REACT JS v16.0

Copyright © 2011 IGATE Corporation (a part of Capegemini Group). All rights reserved. No part of this publication shall be reproduced in any way, including but not limited to photocopy, photographic, magnetic, or other record, without the prior written permission of IGATE Corporation (a part of Capegemini Group).

IGATE Corporation (a part of Capegemini Group) considers information included in this document to be Confidential and Proprietary.

Document Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Revision No. | Author | **Summary of Changes** |
| May-2018 | 1.0 | Kathiresan.N | Lab book exercises created |

Table of Contents

[Table of Contents 3](#_Toc390968894)

[Getting Started 4](#_Toc390968895)

[Overview 4](#_Toc390968896)

[Setup Checklist for Spring Framework 4](#_Toc390968897)

[Minimum System Requirements 4](#_Toc390968898)

[Creating the first Spring application: 4](#_Toc390968899)

[Lab 1. Injecting dependencies into a Spring application 5](#_Toc390968900)

[Lab 2. Spring MVC with JPA 9](#_Toc390968901)

[Lab 3. Injecting cross-cutting concerns 14](#_Toc390968902)

[Appendices 15](#_Toc390968905)

[Appendix A: Class Diagrams 15](#_Toc390968906)

Getting Started

## Overview

This lab book is a guided tour for learning Basic Spring 4.0. It comprises solved examples and ‘To Do’ assignments. Follow the steps provided in the solved examples and work out the ‘To Do’ assignments given.

## Setup Checklist for Spring Framework

Here is what is expected on your machine in order for the lab assignments to work.

## Minimum System Requirements

* Intel Pentium IV or higher
* Microsoft Windows (NT 4.0/XP/2K)
* Memory: 256MB of RAM (512 recommended)
* 500MB hard disk space
* Node JS v6.11.0
* Create-react-app 1.5.2(other plugins are pre-installed)
* Notepad++ or Notepad

**Guidelines for all project below**

## Creating the first React application:

* Create a folder with your project name
* Ensure that node\_module is copied and pasted into your project folder
* Ensure you created a package.json file and src folder ,inside it

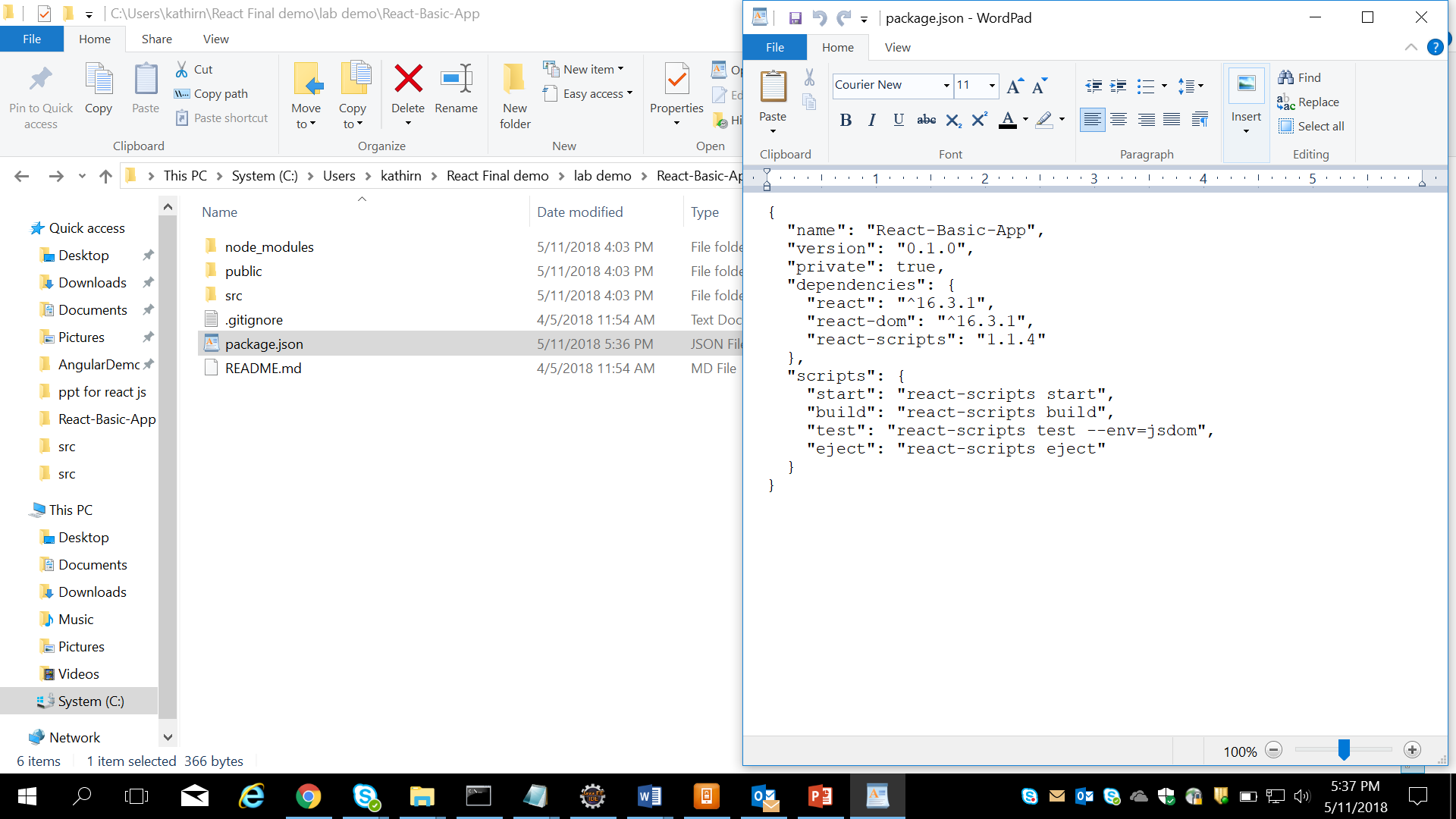
Lab 1 & 2: Creating a basic React App with createElement ,render() , attribute Expression and conditional Rendering

|  |  |
| --- | --- |
| **Goals** | * Create a hello-world app * Create a basic react app with render,attribute expression and conditional rendering |
| **Time** | 60 minutes |

* **Problem statement-1 : create a hello-world App**  using react, use the above guide lines
* **Problem statement-2.1 :**  **render,attribute expression and conditional rendering**

Create a react app with name **“React-Basic-App”. Follow the guidelines stated above and create the project.**

1. **Create a folder “React-Basic-App” and create packages.json and src folder in it**
2. **Add Below content in packages.json**



1. **Inside src folder create a js file with name index.js .**
2. **Inside index.js ,**
3. **Create a element with name ‘App’ and display a image based on conditional rendering & Jsx Expression attributes.**
4. Add images into proj src folder
5. Based on true or false display images, ie.**, display user.jpeg on true , on false display “guest.jpeg” inside a div tag**

(TRY: putting a title for the image like **“this is user”**)

Lab 3: Components,Data & Servers

Lab 3.1 :

|  |  |
| --- | --- |
| **Goals** | * Create a basic react app using createClass |
| **Time** | 30 minutes |

* **Problem statement-2.1 :**  **Create a component using createClass method and render Dom to User**

Like previous demo create a App with name (ie folder) **“React-create-element-demo”**

Ensure you have created package.json, copied node\_mondules and src folder is created

1. Create a file with name index.js and App.js

In App.js , create a element **‘Welcome’** , which displays **‘welcome to Capgemini’ in h1 format**  using createClass and display it in index.js using ReactDOM.render() Method

Lab 3.2 :

|  |  |
| --- | --- |
| **Goals** | * Create a basic react app using React Children |
| **Time** | 90 minutes |

* **Problem statement-3.2 :**  **Create a React application with name** “recat-create-children-api” **and implement the all the functionalities of React children.**

**Create a index.js file,Parent.js**

**In index .js , create a Parent element as below,**

**<Parent>**

**<br/>**

**<button>register</button>**

**<br/>**

**this is a sample content**

**</Parent>**

**a. count number of components ie child components in Parent component.**

**b. print type of child component nested inside the parent component(using property of child component) & map**

**c. Print also the Child Element**

**d. Restrict the parent to have only one child element inside**

Lab 3.3 :

|  |  |
| --- | --- |
| **Goals** | * Create a basic react app using React Children - forEach |
| **Time** | 20 minutes |

* **Problem statement-2.3 :**  **Create a React application with name “react-create-children-forEach” and implement the forEach of React children.**

Iterate and array of string using forEach functionality of React.Children and count number of charaters in each string

1. Print each string separately using forEach
2. Print length of each string using forEach
3. No of strings available in array
4. Print also the index of each element along with the component

Lab 3.4 :

|  |  |
| --- | --- |
| **Goals** | * Create a basic react app using React Props |
| **Time** | 20 minutes |

**Problem statement-2.4:**  **Create a React application with name “react-create-props-movie-list”.Use props ,Store data in props and display the data in index.js , through render method.**

**Guidelines:**

1. **Create 2 js files with Name index.js, MovieList.js**
2. **Index.js is going to render the final output**
3. **Pass the image name,ticket price and name of movie through the tag given in index.js and print image and movie name in the movieList component**

**(Note Movie Name and image name are same , for easy usage)**

**Use : HarryPotter, Avengers as movie Name and same as image name (.jpeg)**

Lab 3.5 :

|  |  |
| --- | --- |
| **Goals** | * Create a basic react app using React Props Types |
| **Time** | 30 minutes |

**Problem statement-2.5 :**  **Create a React application with name “react-create-propTypes-movie-list”.Use props ,Store data in props and display the data in index.js , through render method.**

**Continue Lab 2.5 exercise and implement propTypes , for below scenario**

1. Values of price are mandatory to be passed in props and also check their datatypes
2. If value of name is not provided , then it has to be taken as “DEFAULT PRODUCT”

Lab 3.6 :

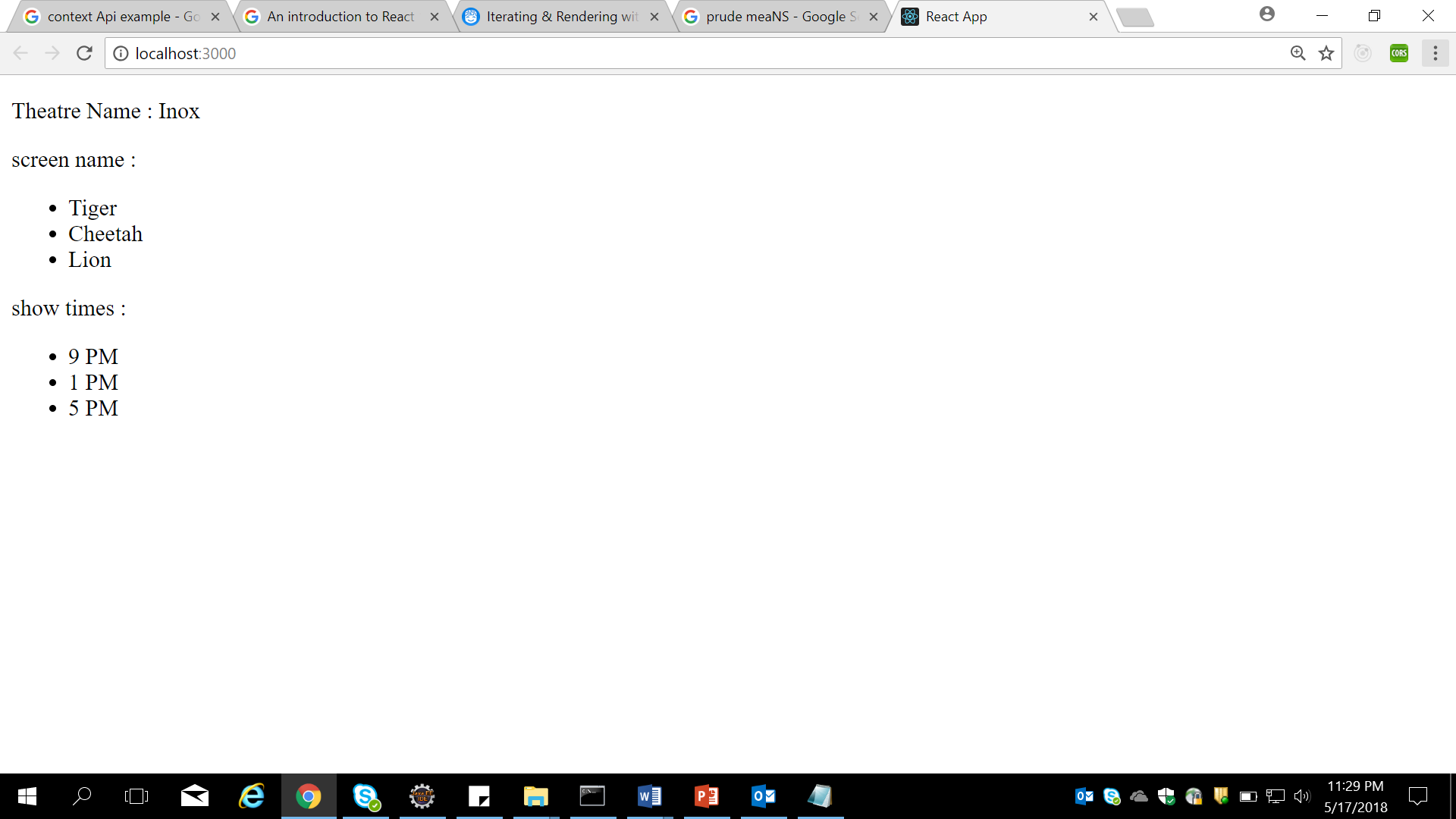
|  |  |
| --- | --- |
| **Goals** | * Create a basic react app using React Context API |
| **Time** | 60 minutes |

**Problem statement-3.6 :**  **Create a React application with name “react-create-context-mall”.Use Context Api and implement the below scenario**

Create 4 js files name as follows index , Mall, Theatre and Screen. Render the final output using index.js(using Mall in index)

Screen is child component of Theatre and Theatre is child component of Mall. Pass the values for variables in Screen through Context API

**Final output looks like :**



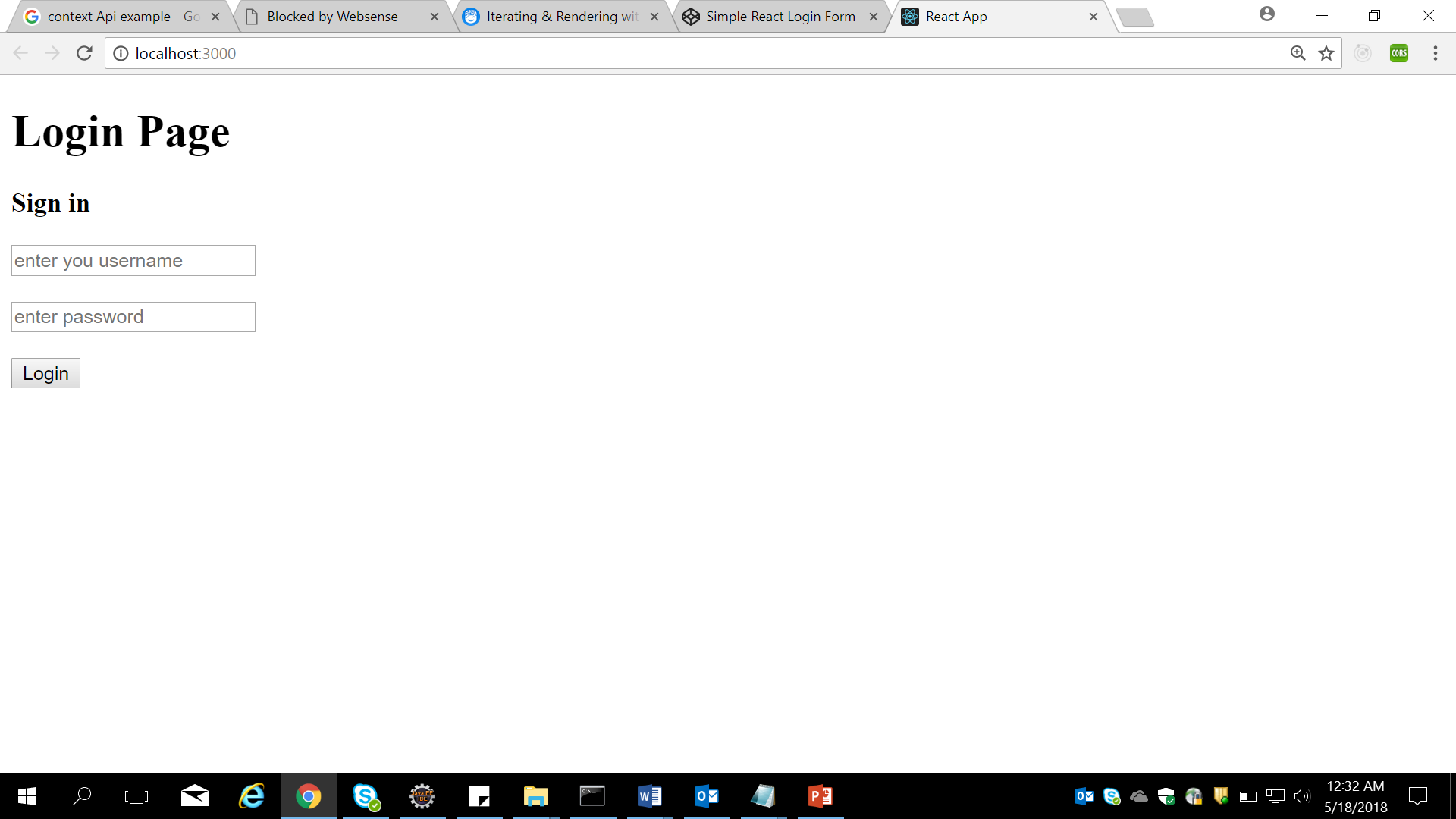
Lab 4. Lists and Forms & Events

Lab 4.1 :

|  |  |
| --- | --- |
| **Goals** | * Create a react LoginForm using React( nested components, nested Components, props, conditional rendering, inline styling, event handling , input refs) |
| **Time** | 120 minutes |

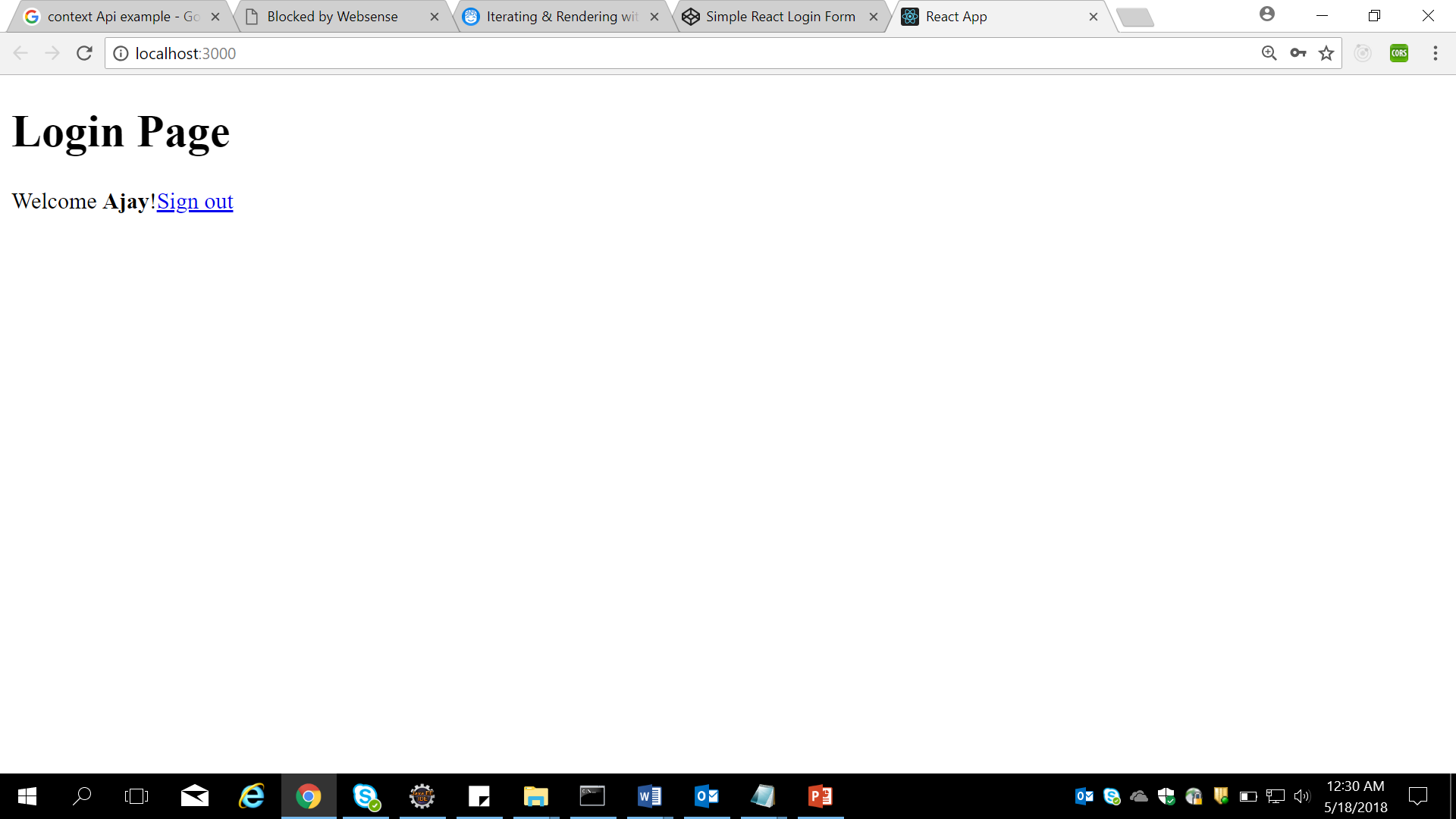
**Problem statement-4.1 :**  **Create a React application with name “react-create-login-form”.**

1. **On running it has to display a login form like below**



**After typing username and password,on submit,**

1. **Store the data of form in state**
2. **It has to render output like below screen**

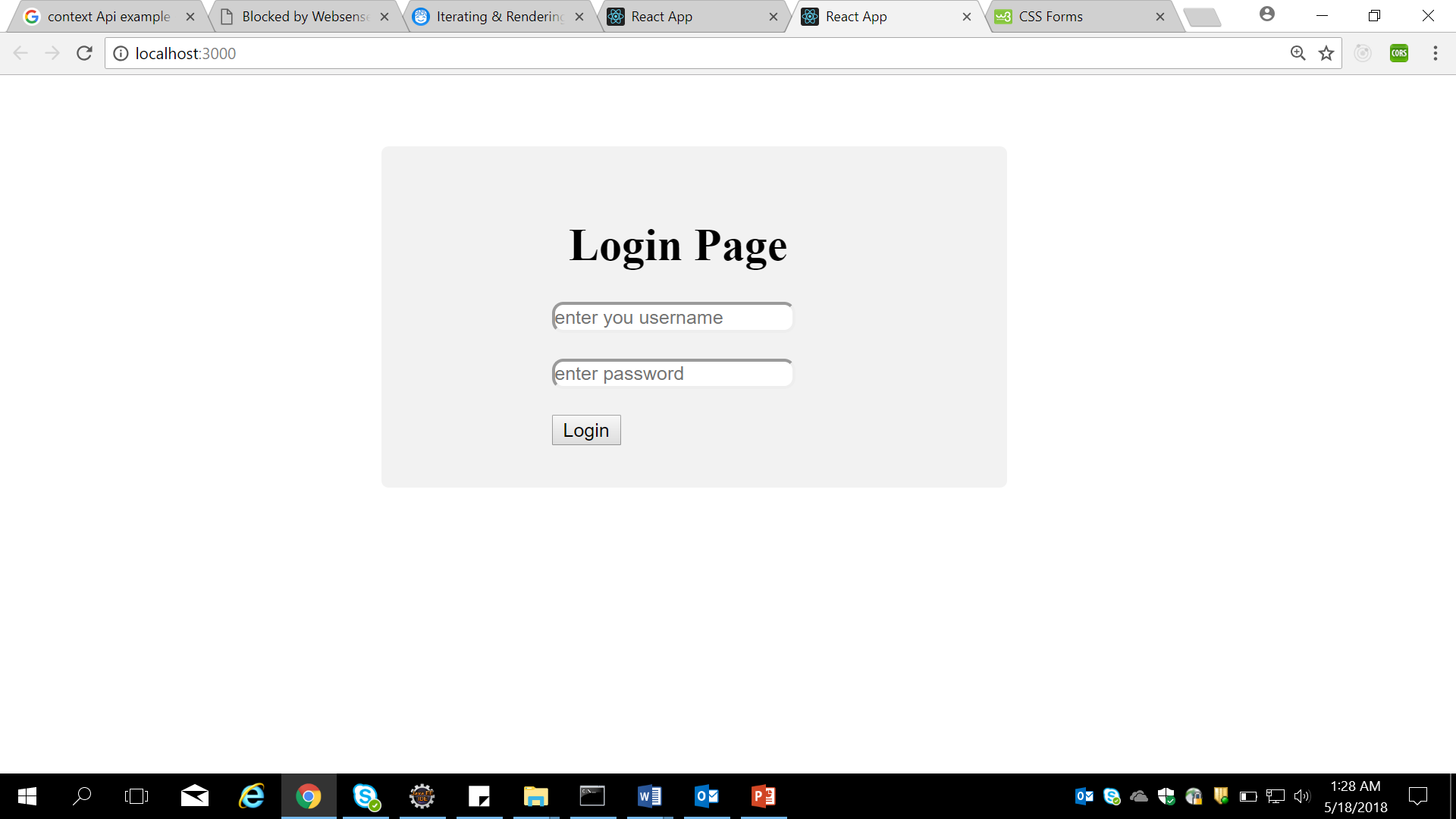


**If clicking signout it has to set values of name and password as null in state and render the sig in component**

**(note we are not using router for this, use conditional rendering)**

1. **Modify the look of the above form using css (asd given below)**

**Final Output of the above code should be:**



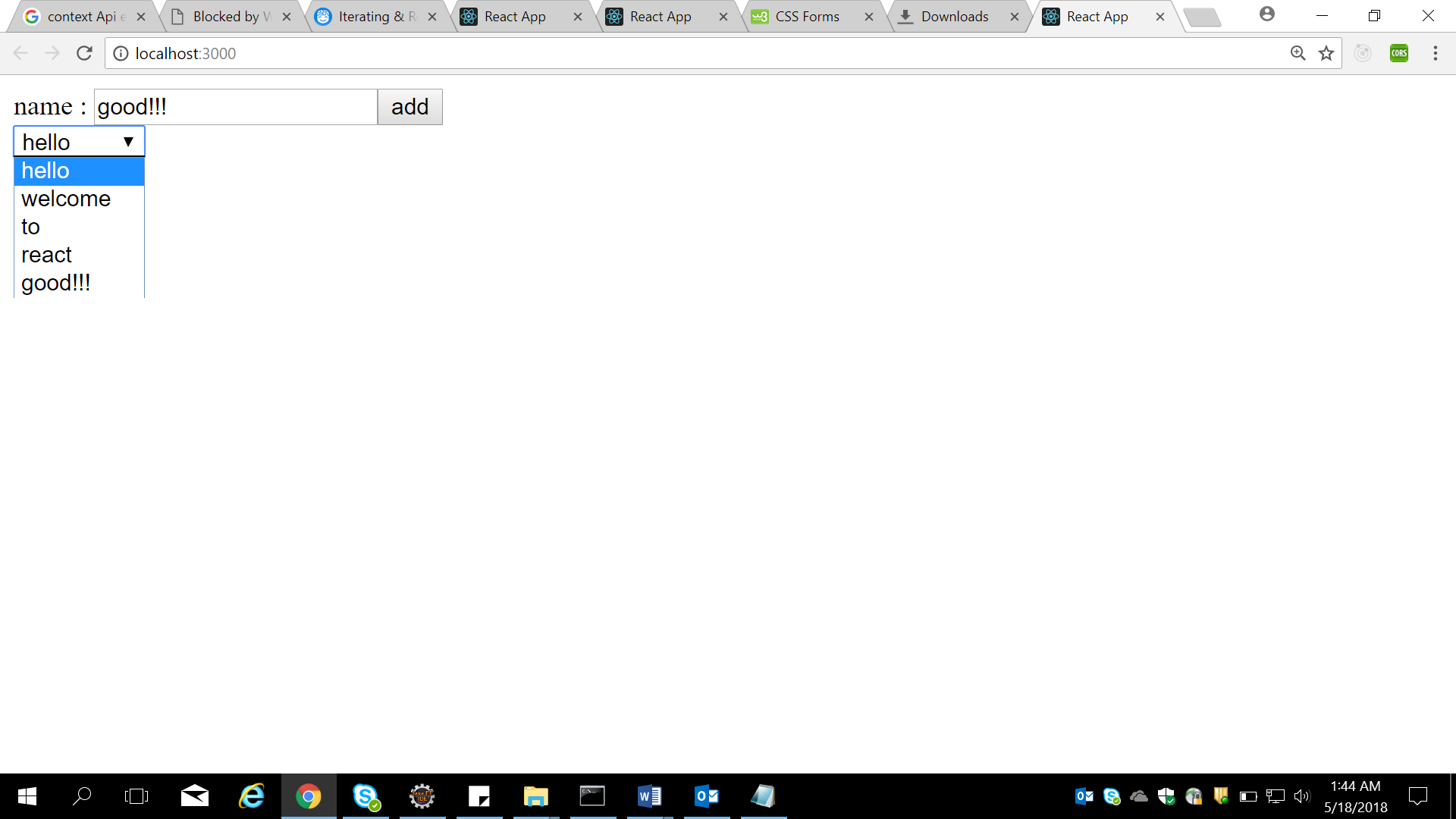
Lab 4.2 :

|  |  |
| --- | --- |
| **Goals** | * Create a react Application using React – dynamic keys to children |
| **Time** | 120 minutes |

**Problem statement-4.2 :**  **Create a React application with name “react-create-dynamic-keys”.**

1. **Create a text box and a button in a js file named FormEg.js**
2. **When user enters input and clicks the button , add the content typed to drop down box on same page**

**Final o/p should look like**



Lab 5. State, Life Cycle, and Events

Lab 5.1 :

|  |  |
| --- | --- |
| **Goals** | * Create a react Application using State |
| **Time** | 90 minutes |

**Problem statement-5.2 :**  **Create a React application with name “react-create-state-clock”.**

**(use statefull component , necessary life cycle methods & props where ever needed)**

1. **Create a ClockTick component in a js and render it through index.js**
2. **The clock has to update for every second and display in browser using state**
3. **When Clock starts it tick the name has to be changed to CAPGEMINI from IGATE.**

Lab 6. React Router

Lab 6.1 :

|  |  |
| --- | --- |
| **Goals** | * Create a Gallery App using react Application using Reacter Router |
| **Time** | 120 minutes |

**Problem statement-6.1 :**  **Create a React application with name “react-create-route-gallery”.**

1. **Create a Application using router for a gallery of mobile application for minimum 3 category of mobiles**
2. **Create 1 component for atleast 3 category, like Samsung ,Mi and Nokia (add needed images in the src folder, minimum 4 )**
3. **Import images in to project and display it using image tag**
4. **Use Navigation.js for creating links to different pages**
5. **Use Main.js for writing the route details**
6. **Finally use App.js to add Navigation and Main then display through index.js**

Lab 6.2:

|  |  |
| --- | --- |
| **Goals** | * Create a react Application using React redux |
| **Time** | 120 minutes |

**Problem statement-6.2 :**  **Create a React application with name “react-create-todo-list”.**

1. **Create a Application using react redux( store dispatcher etc) and implement a to-Do list**
2. **Create a component Form , for Add article , then List.js for printing data in list, use App and index.js as usual print the information**

Lab 7. React Flux, NodeJs and Gulp

Lab 6.1 :

|  |  |
| --- | --- |
| **Goals** | * Create a React App using React flux and gulp |
| **Time** | 140 minutes |

**Problem statement-6.1 :**  **Create a React application with name “react-create-route-gulp-flux”.**

1. **Continue 6.1 or 6.2 application convert it into flux and gulp based program**