

Kumar Simkhada

Transitioning Angular 2 User Interface (UI) into React

Helsinki Metropolia University of Applied Sciences

Bachelor of Engineering

Information Technology

Thesis

April 12, 2017

Author Title	Kumar Simkhada Transitioning Angular 2 User Interface (UI) into React
Number of Pages Date	40 pages + 2 appendices April 12, 2017
Degree	Bachelor of Engineering
Degree Programme	Information Technology
Specialisation option	Software Engineering
Instructor(s)	Markku Karhu, Principal Lecturer
<p>The fundamental goal of this thesis was to appraise the comparison and superiority between the two most favoured front-end JavaScript framework and libraries: Angular 2 and React respectively. Moreover, this thesis is supplemented with the prototype user interface application in Angular 2. The UI thus developed was completely transitioned into React to find out the major differences between them.</p> <p>The study was winded up within three major steps. The initial step in the project was done by collecting data from the developers and finding out the statistics available on the web. Secondly, the features and tools applied in the two technologies (Angular 2 and React) were analysed. Lastly, an identical User Interface application was developed in both technologies in-order to support the study.</p> <p>As a result, React was found as a powerful tool for the development of front-end user interface. Being only responsible for the View component of the application, React was found more suitable to create an interactive user interface.</p> <p>The study was concluded highlighting the six major key points: Performance, Learnability, Templating, Structuring data model, Code reshuffling and Package management. In the study, it was found that React displayed dominance in major aspects. React showed superiority in terms of performance, Learnability, package management. While a tie was found when considering the code reshuffling and structuring data model. However, Angular 2 offered an edge in Templating.</p>	
Keywords	AngularJs, Angular 2, React, HTML, JavaScript, Front-end, Open-source, Bootstrap, JQuery, Virtual DOM, npm, Redux, Packages, Libraries, Frameworks, Flux, Components, Performance.

Contents

1	Introduction	1
2	Background	2
2.1	User Interface	2
2.2	Common Problems in UI Development	2
2.3	Latest Trends in Developing the Web Application UI	3
2.4	Front-end Web Technologies	3
3	Research Benchmarks	5
4	JavaScript UI Frameworks and Libraries	7
4.1	Web Statistics and Surveys	9
4.2	Angular 2	12
4.2.1	Typescript on Angular 2	13
4.2.2	Directives and Components	14
4.2.3	Modules	15
4.2.4	Services	16
4.2.5	Templates and Forms	16
4.2.6	Dependency Injection	17
4.2.7	Data Binding	18
4.3	React	19
4.3.1	JSX	20
4.3.2	React Components, Props and State	21
4.3.3	Virtual DOM	22
4.3.4	Forms	23
4.3.5	Flux Architecture	23
4.3.6	Data Binding in React	25
5	Angular 2.0 User Interface	25
5.1	Specifications	27
5.2	Environment Set-up	27
5.3	Implementation	30
6	UI Transition in React	31
6.1	Environment Set-up	31

6.2	Implementation	32
7	Discussion	33
8	Conclusion	37
	References	38
	Appendices	
	Appendix 1. Survey Questionnaires for Front-end Technologies	
	Appendix 2. Screen Shots of the UIs	

Abbreviations

API	Application Programming Interface
CLI	Command Line Interface
DOM	Document Object Model
ECMA	European Computer Manufactures Association
ES5/6/7	ECMA Script 5/6/7
HTML	Hypertext Mark-up Language
IE	Internet Explorer
IoT	Internet of Things
IT	Information Technology
JS	JavaScript
JSX	JavaScript and XML
MVC	Model View Controller
MVVM	Model View-View Model
NPM	Node Package Manager
SPA	Single Page Application
UI	User Interface
W3C	World Wide Web Consortium
XML	Extensible Markup Language

List of Figures

Figure 1: Front end Spectrum [2].....	4
Figure 2: Most loved technologies of 2017 in Web development [5].....	8
Figure 3: The Survey Results from the Discussion Groups.....	9
Figure 4: Relationship of Typescript with ES5, ES6, and ES7 [11].....	11
Figure 5: Components in Dependency Injection [13].....	15
Figure 6: Virtual DOM Workflow Visual Representations [18].....	20
Figure 7: Data Flow in Flux Architecture [20].....	22
Figure 8: Screenshot of Vacancies Section in Tuubi [21].....	26
Figure 9: Use Case Diagram for Job Portal.....	27

List of Tables

Table 1: Head to head Comparisons between Angular 2 and React.....	7
Table 2: Further Comparison between Angular 2 and React.....	36

1 Introduction

There has been a diverse modification and introduction of new web technologies since last decade. Thereafter, web has been the area of intense focus and research of today because of its unimaginable future. The impact of open-sources and the versatility of the web are accessible and outspread radically into every sector regardless of the hardware or software platforms used by engineers, developers and designers.

The method of developing web applications using HTML (Hyper Text Markup Language), CSS (Cascading Style Sheet) and core JavaScript (Scripting Language) in the presentation layer of a web application is considered as traditional. Later, it was realised by the developers that dynamic and more interactivity in the web is necessary. As a result, the data-access layer of web was a focus point in web application development. Now, the design and development trend has been changing every year, and emphasis has been given to performance, security, cost and efficiency in order to make a better user experience. As a matter of fact, there has been a raise of different JavaScript frameworks and libraries to maintain these aspects. However, a rapid change in the development platform has brought a paradox in selecting the right technologies for developers.

This thesis outlines the comparison of two most popular framework and libraries of this time: Angular 2 developed by Google and React developed by the Facebook. Different methodologies were applied to find out the edge in user interface development between these two technologies. A simple prototype User Interface application was developed in Angular 2 to supplement the thesis and the whole UI was practically implemented into React.

2 Background to Web Technologies

2.1 User Interface

The user interface is the design ground between the machine and the human. This is a specific field of human-computer interaction that permits the functional control and operations of computer operating systems and software, machineries, and other tools. The user interface has a significant role in the web development as it is focused totally on maximising the user experience. The introduction of various web technologies has made the role of UI developer more demanding and challenging.

2.2 Common Problems in UI Development

Content Creation

The main problem seen in UI development is the creation of contents. The content in the UI refers to the visual components, design, or patterns which can be seen at the very beginning of the application. Because of the variable choices of the users, it is hard to make common UI content that would be accepted and liked by every user.

User Input Handling

A normal UI allows users to interact with certain sets of fields where users can play with interactive data. Few years ago, it was not so difficult to manipulate the data with a simple call back, but these days due to the introduction of business logic the user input handling stands as a typical problem in user interface design.

Dialogue Control

Dialogue control creates system-user interaction structures which are followed by series of complex steps. Incoming and outgoing of the data happens when the user's data are processed into the business logic of the application. The user's data passed into the UI interacts with the objects of the business logic layers, which create an issue in UI development.

Merging UI with Business Logic

The development tools designed these days force the UI to be coupled with the business logic. Fast data manipulation has become a matter of concern to increase user experience. This process of coupling the data into the UI is always a focus in UI development.

2.3 Latest Trends in developing the web application UI

The last few decades can be taken as decade with tremendous change in the web development technologies. With the raise of the SPA frameworks and libraries, single-page web applications development has overtaken multiple page web applications. This outlines that traditional UI design and development have literally been ended. Similarly, the use of chat bolt for conversational Web applications, minimalist design for layouts, use of lesser photos for authenticity and use of gif animations in the layouts are the basics trends in the user interface design since last few years. Besides, raise of JavaScript (basically the introduction of Angular 2 and more exploration of React), Data visualization, and the IoT (Internet of Things) on the web are other major key trends. [1]

2.4 Front-end web technologies

In modern web engineering, there are two typical layers present in the web software architecture, namely, the presentation layer which is known as front-end and the data access layer popularly known as back-end. The front-end part involves in every single thing concerned with the user or the clients in a web application. This includes all the things associated with the user interfaces. The front-end of web development is basically managed with the use of standard technologies such as HTML, CSS and JavaScript.

HTML is one of the giant tools and considered as the backbone of the web technologies that work as an interpreter of normal plain text, images, and other audio, visual components into the web browsers. HTML documents are arranged and written with combinations of different tags in the form of different Angular brackets. The fifth major version of HTML was introduced in 2013 as HTML5 and this brought a revolutionary change to web development. HTML provides the fundamental base for all technologies.

Similarly, CSS is another major web technology which is responsible for styling a HTML document. All the visual effects, layout, colours, fonts etc. are handed by CSS. Among all front-end the technologies, CSS has a primary role in user interface design. HTML and CSS are validated with W3C a standardised Markup validator service. Moreover, HTML and CSS are standing as the core technologies required for modern web development. Figure 1 illustrates the spectrum of technologies in front-end web development.

THE FRONT-END SPECTRUM



Figure 1 Front-end Spectrum. [2]

Likewise, another multipurpose and arguably the most important web technology for fully functional and dynamic web development is JavaScript. JavaScript is a high-level, integrated programming language which is standardised with the ECMAScript language specification. JavaScript is often treated as a functional programming language that floats in the entire modern browser. It has brought a revolution to World Wide Web because of its compatibility and adaptability in different layers of the web. With the introduction of different open-source libraries and frameworks in the last few decade, JavaScript has changed the way how developers are involved in building the web software. However, JavaScript is moving and evolving so fast that the developers have to

learn new changes in the libraries and frameworks. This has resulted as a daunting task for every developer to decide which frameworks or libraries to follow and move forward.

3 Research Benchmark

Every new technology offers certain labels for a particular set of users, doing something, in certain conditions. There are a huge number of tools offered by the frameworks and libraries to design the user interface component. The difficulty in choosing the right toolkit will always be at the peak when a new technology arrives to the market. This is always a hard task for the novice developers to make the right choice in order to get rid of development complexity.

While addressing the web frameworks and libraries in user interface development, there is always a need for a mandatory research guide for a specific topic. But, there is always a long inventory of features marketed instead. It is very hard to include huge options available to roll into very few benchmarks. However, this study is bounded with certain sets of limitations for user interface components between different technologies.

Application Architecture

Developing a UI application is a very complex task when it is associated with manipulating, processing and rendering large business data. Developers might get lost anytime into the visual components of the applications containing a large number of user interfaces associated with the business logic. The manipulation of large business data brings chaos to the code organization in the development process. The basic requirement for good code organization focuses on reusable, clear and maintainable codes.

The design of a good user interface application starts with good application architecture. User interface application designs are always associated with specific sets of design problems and patterns. When talking about client-side web development, there are different architectural patterns offered for a good user interface. Design patterns like MVC (Model-View-Controller), MVVM (Model-View-View-Model), and MVP (Model-View-presenter) provides good supporting architecture for modern web UI development.

A very important aspect to note about the UI architecture tool is that it changes very fast. The technologies that are on hype today will no longer be in exist a year after. So, whatever the chaos the new technology brings, it will always be a good idea to initiate work with good application architecture.

Features

A new technology comes with a wide range of features to make the UI more dynamic and versatile. Basically, in web engineering, the technologies offer features such as browser and device compatibility, speed and performance, productivity and testing of the applications. These features offered by the technologies bound the UI development process into certain limits to make the UI more interactive and user friendly.

Data Integrity

Data integrity in user interface relates to the equitable separation of works in the development process. This will fortify adequate control over how research works can be carried out within the limitations life cycle. Also, this will finally result in the stability, validation and accuracy of the development and its result.

Usability Trap

When it comes to the analysis of user interface and its development platforms, there are various measures that need to be fortified. The measures of time estimation to complete a certain level of task, time required for debugging and error handling, are bounded with certain usability protocols. Similarly, the standard evaluation techniques such as heuristic evaluation, first-click testing and system usability scaling provide a good track for the research and development of the user interface.

Flexibility

The flexibility in the UI tools can be taken when the tools or the development platforms allow an excessive change in design which later can be evaluated by users. A certain development platform does not allow the applications to run in run time. Some are not compatible with the browsers, some need to compile before running, some takes an

excessive time to fix the bugs. This incompatibility of the development platforms consumes more time and reduces efficiency.

Development Viscosity

While developing a UI, the tools used in the development process should always decrease development viscosity. In order to increase user experience, the functioning of the tools needs to be faster and efficient.

4 JavaScript UI Frameworks and Libraries

JavaScript frameworks are the application framework designed and written in core JavaScript to provide a generic functionality to a whole new application so that the codes and functionalities of the software can be reused. JavaScript UI frameworks provide support features to different libraries, tools and API. When working in advanced projects, JavaScript frameworks help in architecting codes that make them more readable and secure.

Many frameworks have been introduced over the years. Ember.js, Backbone.js, AngularJs, Angular 2 are the notable names that stand as primary JavaScript frameworks. These frameworks are created with different specifications but they all have common goals to ease the development process.

There are huge advantages of using the JavaScript frameworks for modern client-side web development. A few advantages have been listed below:

- Since, frameworks are the future of web development, use of frameworks in a application reduces time and cost that leads to the supply of quality products to clients.
- The frameworks provide perfect code architecture that helps to make the codes used in the application more understandable and readable.
- Codes written by using frameworks can be reusable i.e. same codes can be reused the application time and again. This helps in the development of complex application faster.

- The frameworks helps to reduce the gap between client-side and server-side web development by making a good link, perfect validation support and communication with servers.
- The features used such as two-way data binding, controllers, components and modules provide superior management support in client side web development.

Similarly, with the demands of JavaScript, there has been dynamic development of JavaScript libraries by the big enterprises such as Microsoft, Yahoo, and Facebook. JavaScript libraries are the piles of pre-written codes in JavaScript which aims to make web-centric application development easier. The main reason behind the development of JavaScript libraries is to render the huge data into the UI component of the web. React, JQuery, vue.js are the notable names of JavaScript libraries.

The JavaScript libraries are small in comparison to the frameworks. Regardless of being small, JavaScript Libraries provides same and even more powerful features than the frameworks. A few advantages of using JavaScript libraries over JavaScript frameworks have been listed below:

- JavaScript libraries can be deployed into any of the advanced JavaScript frameworks such as AngularJs, Angular 2 and Backbone.js that follows the MVC design pattern. JavaScript libraries provide the V i.e. View, in MVC design pattern.
- AJAX support is the most fundamental thing for a smooth user interface and JavaScript libraries fit into this.
- JavaScript libraries have great open-source communities that are constantly updating the changes.
- Comparing with the libraries of other functional programming languages, JavaScript libraries are huge and can handle a huge amount of data the UI components.
- JavaScript libraries are simple and easy to learn for the novice developers.
- None of the JavaScript frameworks provides the same level of performance as JavaScript Libraries.

Both the frameworks and libraries of JavaScript provide a wide range of features for the design of a robust user interface. However, it has been really been a difficult task to select the right technology. A recent trend shows that Angular 2 and React are the most favoured frameworks and libraries for web UI development. [5.] The use of both of these technologies adds superior performance, code clarity and code reusable features in the application.

This thesis incorporates a study and implementation of features of React and the recent version of AngularJs: Angular 2. Angular 2 was introduced in the year 2016 with the initial release of the beta version and many developers actually migrated from its previous version AngularJs. Even though React being was at its best in 2016, developers migrated from React to Angular 2.

4.1 Web Statistics and Survey

Angular and React are the frameworks and libraries from Google and Facebook respectively, but the by-products of both of these technologies are the result of open-source communities. The beauty of open-source communities is that it provides every developer with a chance to contribute and engage in any open-source project. Open-source communities of Angular and React are very large and there are huge piles of contributions from the developers in online communities like Github, Stackoverflow. Github repositories of Angular and React show the popularity of these two technologies over the year. Head-to-head comparison between Angular and React is shown below:

Table 1 Head-to-head comparison between Angular and React [3]

	Angular 2	React
Version	2.0	15.3.2
Stable Release Year	2015	2016
Authored by	Google	Facebook
Focused language	Typescript	JSX
Github Star	21.9k	62.4k
Github contributions	410	956

Note: Above data is abstracted as of 21st March, 2017.

Table 1 shows that React has more popularity and has strong open-source communities over Angular as of 21st March 2017. However, the stable release of Angular 2 was

later than that of React. Angular 2 has not been embraced heavily because it has introduced the whole new style of development which is not compatible with the previous versions. Developers have to learn the Angular 2 syntax and everything from scratch and this might be decisive after some time. But, looking at the data from table 1, Angular 2 will not meet the same level as that of React within the same duration of time.

Similarly, the Stackoverflow survey of 2017 which was resulted from the 64,000 developers shows that React is the most popular than any other web technology. Angular made the sixth position in the list.

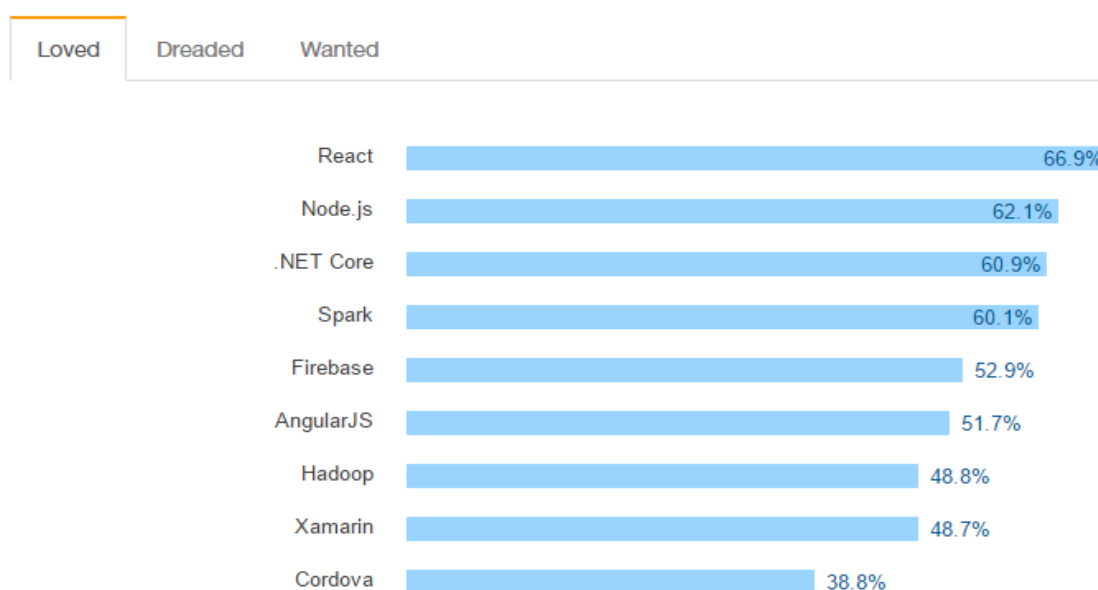


Figure 2 Most loved technologies of 2017 in web development [5]

Figure 2 shows that React is standing in the first position in the survey results of the most loved web technologies in 2017 with 66.9% out of 64,000 developers while Angular stands with 51.7% out of total. [5]

Similarly, a survey was carried out among the members of the front-end developer group in LinkedIn. Out of the total members from the group, 39 responded to the survey and the following results were found:

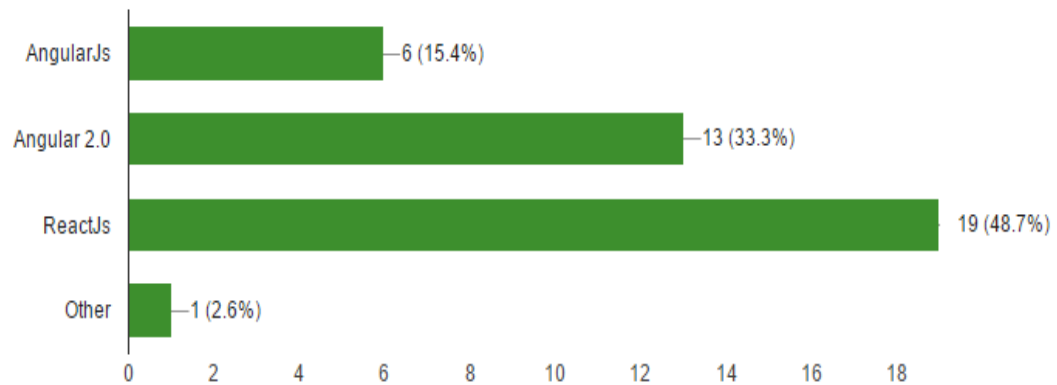


Figure 3 The Survey Results from Discussion Group.

Figure 3 suggests that, out of a total of 39 respondents, 48.7% of the developers prefer to work with the React while the respondents for its counterpart Angular 2 are 33.3%. The reason behind the majority of the respondents preferring React is because of its JavaScript centric features. React allows the developer to write everything in one file and there is no need for separate HTML, CSS and JavaScript. React combines HTML and JavaScript into the brand new features called JSX.

Many developers and companies have adopted these two technologies. The choice of the developer and companies differs by their experience and the backend technologies they follow. The user interface of popular web applications like Netflix, Imgur, Airbnb, Yahoo Mail etc. are written and maintained in React. [6.] Similarly, the page plugins and single page applications like YouTube player, Instagram filters, and YouTube trends are written in Angular 2. [7]

After comprehensive research between Angular 2 and React, it seems that both are trying to maintain their supremacy in the world of front-end web development. Being a fresh technology, Angular 2 is on an expanded hype at the moment. React has managed to stand as a leader starting from its release and it seems that React will be the developers choice for so many years to come from now. However, if communities between AngularJs and Angular 2 repositioned; then they will stand as a superior technology for a long period of time regardless of whatever technologies they face on their way.

4.2 Angular 2

AngularJs has become a fundamental framework of choice for every front-end developer for few years from 2009. The concept of AngularJs was introduced in 2009 by Misko Hevery and Adam Abrons but now it is officially been maintained by the giant enterprise Google. The initial version of AngularJs version 1.0 was released in the year 2012. As of August 2015, AngularJs released its newest version 1.4.4 and this is still considered as the current stable version. [8, 6-11]. The name Angular is introduced from the concept of the HTML tags i.e. HTML tags are Angular and Angular naming came up from the Angular brackets of HTML. [9.]. AngularJs was developed to overcome the vulnerability and issues in a web application such as bug fixing, high quality user interface and security.

After research for approximately two years, the same team from AngularJs released the final version of Angular 2 in late 2016. Angular 2 is focused on providing the best server-side rendering functionalities for perfect user interface. Angular 2 is basically known as the names, Angular or Angular 2. There are many issues overcome by Angular 2 over AngularJs. A few of them are highlighted below:

- Angular 2 provides much better performance over AngularJs. Angular 2 applications put low pressure on the virtual machines because of less memory usage. Superior template reusing functionalities, cache viewing features are among the notable things in Angular 2 that provide better performance.
- Angular 2 has advanced and powerful templating features. The development environment and IDEs are modern in Angular 2 and are better than the runtime of AngularJs. This helps the template writers to find out the errors fast and can save time in the template writing process.
- Angular 2 is more concerned for the future works in comparison with AngularJs, since web technologies are more dedicated on live data capturing with the web works such as web crawling. Angular 2 is focused to fulfil the needs for smooth operation of server side rendering to the user interface.

- Angular 2 has feature support for web platform mobile applications. The native mobile UI framework of the Angular 2 helps to make the fully functional native mobile UI in iOS and android platforms. [10].

4.2.1 Typescript on Angular 2

Typescript is an open-source programming language developed by Microsoft. It is scripting language written in core JavaScript. It is much similar to modern object oriented programming languages such as C#, C++ or Java. Typescript is used to develop the web based JavaScript application on client side or server side.

Although Angular 2 applications can be made with ES5, ES6 and dart, Typescript provides a wide range of features in creating a large Angular 2 application. The main reason behind using Typescript on Angular 2 applications is that it provides great tooling features. Navigating, code refactoring, auto completing, static typing and wealthy syntax generator are among the notable features of Typescript in developing Angular 2 applications. Figure 4 illustrates the relationship between Typescript with the ES5, ES6 and ES7.

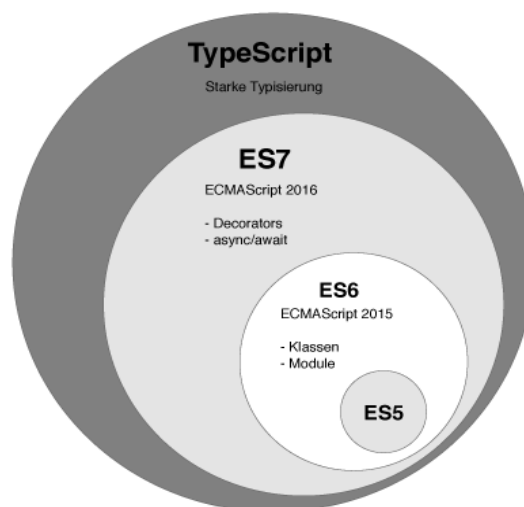


Figure 4 Relationship of Typescript with ES5, ES6, and ES7 [11.].

Starting an Angular 2 project basically starts with the installation of Typescript into a terminal. The Node Package Manager (npm) manages all the Typescript packages in the Angular 2 application. The latest stable version of Typescript in Angular 2 application can be installed with the following command:

```
$npm install -g typescript
```

And, in order to verify the latest version, it can be verified with the command:

```
$tsc -v
```

To start a Typescript program, a project folder should be created and project folders needs to have the files with the files extension .ts and the files with .ts will then be compiled with the file .js file in order to give the programme output.

Considering the research objective, beginning in section 3.2.2, the application architecture of the Angular 2 component required for Angular UI application has been outlined.

4.2.2 Directives and components

Basically, there are three types of directive in Angular 2: attributes, structural and component directives. Component directives handle the templating part to create the user interface, while structural directives change the document object models layout by adding and detaching the DOM object. Similarly, attributes play a role in changing the visual appearances of components and elements.

An example of a code attribute and its application into the Angular 2 directive is shown below:

```
<p myHighlight>Highlight me!</p>
```

Listing 1 Angular 2 code attribute

Here, myHighlight attribute is applied in the directive code shown below:

```
import { Directive, ElementRef, Input } from '@angular/core';
@Directive({ selector: '[myHighlight]' })
export class HighlightDirective {
  constructor(el: ElementRef) {
    el.nativeElement.style.backgroundColor = 'yellow';
  }
}
```

Listing 2 Angular 2 Directive application

The example directives shown in listings 1-2 is supplemented with the import statement which emphasises the Angular 2 core library that has three specific symbols: directive, ElementRef (feature that permits the security attack in DOM object) and Input. The directive provides functionalities to the @Directive decorator. Similarly, ElementRef acts as a constructor for code access in DOM, and input is responsible for data binding. [13]

A component plays a fundamental role in building the Angular 2 user interface because Angular 2 applications are made with the trees of the Angular 2 components. Angular 2 components are basically known as the subset of Angular 2 directives. In order to process the Angular 2 components and make them usable in the runtime environment, Angular 2 component decorator should be used to mark the classes as Angular 2 components.

```
@Component({selector: 'greet', template: 'Hello {{name}}!'})
class Greet {
    name: string = 'World';
}
```

Listing 3 Component in Angular 2

An example of component with a class called Greet is shown in listing 3 to outline the Angular 2 component. [12].

4.2.3 Modules

A route module in Angular 2 is a class that describes how different parts of the application are managed simultaneously in Angular 2. Every Angular 2 application needs to be equipped with at least one Angular 2 module that is called the route module. Naming of the Angular 2 route module can be anything but official Angular 2 documentation has suggested the name “AppModule”. Basically, route modules are used to create a simple application but many applications use the feature modules. Every Angular 2 module, whether it is route or feature, is a class with a decorator called @NgModule.

```

import { NgModule }      from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { AppComponent }  from './app.component';

@NgModule({
  imports:      [ BrowserModule ],
  declarations: [ AppComponent ],
  bootstrap:   [ AppComponent ]
})

export class AppModule { }

```

Listing 4 Angular 2 route module

Above code in listing 4 outlines the root module with its properties. Imports, providers, declarations, exports and bootstrap are the important properties of the root module. [13].

4.2.4 Services

A service in an Angular 2 application is also a class that does not have anything do with itself but an Angular 2 component takes several things from services. Services in Angular 2 are injected with dependency injection which refers to the features such logging, data integration, app configuration etc. Listing 2 illustrates the example code for Angular 2 services.

```

export class Logger {
  log(msg: any) { console.log(msg); }
  error(msg: any) { console.error(msg); }
  warn(msg: any) { console.warn(msg); }
}

```

Listing 5 Login services to the web browser in Angular 2

4.2.5 Templates and forms

Angular 2 templates are much similar to the HTML documents but they actually a form of HTML that guides Angular 2 while rendering the component to its UI component. Angular 2 templates uses all the HTML-based syntaxes and elements such as `<h1>` `<p>` `` etc., but it offers some unique features such as interpolation and expression.

Similarly, Angular 2 forms are the integral part of the user interface for well-integrated, functional and convincing data entry user experience. Angular 2 forms are written in Angular 2 directives based upon the Angular 2 template syntax. A web application without any form is doomed because it plays a vital role to increase user experience. In Angular 2, forms are created with the component and template. Angular 2 models are used for the form for data interactivity and to support two-way data binding. Forms thus created will then be visually designed with CSS and lastly the information will be shared across the HTML element. [13.]

4.2.6 Dependency Injection

Dependency in simple terms is actually a services and dependency injection which is a formal way of providing a service to the Angular 2 component in order to create a new instance of class. A dependency injection is winded up with Angular 2 and they are used widely.

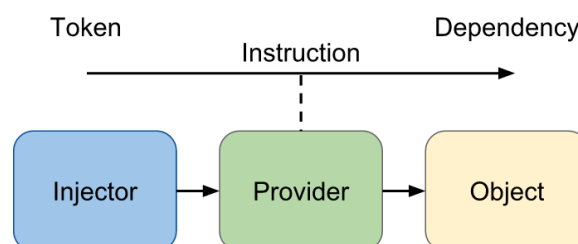


Figure 5 Components in Dependency Injection [13]

Figure 5 illustrates the dependency Injection components in Angular 2. A dependency injection in Angular 2 basically contains three particular things: an injector, provider and dependency.

- An injector: Injector exposes the APIs in order to create dependency instances.
- A provider: Provider takes help from the injector to create dependency instances.
- A dependency: An object is created with the dependencies. [13]

4.2.7 Data Binding and Meta Data

Data binding is the process of synchronization of data between different components. In Angular 2 application, data binding makes the coordination between the component and the templates. Angular 2 applications use two-way data binding which updates the data in the view and controller of the application simultaneously.

Advantages of the data binding in an Angular 2 application are highlighted below:

- Data binding has made the user inputs fast. Earlier, user's responses were pushed into HTML and such methods of push-pull are unvaried and error-vulnerable.
- Data binding provides the efficiency in an Angular 2 application for user input as it works as a communicating bridge between templates and components. [12.]

However, two-way data binding techniques fire some vulnerability in web applications.

- The change in the value in the data binding changes the event.
- When some directives are used for more than once, there is the chance of break in the applied data binding.

Similarly, another most important fundamental architectural tool used in the Angular 2 application is meta-data. Meta-data is responsible for the processing class into Angular 2 applications. A class will remain a class in Angular 2 application and it never behaves as component. Meta-data is needed to process the class into components in Angular 2 application.

Advantages of using Meta data in Angular 2 application is highlighted below:

- Meta-data provides the clear dependency injection style in Angular 2 application
- Make-data makes developers to think the applications as a form of component tree and Angular 2 style.
- Meta data in Angular 2 is future-oriented as it seems to be focused on upgrading of frameworks. [15.]

Likewise, Angular 2 Meta-data also offers few disadvantages:

- The manual linking of the components can lead the users deprived of getting out-most of the user experience.

4.3 React

The narrative of React was given by Facebook in the year 2013. Then it was open-sourced and the final release was made in 2015. React focuses on making the most interactive user Interfaces in comparison to its compatriots AngularJs, Angular 2 and ember. The biggest changes brought by React over other existing frameworks and libraries are the use of the virtual DOM. Literally, React is not just considered a normal front end library, but it has become a paradigm in the world of front-end web development.

React is unlikely similar to the large frameworks like AngularJs and Angular 2. It is based upon the philosophy of UNIX, i.e. combining the simple things to get the best out of it. React allows developers to write the declarative JavaScript components which are far more superior to the traditional way winding up the HTML code templates into JavaScript.

Basically, MVC design patterns are considered the most underlying design pattern for user interface design in front-end web development. MVC patterns contain three components: the model, view and controller. The majority of the front end framework focuses on these three parts. However, React only focuses on the DOM because it is the sematic representation of the View. Literally, React provides the view part “V” in the MVC design pattern.

Moreover, React is absolutely a different concept in web application development. React has taken a unique approach in solving the problems faced by large-scale user interfaces. This is in fact created to solve the large-scale user interface problems of Facebook and Instagram. Along with solving the problems in user interface, React came with the idea of making the development process easier for developers with varied skill levels. [16, 1-2].

Advantages of using React are highlighted below:

- Parsing HTML into JavaScript is so main-stream, while React is JavaScript centric i.e. it allows the developer to write JavaScript codes to generate HTML. This feature is so unique and liked by majority of the web developers. Since, JavaScript technologies are running all over the web applications from front end to backend. So, adjusting with the React will be a good idea.
- Since React applications are made with the arrangement of the components. React application can roll up the style, view and state in one component which is obviously better than the rolling up with components and directives.
- Bugs fixing in React is very fast in comparison to the large frameworks like Angular 2.
- Although it takes a bit more time to develop the React application but later it will save a lot more time when it comes to maintenance.
- React is developed with the philosophy; learn once, write anywhere. React doesn't care about the technology stacks. React components can be rendered using node.js into the server. Also, it allows the developer to add new features in it without the use of any pre-existing codes. [17.].

From section 4.3.1, the features of React will be discussed:

4.3.1 JSX

JSX is an HTML-like syntax extension features introduced by React. JSX is used in the React to describe the UI component of the application. JSX might look something similar to the mark-up language like HTML or XML but it is fully JavaScript centric and fully functional features used to create React application. However, it is not mandatory to use JSX in any React application but the application equipped with JSX provides high level of performance, fast error handling and easy deployment.

JXS contains exactly the similar elements like HTML, such as `<h1>`, `<p>`, `` etc. but these elements need to be nested within one `<div>` element in JXS which later need to be wrapped into a component. Literally, all the JXS elements need to be included inside the React component followed by rendering in document Object Model or exporting it into another component. [17]

4.3.2 React Components, Props and State

React components are the building block in React application development especially when talking about updating the user interface. In general, React component is considered similar to the JavaScript functions. React permits to create the components with a method called `React.createClass()`.

“Props” are the communication channel features used in React component. It acts as a bridge between the parent and child components.

```
import React from 'react';
class MyApp extends React.Component {
  constructor(props) {
    super(props);

    this.state = {
      head: "this is the header",
      "data": "this takes data from state"
    }
  }
  render() {
    return (
      <div>
        <h1>{this.state.head}</h1>
        <h2>{this.state.data}</h2>
      </div>
    );
  }
}
export default MyApp;
```

Listing 6 Representation of components, state and props in React

Similarly, the “State” is known as the heart of the React component. The “State” is the only feature in React component that makes the component more interactive and dynamic. The “State” allows the component to return the same output for the given input. [17].

4.3.3 Virtual DOM

DOM manipulation in the contemporary and dynamic web is considered a primer feature. However, it is very slow in comparison with other JavaScript operations. The reason behind the slowness of the DOM manipulation is because of its regular updates made by the modern JavaScript frameworks. JavaScript frameworks are supposed to update only particular object in DOM but things are not happening in that way. A framework actually rebuilds and then updating the entire object in DOM even if the updates is needed for only particular object. This way of updating from JavaScript frameworks are lowering down the performance of every web applications. Considering this particular problem in the DOM manipulation; React introduced the concept of Virtual DOM.

Virtual DOM is completely based on the DOM object and considered as the light weight copy of DOM. DOM objects are responsible to display the data into the user interface component of the application but virtual DOM Object has nothing to do with the User Interface but it offers special features of changing and updating the objects in the back of the UI.

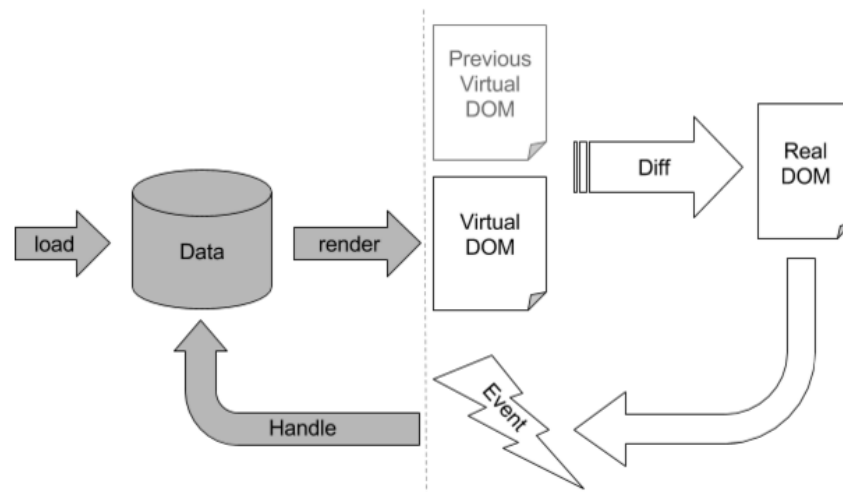


Figure 6 Virtual DOM workflow visual representations [18]

When rendering a JSX element, every virtual DOM gets updated. This does not sound interesting but when it comes to the efficiency and performance, virtual DOM is very powerful than the normal DOM.

4.3.4 Templates and Forms

React offers a simple functioning template that does not require a runtime environment. Syntaxes in React are simple like HTML, which are well supported by all browsers and IDEs. When inspecting the codes in a browser, the inspection in HTML and the coding does not look much different because of the declarative coding. The templates made with React are supportive in all the platforms such as Typescript, ES6, and CommonsJs.

React templates work differently when comparing with the Angular 2 templates. First, template containing the *.rt file needs to be compiled into the JavaScript file. The file thus compiled returns a function and later gives a Virtual DOM based upon the original DOM and user components.

Similarly, React works as a remarkable user interface library because it focuses on the view component of the application which is a user interface component. A view gets updated quickly when there is a change in the state. This phenomenon of updating the view in React makes it a good choice to overcome the form validation error since it

easily displays the errors in the form. However, it is hard to say that forms made in React are fully secured. An additional security measure needs to be applied to the form because a hacker can easily bypass the React codes by simply allowing the common variables right into the server. [19].

4.3.5 Flux Architecture

Flux is an architecture used in the react application. It is not similar to the regular libraries or framework but it provides a well round of features to the unidirectional data flow in a React application. Moreover, Facebook has assisted a library called dispatcher for the Flux architecture. Prototypical Flux architecture holds the dispatcher library in order to ease the event stare by setting up the event system along with the help of Node 'Js eventEmmitter module.

Basically, Flux architecture is made up of four different individual components: Actions, Dispatcher, Store, and Component (Views).

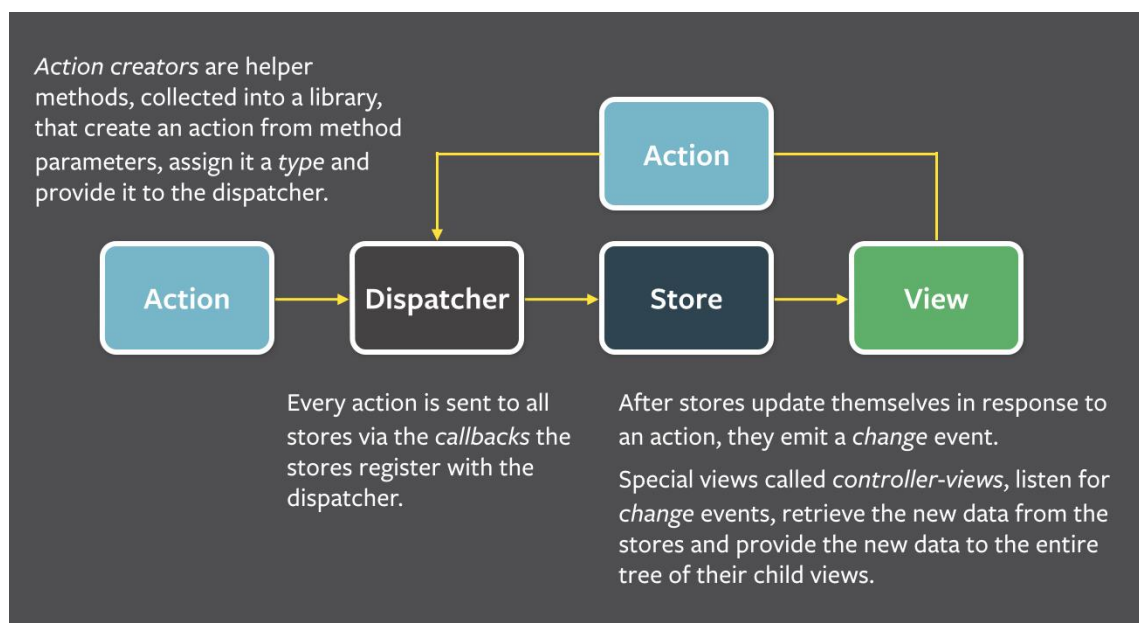


Figure 7 Data flow is Flux Architecture [20]

Figure 7 illustrates the data flow in between the individual components of the Flux architecture. The action in Flux is considered as a helper for the Dispatcher as it helps to pass data into the Dispatcher easily. Likewise, Dispatcher receives the actions and

later it forwards to the call-back function. The data thus forwarded will then be stored in a container for the application “Logic” and “State”. Finally, the Data State from the “Store” will then be passed to the UI component through the React “Props” method.

The data flow in the Flux architecture is unidirectional and considered central to the pattern. The three different components, State, Dispatcher and Views, are unconditional with well-defined Output/ Input. However, Action component in Flux are considered an object with new data and distinguishing property type.

The structure developed in the Flux architecture gives a clear idea about its purpose for a specific flow of data in the application. This gives a sense that it is based upon the reactive functional programming. In reactive functional programming, data flows only in one direction i.e. application state is only perpetuated in the store. [20]

4.3.6 Data binding in React

React offers a unidirectional data binding but it also supports the bidirectional data flow by the help of some external plugins. The unidirectional data flow provides the ground for the transmission of data from parents to child. The “Props” properties in React are responsible to carry the data from parents to child. The change in “Props” changes a React component which leads to the change in Component tree and its properties.

5 Angular 2 User Interface

A prototype user interface was designed in the Angular 2 in order to research the functionalities provided by the Angular 2 framework. The application thus developed was applied in React, to compare the similar features and product output between these two technologies.

Project Idea

The idea of the design was taken from the vacancies section of “Tuubi” an online portal of Helsinki Metropolia University of Applied Sciences. It has been found that there are specific portals available for libraries, IT services and help desk, thesis registration,

courses enrolment etc. However, the Vacancies available for the students are not specifically arranged into a different portal.

As the new system OMA has been introduced and applied recently to replace the “Tuubi” portal. It has become a necessity to separate the Vacancies section into a new portal. However, it seems reasonable to implement the Vacancies section in OMA services portal but making it more specific will make the students easily to get linked with the Jobs available.



Figure 8 Vacancies Section in Tuubi [21].

Figure 8 shows the vacancies section in the “Tuubi” portal. The section does not seem to be user-friendly because everything is included in a small section with very few details available.

A list of the problems found in the Vacancies section in “Tuubi” portal is highlighted below:

- Since “Tuubi” is a very diverse portal for students, teachers and administrators, it includes all necessary details needed by teachers and students. However, including everything into a single portal has made it look messy. Important sections such as, workspaces, vacancies, an announcement needs to be separated and introduced into a new portal. The space provided for the vacancies section is very small and sometimes this section does not even get noticed.

- Vacancies posted into this section are compact and small in letter size. Only the heading can be shown into this allocated space. The vacancies could have been provided with much more details.
- A better environment could have been developed to make the interactivity between the students and the jobs poster/recruiter.
- Every time the jobs post needs to be made by the Tuubi” administrator, or study coordinator or placement coordinator. It would have been much wiser to make a portal that can combine the UI with business logic. This would help the students to find jobs instantly.

5.1 Specifications

The UI Portal developed during the final year project was made considering three use cases. Figure 9 illustrates the use case diagram for the Job Portal which shows that the Job Portal has three primary Actors: An actor of Student, An actor of Recruiter and An actor of Teacher/ Administrator.

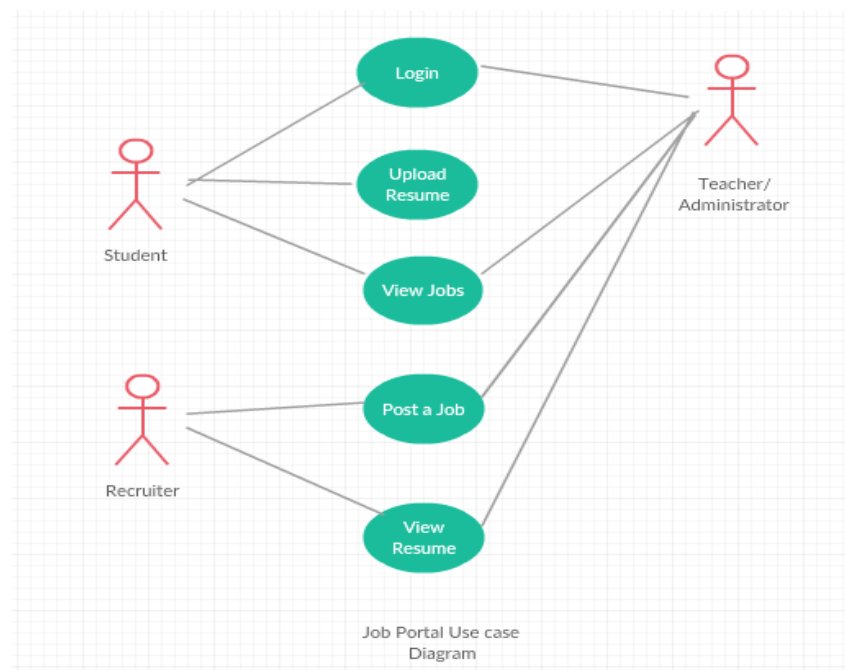


Figure 9 Use case Diagram for Job Portal

- An actor of Student: Students can login to the system, upload the resume to the system, and can view the Jobs in the system.
- An actor of Teachers/Administrators: Teachers and administrators can login to the system, view the jobs and resume availability in the system and can post jobs into the system.
- An actor of Recruiter: Recruiters can view the resumes and post the jobs into the system.

5.2 Environment Set-up

The biggest turnoff in the Angular 2 application development is the environment set up. This is the most essential thing to do while starting the Angular 2 project. The environment set up was done to leverage the ES6 (European Computer Manufacturers Association Script version 6) in browser support. This has to be done because all browsers do not support the ES6 script.

There were two alternatives available for the Angular 2 environment set-up; one is by directly installing the Angular –cli and another is manually setting up the environment with the installation of npm, grunt etc. For this project, the later method of environment setup was done. The project was setup by using the external terminal called Gitbash in Windows Operating System. Likewise, notepad++ text editor was used to write the script. The project was set up into three steps:

The first step was loading up an index.html file which was followed by npm installation of the package.json file. And in order to run the application, lite-server was installed. Two files are created in this step:

```
| -index.html  
| -package.json
```

Lite server installation was done to run the application in a browser.

The second step was involved in installing the Typescript; and it was installed along with the package called “typings”, which is a package that the Typescript used to understand how external libraries are used in the application.

```
|-tsconfig.json
|-typings.json
```

The third step was involved was for the dependencies installation, which is followed by importing the core packages. The necessary packages installed into the application are:

- @angular/core
- @angular/common
- @angular/compiler
- @angular/platform-browser
- @angular/platform-browser-dynamic

Similarly, three additional packages were installed to support the application:

- @angular/router
- @angular/http
- @angular/forms

The above steps outline the development environment setup that was applied to the metropoliaJobs UI development in Angular 2. After the necessary steps applied, the final look of the entire package.json file is shown in listing 7. Package.json file contains all the listing of Packages, Task-runners, Scripts and Dependencies with their respective versions.

```
{
  "name": "metropoliaJobs",
  "version": "1.0.0",
  "description": "MetropoliaJobsPortal",
  "main": "index.js",
  "scripts": {
    "start": "tsc && concurrently \"npm run tsc:w\" \"npmrunlite\"",
    "lite": "lite-server",
    "tsc": "tsc",
    "tsc:w": "tsc-w",
    "typings": "typings",
    "postinstall": "typingsinstall"
  },
```

```

"keywords": [],
"author": "",
"license": "ISC",
"devDependencies": {
  "concurrently": "^3.4.0",
  "lite-server": "^2.3.0",
  "typescript": "^2.2.2",
  "typings": "^2.1.0"
},
"dependencies": {
  "@angular/common": "^2.0.0-rc.5",
  "@angular/compiler": "^2.0.0-rc.5",
  "@angular/core": "^2.0.0-rc.5",
  "@angular/forms": "^0.3.0",
  "@angular/http": "^2.0.0-rc.5",
  "@angular/platform-browser": "^2.0.0-rc.5",
  "@angular/platform-browser-dynamic": "^2.0.0-rc.5",
  "@angular/router": "^3.0.0-rc.1",
  "@angular/upgrade": "^2.0.0-rc.5",
  "core-js": "^2.4.1",
  "reflect-metadata": "^0.1.10",
  "rxjs": "^5.0.0-beta.6",
  "systemjs": "^0.20.11",
  "zone.js": "^0.8.5"
}
}

```

Listing 7 package.json file for the metropoliaJobs

The architecture overview of the application in Angular 2 developed with Typescript is shown in the listing 8.

```

|-metropoliaJobs/
  |-----app.components.ts
  |-----app.module.ts
  |-----app.main.ts

```

```
|-index.html
|-package.json
|-tsconfig.json
|-typings.json
|-systemjs.json
```

Listing 8 Application Architecture for initial environment setup.

5.3 Implementation

The project was bound with the development of some set of “Forms” in Angular 2, a “Form” component was created with the importing from the Angular 2 core package and its component was stylised with the component animation features available in Angular 2.

Finally, after the configuration and installation of the necessary tools for the development environment setup, the UI was developed using a certain set of additional UI technologies. HTML, SAAS, Angular 2, Typescript, Twitter Bootstrap 4 powered by Angular 2, jasmine and karma are the technologies used to design the prototype application in Angular 2. The Lite-server was used to serve the application into the web browsers. The application was tested in the server: <http://localhost:3000/>. The screen shots of the prototype UI developed in Angular 2 are listed in Appendix 2.

6 UI Transition in React

This thesis is focused on to make an analysis of UI designed in two different platforms. The prototype UI application was deployed completely to see how it is implemented into React.

6.1 Development Environment Set-up

The basic development environment set up for the React UI development was done by initiating the npm and the dependencies for the application in a package.json file. The production dependencies; React, React-DOM, Redux, Redux-react, Redux-thunk and babel were installed at the beginning. After that; babel, babel-core, babel-preset-es2017, babel-preset-react, Webpack@2.1.0-beta.27, file-loader, babel-loader,

Webpack-dev-server@2.1.0-beta.11 and react-hot-loader@3.0.0-beta-6 were installed as dev dependencies. Similarly, one entrance of JavaScript file was created for the initial testing of the application. After the import of a few mandatory packages from React library, the application dependencies were ready for deployment.

Application structure:

```
|-metropoliaJobs
  |--src
    |--components
      |----App.js
    |--app.js
    |--index.html
  |--babelrc
  |--package.json
  |--webpack.config.js
```

Listing 9 Application Structure in React

Listing 9 illustrates the application structure in React. The application architecture folder structure contains the necessary files for the development of React UI. All the files have been included into the parent directory called metropoliaJobs.

```
{
  "name": "metropoliaJobs",
  "version": "1.0.0",
  "description": "MetropoliaJobsPortal",
  "main": "index.js",
  "scripts": {
    "build": "webpack",
    "serve": "webpack-dev-server --hot --",
    "inline --content-base ./src --open"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "react": "^15.4.0",
    "react-dom": "^15.4.0",
    "react-redux": "^4.4.5",
    "redux": "^3.6.0",
    "redux-thunk": "^2.1.0"
  },
  "devDependencies": {
    "babel-core": "^6.14.0",
```

```

"babel-loader": "^6.2.5",
"babel-preset-es2015": "^6.14.0",
"babel-preset-react": "^6.11.1",
"eslint": "^3.5.0",
"eslint-config-airbnb": "^11.1.0",
"eslint-plugin-import": "^1.16.0",
"eslint-plugin-jsx-a11y": "^2.2.2",
"eslint-plugin-react": "^6.3.0",
"file-loader": "^0.9.0",
"react-hot-loader": "^3.0.0-beta.6",
"webpack": "^2.1.0-beta.27",
"webpack-dev-server": "^2.1.0-beta.11"
    }
}

```

Listing 10 Dependencies Installation for the React project with their respective Versions

Listing 10 Illustrates all the dependencies required for the React UI application. These dependencies contain all the necessary files, packages, task runners and necessary plugins required for the React UI. The respective versions of the tools have been outlined into the package.json file.

6.2 Implementation

The goal of this project was to develop exactly the same UI in React which was developed in Angular 2. The UI thus developed was completely the same as that of Angular 2. Like Angular 2, React also followed a certain set of protocols and patterns but the UI development in React was based upon following React based tools:

- React
- React-DOM
- Redux
- Redux-react
- Redux-thunk
- Webpack- module loader
- Webpack-dev server
- Babel - to make the application runnable into lower versions of the ES6 browser.

Furthermore, in order to make the React UI more stylised, a recent version of CSS called PostCss was used. PostCss is the newest version of CSS that transforms CSS

with JavaScript. The main aim of using the PostCss here was to make the UI more JavaScript centric.

Finally, after successful completion of the configuration and setups, the UI in React was developed using the React based syntaxes. JSX was used for similar mark-up functionalities. A huge number of React components were deployed to make the final output of the React UI. The UI thus developed was exactly the same as that in Angular 2 which was run and tested in the server: <http://localhost:8000/> with the Webpack dev-server runner. The final version of the transitioned UI can be found in Appendix 2.

7 Discussion

A comprehensive study of two technologies (Angular 2 and React) was carried throughout the thesis considering different methodologies. Three major steps were applied to find out the objectives of the study. Firstly, various sets of key data for two different technologies were collected from different sources. Secondly, the features and tools required for the transition/migration of UI from Angular 2 into React were analysed. Lastly, the implementation of a user interface was successfully done in two different platforms.

Both platforms offered solid features and outlook to the UI application. However, the goal of the study was to form a tie breaker through different methodologies. A certain set of comparable guidelines was introduced and the findings were discussed to make efficient comparison between the technologies.

Is Angular 2 Comparable to React?

This is an important question that could be asked every time about new UI technologies even if the existing technologies are capable of addressing a solution to the complex problems. When it comes to scaling the two different technologies, there might come thousands of criteria's. However, setting up a specific goal can be made to highlight the major key aspects and it can be possible to make the comparison.

Which one offered the better UI: Angular 2 or React?

The answers to the above question is solely analysed by the outcome of this thesis. Six major fundamental points are taken into comparison between the two technologies.

Performance

React wins in performance in comparison with Angular 2. When updating and synchronising the components in the application, React used the virtual DOM in the application even when the application architecture was complex and that resulted in the superior performance, while in Angular 2 UI the regular DOM was involved for the updating all the DOM objects without updating the necessary one, which resulted the slowness in the application.

Templating

When comparing the templating between the two technologies, Angular 2 offered superiority because the UI development does not necessarily need the JavaScript centric templating style. Templating is more concerned with the UI components and even a simple mark-up will offer simplicity in templating than the complex JSX. Angular 2 uses the HTML mark-up in a simple way which is far better than writing the tedious JSX codes. While developing the UI, it was found that the templating style in Angular 2 was far simpler than that of React.

Learnability

Learnability in React is far more superior to Angular 2. Learning Angular 2 is interesting at the beginning but the later stage becomes frustrating because of its complex lifecycle. On the other hand, React is unique because of the JavaScript centric feature and even a novice developer can master it up within a week. The simple ecosystem and strong lifecycle are the considerable features that make the React preferable for the learners. Better learning curves mean there will always a better chance of focusing on the user experience by overcoming the existing problems.

Structuring data model

When it concerns the data modelling comparisons between Angular 2 and React, both technologies offer equality. The performance of Angular 2 is sensitive when it deals with “Scopes” and this suggests that using large models will not be permitted. This results in a more testable and simple codes on one hand and on the other hand forces to break down the normal procedures for rebuilding the back-ups. However, React gives freedom of choice in structuring the data model without downsizing the performance. However, it depends completely on the competencies of developers.

Code reshuffling

There is a tie when comparing the code reshuffling between Angular 2 and React. There are plenty of ready-made libraries available for Angular 2. However, React gives the flexibility to reshuffle the code in its own way.

Package management

React gives more flexibility in package management than Angular 2. React uses Require.js and Webpack to make better package management. But, Angular 2 does not allow the codes to run and deploy more flexibly and this results in the poor packaging of codes.

Other comparable aspects?

The two technologies highlighted in this thesis are further comparable. The common goals of both technologies are more or less the same. They both have been created to make the data rendering easy from the server side of the application to the user interface component.

Table 2 Further Comparison between Angular 2 and React.

	Angular 2	React
Browser Support	Firefox(latest),Mozilla(latest), safari(7-10),IE(9-11),edge(13-14),	Firefox(latest),Mozilla(latest), safari(7-10),IE(9-11),edge(13-14),
Performance	Comparatively slower than React	React is faster
Architecture	MVC architecture support	V layer in MVC, Flux support as an external architecture.
Developers Preferences	Programmers from Object oriented background	JavaScript developers
Future versions	Angular 4	React V16.x.x
Rendering	Server side	Server side
Mobile support	Ionic framework	React Native
Templating	Typescript file	JSX file
Code organization	Good code organization	Complex due to Js centric
Data Binding/data flow	Bi-directional (two way)	Unidirectional

8 Conclusion

Considering all the methodologies applied, the study was finally concluded as React being more preferable technology for the development of a user interface. Nevertheless, Angular 2 might have been in the lead if the study was based upon other criteria besides the one highlighted in this thesis.

The technologies for the development of user interfaces evolve very rapidly like any other relatable field in web development. The way technologies are evolving in the front end frameworks has been changing the logic and methodologies used in the development process. The front-end and user interface were meant for the designers. However, the use of frameworks and libraries in the front end has created dynamism in the web.

The final recommendation of this study is that it suggests developers to go through a practical approach to develop an identical user interface application in both technologies. The findings and outcomes described in this thesis are solely based upon my findings in this study and thus the outcome might differ in regard to the competencies of developers.

References

- 1 Usernap. Web design trends [online]
URL: <http://usersnap.com/blog/web-design-trends-2017/>
Accessed: March 2, 2017
- 2 Front End Masters. Front-end Spectrum [online]
URL: <https://frontendmasters.gitbooks.io/front-end-handbook-2017/content/tools.html> Accessed: March 6, 2017
- 3 Github. Facebook/React [online] URL: <https://github.com/facebook/react>.
Accessed: March 21, 2017
- 4 Github. Angular/Angular [online] URL: <https://github.com/Angular/Angular>.
Accessed: March 21, 2017
- 5 Stackoverflow .Stackoverflow survey [online]
URL: https://stackoverflow.com/insights/survey/2017/?utm_source=social&utm_medium=social&utm_campaign=dev-survey-2017&utm_content=social-share. Accessed: March 21, 2017
- 6 Made with React. Apps collection [online] URL: <http://madewithreact.com/>
Accessed: March 22, 2017
- 7 Apps made in Angular 2.0. Angular Expo [online] URL: <http://angularexpo.com/>.
Accessed: March 22, 2017
- 8 Angular Zone. Angular History [online].
URL: <https://angularzone.wordpress.com/Angularjs-history/>
Accessed: March 22, 2017
- 9 NG-conf. Keynote NG-conf 2014 [online] 2014
URL: <https://www.youtube.com/watch?v=r1A1VR0ibIQ>.
Accessed: February 28, 2017
- 10 AngularJs. Angular 1 and Angular 2 integration [online]
URL: <http://Angularjs.blogspot.fi/2015/08/Angular-1-and-Angular-2-coexistence.html>. Accessed: March 22, 2017
- 11 Typescript for Angular 2 [online].
URL: <http://naywinmyint.com/typescript-for-Angular-2-part-1/>.
Accessed: March 23, 2017
- 12 Angular. Component [online]
URL: <https://angular.io/docs/ts/latest/api/core/index/Component-decorator.html>
Accessed: March 23, 2017

- 13 Angular. Directives [online].
URL: <https://Angular.io/docs/ts/latest/guide/attribute-directives.html>
Accessed: March 23, 2017
- 14 ThoughtTram. Dependency Injection in Angular 2 [online]
URL: <https://blog.thoughttram.io/angular/2015/05/18/dependency-injection-in-angular-2.html>. Accessed: March 22, 2017
- 15 Github. Ng-MetaData [online] URL: <https://github.com/ngParty/ng-metadata>
Accessed: March 22, 2017
- 16 Introduction to React. Gackenhimer Cory. 2015
- 17 React. Documentation [online]. URL: <https://facebook.github.io/react/>
Accessed: March 24, 2017
- 18 Afas.nl. Virtual DOM [online]
URL: <https://www.afas.nl/blog/virtual-dom-how-inefficiency-can-lead-to-better-performance>. Accessed: March 26, 2017
- 19 HTML5 Hive. Form Validation in React. [Online].
URL: <https://html5hive.org/reactjs-form-validation-tutorial/>.
Accessed: March 28, 2017
- 20 Flux Documentation. In depth Overview [online]
URL: <https://facebook.github.io/flux/docs/in-depth-overview.html#content>
Accessed: March 29, 2017
- 21 Tuubi. Vacancies [online].
URL: <https://tuubi.metropolia.fi/portal/group/tuubi/etusivu>
Accessed: February 20, 2017.

Appendix 1

Appendix 1: Survey Questionnaire for Front End technologies

Which web technology do you prefer to use for front end user interface?

☐ AngularJs

☐ Angular 2.0

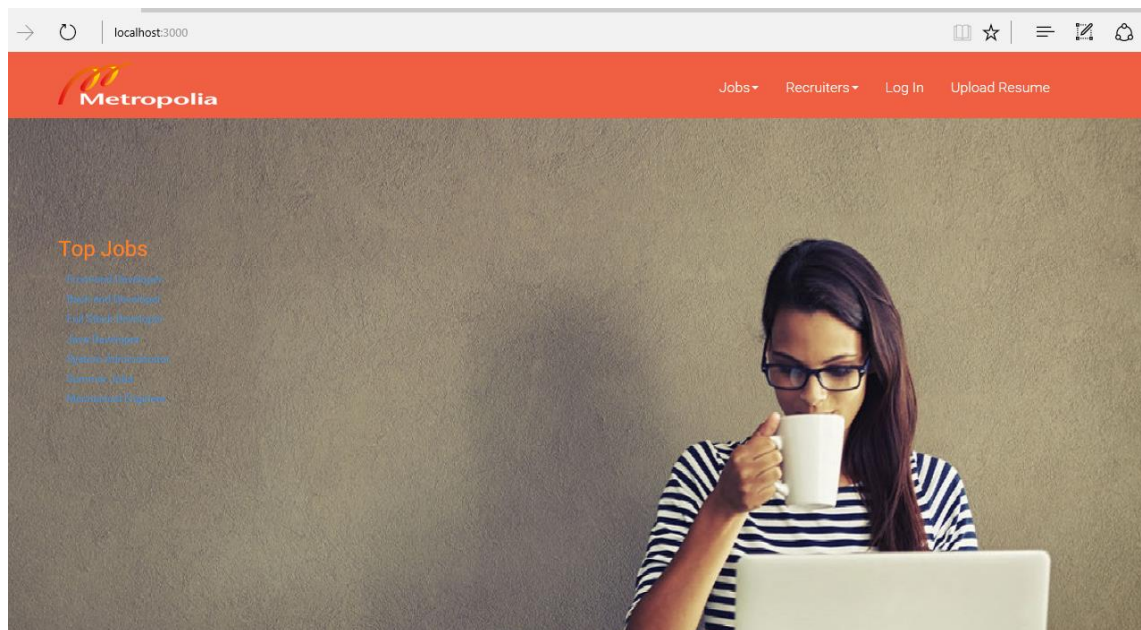
☐ ReactJs

☐ Other (ember, ionic, jQuery etc.)

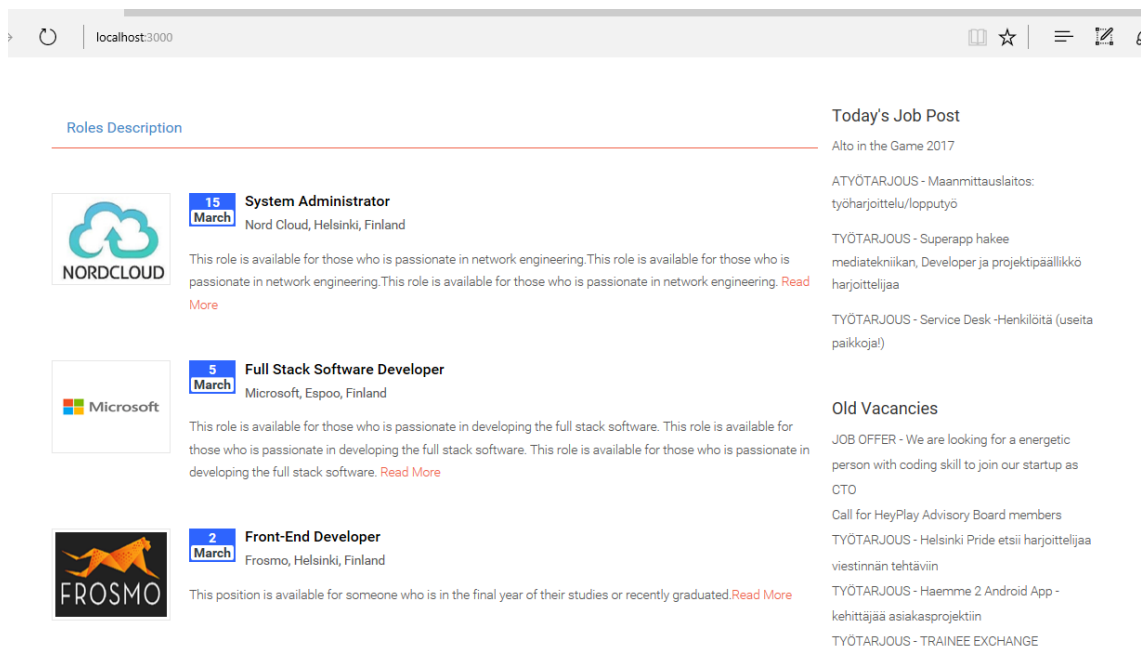
Appendix 2

Appendix 2: Screen Shots of the UIs

Front page /Student Section:




Main cover page




Middle Section

localhost:3000




2 March
Front-End Developer
 Frosmo, Helsinki, Finland

This position is available for someone who is in the final year of their studies or recently graduated. [Read More](#)




27 February
Developer Intern/Thesis Worker
 Nokia House, Espoo, Finland

View the opportunities Nokia offers students and graduates, and get ready to apply. [Read More](#)



26 February
JavaScript Developer
 Frosmo, Helsinki, Finland

This role is available for someone who is very good at programming in JavaScript. Candidates must have the hands on skills in AngularJS, Angular 2.0 or ReactJS. [Read More](#)



24 February
Service Desk Support Specialist
 Nordea, Helsinki, Finland

This role is available for someone who is good at configuring the network protocols. [Read More](#)

Call for HeyPlay Advisory Board members

TYÖTARJOUS - Helsinki Pride etsii harjoittelijaa viestinnän tehtäviin

TYÖTARJOUS - Haemme 2 Android App -kehittäjää asiakasprojektiin

TYÖTARJOUS - TRAINEE EXCHANGE

PROGRAM JOB PLACEMENT Mobility Solutions Korea, Co., Ltd.


TYÖTARJOUS - TRAINEE EXCHANGE

PROGRAM JOB PLACEMENT (Stratio, Inc.)

JOB OFFER - Academic Work is looking for IT talents

Lower middle section

localhost:3000



As a leader in the rapidly growing industry of eCOA technology solutions, CRF Health offers a passionate community of colleagues, ample opportunities for your career path, and the support you need to succeed both individually and as a team. [Read More](#)

« 1 2 3 4 5 »

Services

- Home
- Library Services
- Help Desk
- JobStep

Metropolia Jobs

OMA

Post a Job

Contact Us

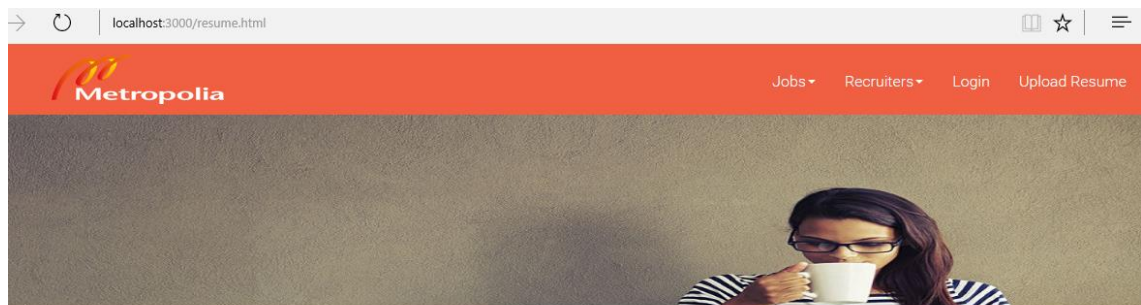
Terms of use

There are employer presentations at the campuses and students attend various fairs and events. Career Services offers meeting forums for employers and students in social media and helps to use Jobstep.net, the career and recruitment service of the Finnish universities of applied sciences.

Copyright © 2017 | Kumar | Metropolia.

Footer section

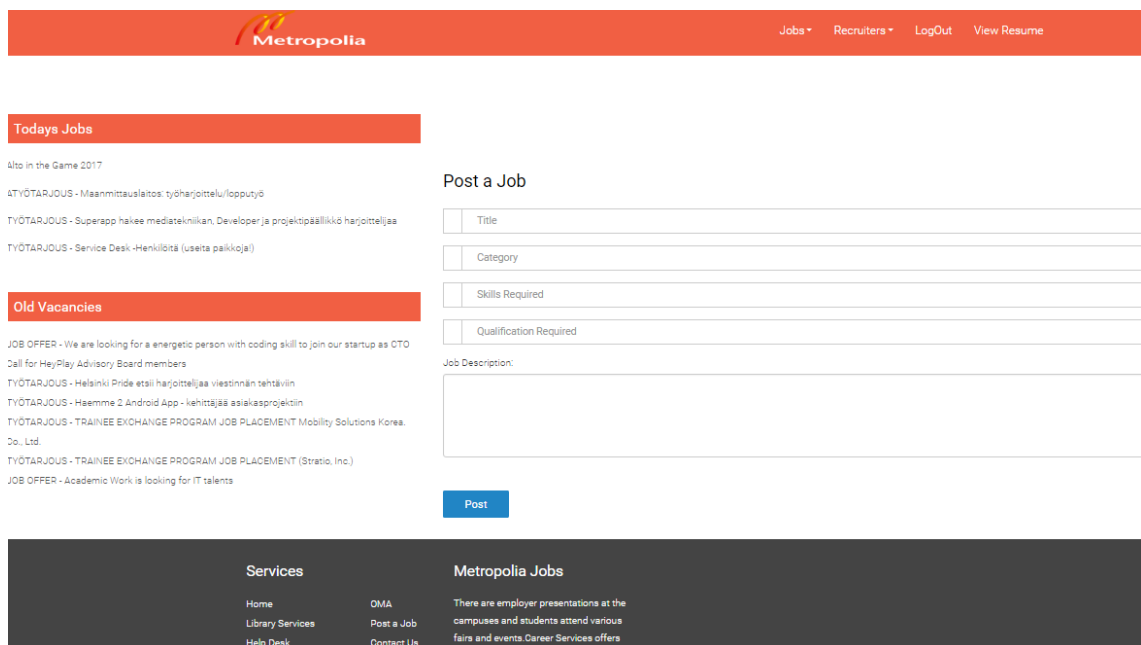
Uploading Resume:



Upload Your Resume




Email Address

Admin Dashboard for Job Posting:



Admin Dashboard


Mobile Responsive:

Top Jobs

- Frontend Developer
- Backend Developer
- Full Stack Developer
- Java Developer
- System Administrator
- Network Engineer
- Marketing Manager

Roles Description



15
March

System Administrator
Nord Cloud, Helsinki, Finland

This role is available for those who is passionate in network engineering. This role is available for those who is passionate in network engineering. This role is available for those who is passionate in network engineering. [Read More](#)



5
March

Full Stack Software Developer
Microsoft, Espoo, Finland

**5 Full Stack Software Developer****March**

Microsoft, Espoo, Finland

This role is available for those who is passionate in developing the full stack software. This role is available for those who is passionate in developing the full stack software. This role is available for those who is passionate in developing the full stack software. [Read More](#)

**2 Front-End Developer****March**

Frosmo, Helsinki, Finland

This position is available for someone who is in the final year of their studies or recently graduated. [Read More](#)

**27 Developer Intern/Thesis Worker****February**

Nokia House, Espoo, Finland

View the opportunities Nokia offers students and graduates, and get ready to apply. [Read More](#)

**26 JavaScript Developer****February**

Frosmo, Helsinki, Finland

This role is available for someone who is very good at programming in JavaScript. Candidates must has the hands on skills in AngularJS, Angular 2.0 or ReactJs. [Read More](#)

**24 Service Desk Support Specialist****February**

Nordea, Helsinki, Finland

This role is available for someone who is good at configuring the network protocols. [Read More](#)

« 1 2 3 4 5 »

Today's Job Post

Alto in the Game 2017

ATYÖTARJOUS - Maanmittauslaitos: työharjoittelu/lopputyö

TYÖTARJOUS - Superapp hakee mediatekniikan, Developer ja projektipäällikkö harjoittelijaa

TYÖTARJOUS - Service Desk -Henkilöitä (useita paikkoja!)

Old Vacancies

JOB OFFER - We are looking for a energetic person with coding skill to join our startup as CTO

Call for HeyPlay Advisory Board members

TYÖTARJOUS - Helsinki Pride etsii harjoittelijaa viestinnän tehtäviin

TYÖTARJOUS - Haemme 2 Android App - kehittäjää asiakasprojektiin

TYÖTARJOUS - TRAINEE EXCHANGE PROGRAM JOB PLACEMENT Mobility Solutions Korea. Co., Ltd.

TYÖTARJOUS - TRAINEE EXCHANGE PROGRAM JOB PLACEMENT (Stratio, Inc.)

JOB OFFER - Academic Work is looking for IT talents

Services

[Home](#)

[Library Services](#)

[Help Desk](#)

[JobStep](#)

[OMA](#)

[Post a Job](#)

[Contact Us](#)

[Terms of use](#)

Thank You
