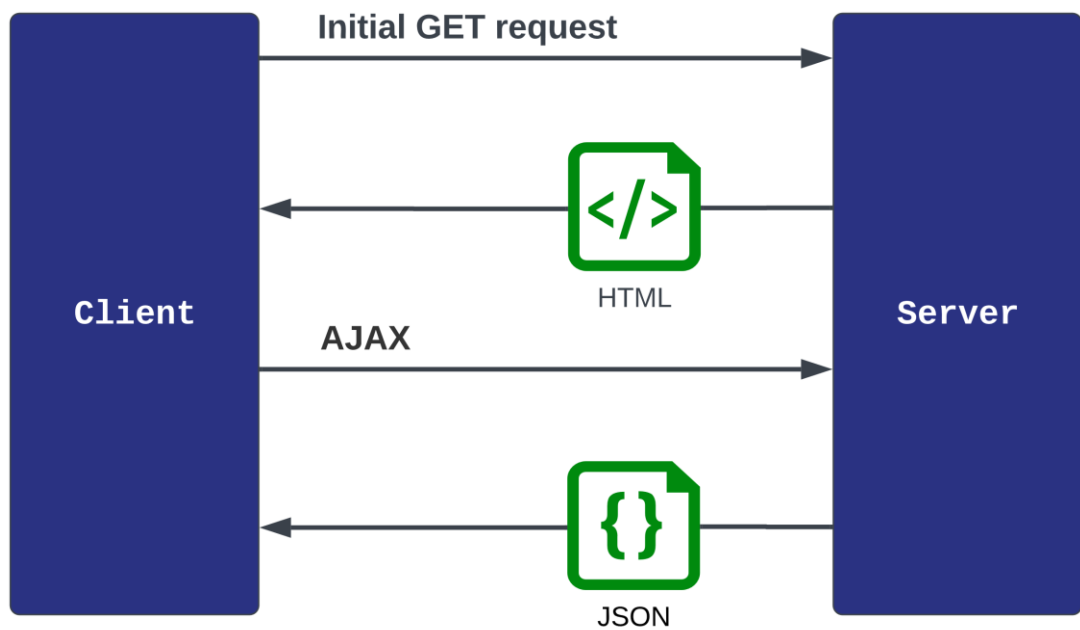


Figure 2. The lifecycle of a single page app. On the initial GET request, the server loads the entire page. Afterwards, responding to subsequent requests, only the necessary data is transferred in the JSON format.

2.4.3 Progressive Web Application (PWA)

Progressive web applications are a rather recent addition to the software development industry, although the concept is not brand new. There have been numerous different kinds of approaches to implement the ideas in the past. The term "Progressive Web App" is not an official name; it was used first by Google for the concept of building a flexible and adaptable application with web technologies. [12]

Briefly, the purpose of PWAs is to combine the advantages of desktop applications and web applications. They are built with the same web development technologies as regular web applications but aim to give a native-like feeling to the user. An application can be classified as a PWA when it implements some particular features or meets specific requirements. For instance, it can be used offline, it is easy to synchronise, and can send push notifications to the user. [12]

2.4.4 Isomorphic Application

Isomorphic is a modern web app architecture used for JavaScript applications that can run both on the client-side and the server-side. It functions so that the JavaScript app is sent to the client browser when the client initially loads the HTML. Afterwards the app starts running like a single page application.

Contrary to a server-side rendered app, the isomorphic web architecture enables good responsiveness, multiple UI and UX options, and quicker data updates and rendering when the server is loaded. Isomorphic architecture also grants an instant display in the browser, user-friendly routing, and search engine optimisation. As a drawback, this type of web app architecture is fully supported only by JavaScript which means that the possible technology stacks are limited to JS-based frameworks and tools. [5]

2.5 Design patterns

There are some software design patterns that are widely used in web applications, most notably the model-view-controller (MVC) pattern. While MVC is arguably the most popular design pattern in web applications, there are also a few other design patterns that are variations of the MVC.

2.5.1 Model-view-controller (MVC)

The MVC design pattern was adopted in software development as a solution to separate presentation from application logic and data storage. A few other design patterns and paradigms also exist that attempt solve the problem, but the focus has been on the MVC model. MVC was generally used for desktop application GUIs in the past but has since become the most popular pattern for designing web applications and mobile apps. [13]

MVC was first envisioned by Trygve Reenskaug in 1979 and introduced in the Smalltalk programming language in 1988. At that time, it was stressed out that there are great benefits to be had if applications are built with modularity in mind. With MVC, the responsibilities in an application are split into three components: model, view, and controller. [14]

- Model component contains the application's business logic. It holds the database and manages all data-related tasks, such as data validation and session state. The model responds to requests from the view and controller.
- View presents the data to the user by managing the GUI. The data required by the view is collected from the model component.
- Controller is the component with the purpose of handling all events. Events can be triggered by user interactions or a system process. The controller instructs the model to update its state accordingly and sends commands to its associated view to change that view's presentation.

The most prominent benefits of using MVC include easier maintenance of code and better extensibility, as well as better support for new kinds of clients. Development of the different components can be done parallelly. It also has the best support for test-driven development, offers good separation of concerns (SoC), and is Search Engine Optimisation (SEO) friendly. [13] Figure 3 shows how some functionality of a simple shopping list app could be implemented using MVC.

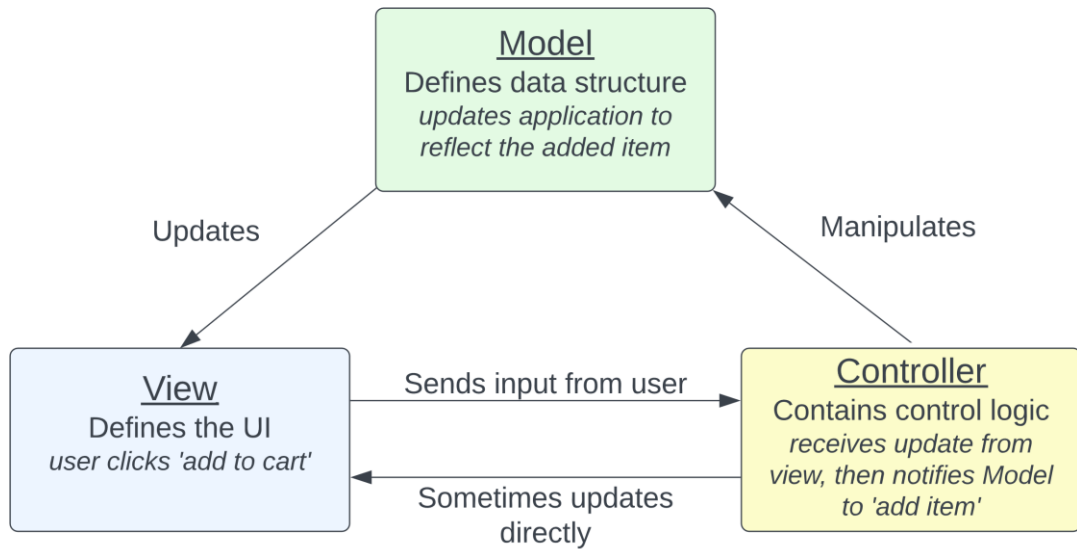


Figure 3. The MVC design pattern with a shopping cart example. [15]

2.5.2 Hierarchical model-view-controller (HMVC)

HMVC is a more modular variation of the MVC, and similar to the presentation-abstraction-control (PAC) pattern. The structure resembles that of traditional MVC, but with the distinguishing factor that HMVC consists of layers of MVCs with clear boundaries for each layer, meaning that a module does not interfere with other modules. These limiting layers are referred to as modules. HMVC is used to increase code modularity, aid reusability, and maintaining a better separation of concerns. [16]

2.5.3 Model-view-adapter (MVA)

MVA is another modification of the MVC design pattern. A key principle of MVA is an absolute separation of the model and view, which is done with adapters. There is a distinct difference in the roles of the view and the adapter: a view focuses on

displaying information to users whereas an adapter processes the change of data. In MVA, there are no direct connections between model and view. [17]

2.5.4 Model-view-presenter (MVP)

The MVP design pattern is also a derivation of the MVC and is used for creating UIs. In MVP the presenter component acts as a link between the model and the view. The presenter retrieves and organises data from the model and formats it to be displayed in the view. The presenter layer must not directly access the controls of the view. The same presenter class can be used by several views but more commonly just one presenter is assigned to each view [18].

What kind of logic is permitted in the view depends on the implementation. At one extreme, the view forwards all interaction operations to the presenter, which means that the view is entirely passive. On trigger of an event method of the view, a method without parameters and return value is invoked on the presenter, which retrieves data from the view via routines that are defined by the view interface. Then, the presenter works as instructed in the model and as the result the view is updated accordingly. There are variations of MVP that allow for some freedom regarding how the responsibilities for particular interactions and events are distributed among classes. [18]

2.6 Frameworks and libraries

Probably the biggest change to web development thus far, along with JavaScript, came with the introduction of web frameworks and frontend libraries. Still approximately ten years ago, all the labour of a web service was done on the server. In contrast, now most of the logic of a web application is handled on the client, which does all the rendering, interactions, and a lot of the data manipulation. Therefore, the front end is mostly responsible for the application's performance. Frameworks have different kinds of approaches to addressing the performance issues.

This chapter introduces the three currently most common frontend frameworks for web development: React, Angular, and Vue.

2.6.1 React

React (also known as React.js or ReactJS) is a declarative and component-based JavaScript library for building user interfaces. It was released in 2013 by Facebook. Along with Facebook, other companies and individual developers also contribute in the maintenance and development of React.

React is fundamentally unopinionated, which means that it provides the basic means of building a user interface with the JSX syntax, and component state management with hooks, among other things. Additional required tools can be freely chosen by the developer. Since React as such does not include all the necessary tools to build complete applications, it is in fact a library instead of a framework. It only provides the 'view' element of the MVC (Model-View-Controller) design pattern, which is popular for designing web applications. Therefore, the 'model' and 'controller' elements must be implemented by other means. MVC is covered more in-depth in a later chapter.

React applications are often reliant upon third-party libraries, so React does not lock the developer into one choice other than React itself, which is arguably an advantage. Frameworks such as Angular and Vue include many more tools bundled within the core package.

2.6.1.1 Components

The core feature of React is entities called components, which are conceptually similar to JavaScript functions. The main responsibility of a component is to process the raw data and produce rich HTML as the output. A component accepts arbitrary

inputs called “props” (short for properties) and returns React elements that describe what is supposed to be shown on the user’s screen. Using components, the UI can be split into independent, reusable pieces.

In addition to props, components have their own state. A component’s state and props together form the raw data that the HTML output is generated from. Props and state are similar in that they are both plain JS objects and they both trigger a render update when changed. They are also deterministic, meaning that the same combination of props and state should not generate different outputs. [19]

Props are a component’s configuration, or options, that are received from parent components. They are immutable concerning the child component that receives them but can be changed by the parent component. In other words, a component cannot change its own props but is responsible for assembling the props of its child components. [19]

State initially holds a default value when a component is mounted and is then changed due to mutations, which result from events triggered by a user’s actions. State is a serialisable representation of one point in time and managed internally by the corresponding component. It does not alter the state of its children, besides setting the initial state. It could be said that state is private, and it is optional to use state. Components without state are preferable since state increases complexity and reduces predictability. [19]

Components can be declared either as class-based components or functional components. A functional component takes props as arguments and returns a React element. A class component requires extending the `React.Component` subclass and including a render function which returns a React element. The most obvious differences between these two component types are the syntax, and the amount of code needed. Figures 4 and 5 show an implementation of a simple counter component as a class and as a function respectively. The component returns a

button and a count that is saved in the component's state which increases by one on every button click.

```
class Counter extends React.Component {
  constructor() {
    super();
    this.state = {
      count: 0
    };
    this.increase = this.increase.bind(this);
  }

  increase() {
    this.setState({ count: this.state.count + 1 });
  }

  render() {
    return (
      <div style={{ margin: "50px" }}>
        <h1>React Class Component Example</h1>
        <h3>Counter:</h3>
        <h2>{this.state.count}</h2>
        <button onClick={this.increase}>Add</button>
      </div>
    );
  }
}
```

Figure 4. A react component declared as class. There is a fair amount of “boilerplate” code.


```

const Counter = () => {
  const [count, setCount] = useState(0);
  const increase = () => setCount(count + 1);

  return (
    <div style={{ margin: "50px" }}>
      <h1>React Functional Component Example</h1>
      <h3>Counter:</h3>
      <h2>{count}</h2>
      <button onClick={increase}>Add</button>
    </div>
  );
};

```

Figure 5. A react component declared in the functional manner. There is no constructor nor render method.

2.6.1.2 JSX

The recommended syntax to be used with React is JSX, which resembles HTML but is in fact a syntax extension to JavaScript. Essentially it is JavaScript and markup combined. JavaScript expressions are put inside curly braces in JSX. An example of JSX syntax is shown in figure 6.

JSX produces React elements that are rendered to the DOM. The UI logic, for example change of state, event handling, and data preparation is inherently coupled with rendering logic. Separation of concerns in React is accomplished with its loosely coupled components that contain both markup and logic. [20]

```

<div style={{ position: "sticky", top: "0" }}>
  {bookmarks && (
    <Button
      className="me-4"
      variant="primary"
      disabled={isBookmarked}
      onClick={() => savePage()}
    >
      {isBookmarked ? "Sivu tallennettu" : "Tallenna tämä sivu"}
    </Button>
  )}
  <OverlayTrigger placement="right" overlay={renderTooltip}>
    <span>
      <Button
        variant="primary"
        disabled={!isBookmarked || addSnippet.isLoading}
        onClick={() => saveSnippet()}
      >
        Tallenna pääkohta
      </Button>
    </span>
  </OverlayTrigger>
</div>

```

Figure 6. An example of JSX syntax.

2.6.1.3 Virtual DOM

React utilises a programming concept called virtual DOM. It is a virtual representation of a UI that is stored in memory and synchronised with the real browser DOM by a library such as ReactDOM. This process is known as reconciliation.

Virtual DOM enables the declarative API of React, which means that React is told what state the UI should be in, and ensures that the real DOM matches that state. The declarative API alleviates the work of developers so that they do not need to worry about exactly what changes on each update. Under the hood it is implemented in React with a diffing algorithm that can predict component updates. [21]

Initially, React promoted its virtual DOM implementation as providing a great increase in application performance. Later the performance claims were discarded since it turned out they were debatable considering that the virtual DOM adds overhead [22]. The subject is somewhat controversial, but it is generally agreed that virtual DOM is valuable because it allows building apps without thinking about state transition, while still providing sufficient performance.

2.6.2 Angular

Angular (commonly referred to as "Angular 2+") is a software development platform created by Google. It is built on the TypeScript language and includes a component-based framework for building web application front ends. Angular is a ground-up rewrite of the old framework AngularJS.

The basic elements in an Angular application are components. Components are written in TypeScript and consist of a decorator, an HTML template, and style definitions. The HTML template contains instructions on how to render the component and a CSS selector defines how the component is used in a template. The template and selector are specified by the decorator. [23] A minimal Angular component is shown in Figure 7.

Each component has an HTML template that is defined inline or by file path. The HTML template declares how the component renders and allows the separation of an application's logic from its presentation. Angular extends HTML with additional syntax that allows the insertion of dynamic values from components. The rendered DOM is updated automatically when a component's state changes. [23]

Angular supports dependency injection, which makes it possible to declare the TypeScript classes without worrying about their instantiation. Dependency injection is strongly recommended as a best practice in Angular, and many aspects take advantage of it to some degree. Angular provides numerous first-party libraries

that are required for most web applications, such as Angular Router, Angular Forms, Angular HttpClient, and Angular Animations. [23]

```
import { Component } from "@angular/core";

@Component({
  selector: "my-app",
  template: `
    <div>
      <h2>{{ appTitle }}</h2>
      <p>This is Angular</p>
    </div>
  `
})
export class AppComponent {
  appTitle: string = "Welcome!";
}
```

Figure 7. A simple Angular component

2.6.3 Vue

Vue (pronounced /vju:/, like “view”) is a progressive front-end JavaScript framework used to build user interfaces. It was created by the Chinese software developer Evan You and was released in February 2014. Vue is designed to be incrementally adoptable, with the core library focusing on the view layer.

At its core Vue uses a system that enables declarative rendering of data to the DOM using a fairly simple HTML-based template syntax. Vue templates are valid HTML and they compile into virtual DOM render functions. Vue also contains a reactivity

system that makes it possible to minimise the number of component re-renderings and DOM manipulations on state changes. [24]

Similar to React and Angular, Vue components are reusable instances that can be inserted as custom elements inside root instances. Being reusable, they accept the same options and lifecycle hooks as a root instance. In order to use a component in a template, it must be registered as global or local. Components that are registered globally can be used in the template of any component within the application. Data can be passed to child components with 'props', which is identical to React. The number of props for a component is unlimited and, by default, a prop can have any desired value. [25] Figure 8 shows an example of a simple Vue component.

```
export default {
  data() {
    return {
      count: 0
    }
  },
  template: `
    <button @click="count++">
      You clicked me {{ count }} times.
    </button>`
}
```

Figure 8. A Vue component defined as a plain JavaScript object.

3 Educational software

The Internet has drastically accelerated the emergence and delivery of new technologies. Along with new technologies and the digital revolution also came the possibility to create new kinds of effective computer applications for educational use. These kinds of applications have been made for both research and teaching purposes. There can be quite significant differences in the implementations depending on the primary purpose.

This thesis discusses software applications which end users are children and adolescents. On the other hand, the target audience is quite narrow, but nevertheless it consists of individuals with diverse levels of computer skills. Some students are very proficient with computers while some have only the most basic skills. The fact that end users' knowledge and skills are on different levels is one thing that should be taken into consideration while designing and developing these kinds of applications, especially in the user interface and user experience design.

Neurone and StudentNet can be categorised as educational software. In short, educational software is software that is designed to implement some computer-based pedagogical settings and contribute to the addressing of some pedagogical objectives [26]. A pedagogical setting can be defined as a setting that is designed to lead learners to develop an activity propitious to the fact that some given targeted pedagogical objective(s) will be achieved [26]. The primary purposes of educational software are teaching and self-learning, and to make some part of education more effective. Improved effectiveness can be achieved with software that provides students a high interactivity level and integrated multimedia content, which in turn help teachers to keep their students interested in a lesson.

3.1 Types

Educational applications can be classified according to the teaching approach they represent. The most important types are presented in the following subchapters, based on the source [27].

3.1.1 Tutorials

Tutorial applications are software that is designed to replace traditional classroom teaching by including complete instructions according to the learning material. Tutorials are generally structured so that they present the contents in small hierarchical steps, each containing theory and practise. Generally, there is a series of “help screens” that pupils can access whenever they need some tips or support. These help screens can aid in solving the most difficult tasks for example by suggesting ways to approach the problem by giving hints and examples. Typically, there is an introduction in the beginning of a tutorial and after that the application continues with the following cycle: Present Information, Question and Response, Judge Response, Feedback or Remediation. After Feedback or Remediation it goes back to Present Information, and this cycle continues until the lesson is complete. StudentNet can be considered as an educational application that belongs in the tutorials and courseware categories.

3.1.2 Courseware

Courseware means software that is designed to be used as supporting tool during courses. This type of educational software can be addressed towards learners of any age: from primary school pupils to university students. Generally, courseware applications display information and set up a dialogue with pupils. Courseware can contain material for teachers to be used during classes, and home study material for pupils.

3.1.3 Edutainment

Interest to combine education and entertainment has existed for some time. The term edutainment, short for educational entertainment, means media that is designed to educate through entertainment. Edutainment software can be described as a combination of traditional educational software and games. In this category also belongs games that offer some educational value but are not designed with the focus in educational content. On the other hand, in many cases the included content is intended to be educational but has incidental entertainment value.

3.1.4 Reference software

The first commercial reference software products were reformulations of existing content, such as print dictionaries and encyclopaedias, into CD-ROM editions. First, these products were just digital versions of already existing prints, but later included multimedia features. Reference software had a great success on the markets and products for specific topics such as music, movies, and encyclopaedia were created and commercialised. In addition to the multimedia content, the main benefit of reference software was the possibility for users to explore the contents via links and hyperlinks during the era before Internet. Since then, educational reference software has migrated from CD-ROMs to the web and become freely available to everyone, thanks to projects such as Wikipedia.

3.2 Benefits of educational software

The main perks of educational software include interactivity, adjustable contents, and the possibility to provide multimedia content. These are also features that distinguish educational software from traditional teaching methods and media. Multimedia content gives the capability of attracting users, especially children, more effectively. Particularly the curiosity of children is stimulated with lively characters that can be interacted with, as well as dialogues, sounds and videos. This kind of content helps keeping children interested in exploring the software.

Software with a high level of interactivity gives users the freedom of deciding what to do: in an open model they can explore the application in their own ways. A previous phase or chapter can easily be revisited if desired, and ideally also the individual phases are provided with minimal restrictions. This kind of open model that gives freedom is the core idea in the design of StudentNet.

The software can be constructed so that it helps users in discovering new arguments and unexpected points of view. Therefore, it can support the development of associative thinking along with the development of having a multidimensional vision of the same problem.

These aforesaid features of educational software contribute to modifying the way how users learn. Ideally a passive approach transforms into a desire to learn. It is a common claim among instructors that children learn best by doing. Therefore, it is essential for improving learning efficiency so that children are given more control and are actively involved in the learning process [28]. This can be accomplished with properly designed software. The primary goal in the design of educational software is to generally improve the learning of students, not to maximise the degree of knowledge on some particular topics [29]. Unquestionably, profound knowledge and good understanding of the topics covered in the application is undeniably a worthwhile side effect. While using an educational computer program, pupils can experience their own ways of thinking and learning. Consequently, there is a good chance that they become conscious of their best individual learning mechanisms.

A communication system is another thing that can be provided in an application. Students could be given the possibility to communicate with each other, and with a teacher. A good communication system allows learners to share their thoughts and experiences conventionally and makes it possible for them to explore the software together. Teachers, on the other hand, can participate in planning of the teaching paths of their pupils. They can also assist their pupils by answering questions in real time. In addition, when teachers interact with their pupils, they can more easily

evaluate their level of knowledge and plan teaching strategies accordingly. The most common technique that allows for users to communicate with each other is an instant messaging system, a form of text-based communication. [27] Other possible means for communication include dedicated email, voice and video calls, and collaborative whiteboard.

It is also important and beneficial that children get to use computers early. Now that all digital everyday tasks can be done on a smartphone, a PC is not an absolutely necessary home device. However, at least basic computer and software skills are expected in many lines of business. Therefore, it is essential that basic computer skills are taught to children rather sooner than later. Lack of computer skills and knowledge is not a good thing in today's, let alone the future's, competitive and technology-oriented world. [27]

Using computer applications in education also makes it possible for students to conveniently self-assess their learning. Self-assessment tests are generally included at the end of each module – if the application is modularly structured – or at the end of a course. With quizzes and questionnaires, pupils can check how well they have learned overall and evaluate their degree of knowledge on a particular topic. Self-assessments also allow pupils to avoid any subjection to the reactions of the teacher and other pupils, contrary to traditional tests done in classrooms. [27]

Educational software has the potential of providing some remarkably useful features that can also offer some advantages directly to teachers. For instance, there can be an authoring system allowing teachers to create new contents and student evaluation tests in the application and to modify existing ones. The content provided in an educational application can be rather easily adjusted. For example, StudentNet is supposed to allow teachers to create and modify exercises and select the available source material for each exercise, as well as create new source material. Educational software also improves teacher's possibility to track their students' learning and advancement, potentially resulting in more time available for tutoring activities [30].

3.3 Design and development

The development of educational software commonly involves several different factors and the cooperation of students and professionals with backgrounds in different fields of study. Both pedagogical and software engineering factors should be considered during the development. The spectrum of activities includes software engineering and programming as well as planning and implementing of the educational contents. In addition, there is no common framework for this purpose. Standard methods and tools can be used in development but it is difficult to establish a standard for the realisation of educational software. [27]

The human resources in this project comprise a pedagogical team and a technical team. The pedagogical team is comprised of professionals and students with a pedagogical background, while the technical team is comprised of software engineering, computer science, and data science students. These teams collaborate in the development of the application as well as in the research, to an extent. Close collaboration is essential since technology-oriented people may face the difficulty of understanding the issues and phenomena that should be considered in educational software projects. Likewise, people whose background is in educational sciences are likely to face the difficulty of understanding issues related to software design and development, and how the workflow happens in practice. Furthermore, technological and pedagogical people share the difficulty of understanding how to relate software dimensions and educational issues. Therefore, misunderstandings may happen, so the aspect of multidisciplinary work is an issue that is good to acknowledge in the beginning of the project. [26]

There are some properties and requirements concerning technical and pedagogical aspects that are of significant importance for educational software. These properties and requirements are described in the following subchapters.

Efficient user interface

The UI of the application should be clear and intuitive. This requirement applies for most software but is extremely important considering educational applications. Students, particularly children, should be able to rather quickly understand how to use the software, i.e., what each of the buttons and other interactable elements shown on the screen are used for. The elements that can be interacted with should be somehow highlighted and images and animations should serve a purpose. The use of some background music and sound effects is generally fine although should be carefully considered and the possibility to mute should be given, whereas speech should be used only when necessary. Finally, highlighting important words that serve as links or have another specific function is recommended. [27]

Quality multimedia contents and interactivity

Although there are books that offer some interactivity and audio content, generally multimedia and interactivity can be considered aspects that distinguish educational software from books and other print material. Multimedia can be fundamentally related to the effectiveness of the software, but the multimedia contents should be carefully chosen so that they are not distracting nor result in a negative effect regarding the degree of attention paid by the users. The quality of multimedia should be considered from both pedagogical and audio-visual point of view. [27]

Appropriate and adjustable contents

The contents of educational software should be appropriate for the audience they are addressed to [27]. In the case of StudentNet the audience is primary school pupils with a rather wide spectrum of know-how. Therefore, the application should allow for adjusting its contents, addressing to individuals of different ages, goals and learning styles. In addition, adjustable support and tutorial should be considered in order to address pupils with different levels of skills.

4 Neurone

NEURONE (oNlinE inqUiRy experimentatiON systEm) is a software platform that provides an isolated search environment in a web browser. Its purpose is to collect data of users' actions as they use the platform. The collected data is used to build profiles or patterns of user navigation behaviours related to their online inquiry skills. [31]

Neurone is released as open source software and was developed as a part of the Finnish-Chilean iFuCo project (2016-2018). It includes four major modules: search, identifying of the main points, critical evaluation of found information, and composing a synthesis. Each module assesses an online inquiry skill. They are built with a set of components, resources, and tools that are deployed in a controlled environment. [32]

The search module is based on a search engine that is connected to a database containing a private collection of Web pages. Within this module users can search for information in an environment which appearance is similar to popular commercial search engines. Found pages can be bookmarked so they can be easily revisited. [32]

The second module makes it possible to access the bookmarked web pages and provides a tool for selecting and saving fragments of texts – snippets - from the pages. The third module provides an interface for users to evaluate the reliability and relevance of the pages they have bookmarked. Finally, the fourth module provides a form to compose a synthesis based on the information in the selected sources. [32] The usage data collected with Neurone includes login information, visited pages, bookmarks, snippets, questionnaire responses, task synthesis, mouse movements, click positions and keystrokes [31].

4.1 Default stages

The Neurone application comprises four distinct phases, or stages, that are traversed linearly when working on an assignment. These stages are briefly described in the following subchapters.

4.1.1 Search and Selection

In this stage, the student is required to make search queries with the search engine module in order to find useful documents (web pages) to be used as source material. The number of relevant pages (usually three) is determined by the experimenter. There is a limited time that can be spent searching, but the number of queries is unlimited. The student can bookmark pages that he thinks are useful, so they are saved for later use in the following stages. Pages can also be unbookmarked. [31]

4.1.2 Collection

In the collection stage the student is assigned to find the most relevant ideas and select them as text fragments (snippets) from their bookmarked pages. Snippets will help the student write a synthesis in the final stage. The student must select a designated number of snippets (around three) from each bookmarked page before he can proceed to the next stage. [31]

4.1.3 Critical Evaluation

This is where the student must evaluate the reliability and relevance of each bookmarked page. The evaluations are given in the form of star icons. The student is also asked to write a brief reasoning for their evaluation.

4.1.4 Synthesis

In this stage the student must compose a synthesis as an answer to the given main assignment for what he was earlier searching information. The student's bookmarked pages and snippets serve as support.

4.2 Technologies

Neurone uses the following technologies:

- NodeJS
- AngularJS
- MongoDB
- Apache Solr
- Meteor

AngularJS is the no longer supported predecessor of Angular, which has already been covered in chapter three. NodeJS is covered in chapter six because it is used in the back end of StudentNet. MongoDB, Apache Solr and Meteor are briefly described in the next subchapters.

4.2.1 MongoDB

MongoDB is the most popular NoSQL database solution for modern web applications [33]. It is a document-oriented database; the data is stored in MongoDB as JSON documents which are organised in collections where they can be queried. In contrast to a relational table, a pre-defined schema is optional for a collection, which allows for the data structures to be evolved rapidly. More importantly, MongoDB allows different data that is often queried together by an app to be saved in the same JSON document. This makes reading the documents rather fast because no join operations are required.

A MongoDB database is easy to scale. Configuring a sharded cluster where data is distributed across multiple servers allows portions of the database, shards, to be configured as replica sets. [34]

4.2.2 Apache Solr

Solr is an enterprise-class open-source search platform that is built on the Apache Lucene search library, also utilised by Elasticsearch. Features of Solr include for example load-balanced querying, replication, distributed indexing, and centralised configuration. [35]

Solr is written in Java and is available as cross-platform software. It runs as a standalone server and is capable of full-text indexing and search. Solr uses REST-like HTTP/XML and JSON APIs, so it is compatible with all of the most common programming languages. Solr can be tailored to many types of applications with its external configuration, without Java coding.

4.2.3 Meteor

Meteor, or MeteorJS, is a full-stack JavaScript framework used for building modern web and mobile applications. It is distributed as free and open-source software. Meteor comes with a build tool and a set of essential technologies and a set of selected NodeJS packages. [36]

Meteor integrates with MongoDB. Data changes to clients are automatically propagated using the Distributed Data Protocol and a publish–subscribe pattern. Meteor can be used to build apps together with any common front-end JavaScript framework or library, such as React, Vue, Angular or Svelte.

An advantage of Meteor is its diverse community of contributors that gives it resilience and longevity. Another advantage is that Meteor is cross-platform, so

implementations of an application for different environments can be done in one place. In addition, Meteor is built in MongoDB handlers, and it supports GraphQL.

Disadvantages of Meteor include possible conflicts with pre-built packages and mobile apps made with Meteor will not run as efficiently as those made with native languages.

Neurone was created using technologies that are now outdated, most notably its front-end framework AngularJS. AngularJS was in extended long-term support until the end of 2021, and since the beginning of 2022 is no longer supported at all. The ending of support for AngularJS was a weighty reason for the decision to create a new application from the ground up.

5 StudentNet

The previous chapter introduced Neurone, which acts as the basis of StudentNet. The primary purposes of Neurone are research, data collection, and measurement. It is an application that can be used for the evaluation of children's internet reading skills. For that reason, the structure of Neurone is linear: the user flow is restricted so that there are stages that must be completed in a specific order. This kind of structure is desirable for research and measurement, but it does not fit so well for learning.

StudentNet is under development with the primary purpose of teaching and to provide a natural environment for improving internet reading skills. In order to provide this experience that is similar to a real-world learning situation, the application should have a nonlinear structure and user flow. The nonlinear structure and nonlinear user flow mean that the user is not restricted to follow some pre-determined order of actions while using the application.

The development of StudentNet has been iterative and incremental with flexible responses to changes in requirements. The changes in requirements were anticipated and taken into account when choosing the right technologies. It was initially planned to implement StudentNet with the MERN stack (MongoDB, Express, React, Node) but eventually an SQL database was considered to be the best option for the project. The NoSQL database MongoDB was replaced by PostgreSQL, so the final technology stack of StudentNet is PERN.

5.1 Technologies

The technologies that form the PERN stack – PostgreSQL, Express, React and Node – are briefly described in this section. In addition, the package manager of Node,

npm, and the search engine Elasticsearch as well as the React libraries React Query and React Table are presented.

5.1.1 Node.js

Node.JS, commonly just Node, is a back-end JavaScript runtime environment. It runs on the Google Chrome's V8 JavaScript engine, which is among the fastest JavaScript implementations [37]. Node.js enables executing JavaScript code outside of a web browser, and thus coding both the frontend and backend of web applications in JavaScript. With Node, JavaScript skills can be reused for general purpose software development on a larger scale of systems.

It is quite beneficial to be able to use the same language in the backend and the frontend [37]:

- The same programmers can work on backend and frontend
- Code migrations between server and client are easier
- The same data format, JSON, can be used between server and client
- There are some common software tools for server and client

Node is asynchronous and event-driven, designed to build scalable network applications. The asynchronous nature makes it possible to handle a large number of connections concurrently. Callback functions are called upon each connection, but if there is nothing to do, Node will sleep. This contrasts with thread-based networking which is inefficient and can be difficult to use. Multiple cores can still be utilised if desired by creating child processes using Node's `child_process.fork()` API. In addition, the cluster module makes it possible to share sockets between processes in order to enable load balancing across the cores.

The traditional approach in servers is to use threads and blocking I/O to retrieve data. However, blocking I/O causes threads to be suspended and waiting for operations to finish, which leads to the situation where the server starts and stops

the threads in order to handle requests. Each suspended thread consumes full stack trace of memory, increasing memory consumption overhead. [37]

Node's design is somewhat similar to systems such as Ruby's Event Machine and Python's Twisted, but there is some notable difference in the event model. In Node, the event loop is presented as a runtime construct instead of a library. Node enters the event loop after executing the input script and exits it after there are no more callbacks to perform. The behaviour is like JavaScript in a browser and with this approach the event loop is not visible to the user. [38]

The backend of this project's software was decided to build with Node mainly because it was already familiar, as was its language JavaScript. Node is also very popular with an active and vibrant community so there are a lot of guides, shared code, and other useful material available. Another thing that spoke for Node was the fact that it does not tie the developers to use any specific framework. Altogether, Node was considered the best fit for this project's needs. Figure 9 shows an example of creating a simple http server with Node using its built-in http module. The presented code creates a web server at localhost on port 3000, with a text content "Hello from Node".

```
const http = require("http");
const hostname = "127.0.0.1";
const port = 3000;

const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader("Content-Type", "text/plain");
  res.end("Hello from Node!");
});

server.listen(hostname, port, () => {
  console.log(`Server is running at http://${hostname}:${port}/`)
});
```

Figure 9. Creating an http server with Node.

5.1.2 Npm

The default package manager for Node.JS is npm (originally short for Node Package Manager). It includes a command line client with a command line interface (CLI) and the npm registry. The registry is an online database containing public and private npm packages. The registry is accessed using the CLI or by browsing and searching for packages on the npm website.

Npm can manage a project's local dependencies as well as globally installed JavaScript tools. All the dependencies of a local project are listed in a `package.json` file and can be installed with one command (`npm install`). In the same file, each dependency can have a range of valid versions defined using the semantic versioning scheme. This enables automatic updating of packages without causing breaking changes. [39]

5.1.3 Express

Express is a Node.JS framework for creating web and mobile applications and APIs. It is designed to be minimal but still provides a robust set of fundamental features. [40] Express is unopinionated, unlike its competitors like Django or Ruby on Rails, which means it does not tell a "best way" to do something. Express provides just the tools that are required to build an application and the developer can install other modules according to their specific needs.

Express was chosen for the back-end framework precisely because of its minimality and flexibility. In addition, it is also hugely popular, so there is a lot of support material available. Other considered back-end framework options were Koa and Meteor, which is used by Neurone.

5.1.4 Bootstrap

Bootstrap is a free and open-source HTML, CSS and JavaScript framework used to build responsive front ends in web and mobile environments. As of April 2022, Bootstrap is the eleventh most starred project on GitHub, with over 156,000 stars.

Bootstrap includes basic HTML and CSS design templates that comprise a big collection of commonly used UI elements. The templates are implemented as CSS classes that can be applied to HTML and which can also be overridden by custom CSS. The use of semantic class names makes UI components easily reusable and extensible. [41]

5.1.5 PostgreSQL

PostgreSQL – commonly referred to as ‘Postgres’ – is a free and open-source object-relational database system. Its core platform has been under active development for over 30 years and has gained a strong reputation for reliability, feature robustness, performance, and extensibility. PostgreSQL supports both SQL (relational) and JSON (non-relational) querying and has a monolithic architecture. It means that the components are completely united, and the database can only scale as much as the machine running it. PostgreSQL can handle various workloads, ranging from single machines to data centres and web servers with a large number of concurrent users. It includes many sophisticated features that are common to enterprise-class database management systems, for example table inheritance, user-defined types, foreign key referential integrity, advanced locking mechanism, and asynchronous replication.

PostgreSQL is available for all major operating systems and has been ACID-compliant since 2001. [42] ACID stands for Atomicity, Consistency, Isolation, Durability which are attributes of database transactions. Their purpose is to guarantee data validity despite any errors, power failures, and other accidents.

PostgreSQL was chosen as the project's database system since it is free and open source and is very popular with good documentation. Another strongly considered option was MongoDB, which is a document-oriented database. It only has the potential for ACID compliance, while PostgreSQL has ACID compliance built in. Eventually a relational database and SQL were the preferred options.

5.1.6 Elasticsearch

Elasticsearch is a distributed search and analytics engine that is built on the Apache Lucene search engine library. Elasticsearch works with most types of data, including textual, numerical, geospatial, structured, and unstructured data. Elasticsearch promotes that the distributed nature, speedy operation, good scalability, and simple REST APIs are its strengths. The search engine is the core component of the Elastic Stack, commonly referred to as the ELK Stack after its three components: Elasticsearch, Logstash and Kibana. Logstash provides a server-side pipeline for processing data before sending it to Elasticsearch, whereas Kibana is a tool for data management and visualisation. [43]

Documents in Elasticsearch are stored in the JSON format. An Elasticsearch index is a collection of JSON documents that are somehow related to each other. Before data is indexed in Elasticsearch, the raw data is parsed, normalised, and enriched. This process is called data ingestion. A document contains a set of keys, i.e., names of fields or properties, and their corresponding values (e.g., string, number, Boolean, date). The data structure is called an inverted index, which lists every unique word that appears in any document and identifies all the documents containing each unique word. [43]

Elasticsearch is used by thousands of companies ranging from small businesses to huge corporations. Netflix, Uber, and Microsoft are some of the most notable companies that use Elasticsearch in their products or businesses. There are numerous possible use cases for Elasticsearch, for example application and website

search, logging and analytics, application performance monitoring, and business analytics.

5.1.7 React Query

In order to understand React Query, it is important to know the difference between client state and server state. Client state means any information that is relevant to a particular web browser session. For example, a user's chosen language or the theme of a web site. These do not have anything to do with the server, they just form the client state for this particular user.

Server state, on the other hand, is data that is stored on the server but is needed to display on the client. This can be, for instance, blog post data that is stored in a database. This data is needed on the client to display to the user but it is stored on the server so that it can be persisted to multiple clients.

React Query is a library that manages the server state of a React application. It allows for caching and updating data in React-based applications in a simple and declarative manner without mutating the global state. Whenever React Query gets data, it stores that data in a cache. So that when react re-renders it is not lost and does not need to be re-fetched. The purpose of React Query in StudentNet is to make the management of asynchronous data more convenient.

React Query maintains a cache of server data on the client. The way it works is so that when React code needs data from the server, it does not get it straight from the server using fetch or axios. Instead, it asks the React Query cache, which is the source of truth for server data. React Query's responsibility is to maintain the data in that cache, based on the configuration of the React Query Client. Developers must indicate when to update the cache with new data from the server; the data on the client's cache must be consistent with the data on the server. This can be ensured in a couple of ways. It can be done either imperatively, by just telling React Query that certain data in the cache should be invalidated and replaced with new data from the

server, or declaratively by configuring how and when to trigger a re-fetch. For example, a re-fetch may be triggered when the browser window is re-focused after 30 seconds or more has passed.

In addition to the cache feature, React Query includes a lot of other tools to help with server state management. For instance, it maintains loading and error states for every query made to the server, so they do not need to be handled manually. React Query also provides tools to fetch data in small pieces, just when it is requested by the user, which is useful for implementing pagination or infinite scrolling. Data can also be pre-fetched and stored in the cache, if it is anticipated that the user is going to need it. The application can then pull that data from the cache, so the user does not need to wait for the application to fetch data from the server.

React Query can also manage mutations, i.e., updates of data on the server. Since queries are identified by a key, React Query can manage requests so that when a page is loaded and several components on that page request the same data, the query is sent only once. If another component requests the data when the original query is being processed, React Query can de-duplicate the requests. React Query can also manage fetch retries in case of errors and provides callbacks so that there are actions to be taken if a query is successful, returns an error, or a callback can be provided to take an action in either case.

There are three ways to handle the asynchronous data in React Query:

1. Re-fetching the data from the server
2. Use the server's response to update the cache to show new data
3. Optimistic UI update: the cache is updated before the server responds so user gets immediate feedback

Data fetching and data mutations in StudentNet are implemented as custom hooks that utilise React Query's `useQuery` and `useMutation` hooks. Each of these custom hooks also include a tracking module in order to log all user actions. The custom

hooks are re-usable among all components, and they keep data fetching and mutations separated from the UI components. Figure 10 shows the implementation of the hook `useAddBookmark`, which is called when a user adds a new bookmark.

```
const trackBookmark = (newBookmark, userId, pageCode) => {
  const data = trackingData("BOOKMARK", userId, {
    text_id: pageCode,
    detail: newBookmark.text_title,
    action_id: newBookmark.action_id,
  });
  tracking(data);
};

export function useAddBookmark(userId, pageCode) {
  const queryClient = useQueryClient();

  return useMutation(
    (newBookmark) => {
      axios.post("/bookmarks", newBookmark);
      trackBookmark(newBookmark, userId, pageCode);
    },
    {
      onSuccess: () => {
        // Invalidate the corresponding query to trigger a refetch and update
        // the client's view
        return queryClient.invalidateQueries([queryKeys.bookmarks, userId]);
      },
    }
  );
}
```

Figure 10. New bookmarks are added with the `useAddBookmark` hook.

5.1.8 React Table

React Table is a collection of React hooks for implementing tables and data grids. The hooks are lightweight, composable, and highly extensible. React Table is a so-called headless UI library which effectively means that it does not render or supply any actual UI elements. Therefore, the developer oversees utilising the state and callbacks provided by React Table's hooks. A headless UI is beneficial since it supports maintaining separation of concerns, makes code updating and maintenance easier, and enables extensibility. In StudentNet, React Table is utilised in the Bookmarks, Snippets, and Synthesis modules.

5.2 Modules

StudentNet consists of different modules that each serve their own purpose in the application. Each major module has its own route endpoint, and a page rendered in the application which can be accessed from the Navigation Bar.

5.2.1 Search

The search module makes it possible to search the database for relevant documents that are used as source material. The documents' text content is automatically indexed into an Elasticsearch cluster with a PostgreSQL extension called ZomboDB.

The core of the search module is implemented with two functional components called Search and SearchResults. Their states are managed with two useState hooks and two custom hooks. The first useState hook is used to save and set the user's search term, while the second one is for keeping track of if a search has been done or not.

The data that is fetched when a search is done is saved in a state called "pages" and is set with "setPages". These are managed with a custom hook called usePersistedState. A custom hook is required instead of a useState hook because we want the data to be preserved when the Search component is unmounted, i.e., when the user exits the search page. Inside the custom hook the React's useEffect hook is utilised to monitor changes in the pages data, and the data is preserved in the browser's local storage when the component is unmounted.

The searchTerm state is set when the input of the search field changes. This caused the problem that the displayed search term changed in real time with the user's input, before it was submitted and the search done. To properly display the submitted search term with the corresponding search results, the term is also saved in another state which is set on submit. Like the search results, also the displayed

search term must be preserved when the component is unmounted, so the usePersistedState hook is used.

For the search to be more user friendly and look like other search engines, the search results are paginated. Pagination can be implemented in the front end, since we are not dealing with a huge amount of data. An additional component, Pagination, and some states are used to implement the pagination. Figure 11 shows a view of the search page after a search has been completed.

StudentNet Tehtävä Haku Tallennetut sivut Pääkohdat Yhteenveto Loppukysely

Hae lähteitä

Vinkki: Voit käyttää hakusanoissa asteriskia (*). Esimerkiksi hakusanaalla tutki* löytyy tutkiminen, tutkijat jne.

Hae

18 tulosta hakusanaalla: silm*

Keuringon kirjaston facebook-sivut, 15.4.2016
Keuringon kirjaston facebook-sivut, 15.4.2016 Laulu harmaista silmistä Kirjastonhoitajana minulta kysytään joskus hyvin yksityiskohtaista tietoa esimerkiksi lauluista. En voi sanoa olevani lauluyriikan asiantuntija, mutt...

Tutkijat löysivät muinaisen nelisilmäisen liskon
Tutkijat löysivät muinaisen nelisilmäisen liskon Lasten tiedelehti, 12.1.2019 Saniwa endisens -varaanilaji kuoli sukupuuttoon noin 34 miljoonaa vuotta sitten. Saksalais-amerikkalaisen tutkijaryhmän tutkiessa ikivanhoja fossiileja selvisi, että kyseisellä varaanilajilla oli ollut...

Russarön majakkaa kutsutaan Hangon silmäksi
Russarön majakkaa kutsutaan Hangon silmäksi Kotimaan majakat --matkailusivusto, 4.6.2020 Russarön saari sijaitsee Hangon edustalla. Ensimmäinen majakka saarelle rakennettiin vuonna 1826 Venäjän keisarin käskystä. Nykyinen kahdeksankulmainen, graniitista ja tiilestä rakennettu maj...

Mä silmät luon ylös taivaaseen --virsi on kansallisrunoilijamme kirjoittama
Mä silmät luon ylös taivaaseen --virsi on kansallisrunoilijamme kirjoittama Pyhäkoulu-blogi, 25.10.2019 Virsi 490, Mä silmät luon ylös taivaaseen, on kansallisrunoilijamme J.L. Runebergin kirjoittama lastenvirsi. Sitä on laul...

Miten ihmisen silmä eroaa eläinten silmistä?
Miten ihmisen silmä eroaa eläinten silmistä? Lasten tiedeuutiset 20.1.2020 Useiden lajien silmät toimivat samoilla periaatteilla ja ne koostuvat samoista osista. Eroja ihmisten ja muiden eläinten silmistä voidaan havaita näkökyvyn ominaisuuksissa, eli siinä, kuinka hyvin ja millä...

Jax-koira sai apua huonoon näköönsä silmälasista
Jax-koira sai apua huonoon näköönsä silmälasista Viihdelehti.fi Silmälasia tarvitsavasta texasilaiskoirasta on tullut somejulkkis! Jax-koiralla on silmänpainetauti ja kaihi, joiden takia se näkee huonosti. Eläinlääkäri oli kuitenkin keksinyt Jaxin ongelmaan ratkaisun ja määrän...

Silmien väri ei ole sattuman kauppaa
Silmien väri ei ole sattuman kauppaa Kysy ihmiseltä ihmisestä-sivusto, 8.9.2008 Mistä silmien erivärisyys johtuu? Biologista syytä silmien erivärisyyteen ei tiedetä, mutta se tiedetään, minkä takia silmät näyttävät eri väre...

Hevosien silmät näkevät laajalle alueelle
Hevosien silmät näkevät laajalle alueelle Eläinmaailman opas Hevosien silmät ovat hyvin tarkat havaitsemaan nopeita liikkeitä. Hevosien silmän näkökenttä on lähes 360°, mutta se ei näe suoraan eteensä tai taakseen kääntämättä p...

1 2 3

Figure 11. Search page displaying search results.

5.2.2 Bookmarks

The bookmarks module allows users to save pages that they find useful and relevant considering the assignment they are working on. On the bookmarks page, the bookmarks are displayed in a table with appropriate information: the page title, and reliability, relevance, and category, which are evaluated by the user. The page title works as a link to the corresponding document page.

Before a document page can be evaluated or used as a source for snippets, it must be bookmarked. Each document page has a bookmarking button in the top left with the text “Tallenna tämä sivu” (Save this page). When a user clicks the button, the page is saved in the bookmarks table in the database. During the saving process (React Query mutation) the button displays “Tallennetaan...” (Saving...) as its text and is disabled. When the page is successfully saved, the button’s text changes to “Tallennettu” (Saved) and it stays disabled. In order to remove a bookmark, users must go to the bookmarks page where they can see all their bookmarks. The bookmarks table, depicted in Figure 12, is created with the React Table library.









Otsikko	Luotettavuus	Hyödyllisyys	Kategoria	
Australia tutuksi kahdessa viikossa			Mainos	Poista
Havumetsä ja tundra ovat pohjoisen pallonpuoliskon kasvillisuusalueita			Tietoteksti	Poista
Australian ja Suomen eläimet ovat sopeutuneet erilaisiin olosuhteisiin			Tietoteksti	Poista
Suomi kärsii ilmastonmuutoksesta - kasvillisuusvyöhykkeet vetäytyvät			Uutinen	Poista

Figure 12. The Bookmarks page displays all bookmarked pages in a table along with their associated evaluations.

5.2.3 Snippets

The snippets feature makes it possible to save selected fragments of text. Users are supposed to find the most relevant parts from their bookmarked pages and save them as snippets. These snippets can be then utilised when writing the synthesis.

Saved snippets are displayed on the synthesis page below the text editor so they are easily available. Snippets are also visible in the snippets page, where they are displayed using the React Table library. The table shows the text title and the snippet text of each snippet, as well as a delete button on the right. The text title works as a link to the page in question. Snippets can be saved from a page after the page is saved as a bookmark. On non-bookmarked pages, the “Save snippet” button is disabled and shows a tooltip when mouse-overed. The tooltip instructs to first bookmark the page in order to save snippets. A screenshot of the snippets table with some saved snippets is shown in Figure 13.

Tekstin otsikko	Pääkohta	
Australian ja Suomen eläimet ovat sopeutuneet erilaisiin olosuhteisiin	Maapallon eläinlajit ovat sopeutuneet elämään erilaisissa olosuhteissa. Osalla eläimistä sen sopeutumiskeinot näkyvät suoraan sen ulkonäöstä: esimerkiksi paksu turkki voi kertoa siitä, että eläin pystyy elämään kylmissä olosuhteissa.	<input type="button" value="Poista"/>
Havumetsä ja tundra ovat pohjoisen pallonpuoliskon kasvillisuusalueita	Esimerkiksi Suomesta suurin osa kuuluu havumetsäalueeseen. Myös Ruotsissa ja Venäjällä on laajoja havumetsäalueita.	<input type="button" value="Poista"/>
Suomi kärsii ilmastonmuutoksesta - kasvillisuusvyöhykkeet vetäytyvät	Suomessa erityisesti havumetsänrajan siirtyminen on aiheuttanut sen, että monet lajit ovat pakotettuja siirtymään. Ongelmaksi muodostuu se, että sopivia elinympäristöjä ja tilaa ei tahdo riittää pohjoisessa kaikille.	<input type="button" value="Poista"/>

Figure 13. Snippets page displays user’s saved snippets in a table.

5.2.4 Evaluation

The evaluation module is located on the right side of a document page. It consists of rating components, a drop-down menu for category selection, a text area for reasoning the evaluation, and button to save the evaluation. The two rating components provide the possibility to evaluate the reliability and relevance of a page. They are created with Material UI’s rating component and consist of five

different looking emoji icons, which have been modified so that they change colour when mouse-overed and selected.

Below the rating components is a drop-down menu that is used to evaluate and select what category the current page belongs to. The category options are nonfiction, advertisement, blog, fiction, news, and other. After selecting the category, the user is supposed to briefly explain his or her evaluation in the provided text area. The whole evaluation module is illustrated in Figure 14.

Luotettavuus:



Hyödyllisyys:



Valitse kategoria ▾

Perustelut:

Tallenna arviointi

Figure 14. The evaluation module is found on the right side of every document page. It is used to evaluate the reliability, relevance and category of a page and briefly explain the evaluation.

5.2.5 Synthesis

Synthesis is the final module and the closing phase in StudentNet. It provides a text area, where the user can compose their answer to the questions of the given assignment. Later this text area is supposed to be replaced with a simple text editor component that allows for basic formatting of text. Pupils are supposed to use their

saved snippets as a support when writing the synthesis. For convenience, the snippets are also shown on the synthesis page, in a table below the text editor.

The user is not limited to using only snippets when composing their answer. They can, whenever they want, go to another page in the application. If one wants to see the whole page where a snippet is taken from, they can go to that page via a link either from the snippets list or the bookmarks list. Users can also still make new searches, bookmarks and snippets while working on the synthesis.

The number of typed words is shown below the editor, along with a button to save the typed text. Text is automatically saved in the client but saving the text to database must be done manually, however autosaving to the database will be implemented later.

6 Experiments

Several controlled experiments involving StudentNet were arranged during the writing of this thesis. These experiments comprised of testing both Neurone and StudentNet, or one of them individually, by a different number of test participants. Two of these experiments are described in this thesis. Each experiment included an assignment that was presented to the testers, and they were supposed to complete that assignment in the application under test in a given time frame. Each participant's actions done while using the application were collected as log data. In addition, a questionnaire and a feedback inquiry were presented before and after the assignment.

The first experiment was a pilot experiment, where StudentNet was tested for the first time. At that time, StudentNet only included the Search module, and there were only 11 participants who completed the assignment. Logging users' actions was possible only in Neurone. In this experiment both StudentNet and Neurone were tested.

The second experiment was the main experiment that involved StudentNet with every main module (Search, Bookmarks, Snippets and Synthesis) as well as data logging capability. Neurone was not tested in this main experiment.

6.1 Pilot experiment

The pilot experiment was the first controlled occasion of testing for StudentNet. It was arranged primarily to verify that there are no problems in the experiment setting or with any of the involved systems, and that the assignments can be completed without technical difficulties. In addition, valuable feedback was

collected from the testers with questionnaires and during an online discussion after the testing.

At the time of this pilot experiment, StudentNet only included the search module. Therefore, an external system was needed to provide the possibility of writing a synthesis and the functionality for collecting information and feedback via questionnaires. The learning platform Moodle proved to be an appropriate environment, so it was used to organise the testing.

The experiment consisted of the following parts:

1. Start questionnaire
2. Assignment A (StudentNet)
3. Feedback inquiry A
4. Assignment B (Neurone)
5. Feedback inquiry B
6. End questionnaire
7. Material about online reading skills (Optional)

The pilot experiment was completed by 11 participants. Despite the small number, some useful information was gained from the questionnaire feedback. Most notably, all testers disagreed with the statement that the search engine of StudentNet works quickly. In contrast, all of them agreed with the same statement considering Neurone. One quite apparent reason for why StudentNet's search was not as fast as Neurone's is the fact that the development build of the application was in use. The React development build includes many helpful warnings that are very useful in development. However, they make the application larger and slower, so the production build should be used for optimised performance [44]. Also, the version of StudentNet at the time was a very early prototype without specific performance optimisations.

The statement “StudentNet’s UI is clear” was disagreed by every tester and the statement “StudentNet is easy to use” was disagreed by three testers. This was quite surprising information, considering that there are very few elements in the UI. Lacking the ability to save pages as bookmarks in the application probably affected the ease-of-use evaluation. Another thing that may have hindered the usability was the requirement to use two different systems at the same time.

The assignment of StudentNet (Assignment A, Vaccine) was found uninteresting by the testers. In contrast, the assignment of Neurone (Assignment B, Mask) was found interesting. However, more feedback must be collected before any possible modifications are made to the assignments.

6.2 Experiment at Rauma Teacher Training School

The first experiment was carried out at Rauma Teacher Training School, which belongs to the Faculty of Education of Turku University. It is a comprehensive school that teaches pupils in basic education from grades 1 to 9. [45]

Two classes of fifth grade pupils participated in the Rauma experiment. This experiment involved only StudentNet instead of both environments. However, StudentNet had more modules ready for testing, and log data could be collected. The modules that were implemented at the time of Rauma experiment were: Search, Bookmarks, Snippets and Synthesis.

User authentication was not yet implemented, so a temporary means to individualise users of StudentNet had to be come up with. The solution was to add an input field on the front page of the app, which is shown in Figure 15. There, users were prompted to enter their user id. Participants of Rauma experiment were instructed to use their own student id. The user id was then saved using React’s Context, which provides an effortless way to pass data through the component tree. User id can then be passed to all database queries where required.

Tervetuloa StudentNettiin!

Kirjoita tähän käyttäjätunnuksesi ja paina "Aloita"

Figure 15. The front page of StudentNet at the time of the experiment in Rauma. Users are welcomed and instructed to enter a username.

6.2.1 Structure

The experiment was structured as follows:

1. Start questionnaire (15 min)

- a. Age
- b. Gender
- c. Class
- d. Self-efficacy
- e. Attitude
- f. Pre-knowledge on internet reading
- g. Pre-knowledge on the topic of the assignment

2. Intervention (45 min)

- a. Teaching and exercises on internet reading

3. StudentNet assignment (45 min)

- a. Search & Bookmark pages
- b. Collect Snippets from bookmarked pages
- c. Write a Synthesis

4. End questionnaire (15 min)

- a. StudentNet evaluation
- b. Own performance evaluation
- c. Project evaluation

6.2.2 Assignment

The assignment in StudentNet consisted of two questions:

1. What are the differences between the eyes of a human and a horse?
2. How should a person take into account the horse's eyesight, for example when caring for a horse?

The time given to complete the assignment was 45 minutes and at least 20 minutes had to be used. During this time the participants were supposed to first search for relevant texts and read them, and bookmark the pages they think are most relevant. Then, select fragments of text from the bookmarked pages and save them as snippets, and finally, compose a synthesis to answer the two presented questions.

6.2.3 End questionnaire

The questionnaires were carried out with Webropol software. The end questionnaire comprised 19 statements with 5-point Likert scale agreement response format. The statements were selected according to the major variables in the Technology Acceptance Model (TAM), which is discussed in the analysis subchapter. The statements are shown in Table 1.

Table 1. The statements of the end questionnaire.

1.	The project "Internet reading" was interesting.
2.	The project's topic "Horse's eyes" was useful.
3.	The appearance of StudentNet is stylish.
4.	It is easy to understand how to use StudentNet.
5.	I understand well how StudentNet works.
6.	StudentNet should give more support, e.g. instructions.
7.	A lot of practice is required to be able to use StudentNet.
8.	Teacher should give more hints regarding the usage of StudentNet.
9.	I learned Internet reading in StudentNet.
10.	StudentNet helps in learning Internet reading.
11.	StudentNet is easy to use.
12.	I like the flexibility of StudentNet: it allows moving freely between different phases. For example, I can search or make a new snippet while writing my synthesis.
13.	StudentNet is an effective application for learning Internet reading.
14.	StudentNet is an effective application for learning topics of biology.
15.	StudentNet is suitable for Internet learning, in my own way.
16.	StudentNet enables Internet reading as I have understood it.
17.	Search works effectively.
18.	Snippets work effectively.
19.	Synthesis works effectively.

6.2.4 Analysis

The collected data was analysed with IBM SPSS software. There was a substantial amount of data missing on statements 13, 14, 15 and 16 due to some practical issues during the experiment, so they were excluded from statistical analysis. The descriptive and frequency statistics of responses along with the statements are shown in Table 2. TAM was eventually only used as a support and a theoretical background for selecting good statements. It was not used as a basis for any analysis, due to the rather small amount of data.

Table 2. Descriptive and frequency statistics of responses.

Statement	Mean	Median	Mode	Std. Deviation
1. The project "Internet reading" was interesting.	2,3	2	2	1,063
2. The project's topic "Horse's eyes" was useful.	2,48	2	2	1,201
3. The appearance of StudentNet is stylish.	2,35	2	2*	1,112
4. It is easy to understand how to use StudentNet.	2	2	2	0,853
5. I understand well how StudentNet works.	1,91	2	1	0,9
6. StudentNet should give more support, e.g. instructions	2,96	3	3	1,147
7. A lot of practice is required to be able to use StudentNet.	3,48	4	5	1,377
8. Teacher should give more hints regarding the usage of StudentNet.	3,13	3	3	1,29
9. I learned internet reading in StudentNet.	2,57	2	1	1,502
10. StudentNet helps in learning internet reading.	2,26	2	1*	1,137
11. StudentNet is easy to use.	1,96	2	1	1,022
12. I like the flexibility of StudentNet: it allows moving freely between different phases. For example, I can search or make a new snippet while writing my synthesis.	2,17	2	2	0,984
17. Search works effectively.	2,3	2	2	1,063
18. Snippets work effectively.	2	2	2	1
19. Synthesis works effectively.	1,83	2	1	0,984
*Multiple modes exist. The smallest value is shown.				
(1) Strongly agree (2) Agree (3) Neutral (4) Disagree (5) Strongly disagree				

6.2.4.1 Reliability

The reliability analysis in SPSS resulted in a Cronbach's alpha value of 0.906. Cronbach's alpha, or coefficient alpha, is a measure of internal consistency, describing how closely related a set of items are as a group. The value for Cronbach's alpha can be between 0 and 1; 0.906 indicates a high level of internal consistency. [46]

6.2.4.2 Technology Acceptance Model

Introduced in 1986, the Technology Acceptance Model (TAM) is considered the most influential and most widely employed theory for describing an individual's acceptance of information systems [47]. TAM is not only used with completely new technologies as it can also be used in cases where existing technologies are utilised in new ways. TAM assumes that an individual's information systems acceptance is determined by two major factors: Perceived Usefulness and Perceived Ease of Use.

Fred D. Davis defines Perceived Usefulness as "the degree to which a person believes that using a particular system would enhance his or her job performance." The definition originates from the meaning of the word useful: "capable of being used advantageously." [48]

Perceived ease of use refers to "the degree to which a person believes that using a particular system would be free of effort". In the background is the definition of the word ease: "freedom from difficulty or great effort". [48]

Perceived usefulness and perceived ease of use together create a specific attitude toward using a piece of technology. The statements of the end questionnaire that reflect perceived usefulness are statements 9, 10 and 13. Statements that reflect perceived ease of use are statements 4, 5 and 11.

7 Discussion

The statistics of the end questionnaire from the experiment in Rauma were presented in the previous chapter. The Likert scale data was gathered from the end questionnaire answers of 23 fifth grade pupils who completed the assignment in StudentNet and gave permission to use their answers in research. The most interesting statements are statements number 6, 8, 11 and 12, considering the research question: How does a nonlinear model work in an educational application, regarding the user experience of children?

Statement 6 stated that StudentNet should give more support, e.g., instructions to the users, and statement 8 stated that teachers should give more hints regarding the usage of StudentNet. These statements received the largest number of neutral responses: 39% of responses to statement 6, and 30% of responses to statement 8 were neutral. Furthermore, there was more deviation than in statements 11 and 12. Considering statement 6 as well as statement 8, there was approximately equal number of agreeing (1 or 2) and disagreeing (4 or 5) responses. The distributions of responses to statements 6, 8, 11, and 12 are illustrated as 100% stacked bar graphs in Figures 16-19. The numbers on the bar graphs indicate the frequency of the corresponding response option. The bar graphs of responses to statements 6 and 8 are shown in Figures 16 and 17 respectively.

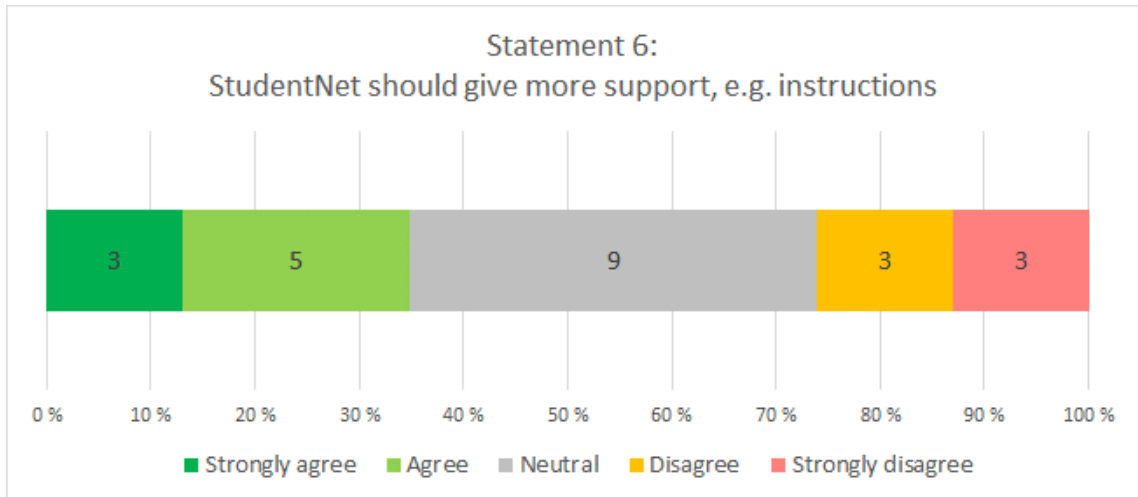


Figure 16. Distribution of responses to statement 6: StudentNet should give more support, e.g. instructions. The most frequent response is Neutral while the other options are quite scattered.

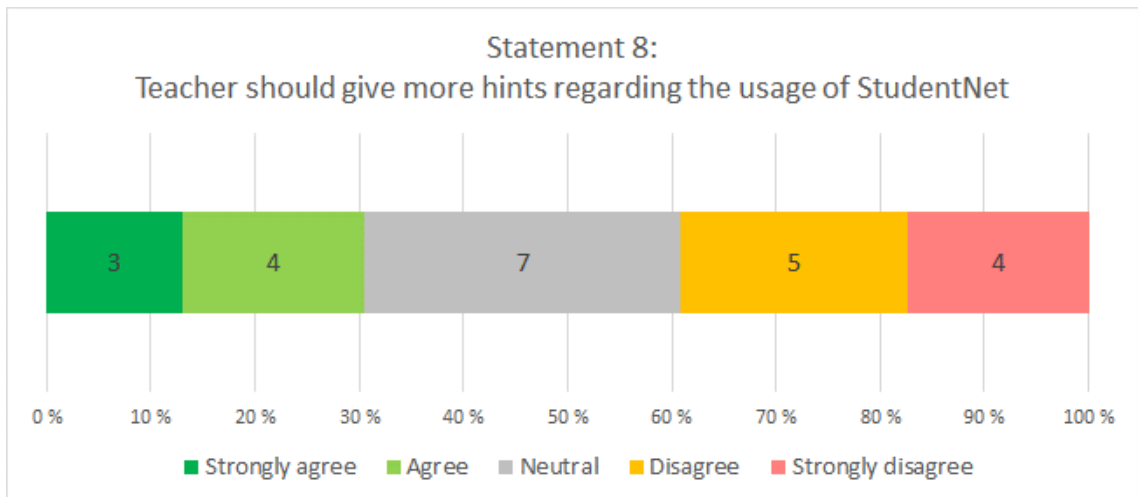


Figure 17. Distribution of responses to statement 8: Teacher should give more hints regarding the usage of StudentNet. The graph looks quite similar to that of statement 6.

Statement 11: StudentNet is easy to use was generally agreed on, the mean being 1.96, median 2 and mode 1. There is one response of value 5, “Strongly disagree” and zero responses of value 4, “Disagree”. 17 out of 23, nearly 74% of participants, agreed with the statement. Nine participants strongly agreed that StudentNet is easy to use, while eight agreed. Five participants chose neutral response to this statement. The frequency and percentage of each response option to statement 11 are illustrated in Figure 18.

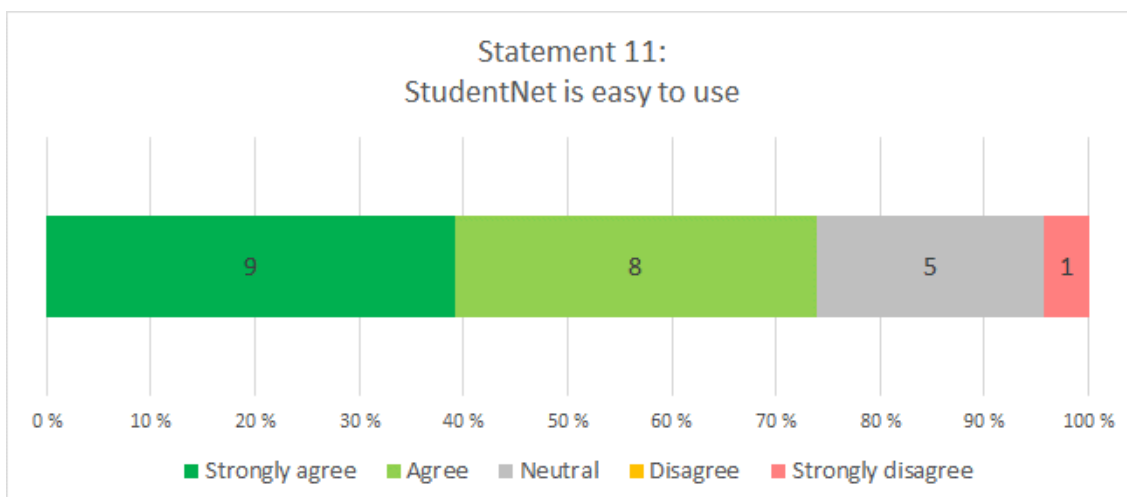


Figure 18. Distribution of responses to statement 11: StudentNet is easy to use. Distinct majority chose an agreeing response.

Statement 12 was broadly agreed by the participants: the mean of responses is 2.17, median is 2 and mode is 2. Again, there is only one disagreeing response (Strongly disagree). However, seven pupils responded “Neutral”. Six pupils gave a response of “Strongly agree”, and nine pupils responded “Agree”. No one responded “Disagree” (4). In total, 65 percent of participant agreed with the statement. The frequency and percentage statistics of statement 12 are shown in Figure 19.

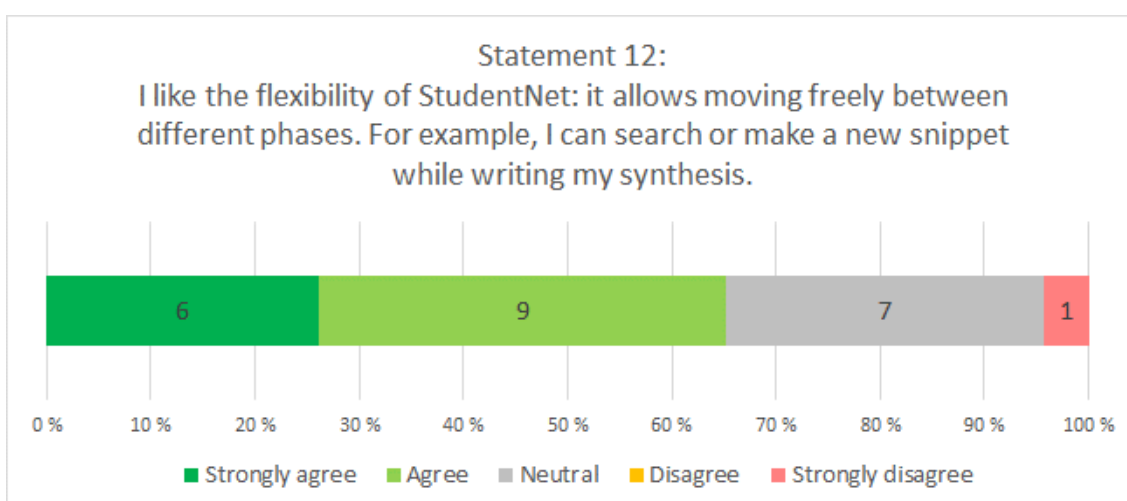


Figure 19. Distribution of responses to statement 12: I like the flexibility of StudentNet: it allows moving freely between different phases. For example, I can search or make a new snippet while writing my synthesis. 65 % chose an agreeing response.

7.1 Evaluation

The results of the experiment at Rauma Teacher Training School were interesting and quite encouraging. The main objective of the experiment was to gain information that can help in finding answers to the research questions:

- What specific things should be considered when developing an application for educational purposes?
- How does a nonlinear model work in an educational application, regarding the user experience of children?

Another objective was to get feedback that can be utilised in the development of StudentNet, both in improving existing features as well as in designing new ones.

The structure of Neurone -application is linear and strict: the user must progress from one stage to another in a pre-determined order. They start at search and bookmarking, proceeding to adding snippets. Then, they evaluate the relevance and reliability of the bookmarked pages, and finally, proceed to composing a synthesis using the bookmarked pages as the source of information. Although the workflow in Neurone is linear, the application still includes comprehensive support and gives clear instructions on how it is supposed to be used. This is one characteristic of an educational application that is targeted for children, as was pointed out in the previous chapter that discussed educational software.

StudentNet works in a totally different manner: the user can move freely between the different stages, although the stages are the same in both applications. It was presumed that this flexible structure of StudentNet provides an altogether better internet literacy experience. However, it was also presumed that this structure may prove to be somewhat more difficult for children to use and understand. It was planned to implement user support and instructions in StudentNet, at least of the same level than those in Neurone. Although the number of participants in the

experiment at Rauma was not very high, the results collected from the end questionnaire give some indication that supports the aforementioned plans.

According to the end questionnaire responses, participants of the experiment seemed to like the flexible model of StudentNet altogether. However, it was also learned that StudentNet was not considered distinctly easy to use. This was not surprising, since the groups of pupils that participated in the Rauma Experiment consisted of children with different levels of computer skills, motivation and concentration.

In addition to the questionnaire feedback, some information was gained from observations during the experiment. It was noticed that there are quite distinct differences between pupils. Some of them could work completely independently, while some needed extensive support. In conclusion, the answer to the second research question is that the nonlinear model works very well for children in general, but some user support and instructions is needed in the application for nonlinear model to be great for all pupils.

7.2 Further development

The development of StudentNet continues so its functionality and features are not limited to those that were described in this thesis. The first implemented module was the search module, followed by all other core modules. Some features that are planned to be implemented in the future are part of the initial plans while others originated from ideas during the development.

The most important and probably the largest entirety among pending features is the user management system. It is important because, for instance, it solves the problem of managing what resources, tools, and information a user has permission to access. It is essential that users are identified and properly authenticated in this kind of system when it is released for normal use. A user management system is also a prerequisite for other important features, such as a teacher's or admin's control

panel. For example, teachers and researchers will be able to create and modify assignments within the application and assign exercises to pupils individually.

A support module is a feature that has been planned since the beginning of the project but was not implemented during the writing of this thesis. It was also found out from the experiment feedbacks that support is a quite desired feature by many users. The development of a support module is starting in the very near future and is the main topic of another thesis.

Regarding the software application, this thesis focused on the technical implementation of StudentNet and on providing the core features that satisfy all the pedagogical needs. Attention was also paid on the user interface and user experience but were not studied or optimised since they were not part of the fundamental topics and scope of this thesis. UI and UX is a large wholeness that can also be addressed as the main topic in another thesis in the future.

8 Conclusion

This thesis is associated with a project where one of the objectives was to develop a completely new web application for the purpose of practicing skills related to internet reading. As a basis for this new application was the software platform called Neurone.

The primary purposes of Neurone are research, data collection, and measurement. For that reason, the structure of Neurone is linear: the user flow is restricted so that there are stages that must be completed in a specific order. This kind of structure is desirable for research and measurement, but it does not fit very well for teaching.

StudentNet is under development with the primary purpose of teaching and to provide a natural environment for improving internet reading skills. To provide this experience that is similar to a real-world learning situation, the application should have a nonlinear structure and user flow. The nonlinear structure and nonlinear user flow mean that the user is not restricted to follow some pre-determined order of actions while using the application.

To be able to design and develop a good educational application, the topic had to be studied first. In chapter 2, web applications were discussed in general. The main concepts related to them, architecture types, and most popular frameworks and libraries were addressed. Chapter 3 introduced the specific software subcategory of educational software. This chapter included the different educational software types and the benefits of using educational software, but the most prominent goal was to find answer to the first research question: What specific things should be considered when developing an application for educational purposes?

In the development of educational software, both pedagogical and computer science factors must be considered, which are normally handled by two distinct people or groups. It was discovered that there are no common frameworks for this purpose and therefore there are no technical characteristics specific to educational software. Instead, the characteristics are related to the functionality and features of an application, e.g., how the application appears to the user and how the user can interact with it. User interface and user experience are fundamental matters in virtually any software targeted to end users in general, but they are quite different in an educational application where the end users are children.

Several practical experiments were arranged during the writing of this thesis. They were arranged in order to conduct pedagogical research and to have the application tested and naturally used in practice. The experiments yielded information that helped in the improvement of the application and in the development of new features.

The experiments included questionnaires that consisted of statements about the performance and appearance, as well as UI and UX of the application. In addition, there were statements with the purpose to provide information that would help in answering the research questions, particularly the second research question: How does a nonlinear model work in an educational application, regarding the user experience of children?

It was discovered that there are no fundamental problems with the nonlinear model altogether. For some individuals it can be a little challenging but that can be solved by providing good tutorials and support integrated in the application. One idea is to create a virtual guide inside the application that instructs what should be done in each situation and can be asked for help whenever needed. However, the distinct majority of users thought that StudentNet is easy to use, and nearly equal majority liked the flexibility of StudentNet. The gathered feedback also indicate that StudentNet was perceived useful, and overall has very good potential as a complementary learning tool.

References

- [1] Statista, "Digital population worldwide," [Online]. Available: <https://www.statista.com/statistics/617136/digital-population-worldwide/>. [Accessed 1 April 2021].
- [2] C. Kiili and L. Laurinen, "Näkökulmia tutkivan nettilukemisen opettamiseen," *Oppimisen ja oppimisvaikeuksien erityislehti*, vol. 29, no. 2, pp. 38-41, 2019.
- [3] F. Shull, J. Singer and D. I. Sjöberg, *Guide to Advanced Empirical Software Engineering*, London, UK: Springer, 2008.
- [4] V. R. Basili, "The Role of Controlled Experiments in Software Engineering Research," in *Empirical Software Engineering Issues*, pp. 33-37, Berlin, Germany: Springer, 2007
- [5] Y. Luchaninov, "Web Application Architecture in 2021: Moving in the Right Direction," 30 July 2021. [Online]. Available: <https://mobidev.biz/blog/web-application-architecture-types>. [Accessed 10 October 2021].
- [6] A. Yaskevich, "Types of Web Applications: From a Static Web Page to a Progressive Web App," 17 August 2018. [Online]. Available: <https://dzone.com/articles/types-of-web-applications-from-a-static-web-page-t>. [Accessed 28 May 2021].
- [7] R. Connolly and R. Hoar, *Fundamentals of Web Development*, Pearson, 2015.
- [8] Digital Skynet Corp., "Desktop App vs Web App: Comparative Analysis," [Online]. Available: <https://digitalskynet.com/blog/Desktop-App-vs-Web-App-Comparative-Analysis>. [Accessed 28 May 2021].
- [9] IBM Cloud Education, "Three-Tier Architecture," 28 October 2020. [Online]. Available: <https://www.ibm.com/cloud/learn/three-tier-architecture>. [Accessed 6 October 2021].
- [10] S. Ivanova and G. Georgiev, "Using modern web frameworks when developing an education application: a practical approach," in *Mipro*, Opatija, Croatia, 2019.
- [11] fullstackopen.com, "Fundamentals of Web apps," [Online]. Available: https://fullstackopen.com/en/part0/fundamentals_of_web_apps#ajax. [Accessed 25 April 2022].
- [12] MDN contributors, "Introduction to progressive web apps," [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps/Introduction. [Accessed 1 June 2021].
- [13] M. Martin, "What is MVC?," [Online]. Available: <https://www.guru99.com/mvc-tutorial.html>. [Accessed 10 June 2021].
- [14] D.-P. Pop and A. Altar, "Designing an MVC Model for Rapid Web Application Development," *Procedia Engineering*, vol. 69, no. 24, pp. 1172-1179, 2014.
- [15] MDN contributors, "MVC," [Online]. Available: <https://developer.mozilla.org/en-US/docs/Glossary/MVC>. [Accessed 4 August 2021].

- [16] J. Cai, R. Kapila and G. Pal, "HMVC: The layered pattern for developing strong client tiers," [Online]. Available: <https://www.infoworld.com/article/2076128/hmvc--the-layered-pattern-for-developing-strong-client-tiers.html>. [Accessed 10 June 2021].
- [17] BungeeLabs, "*Bungee Connect and Model-View-Adapter (MVA): The architectural pattern for next generation interactive cloud-based web applications*", 2009.
- [18] D. Sprowls, "An introduction to model-view-presenter," [Online]. Available: <https://teeps.org/blog/2015/08/14/20-an-introduction-to-model-view-presenter>. [Accessed 25 April 2022].
- [19] O. Cherecheş, R. Schiefer, J. Manning and C. Farwell, "react-guide/props-vs-state," [Online]. Available: <https://github.com/uberVU/react-guide/blob/master/props-vs-state.md>. [Accessed 4 April 2022].
- [20] Meta Platforms, Inc., "Getting Started - React," [Online]. Available: <https://reactjs.org/docs/getting-started.html>. [Accessed 1 June 2021].
- [21] Meta Platforms, Inc., "Virtual DOM and Internals - React," [Online]. Available: <https://reactjs.org/docs/faq-internals.html>. [Accessed 9 June 2021].
- [22] R. Harris, "Virtual DOM is pure overhead," [Online]. Available: <https://svelte.dev/blog/virtual-dom-is-pure-overhead>. [Accessed 9 June 2021].
- [23] Google, "What is Angular?," [Online]. Available: <https://angular.io/guide/what-is-angular>. [Accessed 14 June 2021].
- [24] E. You, "What is Vue.js?," [Online]. Available: <https://v3.vuejs.org/guide/introduction.html>. [Accessed 27 July 2021].
- [25] E. You, "Components Basics," [Online]. Available: <https://v3.vuejs.org/guide/component-basics.html>. [Accessed 4 August 2021].
- [26] P. Tchounikine, Computer Science and Educational Software Design, Grenoble, France: Springer, 2011.
- [27] M. Vannucci and V. Colla, "Educational software as a learning tool for primary school students," in *New Achievements in Technology, Education and Development*, In-Tech, pp. 311-324, 2010.
- [28] J. Castro-Sanchez, E. Del Castillo, J. Hortolano and A. Rodriguez, "Designing and Using Software Tools for Educational Purposes: FLAT, a Case Study," *IEEE Transactions on Education*, vol. I, no. 52, pp. 66-74, 2009.
- [29] M. Guzdial, M. McCracken and A. Ellito, "LCD: a learner centered approach to developing educational software," in *Frontiers in Education: Teaching and Learning in an Era of Change*, Pittsburgh, USA, 1997.
- [30] R. Carver, "Computer assisted instruction for a first course in computer science," in *Frontiers in Education: Technology-based Re-engineering*, Salt Lake City, USA, 1996.
- [31] D. Gacitua, *NEURONE's Experimenter Manual*, Universidad de Santiago de Chile, 2020.

- [32] R. González-Ibáñez, D. Gacitúa, E. Sormunen and C. Kiili, "NEURONE: oNline inqUiRy experimentatiON systEm," *Proceedings of the Association for Information Science and Technology*, vol. I, no. 54, pp. 687-689, 2017.
- [33] MongoDB, Inc, "MongoDB," [Online]. Available: <https://www.mongodb.com/>. [Accessed 25 October 2021].
- [34] MongoDB, Inc, "MongoDB vs MySQL Differences," [Online]. Available: <https://www.mongodb.com/compare/mongodb-mysql>. [Accessed 25 October 2021].
- [35] The Apache Software Foundation, "Solr," [Online]. Available: <https://solr.apache.org/>. [Accessed 25 October 2021].
- [36] Meteor Software, "What is Meteor?," [Online]. Available: <https://guide.meteor.com/>. [Accessed 26 October 2021].
- [37] D. Herron, *Node.js Web Development, Fourth Edition*, Birmingham, UK: Packt Publishing, 2018.
- [38] OpenJS Foundation, "About Node.js," [Online]. Available: <https://nodejs.org/en/about/>. [Accessed 12 April 2021].
- [39] npm, Inc., "About npm," [Online]. Available: <https://docs.npmjs.com/about-npm>. [Accessed 12 April 2021].
- [40] StrongLoop, IBM, and other expressjs.com contributors, "Express," [Online]. Available: <https://www.expressjs.com>. [Accessed 13 April 2021].
- [41] T. Bacinger, "A Simple Bootstrap Tutorial," [Online]. Available: <https://www.toptal.com/front-end/what-is-bootstrap-a-short-tutorial-on-the-what-why-and-how>. [Accessed 24 April 2022].
- [42] The PostgreSQL Global Development Group, "PostgreSQL: About," [Online]. Available: <https://www.postgresql.org/about/>. [Accessed 19 April 2022].
- [43] Elasticsearch, "What is Elasticsearch?," [Online]. Available: <https://www.elastic.co/what-is/elasticsearch>. [Accessed 14 April 2021].
- [44] Meta Platforms, Inc., "Optimizing Performance," [Online]. Available: <https://reactjs.org/docs/optimizing-performance.html>. [Accessed 5 May 2021].
- [45] University of Turku, "Rauman Normaalikoulu," [Online]. Available: <https://sites.utu.fi/rnk/en/>. [Accessed 7 December 2021].
- [46] M. Tavakol and R. Dennick, "Making sense of Cronbach's alpha," *International Journal of Medical Education*, vol. 2, pp. 53-55, 2011.
- [47] Y. Lee, K. A. Kozar and K. R. Larsen, "The technology acceptance model: Past, present and future," *Communications of the Association for Information Systems*, vol. 12, no. 50, pp. 752-780, 2003.
- [48] F. D. Davis, "Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology," *MIS Quarterly*, vol. 13, no. 3, pp. 319-340, 1989.