MAJOR-1 PROJECT

MID TERM REPORT on

“RESUARCHITECT: IMPLEMENTATION AND DEPLOYMENT OF A RESUME BUILDING WEB APPLICATION USING DEVOPS”

Submitted by:

Name	**Enrollment No.**	**Branch**
Manik Khurana	R110216092	B.Tech CSE CCVT
Babanjot Singh	R110216052	B.Tech CSE CCVT
Ekanshu Dargan	R110216063	B.Tech CSE CCVT
Chhavi Sharma	R110216055	B.Tech CSE CCVT

Under the guidance of:

Mr. Harvinder Singh

Assistant Professor
Department of the Virtualization,
School of Computer Science,
University Of Petroleum and Energy Studies.
Dehradun- 248007

UPES

Approved By

(Mr. Harvinder Singh) (Dr. Deepshikha Bhargava)

Project Guide **Department Head**

School of Computer Science
University of Petroleum & Energy Studies, Dehradun
Synopsis Report (2019-20)

1. Project Title

ResuArchitect - Implementation and Deployment of a resume building web application using DevOps.

2. Abstract

In this rapidly developing world, an ample amount of professionalism is required in every work environment. Therefore, any recruitment process nowadays requires a resume but the candidates aren't familiar with the proper format for building one. Though some people even find a format over web and design a resume for themselves but they miss out on some of the crucial details like alignment, line spacing, bulleting and indentation. A resume is a balance of design and content. Our resume builder is recommended for everyone who lacks the time and design skills to create a professional resume from scratch. The software will ask for the desired information and that particular information will be used accordingly in any of the templates available within the application. This will enable anyone to generate multiple resumes for different purposes appropriately. The application will be built based on DevOps methodology and the backend will be based on node.js and the user interface would be made over react.js.

Keywords: JSON, React.js, Node.js, Jenkins, Docker, AWS , EC2, Elastic Stack, Git

3. Introduction

This project is an application - a future product - that can help any student build and download watermark-less resumes and cover letters. This project keeps in mind the basic mindset of students and how well they will adapt to the new software. Using resuArchitect is effortless and secure and its volatility lies in its functionality and its generic character. The resuArchitect is a generic product in its character as the student needs to input her/his details only once and then these details will be fitted in the desired template as required by the student. The resume can then be easily downloaded from the same page and modified as per the student's need. The Project resuArchitect relies on Node.js' Single Thread, non-blocking I/O Technology that uses Event loop to process requests. The frontend functionality is controlled and moderated by React.js and the basics of website portal creation - HTML and CSS. The version releases and the entire development is based on agile methodologies, taken care using the famous DevOps Tool - Jenkins. The development will hence follow Continuous Integration/Continuous Deployment (CI/CD) model. We are planning to host the entire development cycle on Amazon Web Services (AWS)

- **Backend** - Node.js and NPM
- **Frontend** - HTML5, CSS, React.js
- **DevOps Tool -** Jenkins

The Project uses Semantic Versioning System (Semver) for naming the new releases of resuArchitect.

4. Literature Review

Dunlu PENG†, Lidong CAO, Wenjie XU School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China [1] examined that JSON is a light-weight key-esteem style information ex-evolving group. As we probably are aware, the effectiveness of mapping information between various information models is the key point to improve the presentation of web administration applications.

Naimul Islam Naim IMetropolia University of Applied Sciences,30 May 2017 [2] explored that ReactJS as a good stage to be embraced where there are a few choices to pick from. The basics, center engineering, highlights, information dealing with strategies, prominence,what's more, adoptability were examined in this examination. Despite the fact that there is no limitation to utilize ReactJS over different structures it is prescribed to utilize it as a developing web innovation to be embraced by relying upon the idea of the proposed application to be manufactured.

Kenneth Lewenhagen, Anders Åkesson ."Node.js in Open Source projects on Github." *Blekinge Institute of Technology* [3] performed a study with an aim to provide an insight into how Node.js is used and the Node.js technology adaptation in the open source community. This research displays the diversity of Node.js.

Arpitha R & Mrs. Kavitha S N[4] tested the exhibition of the gadgets or item requires significant investment and needs human help. This can be explained utilizing Continuous Integration (CI) method, it is the most generally utilized procedure by the engineers to incorporate their code into vault, CI is the most proficient and quicker approach to computerize the things. Jenkins is a most effective device which gives constant coordination.

Jurgen Cito, Vincenzo Ferme, and Harald C. Gall, *Dept. of ISE, R. V. College of Engineering, Bengaluru, India.*[5] presents how Docker containers can overcome these issues and aid the reproducibility of research artefacts in software engineering and discusses their applications in the field.

Darshita Kalyani, Dr. Devarshi Mehta [6] examined that Elasticsearch is utilized for use cases like analytics store, auto completer, spell checker, cautioning motor, also, as a universally useful archive store; Full content pursuit is one of it. It is a hearty web index that gives a snappy full content inquiry over different records. It look inside full content fields to discover the report and return the most pertinent outcome first. The pertinence of reports is great as Elasticsearch employments boolean model to discover archive

In his article, “Open-source Scholarship”, Kris Shaffer[7] argues that the open-source software model has lessons to offer the academic community. Here, Kris demonstrates how a scholar can put open-source philosophy into practice using a specific tool developed by and for the community: GitHub.

5. Problem Statement

Majority of the resumes are imperfect and they lack a certain appeal that the hirer of any recruiting company looks in a CV or a resume. This is a major reason for unemployment amongst highly skilled engineers.

6. Objective

The primary objective of building resuArchitect is to empower students to make appealing and attractive resumes and cover letters in the least time possible. Thereby giving them more time to focus on non-trivial matters.

7. Methodology

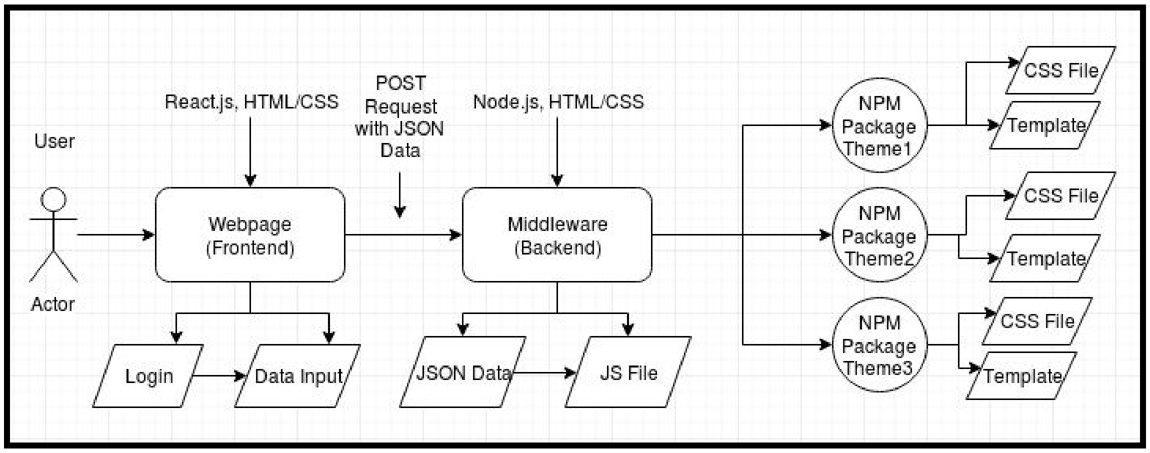


Figure 7.1 Flow Chart of the Entire System

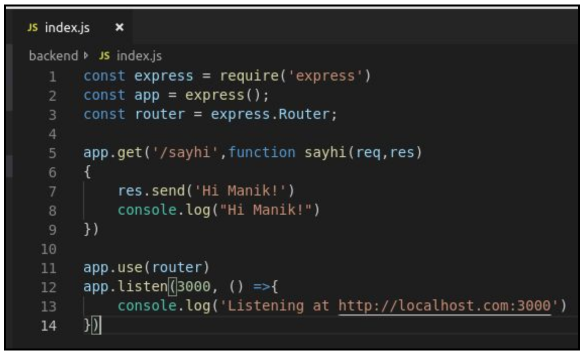
<u>Module 1: Setting up server side application code using Node.js - Express, Sequelize & NPM</u>

Firstly, the server setup will be done using Node.js - Express. This server will enable us to take in all the API Calls from the frontend. We will here encode and expose different functionalities as callback functions to GET/POST Calls via various routes setup. Every time an API is hit, the user will be getting an appropriate response and his/her resume file with all the details filled.

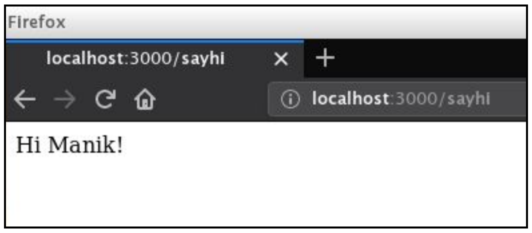
Whenever the user clicks on the download button for any particular resume format, the server will receive the user's details in JSON format in a POST Request. The details will then be sent to the callback function in the request itself and this callback function will then take in all the user details and insert it into the desired template. After insertion is done the resume will be sent to the user after automated testing (at Node end via mocha) via sendFile method.

The backend will also use Sequelize ORM to map the user details into the desired database.For now the dialect will be set to MongoDB as it is really efficient to store JSON Data for it is a NoSQL DB. The database will be used to fetch the User Authentication details and their other details in JSON format as stored.

Here is a snapshot of the basic server setup code and a browser snapshot that shows the output.

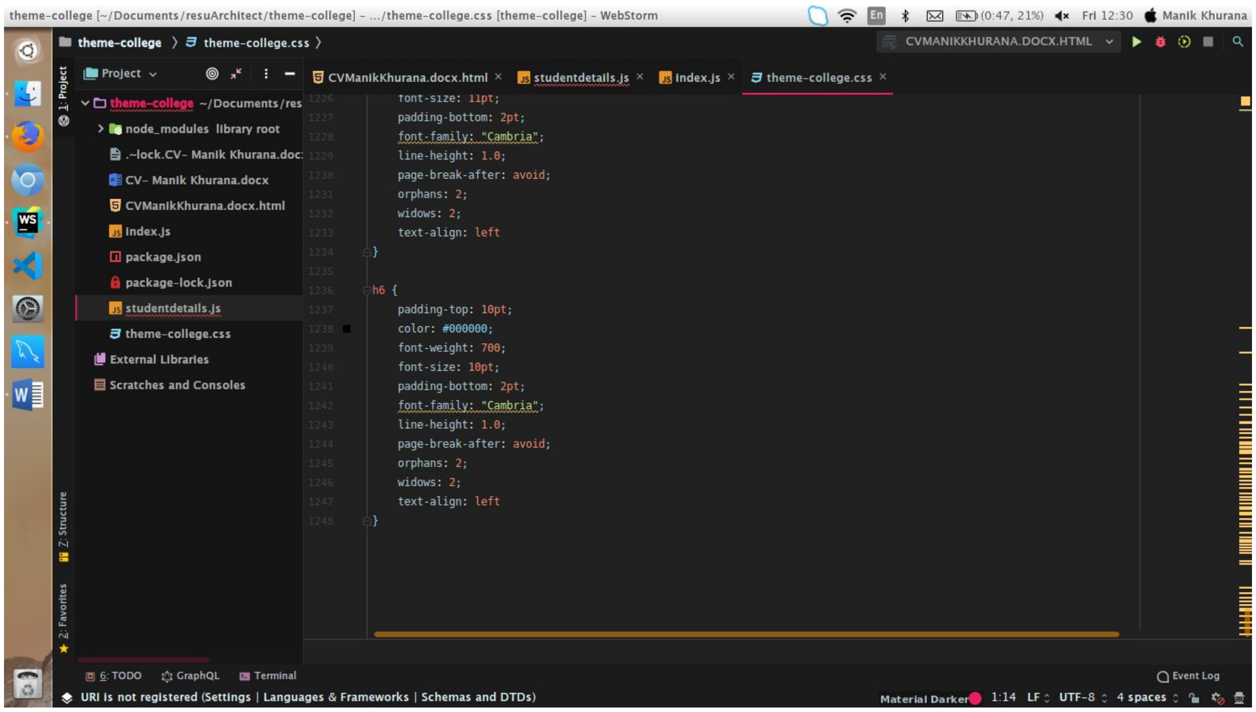


```
const express = require('express')
const app = express();
const router = express.Router;

app.get('/sayhi',function sayhi(req,res)
{
    res.send('Hi Manik!')
    console.log("Hi Manik!")
})

app.use(router)
app.listen(3000, () =>{
    console.log('Listening at http://localhost.com:3000')
})
```



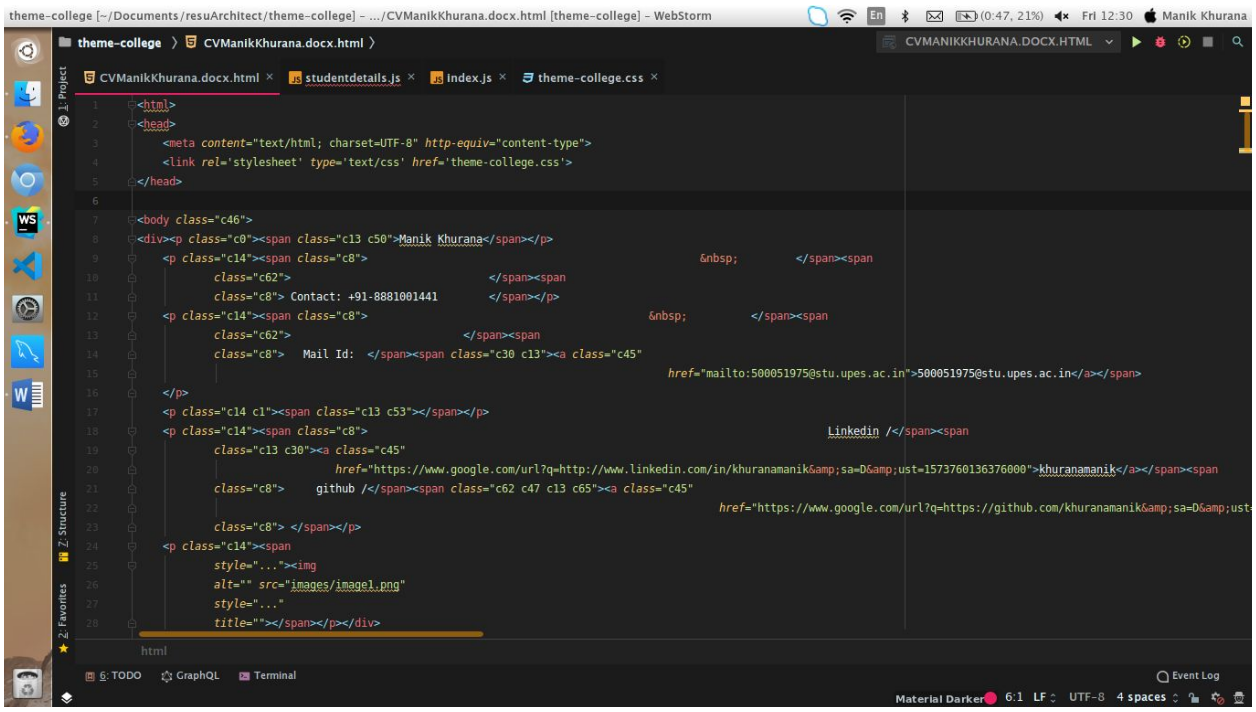
Hi Manik!

Figure 7.1.1 Basic Express Server code snapshot Figure 7.1.2 Logging into the PORT



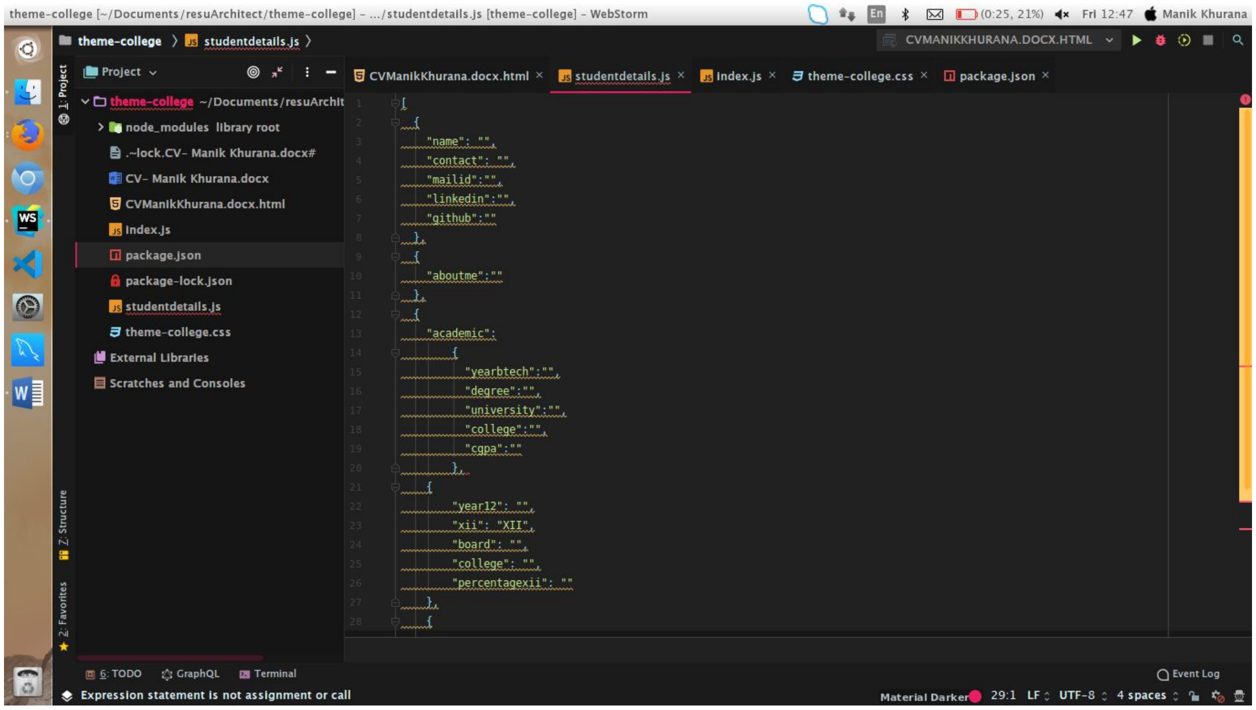
```
    font-size: 11pt;
    padding-bottom: 2pt;
    font-family: "Cambria";
    line-height: 1.0;
    page-break-after: avoid;
    orphans: 2;
    widows: 2;
    text-align: left
}

h6 {
    padding-top: 10pt;
    color: #000000;
    font-weight: 700;
    font-size: 10pt;
    padding-bottom: 2pt;
    font-family: "Cambria";
    line-height: 1.0;
    page-break-after: avoid;
    orphans: 2;
    widows: 2;
    text-align: left
}
```

Figure 7.1.3 The theme-college.css file



```
<html>
<head>
    <meta content="text/html; charset=UTF-8" http-equiv="content-type">
    <link rel='stylesheet' type='text/css' href='theme-college.css'>
</head>

<body class="c46">
<div><p class="c0"><span class="c13 c50">Manik Khurana</span></p>
    <p class="c14"><span class="c8">                                             &nbsp;         </span><span
            class="c62">                              </span><span
            class="c8"> Contact: +91-8881001441        </span></p>
    <p class="c14"><span class="c8">                                     &nbsp;          </span><span
            class="c62">                          </span><span
            class="c8">   Mail Id:  </span><span class="c30 c13"><a class="c45"
                                                                   href="mailto:500051975@stu.upes.ac.in">500051975@stu.upes.ac.in</a></span>
    </p>
    <p class="c14 c1"><span class="c13 c53"></span></p>
    <p class="c14"><span class="c8">                                                                Linkedin /</span><span
            class="c13 c30"><a class="c45"
                       href="https://www.google.com/url?q=http://www.linkedin.com/in/khuranamanik&amp;sa=D&amp;ust=1573760136376000">khuranamanik</a></span><span
            class="c8">     github /</span><span class="c62 c47 c13 c65"><a class="c45"
                                                                          href="https://www.google.com/url?q=https://github.com/khuranamanik&amp;sa=D&amp;ust
            class="c8"> </span></p>
    <p class="c14"><span
            style="..."><img
            alt="" src="images/image1.png"
            style="..."
            title=""></span></p></div>
```

Figure 7.1.4 The HTML File of the CV



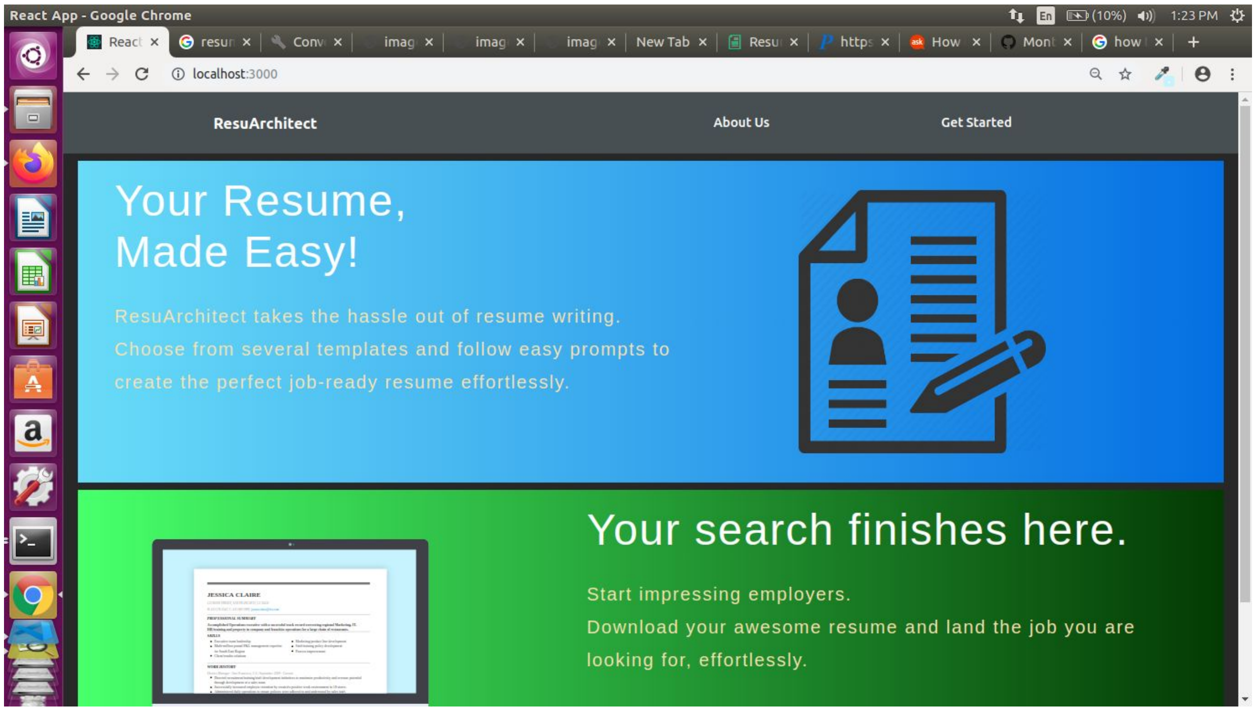
```
[
  {
    "name": "",
    "contact": "",
    "mailid":"",
    "linkedin":"",
    "github":""
  },
  {
    "aboutme":""
  },
  {
    "academic":
          {
            "yearbtech":"",
            "degree":"",
            "university":"",
            "college":"",
            "cgpa":""
          },
      {
          "year12": "",
          "xii": "XII",
          "board": "",
          "college": "",
          "percentagexii": ""
      },
      {
```

Figure 7.1.5 The JSON File used for communication

Module 2: Setting up client side application code using React.js, Redux and HTML/CSS

This is where we build the frontend part of the project. This is where the user will be required to Authenticate himself/herself via various options of Logging in via Google, Linkedin and simply via Email and Password. The user will then be asked various details about his/her Education, Work Experience, etc. These are the details that go into the resume straightaway. In the Downloads option the user will be able to see various themes. The user can then download the resume of his/her choice . When the user will press the download button for a particular theme, his details will be sent in JSON format to the backend API Gateway URL specified as the endpoint for the particular theme.

This way the users can enter their details only once and download any resume format, any number of times. User details will be stored in Database


React App - Google Chrome

localhost:3000

ResuArchitect | About Us | Get Started

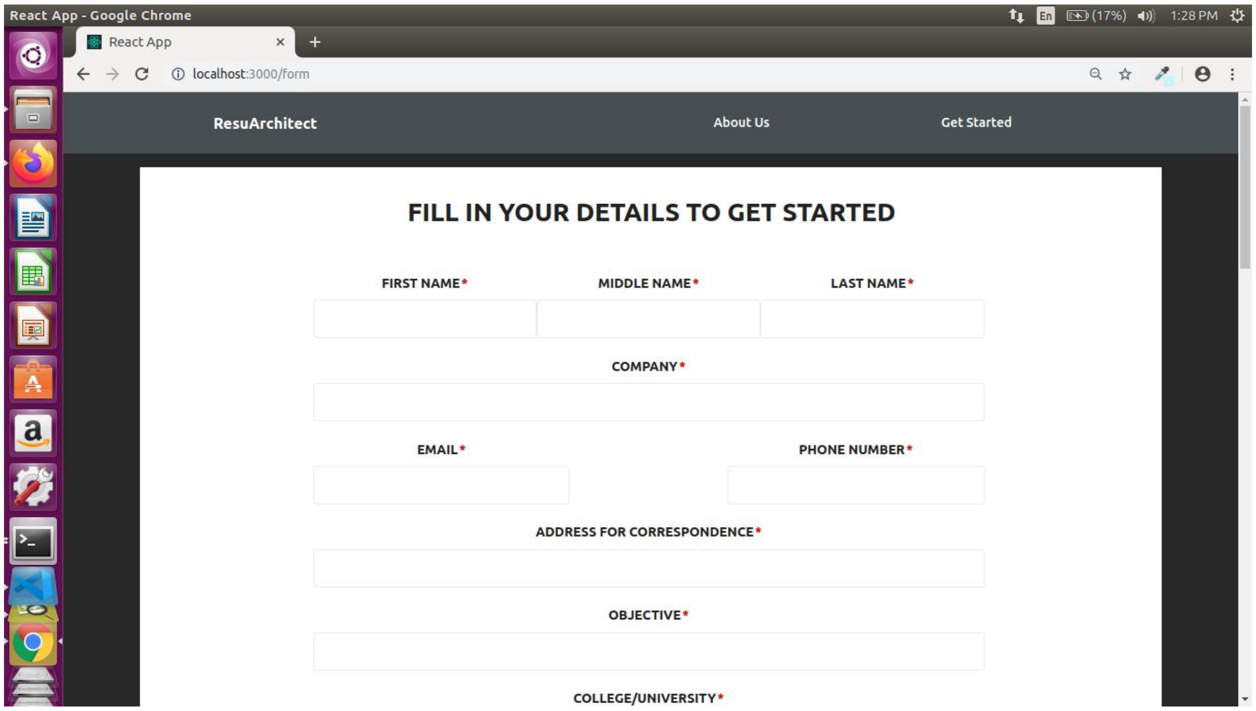
Your Resume,
Made Easy!

ResuArchitect takes the hassle out of resume writing.
Choose from several templates and follow easy prompts to create the perfect job-ready resume effortlessly.

Your search finishes here.

Start impressing employers.
Download your awesome resume and land the job you are looking for, effortlessly.

Figure 7.2.1 Main Homepage


React App - Google Chrome

localhost:3000/form

ResuArchitect | About Us | Get Started

FILL IN YOUR DETAILS TO GET STARTED

FIRST NAME* | MIDDLE NAME* | LAST NAME*

COMPANY*

EMAIL* | PHONE NUMBER*

ADDRESS FOR CORRESPONDENCE*

OBJECTIVE*

COLLEGE/UNIVERSITY*

Figure 7.2.2 Form to input data

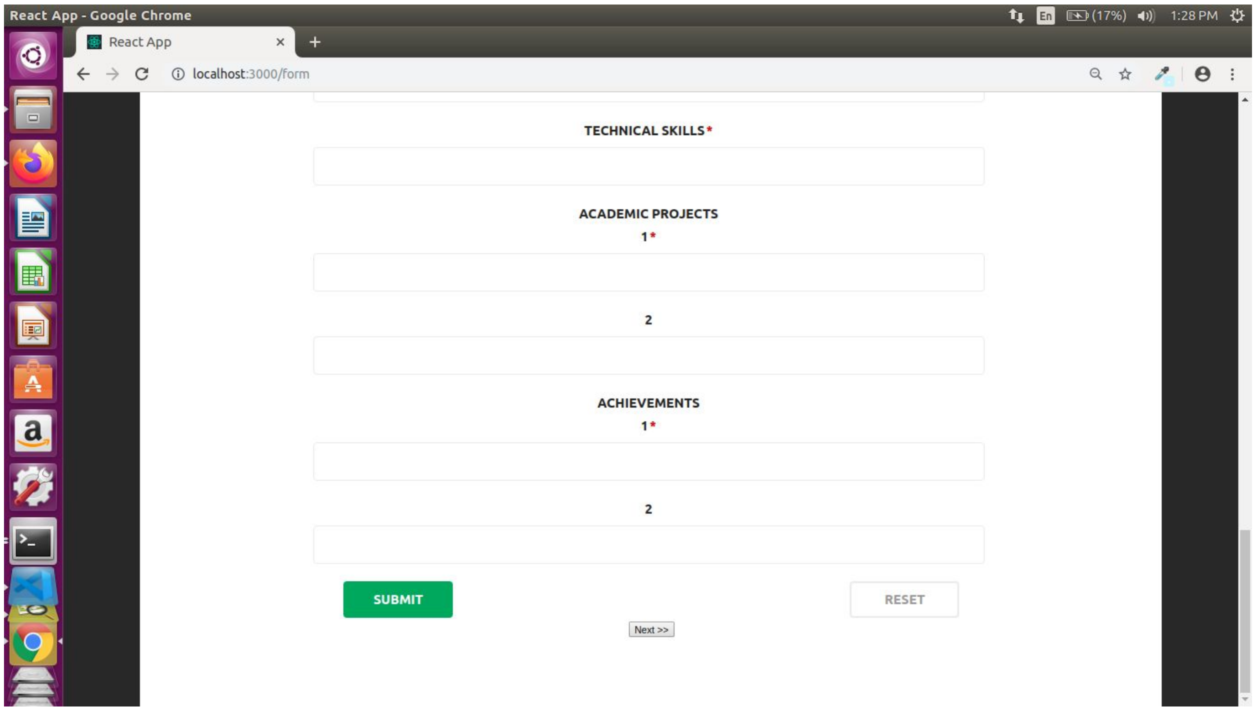
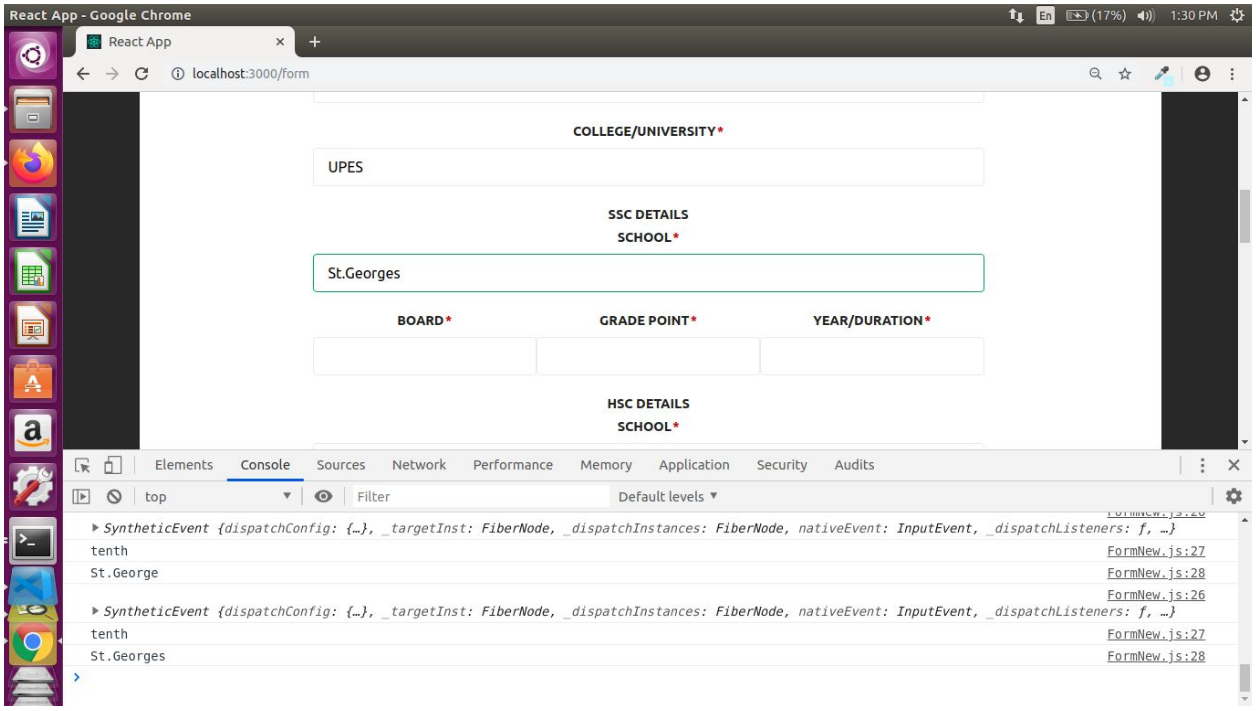


Figure 7.2.3 Submit and Reset Features



COLLEGE/UNIVERSITY*

UPES

SSC DETAILS

SCHOOL*

St.Georges

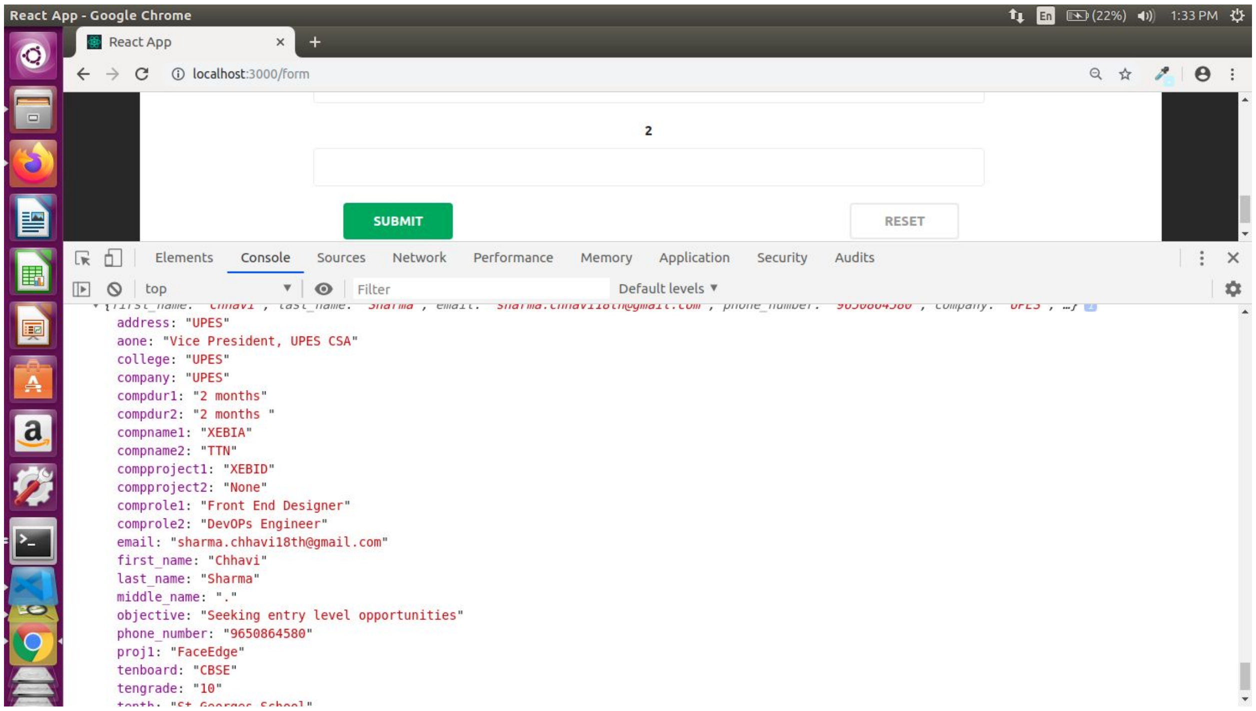
BOARD* | GRADE POINT* | YEAR/DURATION*

HSC DETAILS

SCHOOL*

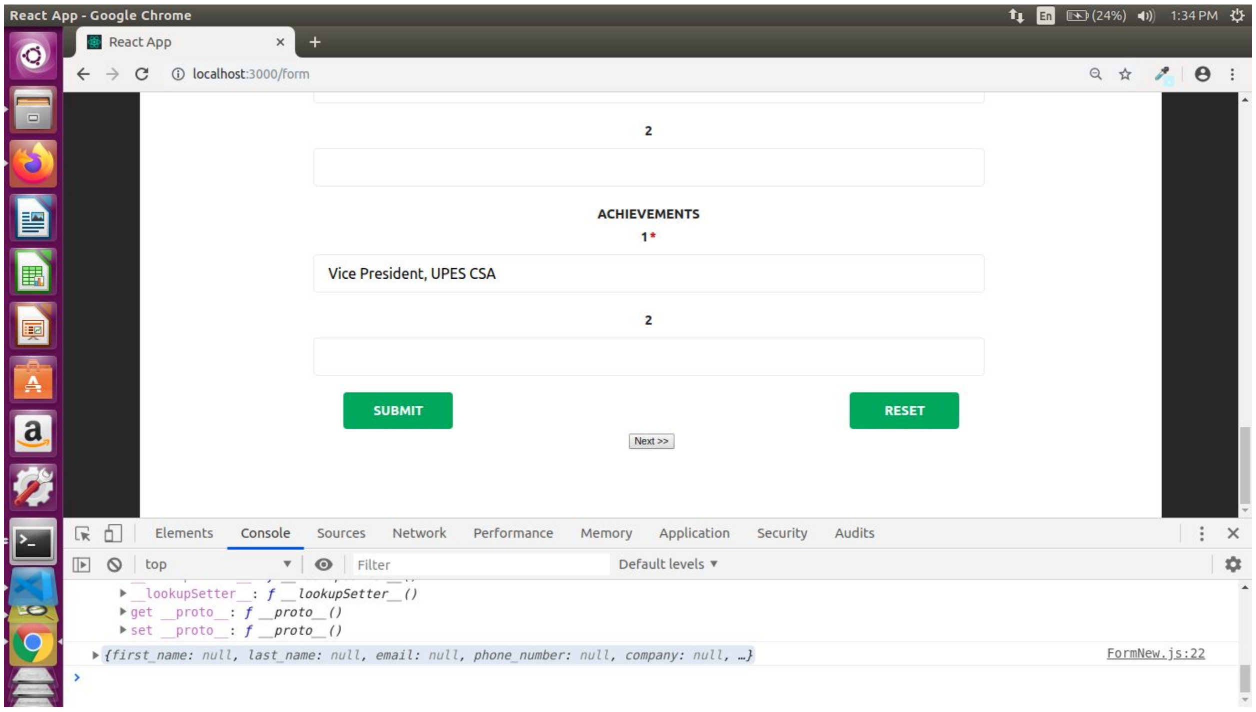
```
▸ SyntheticEvent {dispatchConfig: {…}, _targetInst: FiberNode, _dispatchInstances: FiberNode, nativeEvent: InputEvent, _dispatchListeners: f, …}
tenth                                                                                         FormNew.js:27
St.George                                                                                     FormNew.js:28
                                                                                              FormNew.js:26
▸ SyntheticEvent {dispatchConfig: {…}, _targetInst: FiberNode, _dispatchInstances: FiberNode, nativeEvent: InputEvent, _dispatchListeners: f, …}
tenth                                                                                         FormNew.js:27
St.Georges                                                                                    FormNew.js:28
>
```

Figure 7.2.4 Console View



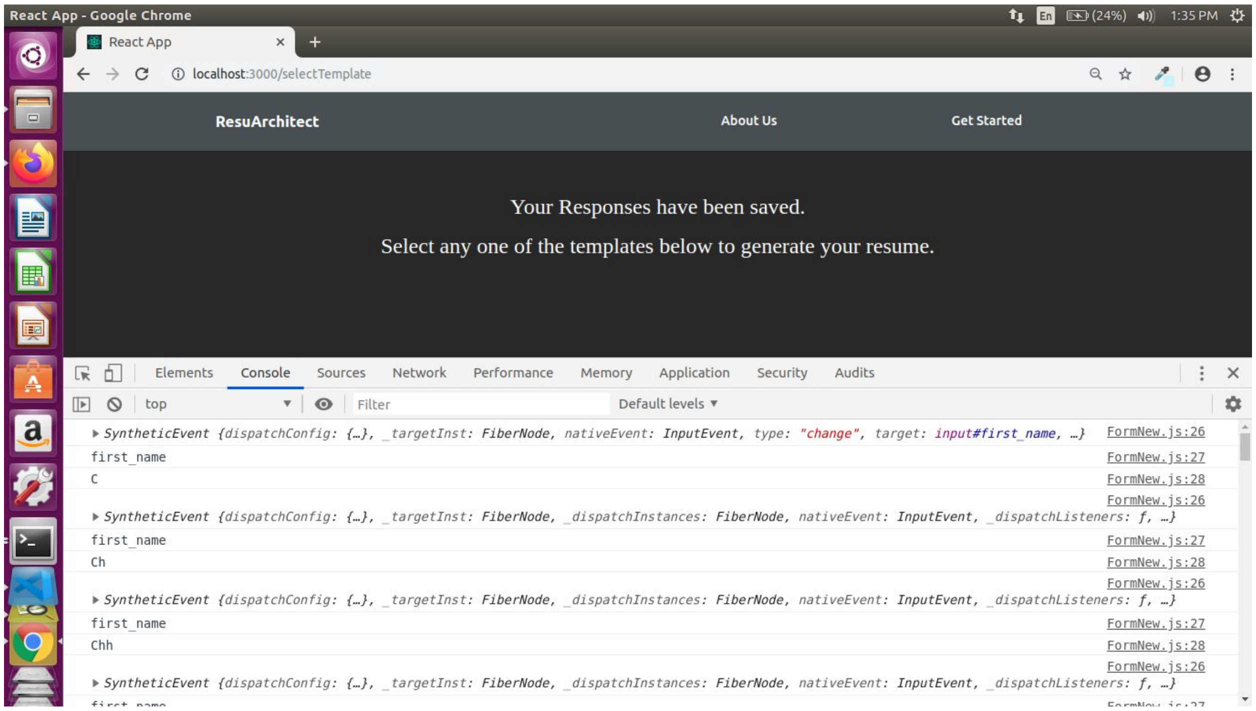
```
address: "UPES"
aone: "Vice President, UPES CSA"
college: "UPES"
company: "UPES"
compdur1: "2 months"
compdur2: "2 months "
compname1: "XEBIA"
compname2: "TTN"
compproject1: "XEBID"
compproject2: "None"
comprole1: "Front End Designer"
comprole2: "DevOPs Engineer"
email: "sharma.chhavi18th@gmail.com"
first_name: "Chhavi"
last_name: "Sharma"
middle_name: "."
objective: "Seeking entry level opportunities"
phone_number: "9650864580"
proj1: "FaceEdge"
tenboard: "CBSE"
tengrade: "10"
```

Figure 7.2.5 JSON Object Creation



```
    ▸ __lookupSetter__: ƒ __lookupSetter__()
    ▸ get __proto__: ƒ __proto__()
    ▸ set __proto__: ƒ __proto__()
▸ {first_name: null, last_name: null, email: null, phone_number: null, company: null, …}        FormNew.js:22
>
```

Figure 7.2.6 Reset Feature

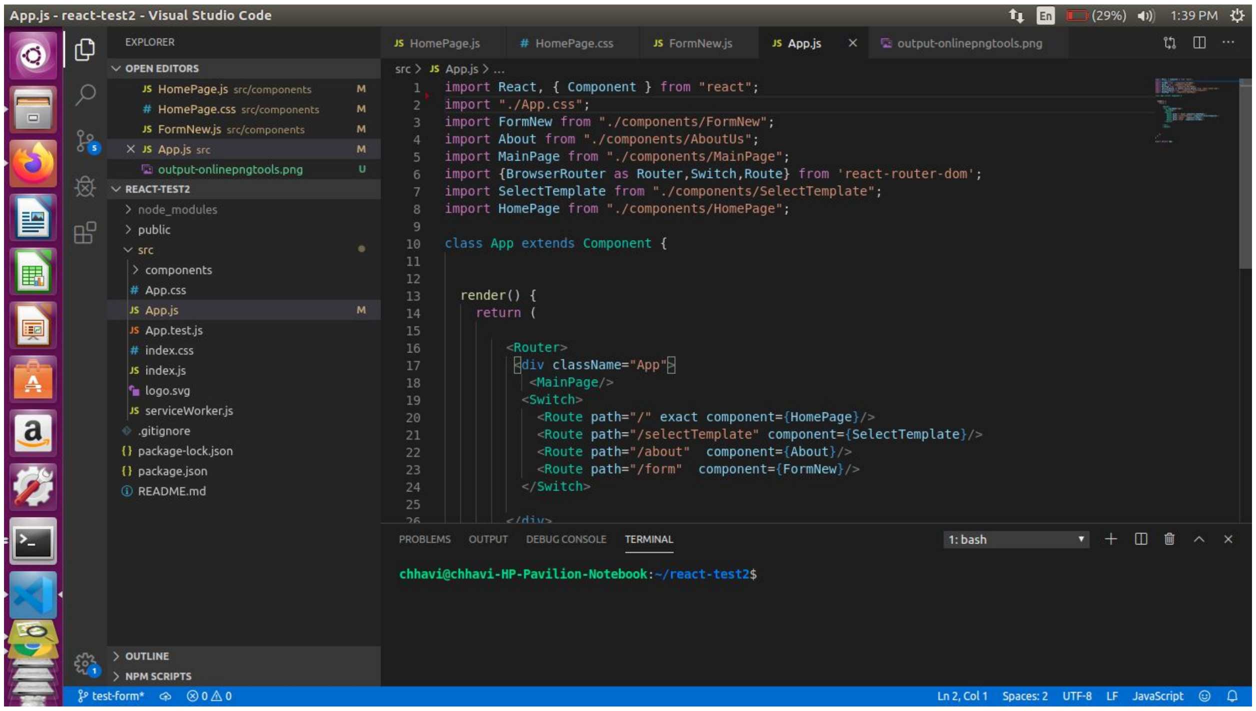


Your Responses have been saved.

Select any one of the templates below to generate your resume.

```
▶ SyntheticEvent {dispatchConfig: {…}, _targetInst: FiberNode, nativeEvent: InputEvent, type: "change", target: input#first_name, …}   FormNew.js:26
first_name                                                                                                                        FormNew.js:27
C                                                                                                                                 FormNew.js:28
                                                                                                                                  FormNew.js:26
▶ SyntheticEvent {dispatchConfig: {…}, _targetInst: FiberNode, _dispatchInstances: FiberNode, nativeEvent: InputEvent, _dispatchListeners: f, …}
first_name                                                                                                                        FormNew.js:27
Ch                                                                                                                                FormNew.js:28
                                                                                                                                  FormNew.js:26
▶ SyntheticEvent {dispatchConfig: {…}, _targetInst: FiberNode, _dispatchInstances: FiberNode, nativeEvent: InputEvent, _dispatchListeners: f, …}
first_name                                                                                                                        FormNew.js:27
Chh                                                                                                                               FormNew.js:28
                                                                                                                                  FormNew.js:26
▶ SyntheticEvent {dispatchConfig: {…}, _targetInst: FiberNode, _dispatchInstances: FiberNode, nativeEvent: InputEvent, _dispatchListeners: f, …}
```

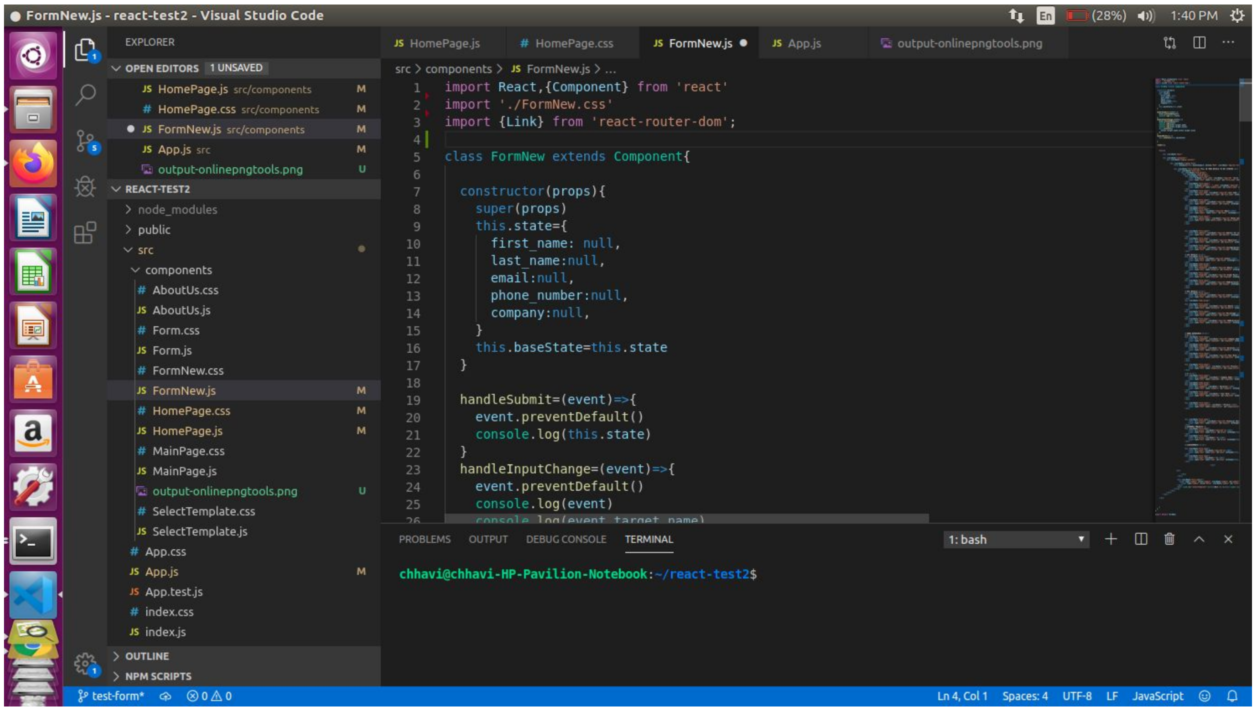
Figure 7.2.7 Responses



```
import React, { Component } from "react";
import "./App.css";
import FormNew from "./components/FormNew";
import About from "./components/AboutUs";
import MainPage from "./components/MainPage";
import {BrowserRouter as Router,Switch,Route} from 'react-router-dom';
import SelectTemplate from "./components/SelectTemplate";
import HomePage from "./components/HomePage";

class App extends Component {


  render() {
    return (

        <Router>
         <div className="App">
          <MainPage/>
         <Switch>
           <Route path="/" exact component={HomePage}/>
           <Route path="/selectTemplate" component={SelectTemplate}/>
           <Route path="/about"  component={About}/>
           <Route path="/form"  component={FormNew}/>
         </Switch>

        </div>
```

```
chhavi@chhavi-HP-Pavilion-Notebook:~/react-test2$
```

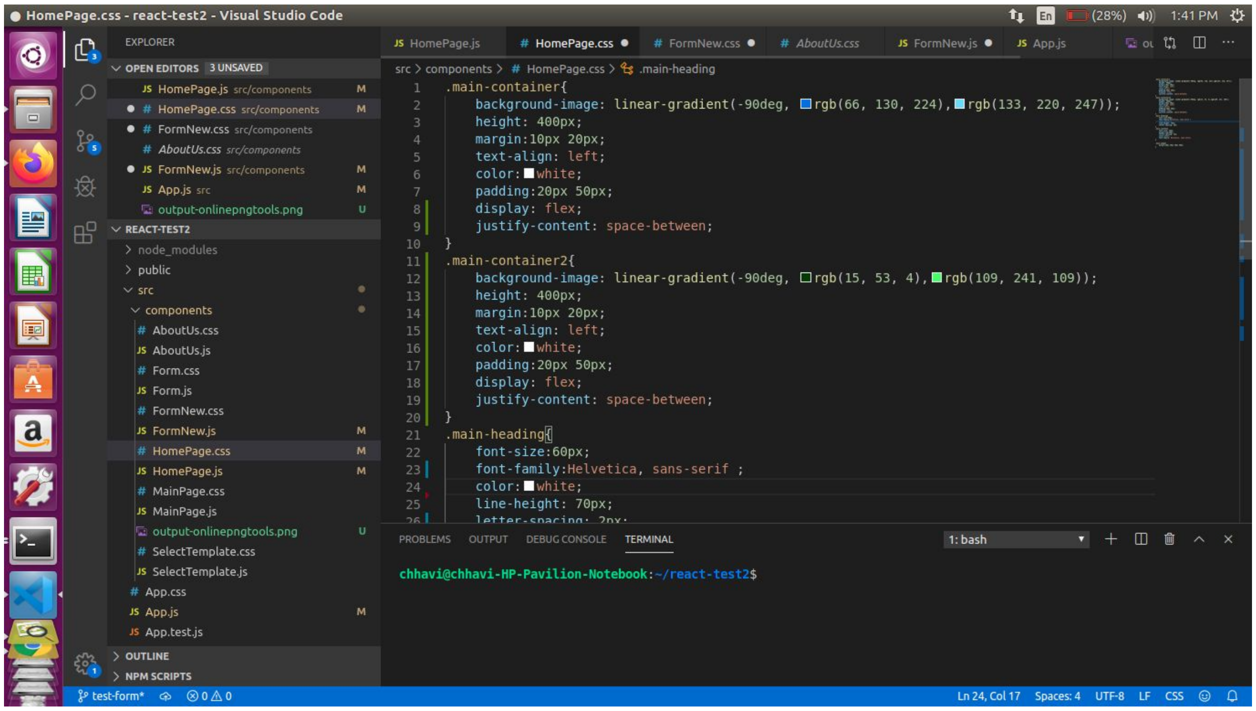
Figure 7.2.8 App.js


FormNew.js - react-test2 - Visual Studio Code
EXPLORER
OPEN EDITORS 1 UNSAVED
HomePage.js src/components
HomePage.css src/components
FormNew.js src/components
App.js src
output-onlinepngtools.png
REACT-TEST2
node_modules
public
src
components
AboutUs.css
AboutUs.js
Form.css
Form.js
FormNew.css
FormNew.js
HomePage.css
HomePage.js
MainPage.css
MainPage.js
output-onlinepngtools.png
SelectTemplate.css
SelectTemplate.js
App.css
App.js
App.test.js
index.css
index.js
OUTLINE
NPM SCRIPTS
src > components > FormNew.js > ...
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: bash
test-form* Ln 4, Col 1 Spaces: 4 UTF-8 LF JavaScript

```
import React,{Component} from 'react'
import './FormNew.css'
import {Link} from 'react-router-dom';

class FormNew extends Component{

  constructor(props){
    super(props)
    this.state={
      first_name: null,
      last_name:null,
      email:null,
      phone_number:null,
      company:null,
    }
    this.baseState=this.state
  }

  handleSubmit=(event)=>{
    event.preventDefault()
    console.log(this.state)
  }
  handleInputChange=(event)=>{
    event.preventDefault()
    console.log(event)
    console.log(event.target.name)
```

```
chhavi@chhavi-HP-Pavilion-Notebook:~/react-test2$
```

Figure 7.2.9 Form.js file


HomePage.css - react-test2 - Visual Studio Code
EXPLORER
OPEN EDITORS 3 UNSAVED
HomePage.js src/components
HomePage.css src/components
FormNew.css src/components
AboutUs.css src/components
FormNew.js src/components
App.js src
output-onlinepngtools.png
REACT-TEST2
node_modules
public
src
components
AboutUs.css
AboutUs.js
Form.css
Form.js
FormNew.css
FormNew.js
HomePage.css
HomePage.js
MainPage.css
MainPage.js
output-onlinepngtools.png
SelectTemplate.css
SelectTemplate.js
App.css
App.js
App.test.js
OUTLINE
NPM SCRIPTS
src > components > # HomePage.css > .main-heading
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
1: bash
test-form* Ln 24, Col 17 Spaces: 4 UTF-8 LF CSS

```
.main-container{
    background-image: linear-gradient(-90deg, rgb(66, 130, 224),rgb(133, 220, 247));
    height: 400px;
    margin:10px 20px;
    text-align: left;
    color:white;
    padding:20px 50px;
    display: flex;
    justify-content: space-between;
}
.main-container2{
    background-image: linear-gradient(-90deg, rgb(15, 53, 4),rgb(109, 241, 109));
    height: 400px;
    margin:10px 20px;
    text-align: left;
    color:white;
    padding:20px 50px;
    display: flex;
    justify-content: space-between;
}
.main-heading{
    font-size:60px;
    font-family:Helvetica, sans-serif ;
    color:white;
    line-height: 70px;
    letter-spacing: 2px;
```

```
chhavi@chhavi-HP-Pavilion-Notebook:~/react-test2$
```

Figure 7.2.10 CSS File

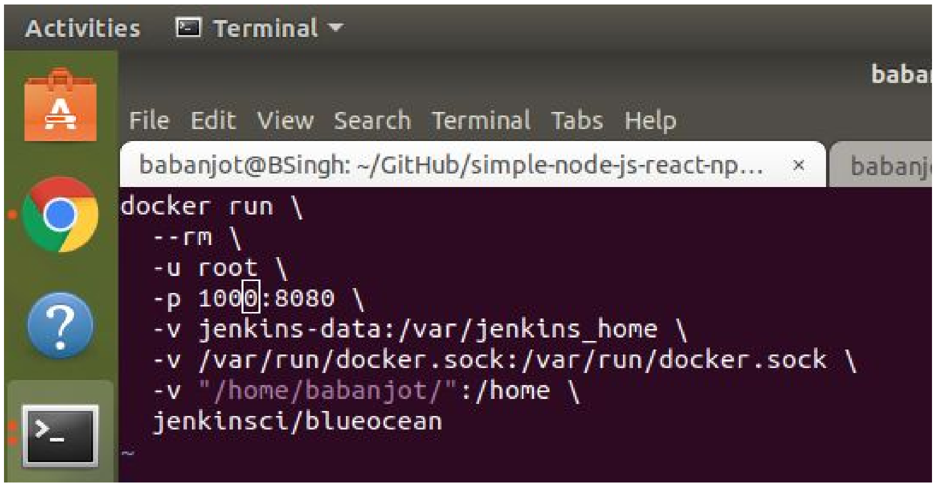
Module 3: Automation of release and versioning control using Jenkins and Github

In this module we will implement the building of our web application using Jenkins in Docker. We will pull the Node.js and React.js codes from our GitHub repository which will be accompanied by a test to check that the application renders satisfactorily.We will define the ports on which our web application and jenkins will run. Following steps will be followed :-

1. **Run Jenkins on Docker:** We will run Jenkins as a Docker container from the jenkinsci/blueocean Docker image.
2. **Accessing the Jenkins/Blue Ocean Docker container:** Here we will access Jenkins and start the wizard on local host to start it up and do further build operations. We will add the plugins that will be required and suggested and will create the first administrator user. After this we will save and finish.
3. **Fork and clone our repository on GitHub:** In this we will clone our GitHub repository by signing in github and then we will pull the source code of our web application.
4. **Create Pipeline project in Jenkins:** Now we will create our Pipeline that will automate building your Node.js and React application in Jenkins. This Pipeline will be created as a Jenkinsfile, which will be committed to locally cloned Git repository and also it will create a docker image for our Node.js and React.js
5. **Defining Stages in Jenkins:** Here will create all the stages for our pipeline such as Build,Test and Deliver in which all the steps will be covered for the continuous integration to delivery.

Steps to Create Initial Pipeline as JenkinsFile

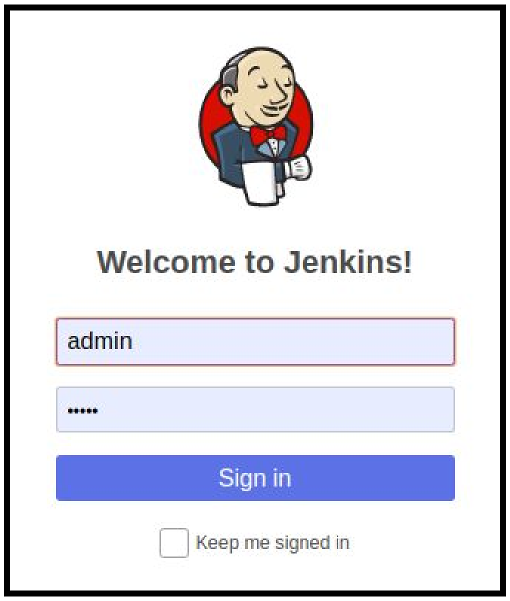
Step 1: A docker file to pull the jenkins container from dockerhub repository


Activities
Terminal
File Edit View Search Terminal Tabs Help
babanjot@BSingh: ~/GitHub/simple-node-js-react-np...

```
docker run \
  --rm \
  -u root \
  -p 1000:8080 \
  -v jenkins-data:/var/jenkins_home \
  -v /var/run/docker.sock:/var/run/docker.sock \
  -v "/home/babanjot/":/home \
  jenkinsci/blueocean
~
```

Figure 7.3.1 Docker File

Step 2: Running Docker container and setting up Jenkins on a local host machine



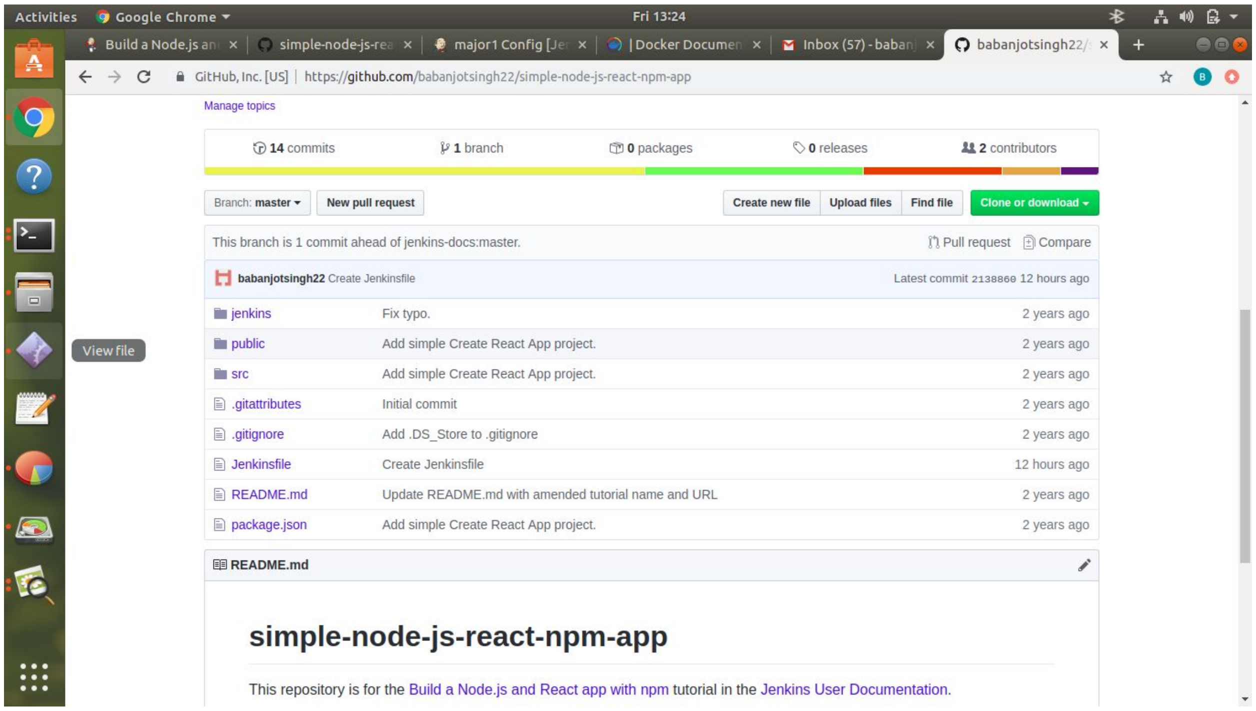
Welcome to Jenkins!

admin

•••••

Figure 7.3.2 Welcome to Jenkins

Step 3: Fork and then Clone the sample repository on GitHub Account



GitHub, Inc. [US] | https://github.com/babanjotsingh22/simple-node-js-react-npm-app

This branch is 1 commit ahead of jenkins-docs:master.

babanjotsingh22 Create Jenkinsfile — Latest commit 2138860 12 hours ago

jenkins	Fix typo.	2 years ago
public	Add simple Create React App project.	2 years ago
src	Add simple Create React App project.	2 years ago
.gitattributes	Initial commit	2 years ago
.gitignore	Add .DS_Store to .gitignore	2 years ago
Jenkinsfile	Create Jenkinsfile	12 hours ago
README.md	Update README.md with amended tutorial name and URL	2 years ago
package.json	Add simple Create React App project.	2 years ago

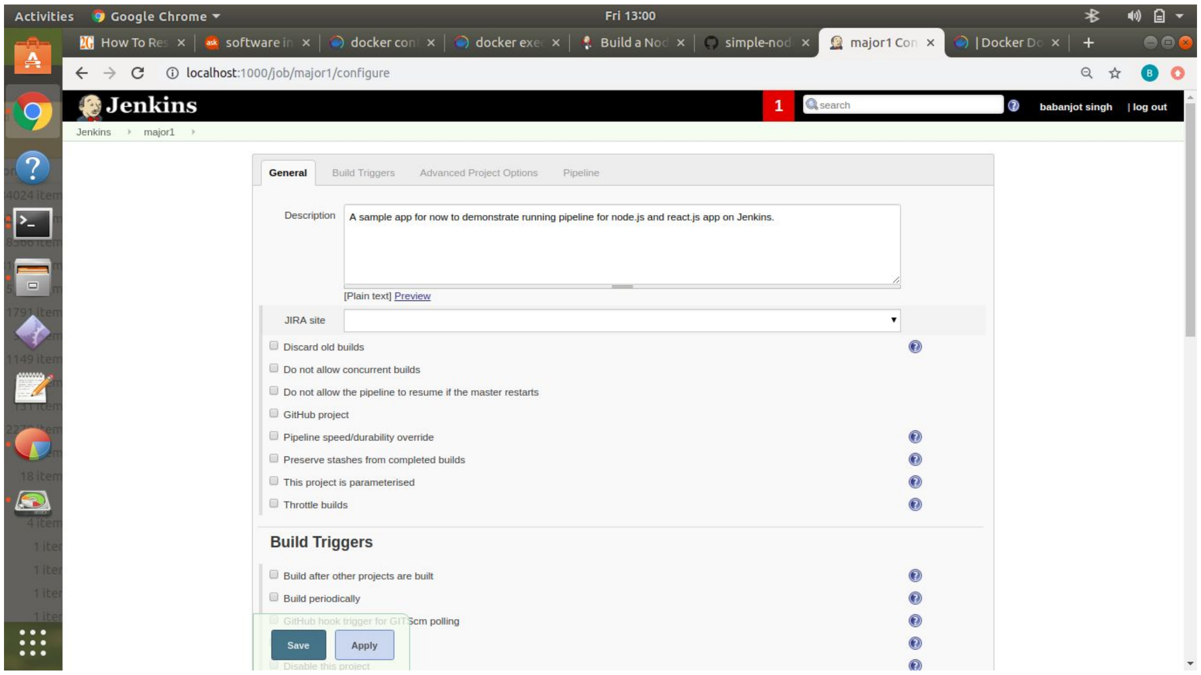
README.md

simple-node-js-react-npm-app

This repository is for the Build a Node.js and React app with npm tutorial in the Jenkins User Documentation.

Figure 7.3.3 Cloning GitHub

Step 4: An Entry level Pipeline demonstrating How to use Jenkins to build a simple node.js and react.js application with NPM



localhost:1000/job/major1/configure

Description: A sample app for now to demonstrate running pipeline for node.js and react.js app on Jenkins.

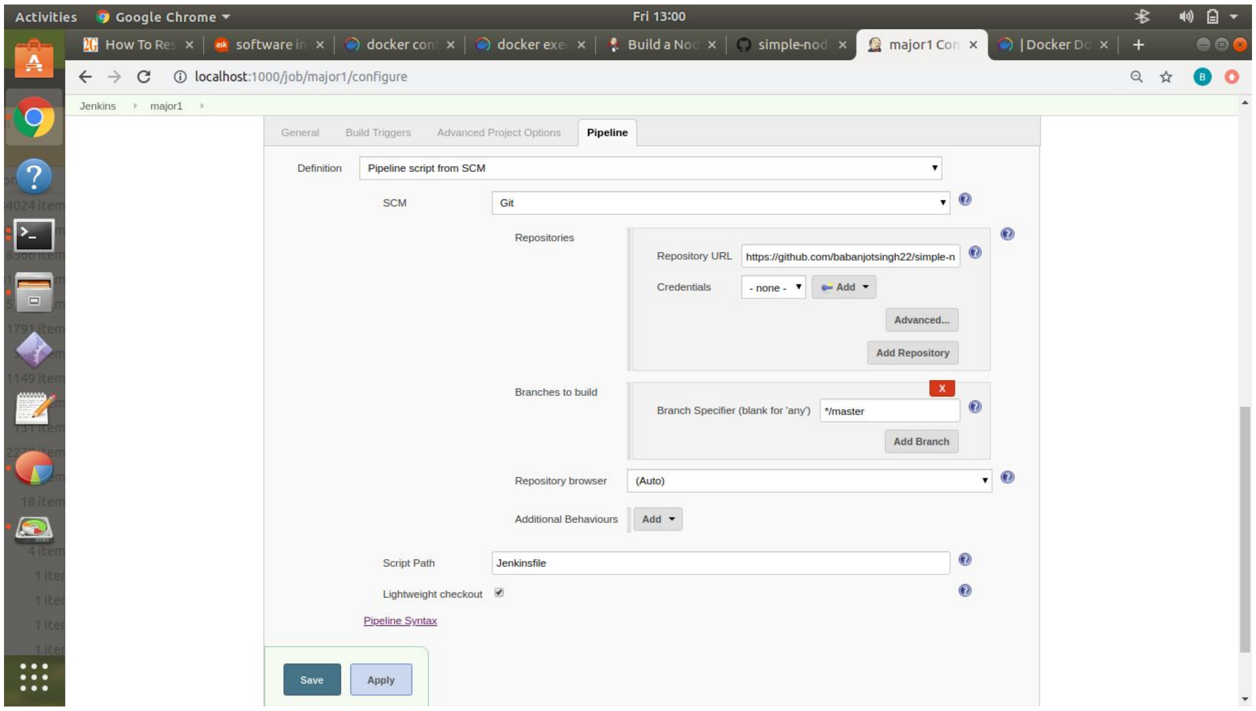
JIRA site

- Discard old builds
- Do not allow concurrent builds
- Do not allow the pipeline to resume if the master restarts
- GitHub project
- Pipeline speed/durability override
- Preserve stashes from completed builds
- This project is parameterised
- Throttle builds

Build Triggers

- Build after other projects are built
- Build periodically
- GitHub hook trigger for GITScm polling

Figure 7.3.4.1 Pipeline in Jenkins



localhost:1000/job/major1/configure

Definition: Pipeline script from SCM

SCM: Git

Repositories

Repository URL: https://github.com/babanjotsingh22/simple-n

Credentials: - none -

Branches to build

Branch Specifier (blank for 'any'): */master

Repository browser: (Auto)

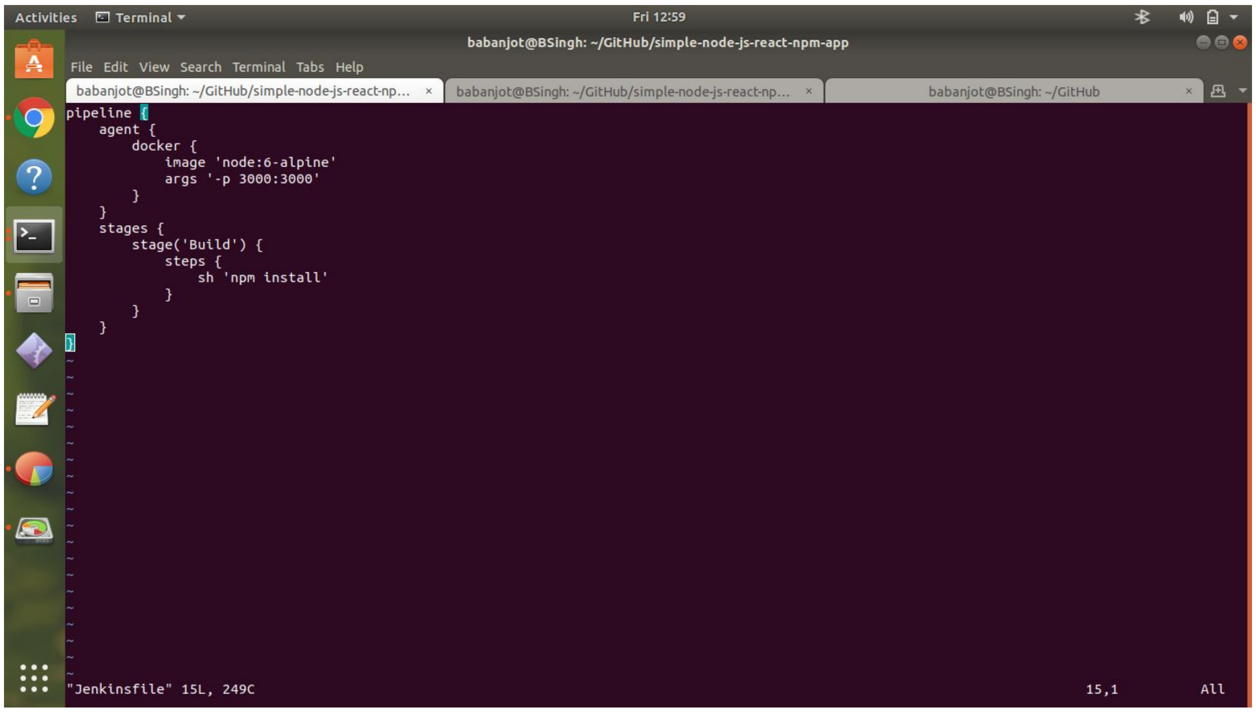
Additional Behaviours

Script Path: Jenkinsfile

Lightweight checkout ✓

Figure 7.3.4.2 Choosing Pipeline Script from SCM

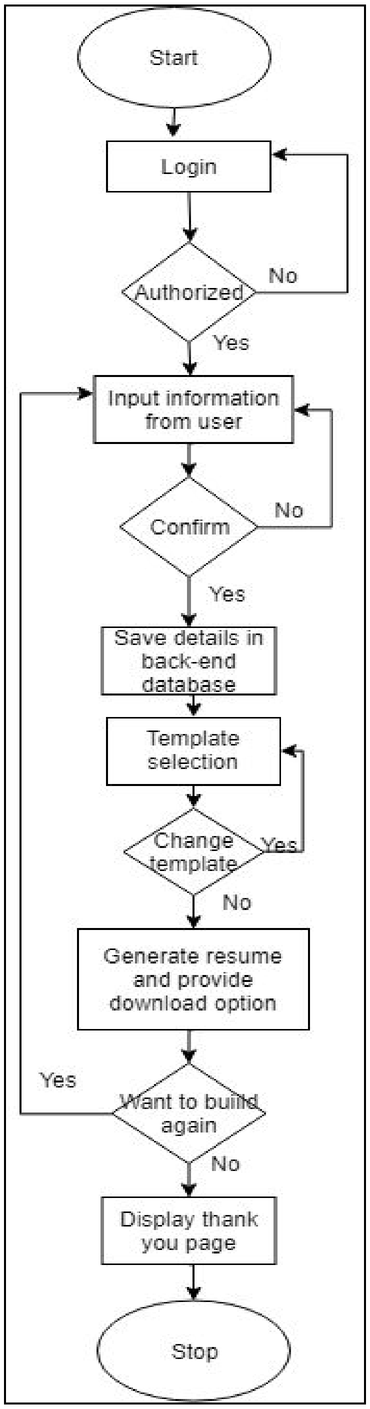
Step 5: Creating Initial Pipeline as JenkinsFile


Activities
Terminal
Fri 12:59
babanjot@BSingh: ~/GitHub/simple-node-js-react-npm-app
File Edit View Search Terminal Tabs Help
babanjot@BSingh: ~/GitHub/simple-node-js-react-np...
babanjot@BSingh: ~/GitHub/simple-node-js-react-np...
babanjot@BSingh: ~/GitHub

```
pipeline {
    agent {
        docker {
            image 'node:6-alpine'
            args '-p 3000:3000'
        }
    }
    stages {
        stage('Build') {
            steps {
                sh 'npm install'
            }
        }
    }
}
"Jenkinsfile" 15L, 249C                                   15,1          All
```

Figure 7.3.5 JenkinsFile

8. Design



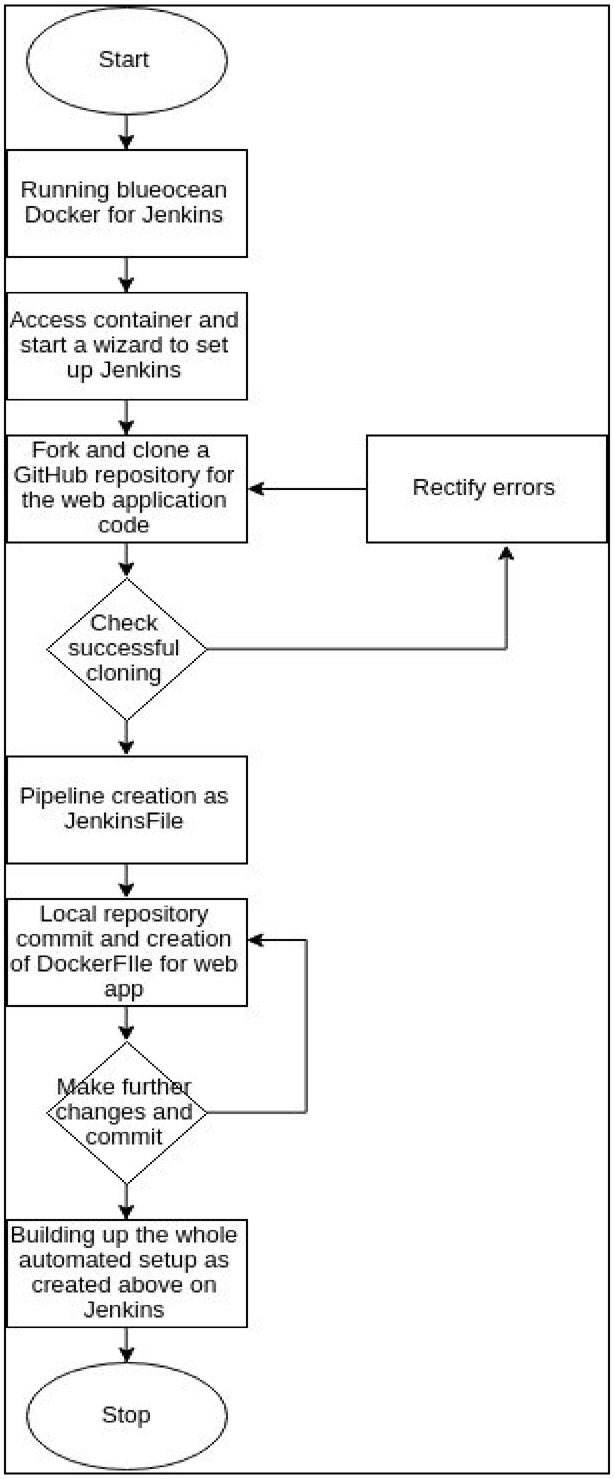
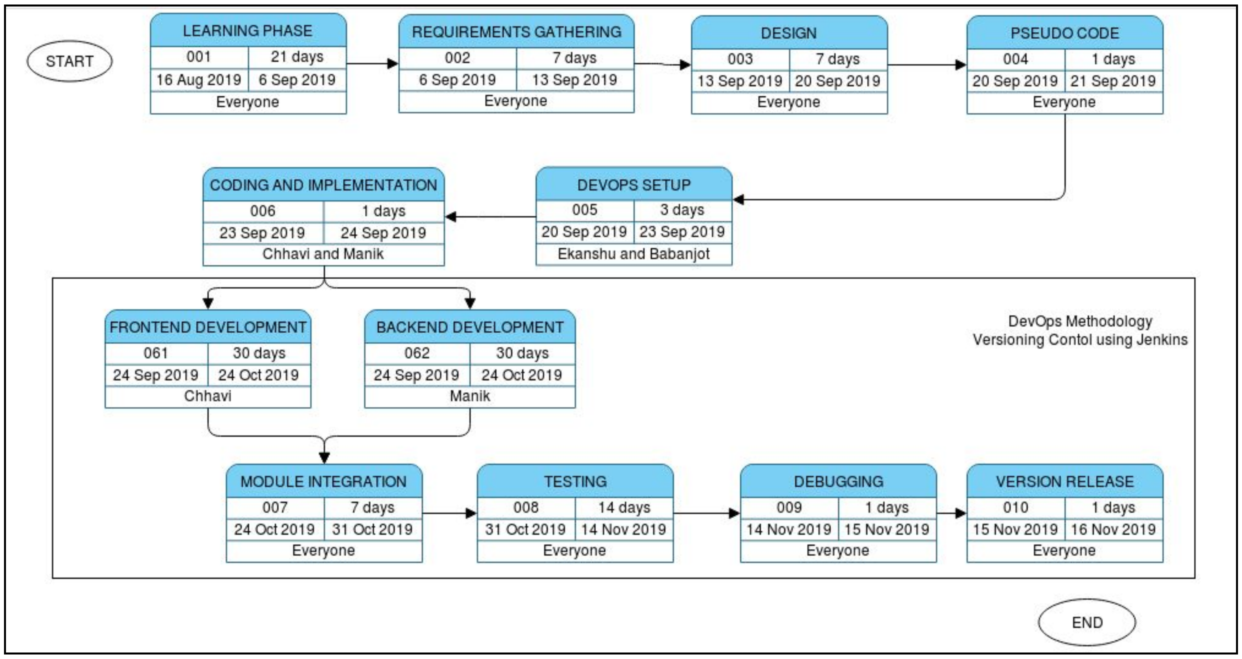

Start
Running blueocean Docker for Jenkins
Access container and start a wizard to set up Jenkins
Fork and clone a GitHub repository for the web application code
Rectify errors
Check successful cloning
Pipeline creation as JenkinsFile
Local repository commit and creation of DockerFIle for web app
Make further changes and commit
Building up the whole automated setup as created above on Jenkins
Stop

Fig 8.1. Creation of web application Fig 8.2. Deployment of web application using DevOps

9. System Requirements (Software/Hardware)

- **Hardware:**
  - Intel(R) Core(TM) i3-3200 CPU @ 1.5 GHz
  - 2 GB RAM
  - System type is 32/64-bit Operating System
- **Software:**
  - Linux based operating System
  - Docker
  - Jenkins

10.**Schedule** (**P**roject **E**valuation and **R**eview **T**echnique Chart**)**



11.References

[1] Dunlu PENG† , Lidong CAO, Wenjie XU "Using JSON for Data Exchanging in Web Service Applications ", *University of Shanghai for Science and Technology, Shanghai 200093, China* , 2011.

[2] Naimul Islam Naim,"ReactJS: An Open Source JavaScript Library for Front-end Developement" *IMetropolia University of Applied Sciences*,30 May 2017.

[3] Kenneth Lewenhagen, Anders Åkesson ."Node.js in Open Source projects on Github." *Blekinge Institute of Technology*

[4]. Arpitha R & Mrs. Kavitha S N,"Automation Using Jenkins: Plugins,Test Design , Test Test Execution and Reporting" *University of Zurich, Zurich, Switzerland.*

[5]. Jurgen Cito, Vincenzo Ferme, and Harald C. Gall, "Using Docker Containers to ImproveReproducibility in Software and WebEngineering" *Dept. of ISE, R. V. College of Engineering, Bengaluru, India.*

[6]. Darshita Kalyani, Dr. Devarshi Mehta,"Paper on Searching and Indexing Using Elasticsearch" *International Journal Of Engineering And Computer Science ISSN:2319-7242*

[7]. Kris Shaffer ,"Push, Pull, Fork: GitHub for Academics", May 26, 2013

Links:

[a].https://nodejs.org/en/doc/

[b].https://reactjs.org/docs/getting-started.html

[c].https://jenkins.io/doc/

[d].https://docs.docker.com

[e].https://docs.aws.amazon.com

[f]. https://guides.github.com

[g].https://www.elastic.co/guide