VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES

Zhihao Tan

Establishing a Web Application to Generate Model Canvases for Small Businesses

Information Technology
2020

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Information Technology

ABSTRACT

Author	Zhihao Tan
Title	Establishing a Web Application to Generate Model Canvases for Small Businesses
Year	2020
Language	English
Pages	56
Name of Supervisor	Pirjo Prosi

This thesis comes up with the design and implementation solution of the Creve application business model Canvas based on front and rear separation. The thesis is done as part of the project work offered by Muova. It is known that small businesses strive to realize their targets but may face more pressure to get solutions, customers, resources, or money. Therefore, the thesis aims at establishing a web application to help users from small businesses in Finland to analyze their situation, build business models, communicate, and improve knowledge based on the created lessons.

For technologies, Single Page Applications (SPAs) is selected to be used, because it has been developing rapidly and studied widely. In addition, SPAs are fast, flexible, and user-friendly. Furthermore, ReactJS is used for the front-end development, Nodejs and Express are used for back-end development, and PostgreSQL is used for database management system.

The result showed that the web application was successfully built, since all the required functions were realized, and the application could run fluently. This thesis discussed the justification of the topic, explained relevant theories, clarified requirement analysis and deployment, as well as assessment of testing and conclusion.

Keywords Front and rear separation, React, NodeJS, Express, PostgreSQL

CONTENTS

LIST OF FIGURES AND TABLES

List of Abbreviations

API	Application Programming Interface
CLI	Command-line Interface
CRUD	Create, Read, Update, and Delete
HTML	Hyper Text Markup Language
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure
JSON	JavaScript Object Notation
RDBMS	Relational Database Management System
REST	Representational State Transfer
SPA	Single-page Application
SQL	Structured Query Language
URL	Uniform Resource Locator

1 INTRODUCTION

1.1 Thesis Objectives

The objectives for the thesis can be summarized into two parts, which include building a web application and writing a thesis. For the former part, the objectives are: (1) Creating a website, which can accumulate information from users and use this information to contribute the business model (can be saved as a PDF file), and mentor can give feedback at the end; (2) Realizing functions such as making different users able to generate multiple business models (this is the key part), offering lessons, and providing a discussion platform; and (3) Offering different views of business models for users to check. In other words, the web application-building part focuses on the establishment of the website and make it function.

For the written thesis, the objectives are: (1) Reviewing and explaining the theories of technologies that have been used for constructing the website; (2) Reporting the processes about how the web application-building was realized step by step, which include analysis, design, implementations and deployment; and (3) Following the writing guidance for bachelor's thesis and writing the thesis in an academic way to graduate successfully.

The thesis is done as part of the project offered by Muova. For the project, it is necessary to build a website to be used in Finland area that can benefit entrepreneurs or companies in the various industries to achieve new business based on the exploitation of creative know-how. In order to realize the project target in a more effective and efficient way, a bachelor's thesis is decided to be part of the project for Muova. The main contact person is Jari Ratilainen.

1.2 Thesis General Background

Today, with the continuous development of web technology, network platform and webpage supported systems are increasingly used in business activities to improve company efficiency and competitiveness. By establishing a single page web application, data can be stored and managed more effectively, customers can be connected in a better way, and companies may grow faster. (Evergreen 2020.)

Technologies are developing fast. Take front-end, for example, technologies are continuously developing, but the working efficiency of developers is limited since traditional code development methods work slowly and require high technology background. From 2013 when React was launched, the number of users has increased significantly since React helps making the process of writing components easier and more stable, creating higher productivity. Also, it is easier to learn and use (Ivanovs 2019).

Being a technological and innovative country, Finland provides an advantageous innovation environment for companies to develop, including the technologies such as artificial intelligence, wireless technology, Linux and 5G. In addition, Finland also have high macroeconomic and financial system stability. Employees in Finland are highly educated and skilled, international-minded, and fluent in English. (Business Finland: 2020a.)

Small and medium-sized enterprises (SMEs) and micro enterprises are regarded as the key players in Finnish economy since they include 99% Finnish companies. A micro enterprise means a company with less than 10 employees, and a SME means a company with more than 10 but less than 250 employees. These companies contribute about 40% of Finnish GDP. (Yrittäjät: 2020.)

As another rapid developing part of economy, there are about 4,000 start-ups established in Finland each year (Business Finland: 2020b). SMEs are more structured and organized to make profits, while start-ups are to a larger extent on the way to search for solutions to find customers, sell products and create value. (Baskerville 2015.)

Small businesses (which means SMEs and startups in this thesis) may face great challenges due to the decentralization of business services, or the lack of expertise and channels. Therefore, it can be interesting and useful to help small businesses to develop, such as creating a web application to analyze their situation and make possible improvements in the future. After a short discussion, since I am interested in topics that are relevant to web application, and I hope to practice my skills and graduate, Muova decided to offer me an opportunity to do a thesis to realize the combination of web technology and the needs of business.

1.3 Thesis Needs and Project Objectives

About the company:

Muova (official full name: Western Finland Design Center Muova) was generated from an independent institution of the University of Art and Design Helsinki in 1988. Since then, Muova started building cooperation relationships with Aalto University as well as companies. In 2014, the administration of Muova transferred to Vaasa University of Applied Sciences, and it is located in Palosaari Business Center, Wolffintie 36 F. (Wikipedia 2020a.)

Muova defines itself as "*a joint research and development platform between VAMK, Aalto University and the University of Vaasa*." Muova has employed about ten people, which include experts in designing, researching, as well as marketing and communication directions. Muova aims at offering innovative solutions for companies about the following development areas: (1) Designing research projects for companies to meet the market needs; (2) Organizing educational activities and giving local students the possibilities to help companies to get resources and new ideas; and (3) Providing international projects to generate ideas, knowledge, solutions and developing models in the national as well as international environment. (Muova 2020.)

About the project:

Creve 2.0 digital is a project funded by European Social Fund, which aims at improving the supply and quality of education. It is a joint project implemented by several organizations, which include Turku University of Applied Sciences, Aalto University, Häme University of Applied Sciences, University of Lapland, Österbottens Hantverk, and Muova. The project is guided by people from different organizations, for example, Petra Tarjanne from TEM, Kirsi Kaunisharju from OKM and Sami Häikiö from Business Finland. (Creve 2020.)

It is known that small and medium-sized enterprises are facing challenges such as (1) Decentralized business services; (2) Lack of creative and professional knowledge and productization skills; (3) Lack of export expertise and channels; and (4) Lack of digital capabilities in production and distribution. Therefore, the target of the project is offering training and advising services about assessing entrepreneurship and analyzing business potentials to meet the needs of the creative industries in Finland. The target group of the project includes entrepreneurs in creative industries, business consultants, and teachers of entrepreneurship. (Creve 2.0 2020.)

In order to realize the project target, the main tasks can be summarized into four key areas: (1) Establishing a national service website which can offer advisory, support, and development services for companies with creative ideas from different industries; (2) Creating an evaluation model of products/services, providing help for developing ideas, and offering courses for companies to increase their potential; (3) Providing knowledge-based short trainings to offer people and organizations to develop competences; and (4) Offering communication possibilities for companies to get ideas and information from other users or customers. (Creve 2.0 2020.)

In the following chapters, the thesis will firstly include a literature review of technologies that have been used. After that, the requirement of the application will be analysed. Then, the thesis will explain the reasons and processes of user interface design, implementation, and deployment. Next, a brief testing will be made. Finally, the results will be concluded, and the reliability and the limitations of the thesis will be discussed.

2 RELEVANT TECHNOLOGIES

A brief review of relevant technologies and framework involved in the whole project will be discussed in this chapter.

2.1 Single-Page Application

Single-page applications， known as SPAs, can dynamically rewrite the current page to interact with the user without reloading the entire page in browsers when the URL changes (Flanagan 2006). SPAs are front end web applications using HTML, JavaScript and CSS (Facebook for Developers 2017).

Single-page applications have been widely used, and many traditional websites are or have been transformed into single page web applications, such as Gmail and Trello. Currently, the fashionable front-end frameworks such as React.js, Vue.js, and Angular all adopt the SPA principle (Wikipedia 2020b). Single-page applications mean that the front end has the initiative, and it also makes the front-end code more complex. That is why modularization, componentization, and architecture design have become more and more important (Skenix 2019).

Compared to traditional web applications, single-page applications achieve front-end and back-end separation. The back end is only responsible for processing the data and provide the interface. The page logic and page rendering are handled by the front-end. SPAs still provide the user with the normal user experience, it shows different page with different URL, instead to have multiple html files. It uses JavaScript to render different pages for different path, the parts of that single page or the entire single page rendered depending on which path the user navigated to in the application. (Microsoft 2019.)

2.2 React Related Technology

React is one of the most popular front-end libraries today, and its popularity cannot be underestimated. From a narrow perspective, react is a good JavaScript framework. From a generalized perspective, the react contains a complete front-end technology ecosystem (React 2020). This section will explain the various React related technology used by this thesis.

2.2.1 React

React is an open source JavaScript library for building user interface. It has the significant features of declarative design and componentized design. Using a declarative way to design a user interface makes the code more predictable and easier to debug (React 2020). React also builds each page as a component, and each component independently maintains its own internal state. It can be more reasonably and easy to transfer data between pages in application.

2.2.2 Redux

Redux is a standalone third-party library often used in React project in order to manage the states. (Redux 2020). The key concept of Redux consists of three parts: Action, Reducers and Store. Actions are information packages mostly used to send data from user interface to the redux. After user interaction, internal events or API calls, the page needs to send data to the store to modify the current page state. Reducer is used to modify the data in the store according to the processing logic of the Action. Reducer receive the action and old state as input and return updated states. Reducer describes how to update Store after responding to actions, it is avoiding modify the state directly. Store is the central object that manages the state of the entire web application state, the updated state returned from reducer will be saved in the store (Dev 2020.) An example of redux flow can be found in Figure 1.

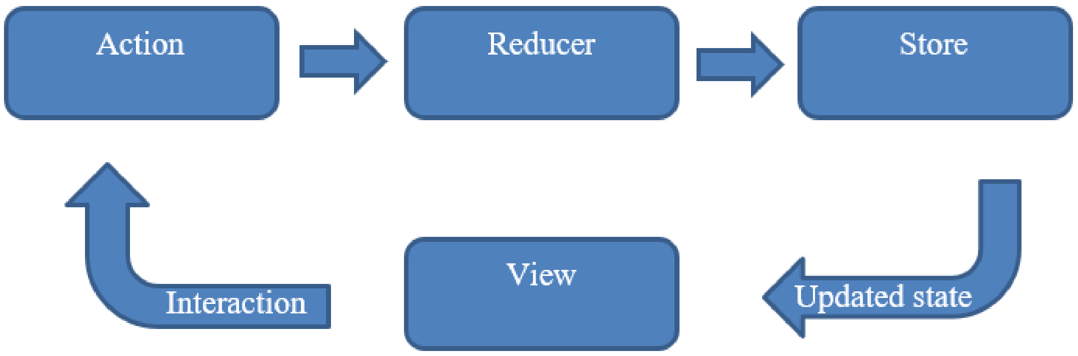


Figure 1 Redux flow

2.2.3 React-Router

React-router is a routing solution of the React system, and the routing mechanism is one of the most important elements of building a SPAs. Through the routing in the front-end, the user experience can be optimized without having to get all the data from the server every time, so that the page can be quickly displayed to the user. (FreeCodeCamp 2016.)

React-Router keeps the user interface in sync with the URL, it contains a series of powerful functions such as dynamic route matching, establishing the correct position transition processing, code buffer loading and so on. On-demand loading of the pages can increase page loading speed for these large websites. During the rendering of the front-end page, react router will only load the page that the user currently selected. After the user changes the page URL, the react-router will dynamically follow the page’s needs according to the routing matching rules defined by the developer. Then it will re-render web page into the browser. (Dabit 2016.)

2.3 Axios

In traditional front-end projects, most HTTP requests are completed by Ajax XMLHttpRequest, its API design is rough and the configuration and calling methods are also hard to use. Therefore, a more powerful and flexible third-party data fetching library Axios is chosen to use in this application. Axios, which works similar to Fetch API is a light weight promise based HTTP client for the browser and NodeJS (Npmjs 2020). It is simple to use Axios send HTTP requests to RESTful API and perform CRUD operations and receive the automatically parsed returned json data.

2.4 ECMAScript 6

ECMAScript 6 short as ES6 is a new generation standard of JavaScript language which was officially released on 2015. Its appearance makes the JavaScript language an enterprise-level development language, giving JavaScript code ability to write more complex and large web applications. There are also some new syntax features in the ES6 version of JavaScript, which including arrow function, the proxy and reflect, promise object, module for example. (W3schools 2020.)

2.5 CSS module

The rules of CSS are defined in a global way, and the style rules of any component are valid for the entire page. People who write CSS may encounter the problem of style conflict and naming confusion (Time 2020). Therefore, tools like CSS module, Sass and Less have appeared.

Since CSS module can automatically create a unique class name, the class name will not be duplicated with other selectors when generating a local scope. (CreateReactApp 2020). The class name is generated with the following format:

[filename]_[classname]__[hash]

Here is an example of automatically generated unique class name by CSS module:

class="modelView_option__EYsws"

CSS module helps to scoping and applying style to the certain component instead of the entire application, it also gives the possibility to use the same CSS class name in multiple files without worry about naming clashes (CSSTricks 2017). As a result, the CSS module is chosen to be used in this thesis application.

2.6 NPM

NPM, short for Node Package Manager, is two things: first and foremost, it is an online repository for the publishing of open-source Node.js projects; second, it is a command-line utility for interacting with said repository that aids in package installation, version management, and dependency management. (Nodejs 2011). NPM will manage the required modules of the project based on the "package.json" file of the project and automatically maintain the dependencies. The available version of each module specified in "package.json" file is to avoid compatibility issues caused by module updates. Furthermore, NPM is very convenient for developers to find and use various dependent packages.

2.7 Express

Express was initially released in November 2010 and it is the most popular Node web framework. It provides a fast, open and simple way to help developers build web server applications. Although Express itself is minimalist, the developers have created and provided a wealth of HTTP tools and compatible middleware packages, so that we have the flexibility to use various middleware to build web applications quickly. (MDN Web Docs 2020.)

2.8 NodeJS

NodeJS is developed by Ryan Dahl based on chrome V8 engine and it is released in May 2009. It is an open source, cross platform runtime environment that allows developers to create all kinds of server-side tools and applications in JavaScript. (MDN Web Docs 2020.)

Before NodeJS was created, JavaScript language has been widely used as a client-side programming language, and programs written in JavaScript were often executed on browsers. The emergence of NodeJS brought JavaScript into server-side development. It enables developers to use NodeJS as a JavaScript runtime environment to establish server-side applications and client-side applications with the same language. Since most of the browsers can support JavaScript, this can enable the web application to be established in a simpler way. (Developers 2020.)

2.9 PostgreSQL

PostgreSQL is chosen to be used in this thesis, because it is an open source and full-featured object-relational database management system (RDBMS). PostgreSQL supports most of the SQL standards and provides many other modern features for developers, such as complex queries and multi-version concurrency control. (PostgreSQL 2020.)

PostgreSQL is cross-platform, compatible with different operating system, for example, Linux, OS X, and Microsoft Windows. Open source can be regarded as a main advantage of PostgreSQL and the source code can be seen by everyone. PostgreSQL is not possessed by any organizations or companies. Instead, it has been developing and maintaining by the public globally. (Douglas 2003.)

2.10 Heroku

Heroku is a PaaS cloud platform based on AWS and it is popular to be used by developers. This service is designed based on Git workflow. This service was originally designed to host Ruby applications, but Heroku later added support for languages such as Node.js, Clojure, Scala, Python, and Java (Heroku Dev Center 2020).

The advantage of Heroku is that it has many third-party add-ons, such as Postgres, MySQL, and Redis. (Heroku 2020). The deployment of the application is done through git, and the compilation and operation can be done automatically. Another reason that Heroku is chosen to be used to deploy the application since its basic services are free.

3APPLICATION REQUIREMENT ANALYSIS

In this chapter, a description of project target, needed functions, application roles and dataflow, the view of component tree, and database design will be explained. The functional requirements will be analyzed based on three different roles, which include user, mentor, and admin.

3.1 Visual Project Target

Based on the discussion with project team, the visual target of the website is to realize the model that users can create, edit, save, and delete. The model is designed for entrepreneurs to write down their thinking and generate thinking about their business roadmap. In order to realize the model, a well-designed data flow and structured database is needed. Figure 2 shows how the model looks.

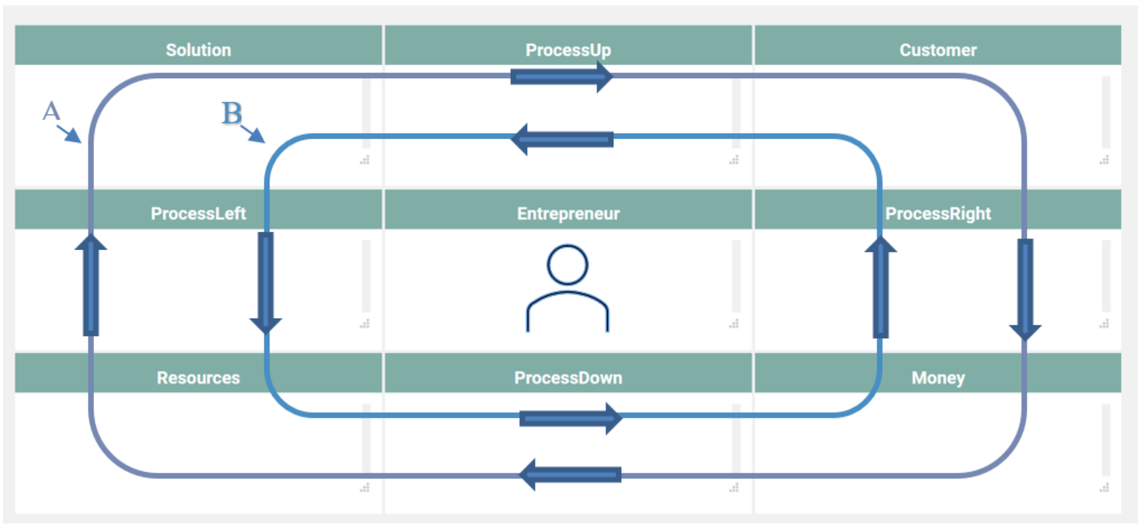


Figure 2 Model canvas data flow

As an entrepreneur, he/she needs to think about four key things when they are running the business – “solution”, “customer”, “money” and “resources”. Maybe the entrepreneur has only one or more of the four things. Therefore, the entrepreneur can use the model to help generate the organization's thinking about the process of how to use the existing thing to create something new, and finally form a circulation of the four things. In other words, an entrepreneur can start from any of the four things and think about how to realize the circulation based on a clockwise direction or a counterclockwise direction.

For example, entrepreneur A owns a high technological product, which can be regarded as "solution". Therefore, he needs to think about making efforts so that he can reach to "customers". When he starts to have "customers", he can consider about how to get "money" from "customers". Then, he can try to use the money to get more "resources" and use the "resource" to improve his technology and get a better "solution" for the business. As a result, a circulation can be formed, and his business can keep growing.

3.2 Functions that Need to Be Realized

Functions of the web application that should be realized can be regarded as the needs, prospect, or targets. Based on the project, the functional requirements are decided as following:

Must-have requirements include: (1) To include general website functions such as "login account" , "sign up account" and "edit account"; (2) To access two different views: "model view" and "process view" so that the users can focus on the results that they hope to check; (3) To generate PDF/PNG model canvases; and (4) To give mentor the right to offer feedback to the model that were created by user.

Should have requirements include: (1) To make the website possible to have notes and comments on model view; (2) To access available lessons list; (3) To reload saved model view. Nice to have requirements include: (1) Users can have their own setting page; (2) To add different users into the same model; and (3) Save model/process view as PDF report.

3.3 General Description of Roles

There are three main types of system users, which include user, mentor, and administrator. As in Figure 3, Creve application is mainly implemented using the following functions: modify model view, modify process view, access lesson list, generated PDF report, write and review feedback.

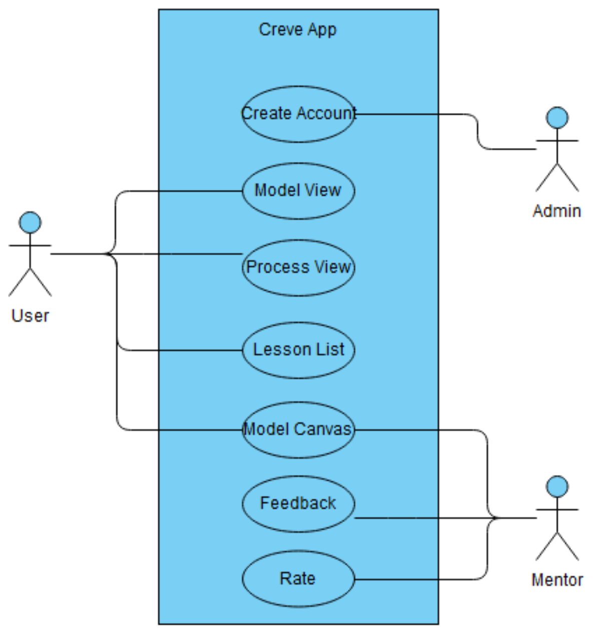


Figure 3 Use case diagram of user role

The “user” can get to the pages of model view, process view, lessons list and generated report.

The “Admin” can generate user account based on user role.

The “Mentor” can get the model from the user and give the rate and feedback to the user

3.4 Description of Data Flow

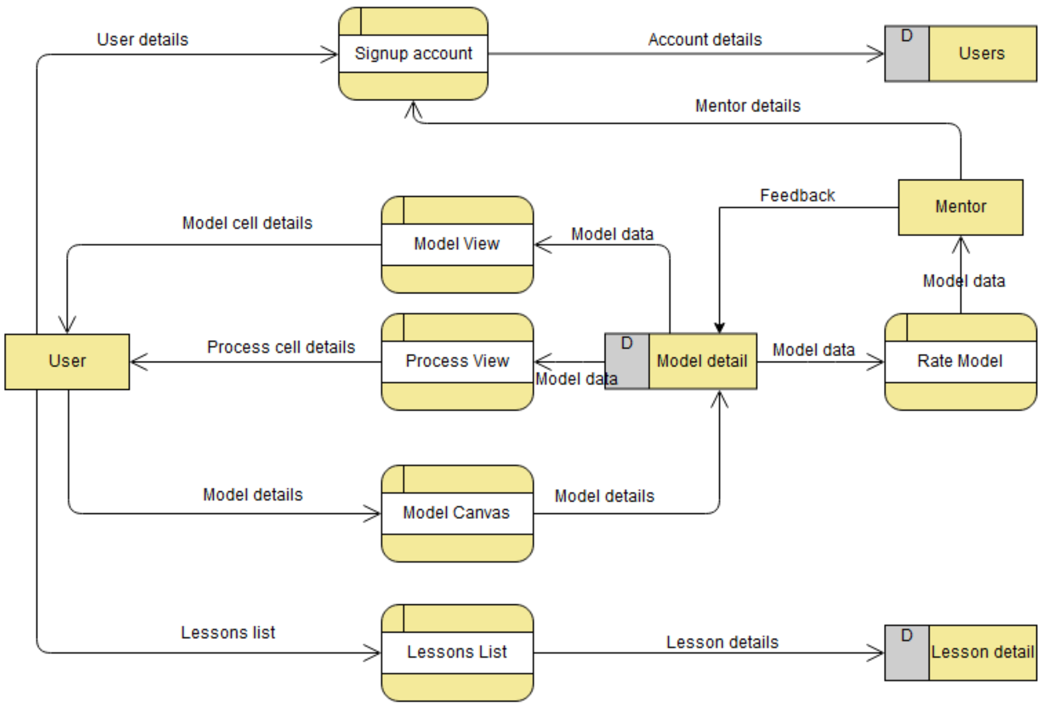


Figure 4 Data flow diagram

The data flow diagram (Figure 4) presents how the data goes into the application. The user has the right to access model view, process view and lesson list. The user can directly modify the model view and the process view. The user can also attempt lessons and put the selected notes into cell/box in the view page afterwards. When the model view has the needed information, it can be saved as a pdf report. The mentor has the right to access the user's model and leave feedback on the model.

3.5 Component tree

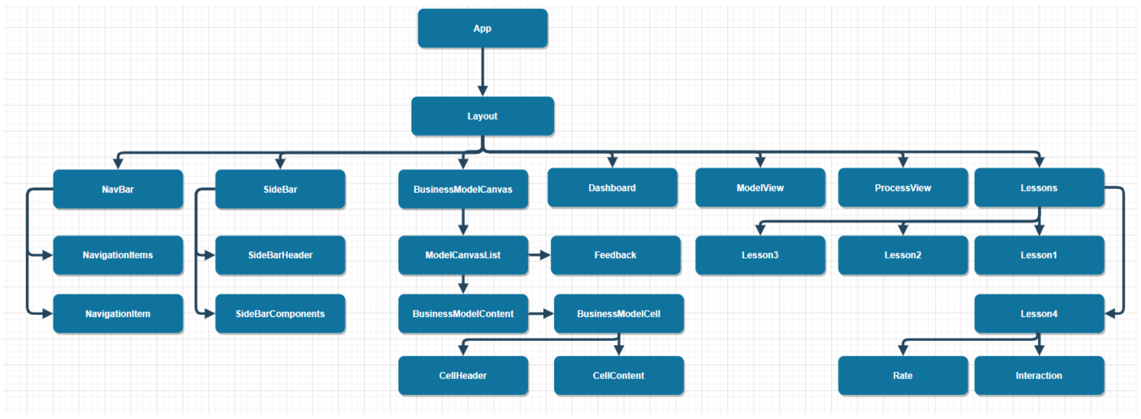


Figure 5 Component tree

The project component tree is shown in Figure 5. It is needed to define what the data flow of the application actually is and what it should be able to do. The goal of building the application is to enable the users to add and build their own model canvases. To build the component tree, the “App” component should be structured, and the “Layout” component is defined under it.

Based on the “Layout” component”, there are several other components nested, for example, “Dashboard” component, “ModelView” component, and “Lessons” component. In addition, there are also sub-components. Take “NavBar” component for example, it represents navigation, and it includes “NavigationItems” and “NavigationItem” component.

3.6 Database design

After an analysis of the application, the relationship among database tables are created in the following way so that data can be saved properly.

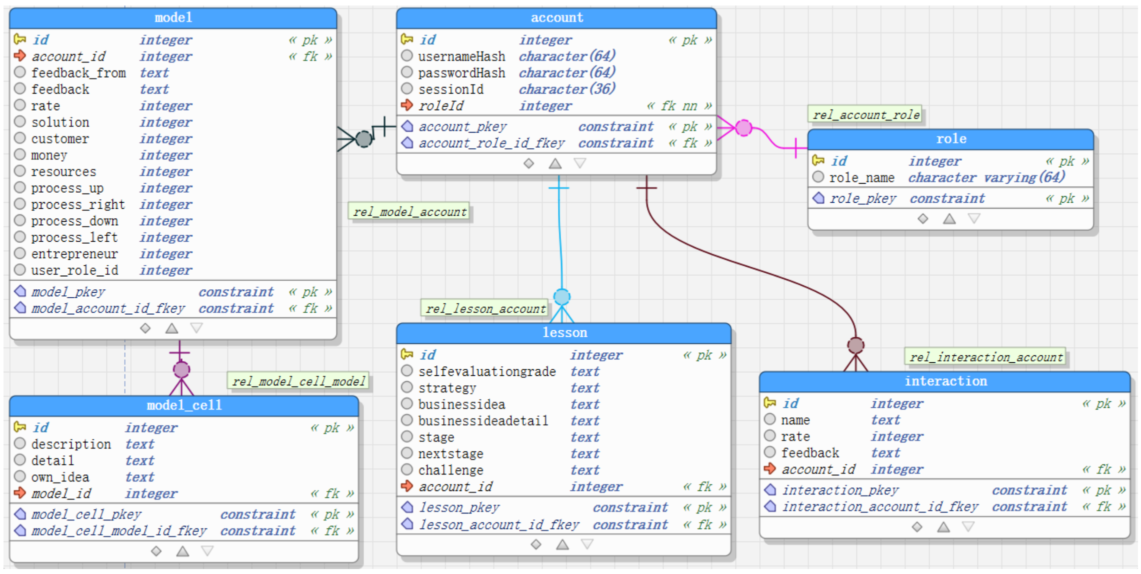


Figure 6 Database design

The Postgresql is used in the application. There are mainly 6 tables included, which are “account” table, “model” table, “model_cell” table, “lesson” table, “role” table and “interaction” table. Figure 6 shows the relationship among them:

Take the “role” table and “account” table as example, the “role” table defines the types of roles, which includes such as student, mentor, and admin. The “account” table includes account information, the hashed username and hashed password, and saved session ID here. The account instance has a role, so that it contains a foreign key referring to the id field of “role” table.

Take the “model” table and “account” table as example. One account may have different models, therefore, there is a foreign key “account_id” stored in the model table, the relation one to many. Similarly, each model can contain many “model_cell”, and hence there is a foreign key “model_id” stored into “model_cell” table. In the same way, “lesson” table, “interaction” table are linked to “account table”. It means that one account may have many lessons and interactions.

4 USER INTERFACE DESIGN AND IMPLEMENTATION

The front-end of this application is build using create-react-app tool, and the backend is using NodeJS with Express. In development stage, Postgres is running locally.

4.1 Login/signup page

The login/signup page is showing in Figure 7, it contains a signup item and a login item in the navigation bar and a login box in the middle.

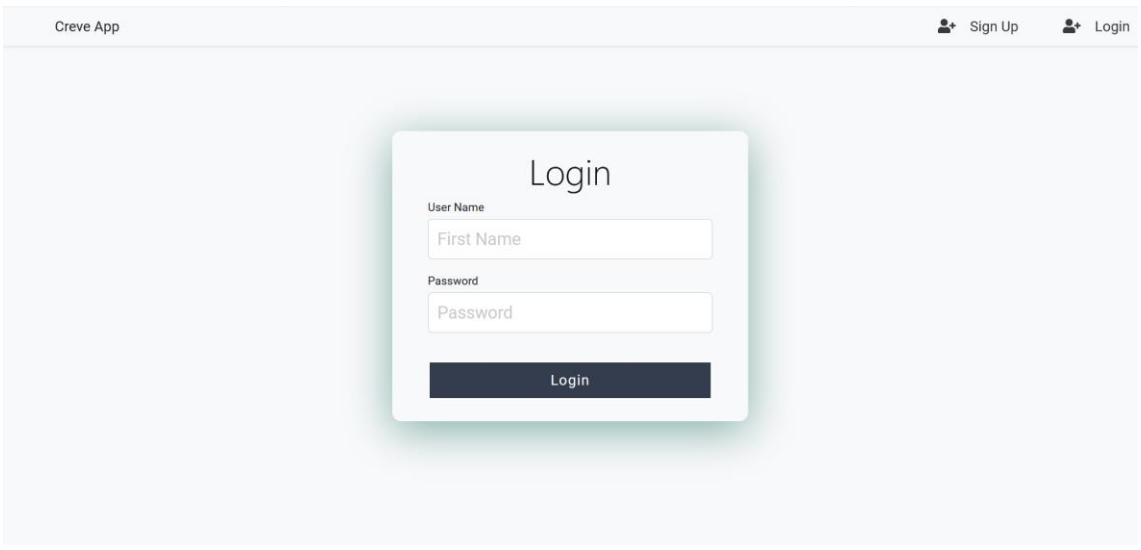


Figure 7 Login/signup page

The signup function: The username requires at least 6 chars and password requires at least 8 chars to improve the security of the application. The username and password will be send to NodeJS backend with a http post request when the user clicks the signup button. In the application backend, a defined application secret key will be appended to the received username value and password value in the request body.

The username/password value and application secret string will be hashed into 64 characters together by using the SHA256 hashing function provided by “crypto” library. That is to say, the username/password will not be saved as plaintext into database. Therefore, the application security will be improved.

After hashing the username and password result looks like: e.g.

‘1a7b487bbc54eb9420e59d8a41ee1c9e33f594d878c947e4616d63c4d4bd7067’

When the user clicks the login button and the account authentication is valid, the session will be generated by using “uuid” library and data will be saved into the database in the application backend. At the same time, the account username appends the unique id created by “uuid” library and they will be hashed together. Then the hashed string will be set as cookie to client. In addition, the “httpOnly” property inside the set cookie function provided by Express needs to be marked as true, which allows the cookie only access by the server. Furthermore, the “secure” property inside the set cookie function needs to set as true for HTTPS connection.

When the sign up or log in triggered, the dispatch method takes the type of action and the result from the returned json format data to the reducer, then a property named ‘loggedIn’ with the type of Boolean is set. The state will be saved in the store, and it is used to check is the application authenticated or not when the page rendering.

As a result, if the user logs in successfully, the layout component will be rendered. If the account is invalid, such as a wrong password, the error message will be shown in the box.

4.2 Dashboard page

Figure 8 displays the dashboard page which includes calculated model canvases that have been completed, the number of received feedback, and the list of created model canvases.

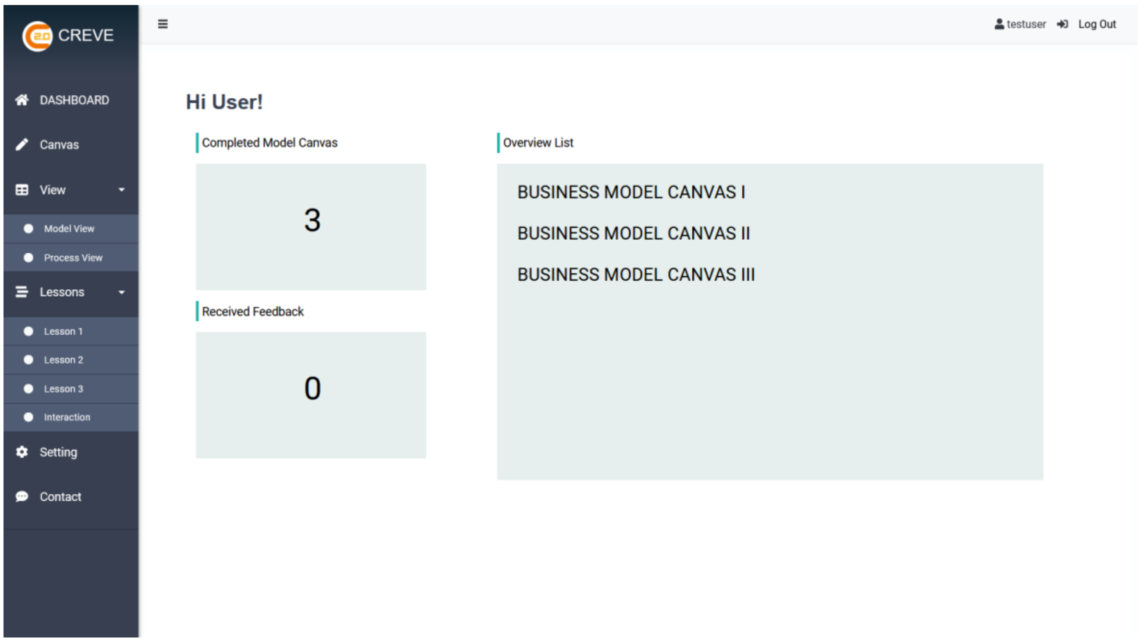
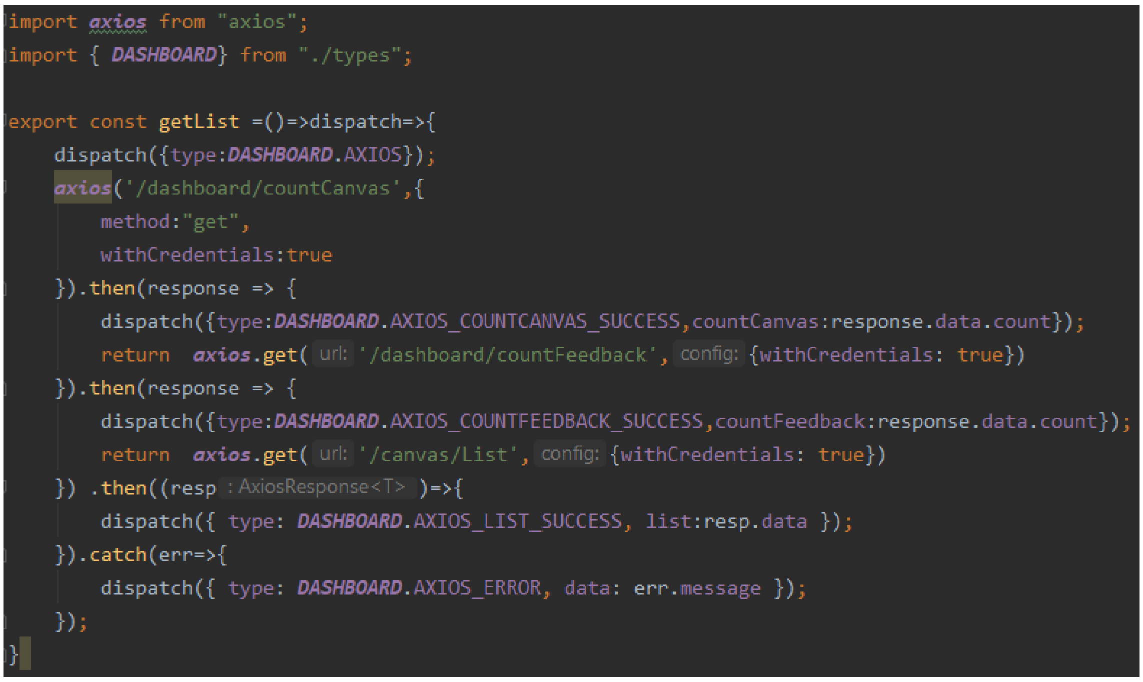


Figure 8 Dashboard page

The dashboard component is a functional component. When it is rendering, the function wrapped within "useEffect" will be called to send a get request to dashboard API provided by the application NodeJS backend. The returned Json data will be saved into store. Then, the state returned from reducer will be matched to props and showing in the dashboard specific field.

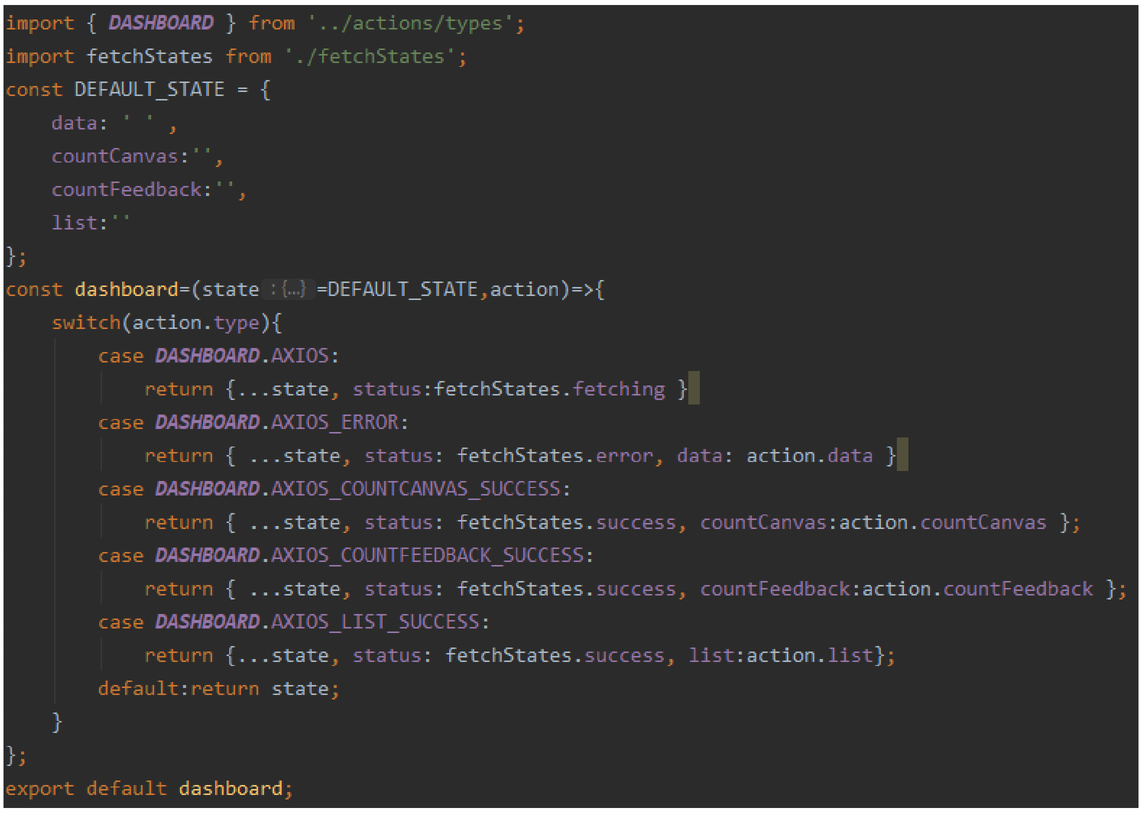
As one example of a dashboard redux flow shows in Figure 9 and Figure 10, when the "getList" function is called, it dispatches different action with its type and payload to the reducer. Then the reducer, which is a pure function, will take the action and the previous state into it. The updated state will return from the reducer and will be saved into the store. Then, the component which is connected to the store can use the state by match it into props.



```
import axios from "axios";
import { DASHBOARD} from "./types";

export const getList =()=>dispatch=>{
    dispatch({type:DASHBOARD.AXIOS});
    axios('/dashboard/countCanvas',{
        method:"get",
        withCredentials:true
    }).then(response => {
        dispatch({type:DASHBOARD.AXIOS_COUNTCANVAS_SUCCESS,countCanvas:response.data.count});
        return  axios.get( url: '/dashboard/countFeedback', config: {withCredentials: true})
    }).then(response => {
        dispatch({type:DASHBOARD.AXIOS_COUNTFEEDBACK_SUCCESS,countFeedback:response.data.count});
        return  axios.get( url: '/canvas/List', config: {withCredentials: true})
    }) .then((resp : AxiosResponse<T> )=>{
        dispatch({ type: DASHBOARD.AXIOS_LIST_SUCCESS, list:resp.data });
    }).catch(err=>{
        dispatch({ type: DASHBOARD.AXIOS_ERROR, data: err.message });
    });
}
```

Figure 9 Example of dashboard action



```
import { DASHBOARD } from '../actions/types';
import fetchStates from './fetchStates';
const DEFAULT_STATE = {
    data: ' ' ,
    countCanvas:'',
    countFeedback:'',
    list:''
};
const dashboard=(state : {...} =DEFAULT_STATE,action)=>{
    switch(action.type){
        case DASHBOARD.AXIOS:
            return {...state, status:fetchStates.fetching }
        case DASHBOARD.AXIOS_ERROR:
            return { ...state, status: fetchStates.error, data: action.data }
        case DASHBOARD.AXIOS_COUNTCANVAS_SUCCESS:
            return { ...state, status: fetchStates.success, countCanvas:action.countCanvas };
        case DASHBOARD.AXIOS_COUNTFEEDBACK_SUCCESS:
            return { ...state, status: fetchStates.success, countFeedback:action.countFeedback };
        case DASHBOARD.AXIOS_LIST_SUCCESS:
            return {...state, status: fetchStates.success, list:action.list};
        default:return state;
    }
};
export default dashboard;
```

Figure 10 Example of dashboard reducer

4.3 Model canvas

The canvas page includes the function of creating model canvases, loading model canvas lists, updating the selected model canvas, and deleting the chosen model canvas.

Figure 11 shows how to add a model canvas. When the canvas component is rendering, the list of model canvases which have been created before will be shown on the page. To create model canvas, the user needs to click add button located in the top right corner, and it will trigger the card component to ask user to enter the canvas name. The delete icon is located right after the created canvas name. When the user clicks the red cross delete icon, the selected canvas will be deleted.

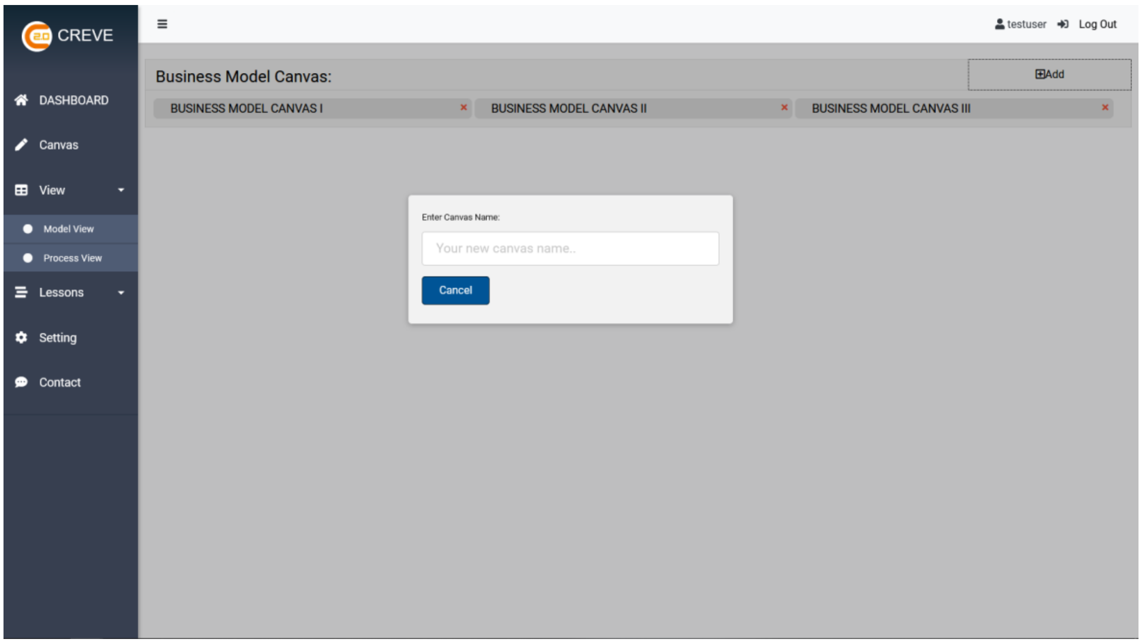


Figure 11 Add a canvas

When the user selects the name of the model canvas from the list (for example, business model canvas III), the chosen model canvas component will be rendered and displayed. It includes several model cells. If the user clicks the specific model cell, the user is able to view the details of the model cell and modify them. As shown in Figure 12, each model cell includes three fields that the user needs to fill in (for example, "Most important points", "Detail", and "Own ideas"). The words that the user have written in "Most important points" will be shown in the specific cell, while the other can only be shown when the user clicks on the specific cell. This setting is designed for only displaying the key points on the canvas page. In addition, after the user has completed his/her own canvas, the user can download the canvas model page in PDF/PNG format.

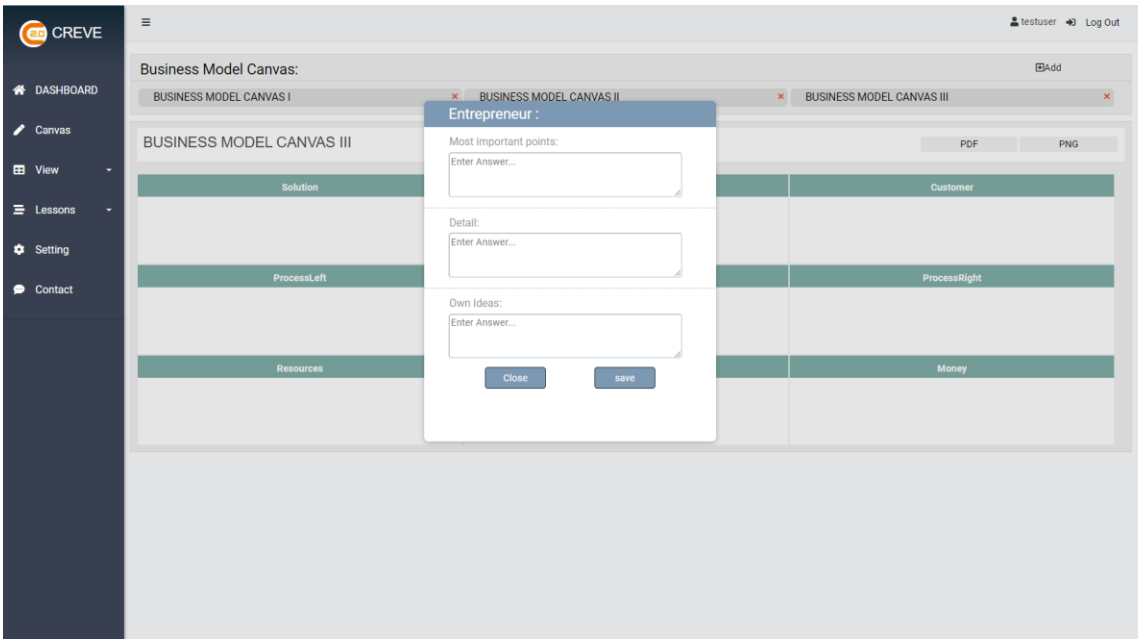


Figure 12 Edit data cell from selected canvas

4.4 Model view

After the user has completed his/her model canvas, there are two views designed for he/she to review his/her answers in a better way.

The model view (Figure 13) concentrates on the four key parts that the user (entrepreneur) has or seeking for when doing his/her business.

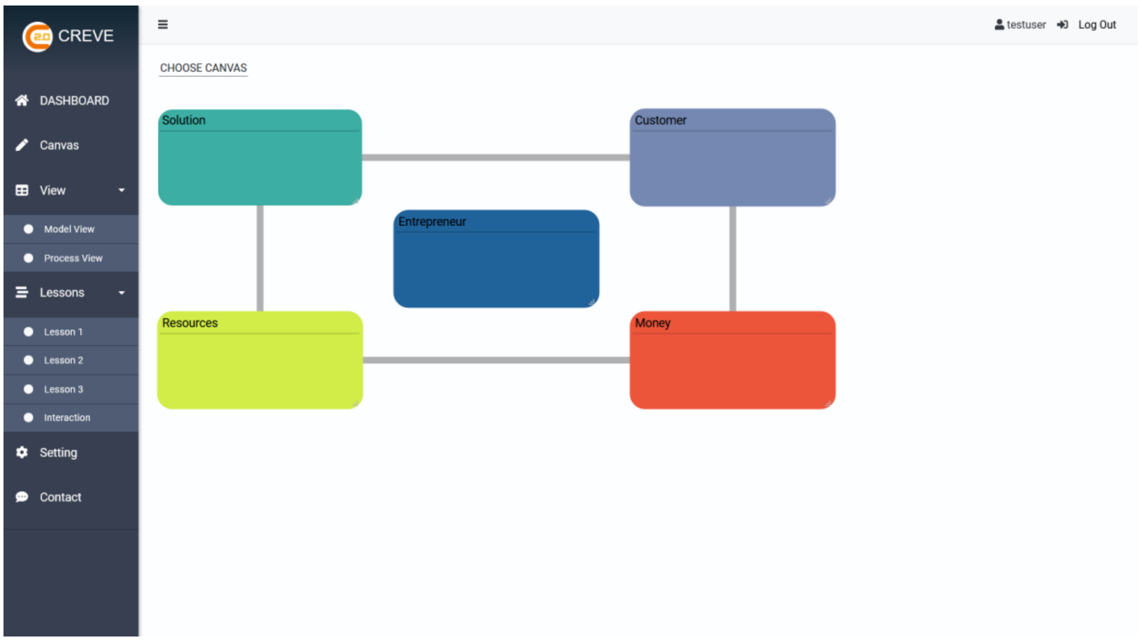


Figure 13 Model view

The process view (Figure 14) focuses on the process and connection between the four key points. This design aims for showing the user (entrepreneur) a clearer version to check if he/she can realize the circulation of the four key points to run his/her business.

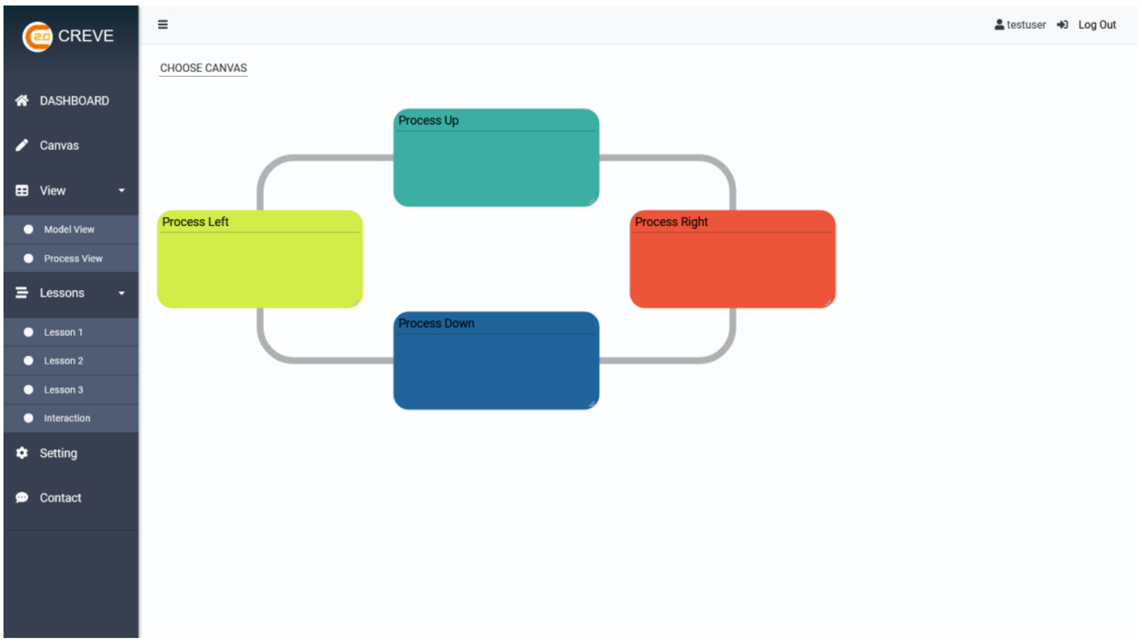
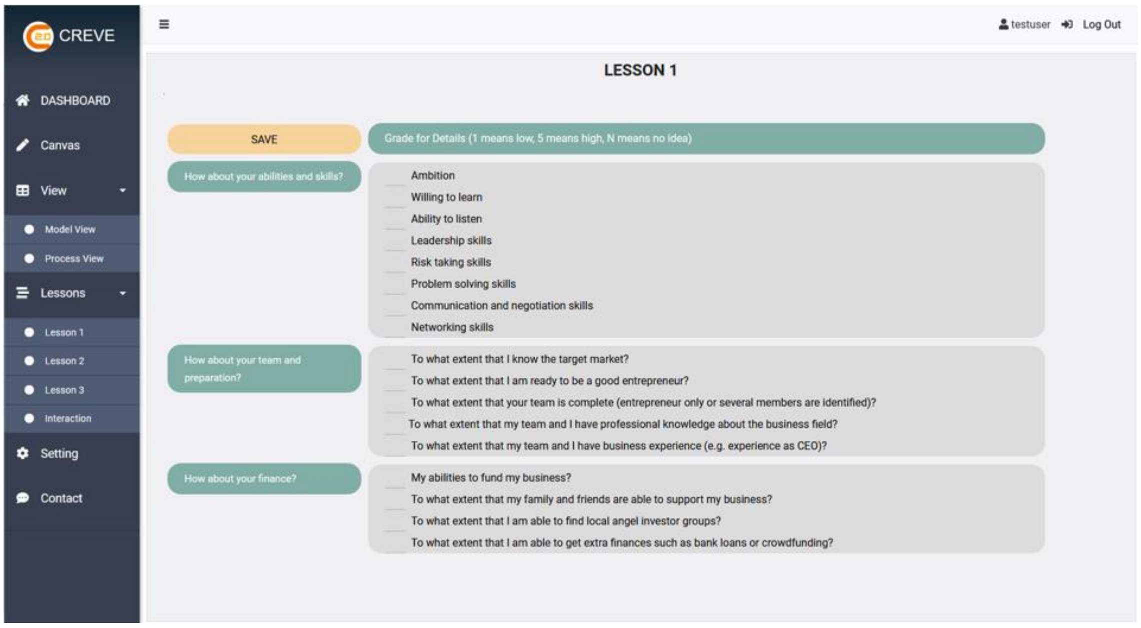


Figure 14 Process view

4.5 Lessons design

Lessons are designed for entrepreneurs (users) to take an overview about their business. Since entrepreneurs may only have an idea, a product/technology, or a simple plan for their business, which has more possibility to lead to a failure from a long-term perspective (Fortnum 2012). Therefore, the three lessons give entrepreneurs (users) an opportunity to check their abilities, market strategy, and preparation for the future.

From the first lesson, entrepreneurs (users) can give a grade about themselves. The lesson includes assessment of personal skills, team members' background, and financing for business. Financing is a key thing for businesses since it can support the whole processes from innovation to after-sales services, and having experienced and talented employees can better help the company to solve business challenges and can help the company to be more competitive (Fortnum 2012). In addition, it is important for entrepreneurs to review themselves so that they can build confidence of doing business, make decisions about developing the strengths, and find out as well as improve the aspects that they do not have/know (Zwilling 2015). Figure 15 shows the specific questions for user to answer.



LESSON 1

Grade for Details (1 means low, 5 means high, N means no idea)

How about your abilities and skills?

Ambition
Willing to learn
Ability to listen
Leadership skills
Risk taking skills
Problem solving skills
Communication and negotiation skills
Networking skills

How about your team and preparation?

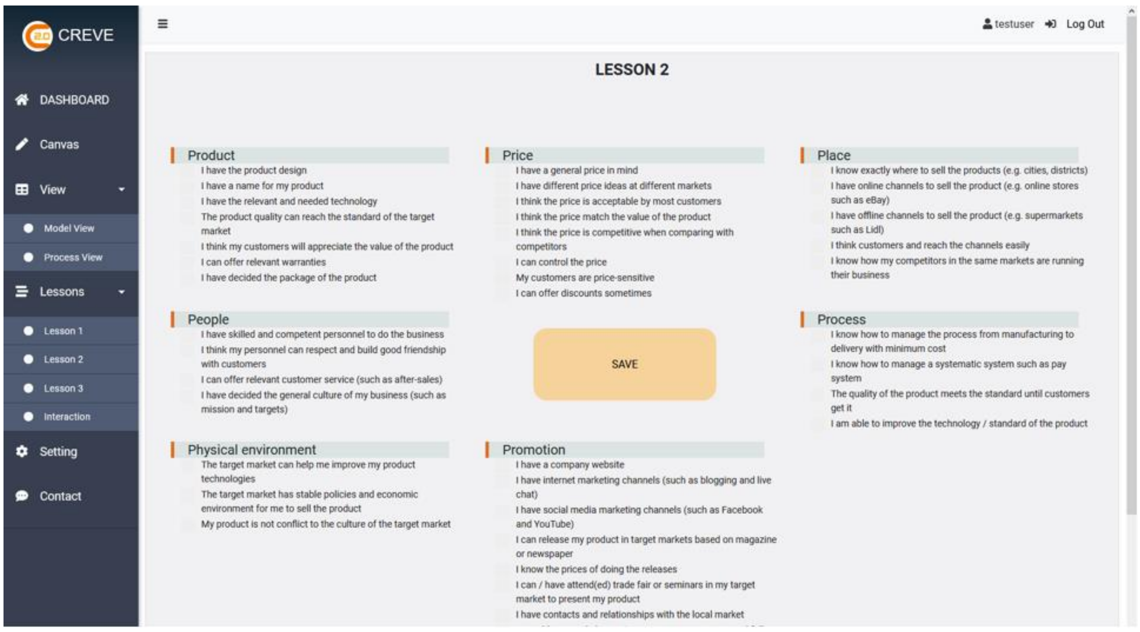
To what extent that I know the target market?
To what extent that I am ready to be a good entrepreneur?
To what extent that your team is complete (entrepreneur only or several members are identified)?
To what extent that my team and I have professional knowledge about the business field?
To what extent that my team and I have business experience (e.g. experience as CEO)?

How about your finance?

My abilities to fund my business?
To what extent that my family and friends are able to support my business?
To what extent that I am able to find local angel investor groups?
To what extent that I am able to get extra finances such as bank loans or crowdfunding?

Figure 15 Designed lesson 1

From the second lesson, users are supposed to mark the points that they have achieved in the list created based on marketing mix 7 Ps (product, price, place, promotion, people, process, and physical environment). Marketing is important since it includes a series of activities from creating, transferring, and distributing the goods to customers. While marketing mix can offer a simple way for a business to know its competitive strengths and possible weaknesses (Chai 2009), the check-list in Figure 16 can guide users to answer as well as develop the main aspects of their businesses.



LESSON 2

Product
I have the product design
I have a name for my product
I have the relevant and needed technology
The product quality can reach the standard of the target market
I think my customers will appreciate the value of the product
I can offer relevant warranties
I have decided the package of the product

Price
I have a general price in mind
I have different price ideas at different markets
I think the price is acceptable by most customers
I think the price match the value of the product
I think the price is competitive when comparing with competitors
I can control the price
My customers are price-sensitive
I can offer discounts sometimes

Place
I know exactly where to sell the products (e.g. cities, districts)
I have online channels to sell the product (e.g. online stores such as eBay)
I have offline channels to sell the product (e.g. supermarkets such as Lidl)
I think customers and reach the channels easily
I know how my competitors in the same markets are running their business

People
I have skilled and competent personnel to do the business
I think my personnel can respect and build good friendship with customers
I can offer relevant customer service (such as after-sales)
I have decided the general culture of my business (such as mission and targets)

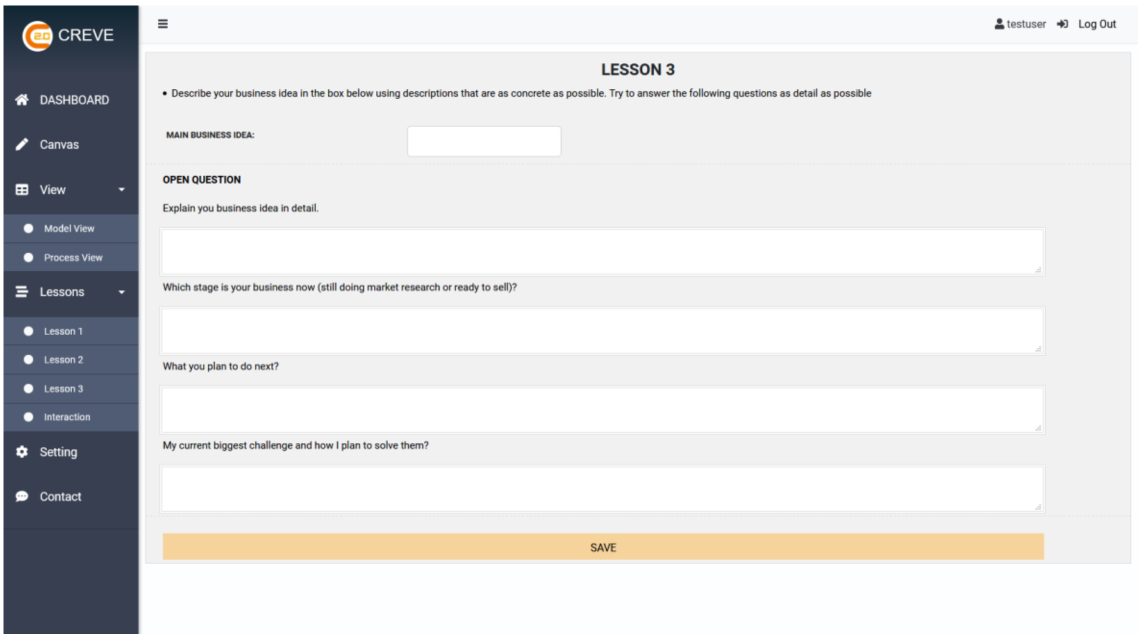
Process
I know how to manage the process from manufacturing to delivery with minimum cost
I know how to manage a systematic system such as pay system
The quality of the product meets the standard until customers get it
I am able to improve the technology / standard of the product

Physical environment
The target market can help me improve my product technologies
The target market has stable policies and economic environment for me to sell the product
My product is not conflict to the culture of the target market

Promotion
I have a company website
I have internet marketing channels (such as blogging and live chat)
I have social media marketing channels (such as Facebook and YouTube)
I can release my product in target markets based on magazine or newspaper
I know the prices of doing the releases
I can / have attend(ed) trade fair or seminars in my target market to present my product
I have contacts and relationships with the local market

Figure 16 Designed lesson 2

From the third lesson (Figure 17), users can give answers to some open questions, such as describing their current business, plans, and challenges. Making plans about businesses is one of the keys to success since it can help entrepreneurs to review their critical thinking before making decisions. For example, entrepreneurs can better avoid big mistakes, improve the viability of the thinking, set more suitable targets, and manage risks. (Lindzon 2019). By answering the questions, users (entrepreneurs) will be clearer about their thinking and the difficulties that they are currently facing, and they can be able to share the information with other users in order to gain some suggestions.



LESSON 3

• Describe your business idea in the box below using descriptions that are as concrete as possible. Try to answer the following questions as detail as possible

MAIN BUSINESS IDEA:

OPEN QUESTION

Explain you business idea in detail.

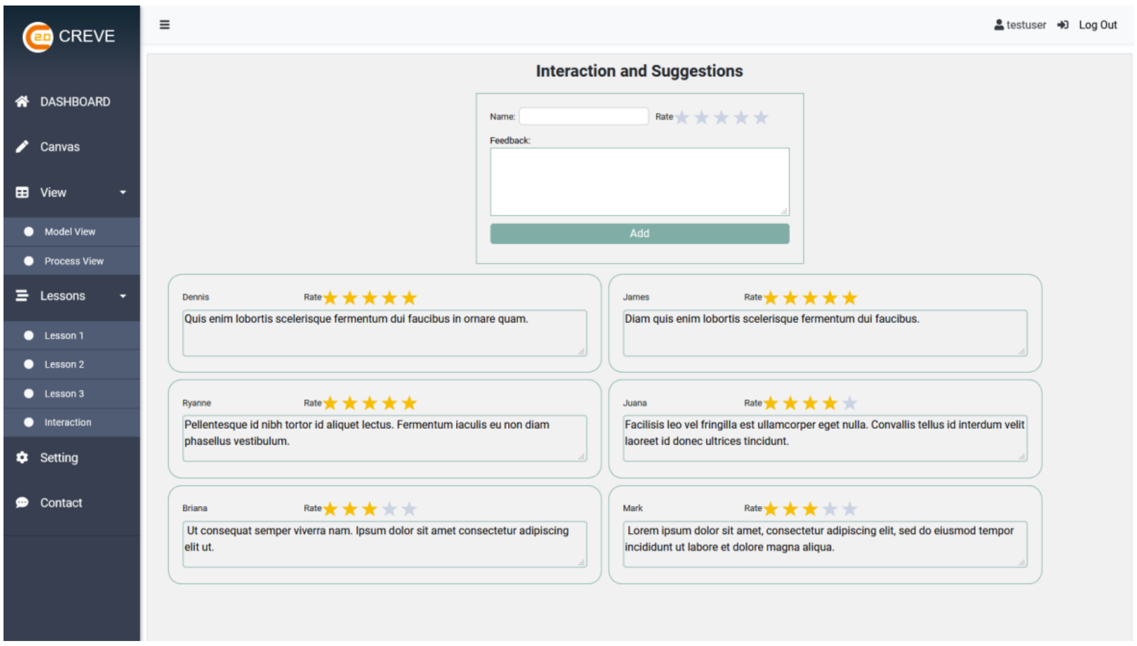
Which stage is your business now (still doing market research or ready to sell)?

What you plan to do next?

My current biggest challenge and how I plan to solve them?

Figure 17 Designed lesson 3

In order to offer users a platform to discuss and accumulate ideas from others, a platform is designed for offering feedback. Based on the information from the models that users have created and the lessons that the users have taken, other users can give suggestions for the overall business idea or a single point about how to improve the business. Different users can establish cooperation relationships or even become customers of others. Figure 18 shows the outlook of the interaction page, from which users can view the rating and comments from other users.



Interaction and Suggestions

Dennis — Rate ★★★★★
Quis enim lobortis scelerisque fermentum dui faucibus in ornare quam.

James — Rate ★★★★★
Diam quis enim lobortis scelerisque fermentum dui faucibus.

Ryanne — Rate ★★★★★
Pellentesque id nibh tortor id aliquet lectus. Fermentum iaculis eu non diam phasellus vestibulum.

Juana — Rate ★★★★☆
Facilisis leo vel fringilla est ullamcorper eget nulla. Convallis tellus id interdum velit laoreet id donec ultrices tincidunt.

Briana — Rate ★★★☆☆
Ut consequat semper viverra nam. Ipsum dolor sit amet consectetur adipiscing elit ut.

Mark — Rate ★★★☆☆
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Figure 18 Interaction page

4.6 Setting

Figure 19 shows the setting page of the application which handles the password resetting. The edit password icon will trigger the card for the user to enter the new password.

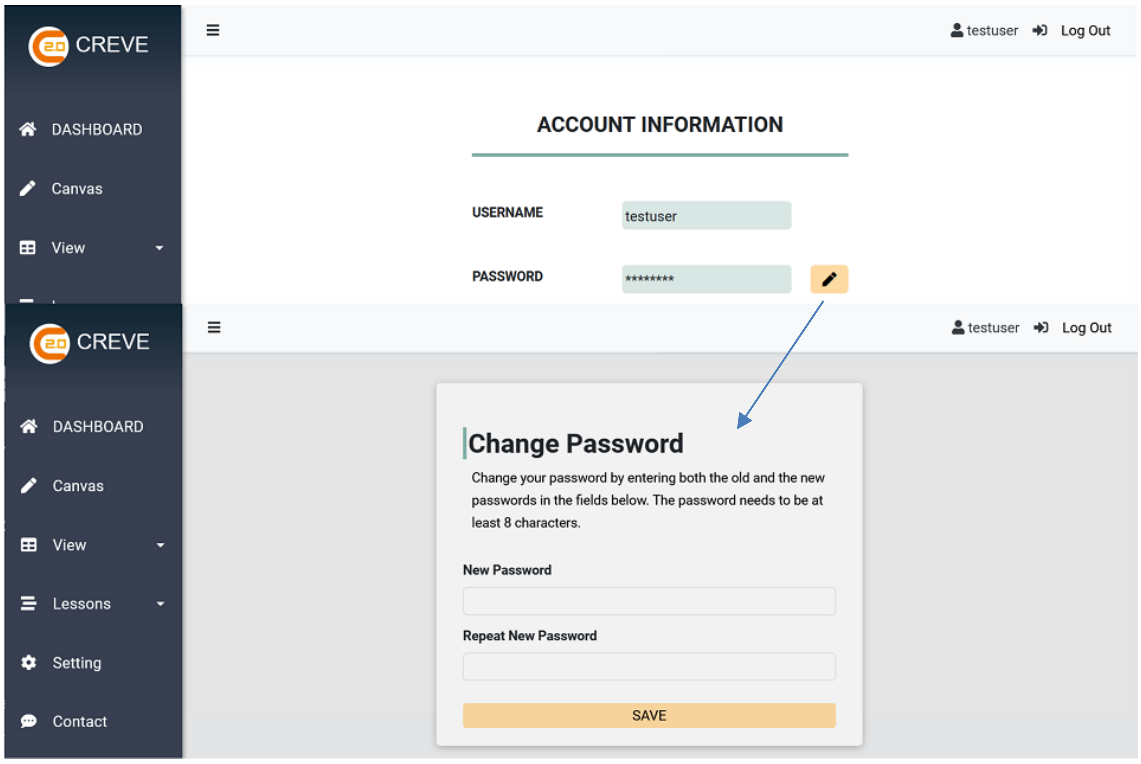
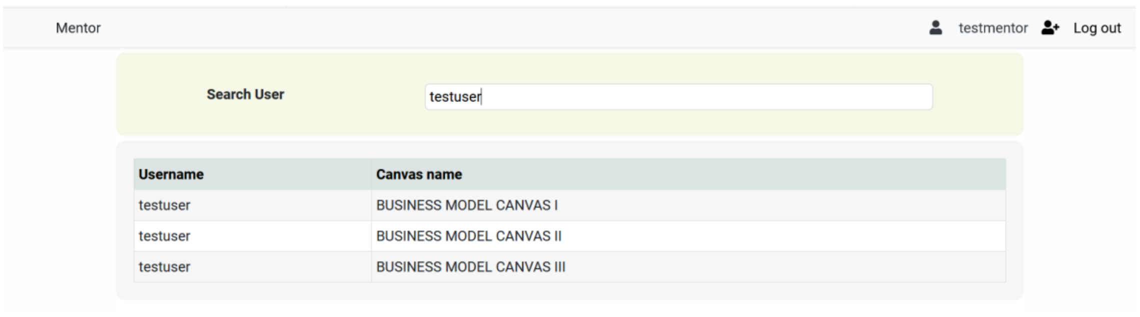


Figure 19 Setting page

4.7 Login as mentor

The user role ID will be saved into store when the user successfully logs in. Different user role will link to the different page.

When it comes to the mentor role: After the mentor logs in to the application, the mentor has the right to search for the user by entering the name of the user in the search box (Figure 20). The list of canvases created by the searched user will be displayed below the search box. The mentor can click any of the canvases from the list to check the details that users have created and saved.


Mentor

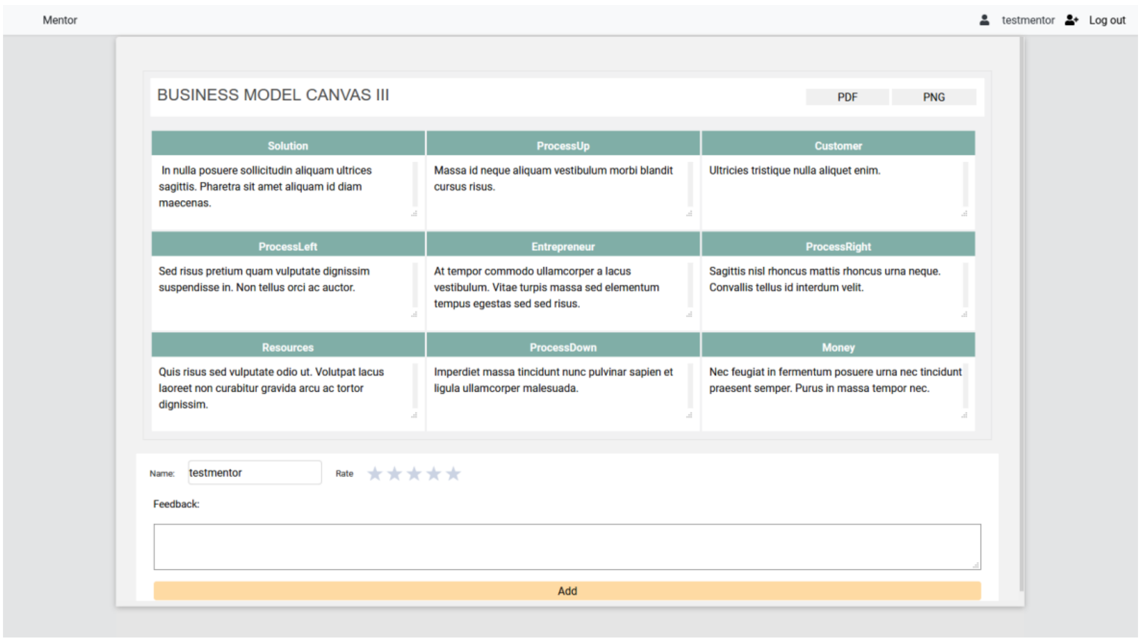
testmentor Log out

Search User testuser

Username	Canvas name
testuser	BUSINESS MODEL CANVAS I
testuser	BUSINESS MODEL CANVAS II
testuser	BUSINESS MODEL CANVAS III

Figure 20 Search user data from mentor page

By selecting one of the model canvases, as shown in Figure 21, the mentor is able to see the “most important points” that the user has filled. At the same time, the mentor can rate and give feedback for the selected model canvas.


Mentor

testmentor Log out

BUSINESS MODEL CANVAS III

PDF PNG

Solution	ProcessUp	Customer
In nulla posuere sollicitudin aliquam ultrices sagittis. Pharetra sit amet aliquam id diam maecenas.	Massa id neque aliquam vestibulum morbi blandit cursus risus.	Ultricies tristique nulla aliquet enim.
ProcessLeft	**Entrepreneur**	**ProcessRight**
Sed risus pretium quam vulputate dignissim suspendisse in. Non tellus orci ac auctor.	At tempor commodo ullamcorper a lacus vestibulum. Vitae turpis massa sed elementum tempus egestas sed sed risus.	Sagittis nisl rhoncus mattis rhoncus urna neque. Convallis tellus id interdum velit.
Resources	**ProcessDown**	**Money**
Quis risus sed vulputate odio ut. Volutpat lacus laoreet non curabitur gravida arcu ac tortor dignissim.	Imperdiet massa tincidunt nunc pulvinar sapien et ligula ullamcorper malesuada.	Nec feugiat in fermentum posuere urna nec tincidunt praesent semper. Purus in massa tempor nec.

Name: testmentor Rate ☆☆☆☆☆

Feedback:

Add

Figure 21 Display selected user data and give feedback

4.8 Login as admin

When it comes to the admin role: After the admin role account login to the admin page, he/she is able to create the user (Figure 22).

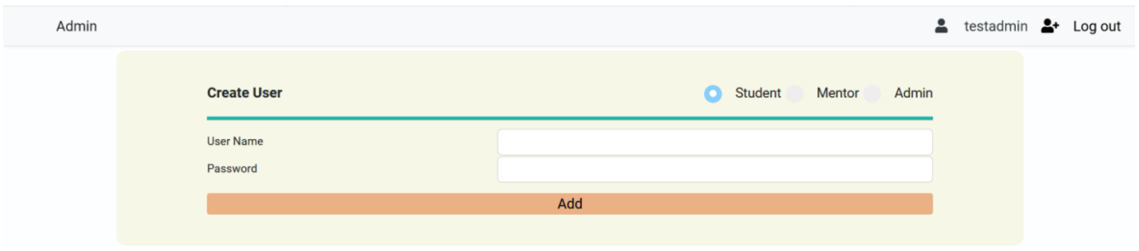


Figure 22 Admin page

4.9 RESTful routing

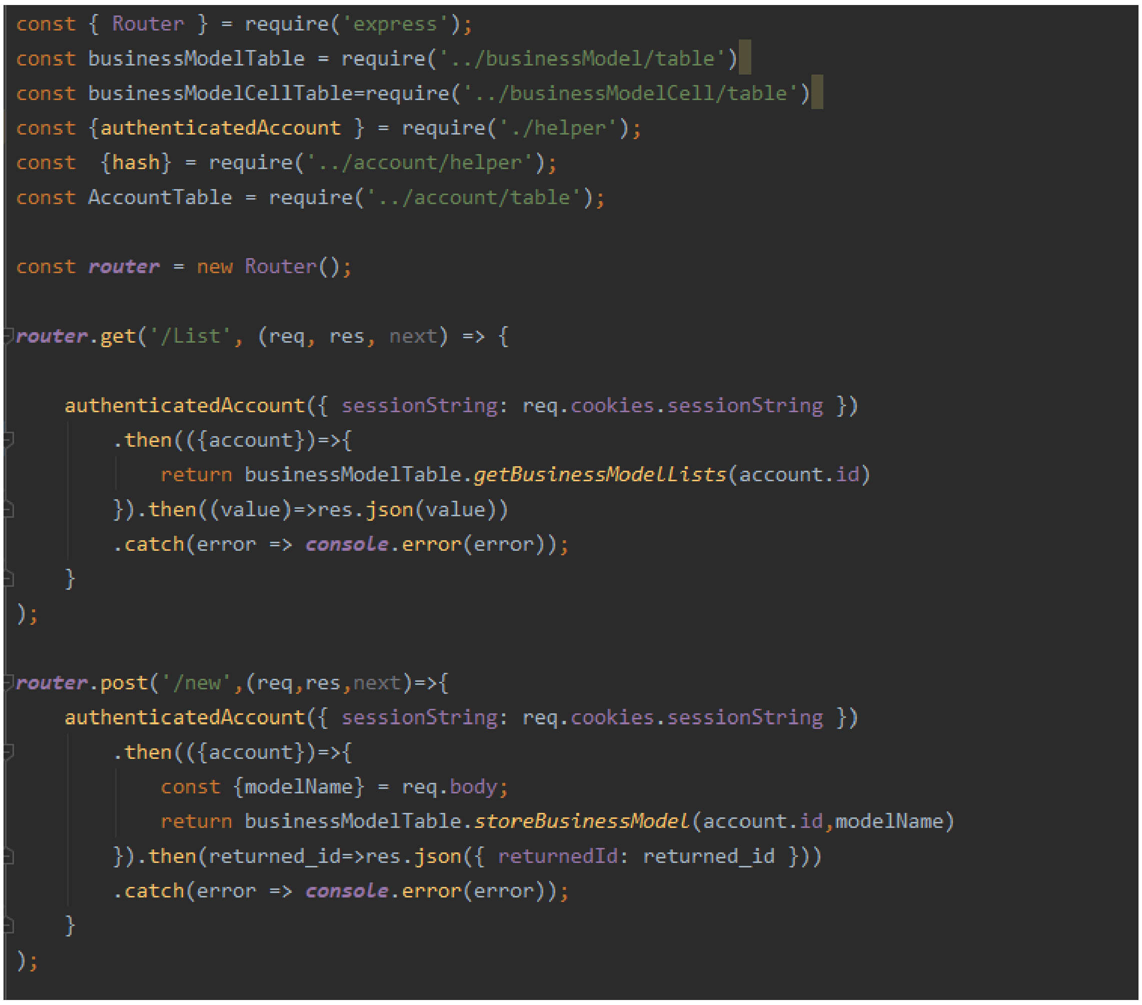
The RESTful pattern is using in this application the backend to map HTTP routes and CRUD, and the API design is done in a modularized way according to the feature they are related to. The design can be seen from Table 1.

Table 1 API design

HTTP Verb	Path	Purpose
POST	/account/ signup	Signup account
POST	/account/login	Login account
POST	/account/logout	Logout account
GET	/account/authenticated	Authenticate account
PUT	/account/update	Update password
GET	/canvas/list	Get canvas list
POST	/canvas/new	Create new canvas
POST	/canvas/saveModelCell	Save model cell data
GET	/canvas/getModelCell/:id	Get model cell by ID
DELETE	/canvas/deleteModel/:id	Delete model by ID
PUT	/canvas/updateModelCell	Update model cell data
GET	/canvas/model/:id	Get model data by ID
GET	/dashboard/countCanvas	Get counted canvas list
GET	/dashboard/countFeedback	Get counted feedback
PUT	/lesson/grade	Update the grade
GET	/lesson/gradeList	Get the grade list
GET	/lesson/strategyList	Get the strategy list
PUT	/lesson/strategy	Update the strategy
GET	/lesson/openQuestion	Get open question answer
PUT	/lesson/updateOpenQuestion	Update open question answer

POST	/lesson/interaction	Create new interaction
GET	/lesson/getInteraction	Get interaction list

The web requests have been split into separate files. This will be accomplished by using a special “Router” class offered by Express. It is needed to require the “Router” from Express module and create a local instance of the class in order to access it.



```
const { Router } = require('express');
const businessModelTable = require('../businessModel/table')
const businessModelCellTable=require('../businessModelCell/table')
const {authenticatedAccount } = require('./helper');
const  {hash} = require('../account/helper');
const AccountTable = require('../account/table');

const router = new Router();

router.get('/List', (req, res, next) => {

    authenticatedAccount({ sessionString: req.cookies.sessionString })
        .then(({account})=>{
            return businessModelTable.getBusinessModelLists(account.id)
        }).then((value)=>res.json(value))
        .catch(error => console.error(error));
    }
);

router.post('/new',(req,res,next)=>{
    authenticatedAccount({ sessionString: req.cookies.sessionString })
        .then(({account})=>{
            const {modelName} = req.body;
            return businessModelTable.storeBusinessModel(account.id,modelName)
        }).then(returned_id=>res.json({ returnedId: returned_id }))
        .catch(error => console.error(error));
    }
);
```

Figure 23 Part of business model API

As one example shown in Figure 23, for “/list” endpoint, the get request is getting the model list data, and the callback has “req”, “res” parameters. The parameter “res” stands for response, which will send back some json data. The parameter “req” stands for request.

Next, the exported router module will be required in the root index.js file, and the main application will use the set of routes from the “canvasRouter”.

5 DEPLOYMENT

In this chapter, a brief explanation about the process of how to deploy the application to Heroku will be discussed.

5.1 Set up Heroku

The Heroku Command Line Interface, CLI for short, is used to manage Heroku apps from the terminal directly (Heroku Dev Center. 2020b). It can be used to login Heroku as well as create a Heroku application. After the Heroku application is created, it is needed to add a remote to a local Git repository which has been initialized so that the application can be pushed from local to Heroku server.

5.2 Set up Postgres add-on

Heroku offers two ways to add-on a Postgres database with the deployed backend. The first way is to use Heroku CLI. The command is:

```
“heroku addons:create heroku-postgresql:hobby-dev"
```

This command tells Heroku to create a Postgres database with the application and use the hobby-div plan, which is a free plan that offers limited database storage.

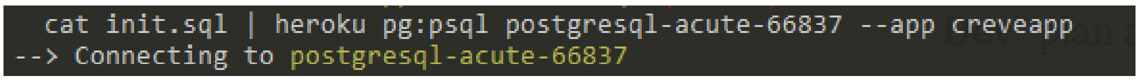
The second way of adding a Postgres database addon is to use the add-ons option from Heroku website and choose the database manually.

5.3 Create database tables and initial data

The automatically generated name (e.g. Postgresql-acute-66837) is returned after the Postgres is added to the application successfully. By using the generated name, it will be able to login and connect to PostgreSQL with Heroku command CLI:

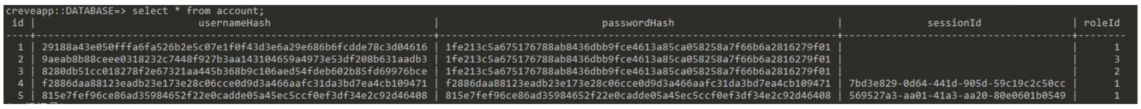
“heroku pg:psql postgresql-acute-66837 --app creveapp”

Heroku allows the user to create a table and entries based on the provided file. It means that Postgres will get the contents from the file user provided and use the psql console to run the initial command inside the file to create tables.



```
  cat init.sql | heroku pg:psql postgresql-acute-66837 --app creveapp
--> Connecting to postgresql-acute-66837
```

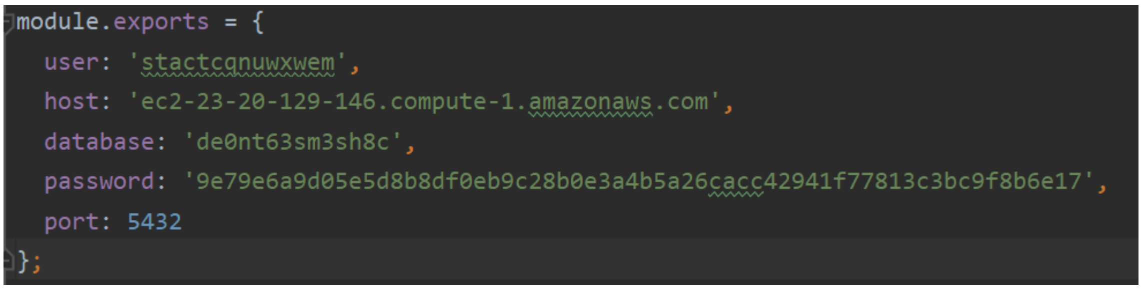
Figure 24. Create table with initial SQL data



```
creveapp::DATABASE=> select * from account;
 id |                           usernameHash                           |                           passwordHash                           |              sessionId               | roleId
----+------------------------------------------------------------------+------------------------------------------------------------------+--------------------------------------+--------
  1 | 29188a43e050fffa6fa526b2e5c07e1f0f43d3e6a29e686b6fcdde78c3d04616 | 1fe213c5a675176788ab8436dbb9fce4613a85ca058258a7f66b6a2816279f01 |                                      |      1
  2 | 9aeab8b88ceee0318232c7448f927b3aa143104659a4973e53df208b631aadb3 | 1fe213c5a675176788ab8436dbb9fce4613a85ca058258a7f66b6a2816279f01 |                                      |      3
  3 | 8280db51cc018278f2e67321aa445b368b9c106aed54fdeb602b85fd69976bce | 1fe213c5a675176788ab8436dbb9fce4613a85ca058258a7f66b6a2816279f01 |                                      |      2
  4 | f2886daa88123eadb23e173e28c06cce0d9d3a466aafc31da3bd7ea4cb109471 | f2886daa88123eadb23e173e28c06cce0d9d3a466aafc31da3bd7ea4cb109471 | 7bd3e829-0d64-441d-905d-59c19c2c50cc |      1
  5 | 815e7fef96ce86ad35984652f22e0cadde05a45ec5ccf0ef3df34e2c92d46408 | 815e7fef96ce86ad35984652f22e0cadde05a45ec5ccf0ef3df34e2c92d46408 | 569527a3-aa01-41a3-aa20-80e0601b0549 |      1
```

Figure 25 Account data sample saved in Heroku Postgres

The process of frontend deployment with Heroku is similar to backend deployment, which can be created by using the same method. As a result, the Node.js Express application backend with a PostgreSQL database can be deployed on Heroku properly.



```
module.exports = {
   user: 'stactcqnuwxwem',
   host: 'ec2-23-20-129-146.compute-1.amazonaws.com',
   database: 'de0nt63sm3sh8c',
   password: '9e79e6a9d05e5d8b8df0eb9c28b0e3a4b5a26cacc42941f77813c3bc9f8b6e17',
   port: 5432
};
```

Figure 26 Database pool setting based on Heroku Postgres credentials

In addition, the database connection pool in the application backend need to change the setting based on the generated data from Heroku Postgres.

6 TESTING

The process of testing will be discussed in this chapter.

6.1 Browser-based testing

The browser-based testing has been made by concentrating more on functionality of the user interface and responsiveness of performance.

By using the user role account, the following functions will be tested one by one, which including signup account, login account, logout account, get into dashboard page, get into canvas model page to edit model, download a canvas as PDF/PNG, get into lessons list, and get into setting.

By using the mentor role account, the following functions will be tested one by one, which including login account, logout account, search for user information, check the selected canvas, rate the canvas, and give feedback.

By using the admin role account, the following functions will be tested one by one, which including login account, logout account, create account based on different role of this application.

All the functions are expected to run fluently. The operated results match expected results.

6.2 Jest automated testing

The front-end application of this thesis is bootstrapped with create-react-app. It is an advantage that the create-react-app comes with a pre-configured testing environment and Jest is already installed and used as its test runner in the application.

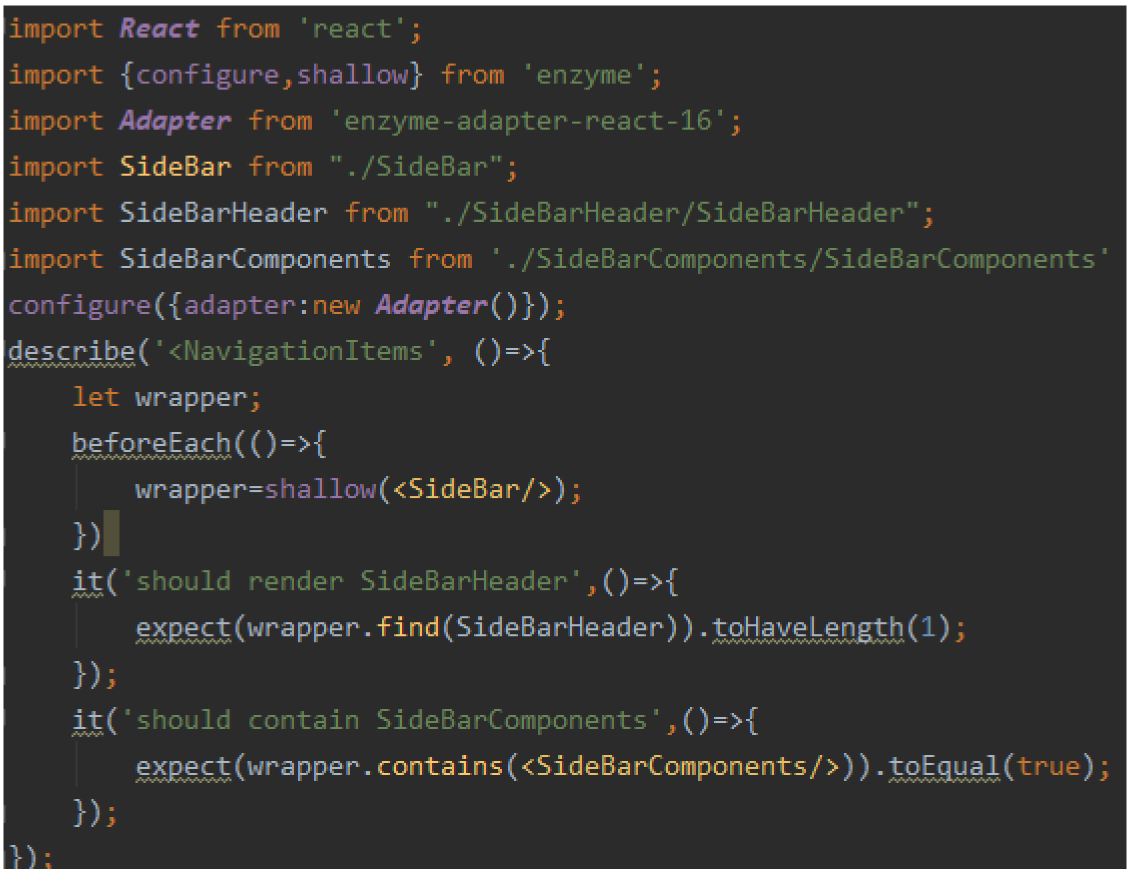
Jest, which is a popular JavaScript testing tool, is regarded as a node-base runner (Jestjs 2020), meaning the tests are operated in a Node environment instead of running in browser. The test runner is responsible for executing the test and providing a validation library to do comparisons and throw potential error. Beside Jest, another needed tool is called Enzyme, which allows the unit tests to be done without render the complete application.

Testing with Jest includes created components, created reducers and isolated unites, and the steps of testing will be explained in the following sub-chapters.

6.2.1 Test component

There are a few methods to define a test. The “describe” is a function which takes two arguments and it is used to group similar tests. The function “it” offers the possibility to write one individual test, which also takes two arguments. The first argument is typically a description string which will appear in the console. The actual testing logic is written in the arrow function as the second argument.

Enzyme offers a method named “shallow”, which helps to render the component with all its content, but the content is not deeply rendered. It means that some of the components inside a testing component are rendered as placeholders, and the contents of them are not rendered. “Shallow” helps in creating an isolated test and it does not render a whole sub tree of a specific component. One example is shown in the Figure 27.



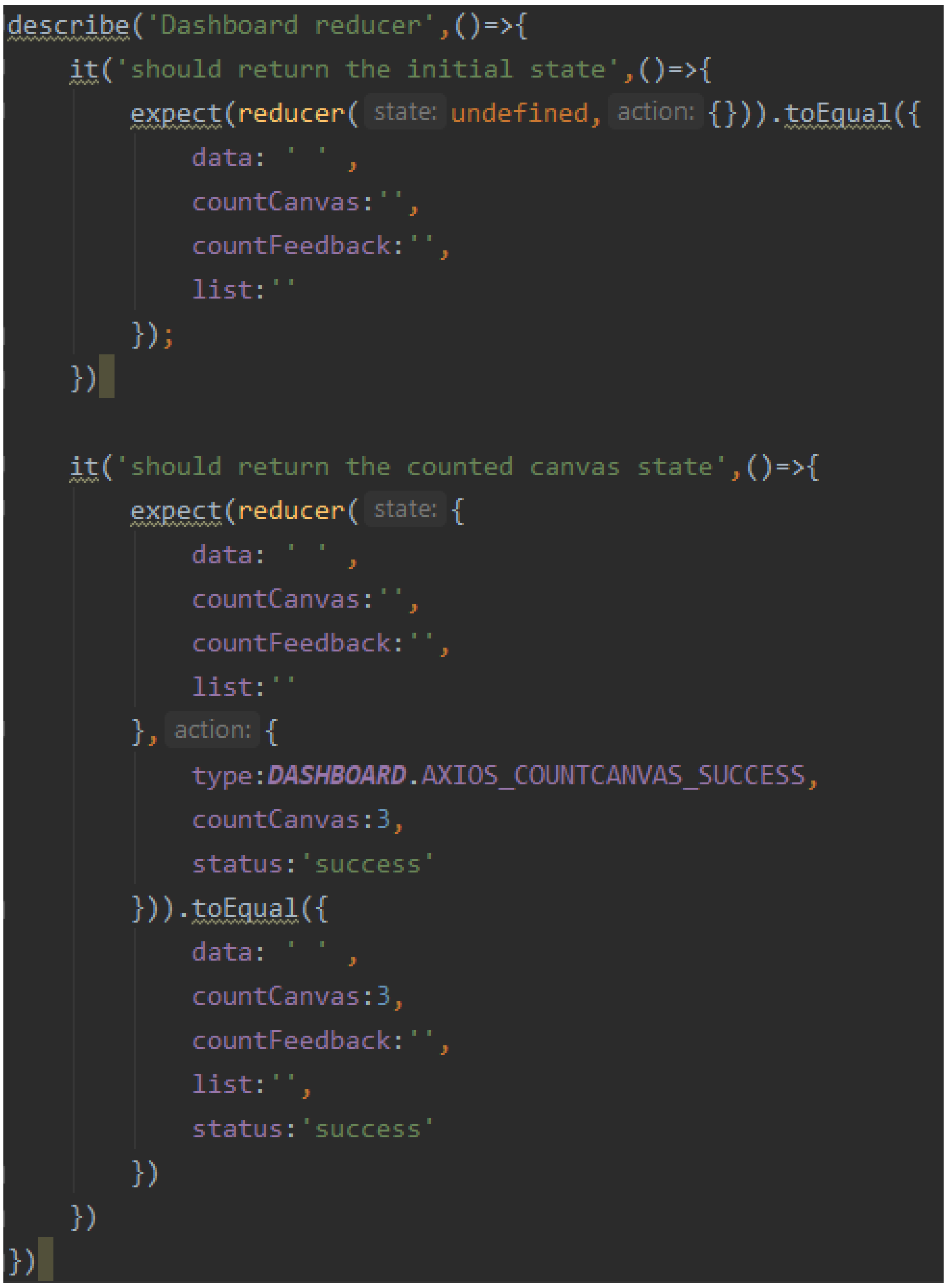
```
import React from 'react';
import {configure,shallow} from 'enzyme';
import Adapter from 'enzyme-adapter-react-16';
import SideBar from "./SideBar";
import SideBarHeader from "./SideBarHeader/SideBarHeader";
import SideBarComponents from './SideBarComponents/SideBarComponents'
configure({adapter:new Adapter()});
describe('<NavigationItems', ()=>{
    let wrapper;
    beforeEach(()=>{
        wrapper=shallow(<SideBar/>);
    })
    it('should render SideBarHeader',()=>{
        expect(wrapper.find(SideBarHeader)).toHaveLength(1);
    });
    it('should contain SideBarComponents',()=>{
        expect(wrapper.contains(<SideBarComponents/>)).toEqual(true);
    });
});
```

Figure 27 Test component with Jest

As shown in Figure 27, the "SideBar" component is passed to the "shallow" method, which renders the "SideBar" component and stores the result in the "wrapper" variable. The "expect" method is used to make an assertion. The "wrapper" variable inside "expect" method are able to use some functions provided by Enzyme, such as "find" function and "contains" function. Then, it follows with an expectation, for example "toHaveLength" function, and "toEqual" function, and these functions are offered by Jest.

6.2.2 Test reducer

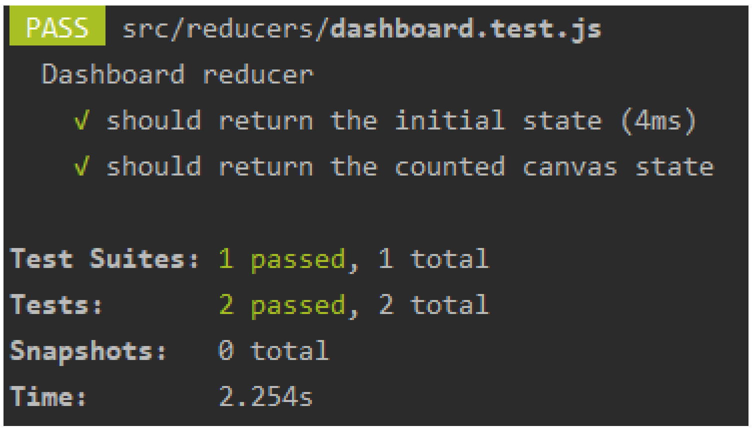
Reducer is pure synchronous function and it does not need to be handled with asynchronous related things. The reducer gets an initial state and action into it and returns the updated state. To make a test for the reducer, it is needed to import the reducer and action types, and then write the test. For example, when an empty action object is added into reducer, it will receive an invalid action type. In this case, the expected state should equal to the initial state. Take another example, when the reducer is executed with the initial state, the valid type, and payload of an action, the expected result is the equalization between the updated state and the expected state. The examples and relevant results are shown in Figure 28 and Figure 29.



```
describe('Dashboard reducer',()=>{
    it('should return the initial state',()=>{
        expect(reducer( state: undefined, action: {})).toEqual({
            data: ' ' ,
            countCanvas:'',
            countFeedback:'',
            list:''
        });
    })

    it('should return the counted canvas state',()=>{
        expect(reducer( state: {
            data: ' ' ,
            countCanvas:'',
            countFeedback:'',
            list:''
        }, action: {
            type:DASHBOARD.AXIOS_COUNTCANVAS_SUCCESS,
            countCanvas:3,
            status:'success'
        })).toEqual({
            data: ' ' ,
            countCanvas:3,
            countFeedback:'',
            list:'',
            status:'success'
        })
    })
})
```

Figure 28 Test reducer with Jest



```
 PASS  src/reducers/dashboard.test.js
  Dashboard reducer
    √ should return the initial state (4ms)
    √ should return the counted canvas state

Test Suites: 1 passed, 1 total
Tests:       2 passed, 2 total
Snapshots:   0 total
Time:        2.254s
```

Figure 29 Result in console

Figure 29 shows the result of test example in the console. The “1 passed” Test Suits refer to the “describe” function and “2 passed” Tests refers to the “it” functions. As a result, the test passed as it is expected.

7 CONCLUSION

In this thesis, a Creve application is built, in which entrepreneurs (users) can fill in the information, and then generate their own business model Canvases. This thesis aims at creating real value for small businesses in Finland to generate their business models in order to make possible improvements. The above chapters have explained the design, implementation, process of the web application establishment.

The general process of the thesis can be summarized into three steps. Firstly, the topic was decided in October 2019 and then confirmed as building a web application as the topic. Secondly, after a short planning of the general technologies that should be used, the general web application was built, improvements were made, and the required functions were realized. Thirdly, the thesis was written, some small improvements for the application were made, and testing went fluently.

By using the application, entrepreneurs (users) can review their business based on four aspects, which include solutions, customers, resources, and money. By using the model, entrepreneurs (users) can check the aspects that they do not have and think about the connection and transformation among the four business aspects. Lessons and an interaction platform are created for entrepreneurs (users) to gather their ideas to complete answering the model. The model page is able to download as PDF or PNG format, and mentors can check and give comments for entrepreneurs (users) to make further improvements.

The professional knowledge and skills that are used in this thesis include the following aspects: Firstly, programming language skills such as JavaScript. In the thesis, React Library was used in front-end application, and NodeJS as well as Express were used in back-end application. Secondly, database and data management skills are practiced. Thirdly, software engineering methods are used. In addition, I also practiced my communication skills, project management skills, and I gained some knowledge about combining the technology development with real business needs, such as using 7 Ps marketing mix strategy to design lessons for entrepreneurs.

For the website building, the general ideas originated from Muova, and I have designed the layout, I made decisions about how to realize the functions by myself. Furthermore, I practiced my business knowledge to some extent in order to make the lessons more useful and meaningful. For the thesis writing, the introduction about Muova and its project, as well as the applied theories are explained by using academic articles and official websites. Therefore, it can be said that the thesis is reliable.

The thesis has some limitations as well, which can be summarized into three points: (1) Time is limited: The duration of the project is 1.5.2018 - 31.12.2020, and the thesis was decided to be completed half a year before the project end. Luckily, all the required functions of the web application were realized; (2) Communication is limited: The communication is mostly conducted through email; and (3) Resources are limited: the thesis aims at offering a web application for small businesses, but there were not any opportunities to get in touch with real businesses and collect ideas from them to further develop the web application.

Therefore, in the near future, there are a few points that Muova can move forward based on this thesis. For example, Muova can edit the web application, change, or add more functions based on the project needs. It would also be interesting to investigate more about the small businesses to improve the functions of the web application.

As a conclusion, all the required functions of the application are realized, the application can run fluently, and the objectives of the thesis are achieved. It can be said that this application matches the concept of Creve project. Thanks for Muova for offering me the opportunity.

REFERENCES

Baskerville, P. 2015. What's the Difference between a Startup and an SME (Small Medium Enterprise)? Accessed 8.5.2020. https://www.quora.com/Whats-the-difference-between-a-startup-and-an-SME-small-medium-enterprise

Business Finland 2020a. Business Environment. Accessed 8.5.2020. https://www.businessfinland.fi/en/do-business-with-finland/finnish-business-environment/why-finland/

Business Finland 2020b. Take a Look at Finnish Startup Companies. Accessed 8.5.2020. https://www.businessfinland.fi/en/do-business-with-finland/finnish-business-environment/startup-environment/

Chai, L. G. 2009. A Review of Marketing Mix: 4Ps or More? Accessed 8.5.2020. http://student.bms.lk/CBM/Slides/MarJourArticle/Main%20criticsm%20of%20marketing%20mix.pdf

CreateReactApp. 2020. Adding a CSS Modules Stylesheet. Accessed 8.5.2020. https://create-react-app.dev/docs/adding-a-css-modules-stylesheet/

Creve. 2020. Creve Info. Accessed 8.5.2020. https://www.creve.fi/hankkeet/creve-2-0/

Creve 2.0. 2020. Project Base and Scope. Accessed 8.5.2020. https://www.creve.fi/wp-content/uploads/sites/24/2018/10/Creve2.0_Esittely-2018.pdf

CSSTricks. 2017. What are CSS Modules and why do we need them? Accessed 8.5.2020. https://css-tricks.com/css-modules-part-1-need/

Dabit, N. 2016. Beginner’s Guide to React Router. Accessed 8.5.2020. https://medium.com/free-code-camp/beginner-s-guide-to-react-router-53094349669

Dev. 2020. Redux Data Flow and React Component Life Cycle. Accessed 8.5.2020. https://dev.to/oahehc/redux-data-flow-and-react-component-life-cycle-11n

Douglas, K. (2003). PostgreSQL. Accessed 8.5.2020. https://learning.oreilly.com/library/view/postgresql/0735712573/ch01.html

Evergreen. 2020. 5 Key Benefits of Web Applications for Business. Accessed 8.5.2020. https://evergreencomputing.com/blog/5-key-benefits-of-web-applications-for-business/

Developers. 2020. Why The Hell Would I Use Node.js? A Case-by-Case Tutorial. Accessed 8.5.2020. https://www.toptal.com/nodejs/why-the-hell-would-i-use-node-js

Facebook for Developers. 2017. Tagging Single Page Applications with the Facebook Pixel. Accessed 8.5.2020. https://developers.facebook.com/ads/blog/post/v2/2017/05/29/tagging-a-single-page-application-facebook-pixel/

Flanagan, D. 2006. JavaScript: The definitive guide. Accessed 8.5.2020. https://learning.oreilly.com/library/view/javascript-the-definitive/0596101996/ch01.html. ISBN: 9780596101992.

Fortnum, D. 2012. THAT'LL NEVER WORK: SIX IMPORTANT LESSONS FROM SUCCESSFUL ENTREPRENEURS. Ivey Business Journal (Online), p. N_A. Accessed 8.5.2020. https://search-proquest-com.proxy.uwasa.fi/docview/1038945025/fulltext/5D1BAC16A6014253PQ/1?accountid=14797

FreeCodeCamp. 2016. Beginner’s Guide to React Router. Accessed 8.5.2020. https://www.freecodecamp.org/news/beginner-s-guide-to-react-router-53094349669/

Heroku. 2020. Heroku Add-ons. Accessed 8.5.2020. https://elements.heroku.com/addons

Heroku Dev Center. 2020a. Getting Started on Heroku with Node.js. Accessed 8.5.2020. https://devcenter.heroku.com/articles/getting-started-with-nodejs?singlepage=true#introduction

Heroku Dev Center. 2020b. The Heroku CLI. Accessed 8.5.2020. https://devcenter.heroku.com/articles/heroku-cli

Jestjs. 2020. Testing React Apps. Accessed 8.5.2020. https://jestjs.io/docs/en/tutorial-react

Ivanovs, A. 2019. Know the Amazing Advantages of React Library. Accessed 8.5.2020. https://codecondo.com/know-the-amazing-advantages-of-react-library/

Lindzon, J. 2019. The Importance of a Business Plan. Accessed 8.5.2020. https://www.waveapps.com/blog/entrepreneurship/importance-of-a-business-plan

MDN Web Docs. 2020. Express/Node introduction. Accessed 8.5.2020. https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/Introduction

Microsoft. 2019. Choose Between Traditional Web Apps and Single Page Apps (SPAs). Accessed 8.5.2020. https://docs.microsoft.com/en-us/dotnet/architecture/modern-web-apps-azure/choose-between-traditional-web-and-single-page-apps

Muova. 2020. Company Website. Accessed 8.5.2020. www.muova.fi/en/

Nodejs. 2011. What is npm? Accessed 8.5.2020. https://nodejs.org/en/knowledge/getting-started/npm/what-is-npm/

Npmjs. 2020. Axios. Accessed 8.5.2020. https://www.npmjs.com/package/axios

PostgreSQL. 2020. What is PostgreSQL? Why use PostgreSQL? Accessed 8.5.2020. https://www.postgresql.org/about/

React. 2020. React. A JavaScript library for building user interfaces. Accessed 8.5.2020. https://reactjs.org/

Redux. 2020. Getting Started with Redux. Accessed 8.5.2020. https://re-dux.js.org/introduction/getting-started

Skenix. 2019. Advantages & Disadvantages of Single Page Application. Accessed 8.5.2020. https://www.skenix.com/advantages-disadvantages-of-single-page-ap-plication/

Time. 2020. Resolving CSS Conflicts. Accessed 8.5.2020. https://time.ly/re-sources/troubleshooting/resolving-css-conflicts/

W3schools. 2020. ECMAScript 6 - ECMAScript 2015. Accessed 8.5.2020. https://www.w3schools.com/js/js_es6.asp

Wikipedia 2020a. Muova. Accessed 8.5.2020. https://fi.wikipedia.org/wiki/Muova

Wikipedia. 2020b. Single-page application. Accessed 8.5.2020. https://en.wikipe-dia.org/wiki/Single-page_application

Yrittäjät 2020. Entrepreneurship in Finland. Accessed 8.5.2020. https://www.yrit-tajat.fi/about-federation-finnish-enterprises/small-and-medium-sized-enterprises-526261

Zwilling, M. 2015. Why You Must Really Know Yourself Before Starting a Busi-ness. Accessed 8.5.2020. https://www.entrepreneur.com/article/248796