

# **PASSWORD MANAGER**

(Mar 2022-July 2022)

Internship report submitted in partial fulfilment of the requirement for  
the degree of Bachelor of Technology

in

**Computer Science and Engineering**

By

Shradha Agarwal (181440)

Under the supervision of

Dr. Monika Bharti

To



Department of Computer Science & Engineering and Information  
Technology **Jaypee University of Information Technology**  
**Waknaghat, Solan-173234, Himachal Pradesh**

# CERTIFICATE

I hereby declare that the work presented in this report entitled “**Password Manager**” in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** submitted in the department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology Wanknaghat is an authentic record of my own work carried out over a period from March 2022 to July 2022 under the supervision of Dr. Monika Bharti, Assistant Professor, CSE & IT.

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Shradha Agarwal(181440)

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

**Dr. Monika Bharti**

**Assistant Professor(SG)**

Computer Science & Engineering and Information Technology Department  
Jaypee University of Information Technology

## **DECLARATION**

I hereby declare that the work presented in this report entitled “**Password Manager**” in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** submitted in the department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology Wanknaghat is an authentic record of my own work carried out over a period from March 2022 to July 2022 under the supervision of Dr. Monika Bharti, Assistant Professor, CSE & IT. The matter embodied in the report has not been submitted for the award of any other degree or diploma.

**Submitted by:**

**Shradha Agarwal,**

**181440**

CSE&IT

Jaypee University of Information Technology

**Project Supervisor:**

**Dr. Monika Bharti**

Assistant Professor(SG)

## ACKNOWLEDGEMENT

The success and outcome of this project required a lot of guidance and assistance from many people, and I am extremely privileged to have got this all along the completion of my project. All that I have done is only due to such supervision and assistance and I would not forget to thank them.

I thank Mr. Ankit Santosh Bhimte for providing me an opportunity to do the project work at Hashedin by Deloitte and giving me all support and guidance, which made me complete the project duly.

I am thankful to and fortunate enough to get constant encouragement, support and guidance from my supervisor Dr. Monika Bharti and all Teaching staff of the Computer Science and Engineering Department, which helped me in successfully completing my project work.

# TABLE OF CONTENTS

<b>Content</b>	<b>Page No.</b>
<b>Certificate</b>	<b>1</b>
<b>Declaration</b>	<b>2</b>
<b>Acknowledgement</b>	<b>3</b>
<b>List of Figures</b>	<b>4</b>
<b>Chapter 01: INTRODUCTION</b>	<b>5</b>
<b>1.1 Introduction</b>	<b>6</b>
<b>1.2 Job Description</b>	<b>6</b>
<b>1.3 Objectives</b>	<b>7</b>
<b>1.4 Internship Schedule</b>	<b>8</b>
<b>Chapter 02: COMPANY AND PROJECT DESCRIPTION</b>	<b>9</b>
<b>2.1 About the Company</b>	<b>9</b>
<b>2.2 About the Project – Password Manager</b>	<b>10</b>
<b>Chapter 03: TOOLS AND TECHNOLOGY USED</b>	<b>13</b>
<b>3.1 React</b>	<b>13</b>
<b>3.2 SpringBoot</b>	<b>17</b>
<b>3.3 MySQL</b>	<b>18</b>
<b>3.4 GitHub</b>	<b>19</b>
<b>3.5 Postman</b>	<b>20</b>
<b>Chapter 04: LIVE PROJECT</b>	<b>23</b>
<b>4.1 Description</b>	<b>23</b>
<b>4.2 Code</b>	<b>23</b>
<b>4.3 Result</b>	<b>30</b>
<b>Chapter 05: CONCLUSION</b>	<b>33</b>
<b>5.1 Conclusion</b>	<b>33</b>
<b>5.2 Mentor’s Review</b>	<b>33</b>
<b>REFERENCES</b>	<b>34</b>
<b>PLAGIARISM REPORT</b>	<b>35</b>

## LIST OF FIGURES

Figure No	Figures	Page No.
1	Process of Software Development	6
2	Growth Trajectory of Hashedin by Deloitte	8
3	A simple react component	14
4	Features of ReactJS	14
5	What is SpringBoot?	16
6	GitHub Repo of backend part of the project	19
7	GitHub Repo of frontend part of the project	19
8	Postman Application	21
9	App.jsx	22
10	Login.jsx	23
11	Security Question component	24
12	Signup Component	25
13	Info Card Component	26
14	Navbar.jsx	27
15	Dashboard.jsx	28
16	Dashboard Screen	29
17	Navbar	29
18	Add Account Modal	30
19	Login Screen	30
20	Security Questions Screen	31
21	Signup Screen	31

# **Chapter 01:**

## **INTRODUCTION**

### **1.1 INTRODUCTION**

An internship is a professional knowledge experience that offers significant, practical work related to a student's field of study or career interest. An internship gives a pupil the chance for career disquisition and growth, and to learn new experience. This report is a description of the internship that I am currently going through at Hashedin by Deloitte, Bengaluru (Bangalore). This internship report particulars the conditioning that supported me attain a lot of my given targets. I was assigned the role of "Software Developer Intern" for my internship. We were asked to learn about different languages and frameworks in the first month of the internship including cloud, React, Angular, Springboot, Advanced Java, Advanced Angular and other. In the second month of the internship, we were assigned to different amazing projects in a team of four people with coaches to support. Currently, I am working on the same majorly on the front-end of the project using React and JavaScript.

### **1.2 JOB DESCRIPTION**

The foreword of engineering generalities to the creation, performance, and specialized operation is known as software engineering. Software engineering was designed to address the challenges of low- quality software enterprise. Problems arise when software exceeds schedules, budgets, and quality prospects. It assures that the software is erected in a harmonious, correct, timely, cost-effective, and specification-compliant manner. Software engineering came vital to keep up with the rapid-fire- fire- fire depending on stoner conditions and the terrain in which the program is anticipated to operate.

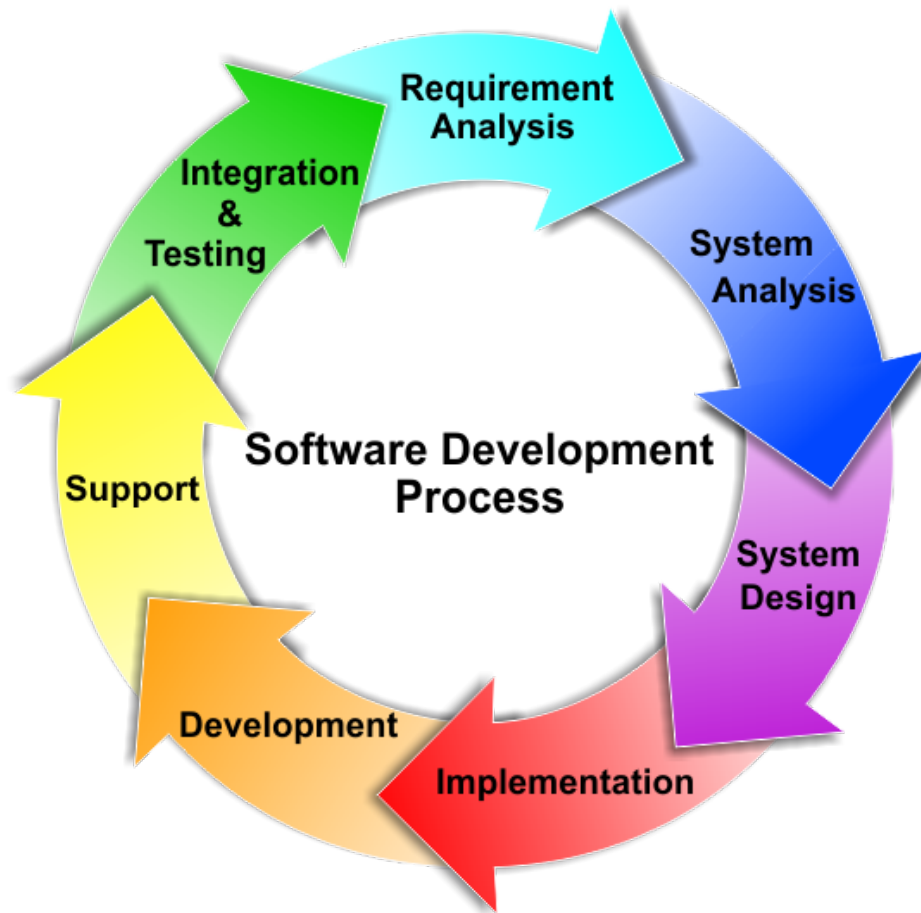


Fig 1. Process of Software Development

The Software Mastermind Trainee is responsible to help in the design and development of software. The Software Mastermind Trainee works with other members of the platoon to develop secure and reliable software results. The duties and arrears of this position are as follows

- Operation development (coding, programming)
- Program debugging and testing
- New software operation attestation and testing
- Probing, diagnosing, and resolving a wide range of specialized challenges
- Working with senior professionals.
- Finding issues and developing their fixes.
- Learning about modern technology as they are evolving

### 3. OBJECTIVES

The objective of the internship is to learn and gain experience about technologies and frameworks like React, SpringBoot, JavaScript, MySQL. These are the main tech stack of the project that I am currently working on. After working on this project, we



all will be assigned to a live project that will help me gain more and better experience of the corporate work. This apprenticeship provides expertise to the freshers so that they can master further about the industriousness. The viewpoint was to gather

enough knowledge so that we can work smoothly on the project tasked to us.

#### **4. INTERNSHIP SCHEDULE**

to the freshers so that they can master further about the industriousness. The viewpoint was to gather enough knowledge so that we can work smoothly on the project tasked to us.

The internship plan was as follows:

##### ■ March 2022

My internship started on 14<sup>th</sup> March, in the first week, only onboarding process and formalities took place. In the second week, we were thought cloud and it is use, we were also supposed to complete a project based on it. In the third week, we were supposed to study about HTML, CSS, JavaScript and React, also needed to complete 3 mini projects and 1 major project based on this.

##### ■ April 2022

First week was for learning Java and SpringBoot and to complete a project based on this. Second week of the month was for Advanced Java, in this we learned about security in web applications and completed a project based on this. Third week of the month was designated to learn about Advanced Angular and did a project based on this. In the last week of the month, we

were assigned to different projects in a team of four people where we have to a build a product from scratch using different frameworks.

##### ■ May 2022

In this month, I developed the front-end of the project including login page, signup page, dashboard, navbar, account details and implemented few other functionalities.

# Chapter 02: COMPANY & PROJECT DESCRIPTION

## 2.1 ABOUT THE COMPANY

HashedIn by Deloitte specialize in native cloud development and software engineering. In 2020, we joined Deloitte Consulting to synergize and gauge our capabilities. Since joining Deloitte Consulting, our innovative cloud native results and our ultramodern cloud delivery model have helped top Fortune 500 companies achieve significant business value.

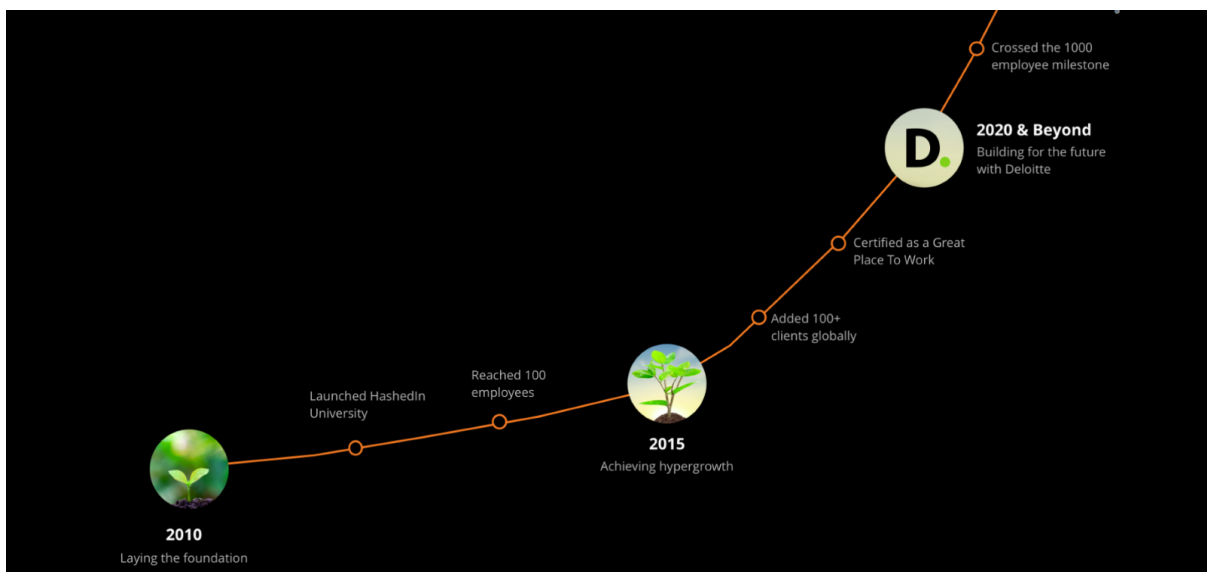


Fig 2: Growth Trajectory of Hashedin by Deloitte

HashedIn by Deloitte is a born in the cloud, high- end software engineering and product development company. We develop request- leading results by using cloud-native technologies, AI capabilities and a pod- based delivery model. HashedIn by Deloitte is structured around small, dynamic brigades working collaboratively to deliver slice- edge results for our guests. We make leaders of the future by encouraging individualities to take extreme power of their work and furnishing the inflexibility to work on systems that prop literacy and development.

## 2.2 ABOUT THE PROJECT – PASSWORD MANAGER

### 1. What is a Password Manager?

From capital letters to punctuation and figures, utmost spots bear long and complex watchwords to stylish cover stoner information. While the benefits of added security are accessible, trying to keep track of the complicated watchwords used across dozens of websites and operations can be easier said than done. Studies estimate that the average business stoner has 191 watchwords to keep straight. For those who struggle to manage watchwords across dozens of spots, a word director can be an excellent investment. These programs are designed to manage security by creating strong watchwords and keeping them organized, and both business and individualities likewise can make use of the protections they offer. From securing non-public commercial content to keeping logins secure, a word director can be a veritably precious tool.

### 2. Benefits of a Password Manager

The biggest vantage for individual and business password manager users is perfecting online safety by removing the need to remember complicate watchwords across in numerous spots. But there are other added benefits for using a password management system, involving:

- **Auto-generating secure passwords**

With the growing threat of cybercrime, assuring passwords are self-assured and tough to guess is more important than ever. Still, creating distinct yet memorable watchwords that include capital letters, lowercase letters, special characters, and figures for every point with an enrolment can be easier said than done.

A word director can take the pressure off word creation. When subscribing up for a new point or service, your word director can suggest a randomized word that meets specific platform conditions. This word will also be stored within the program's memory, performing in the perpetration of a word that is hard to crack but easy to pierce.

- **Alert you to phishing spots.**

Phishing spots, or fraudulent spots posing as licit spots in a trouble to steal information, are each too common on the web. Those who fall for phishing spots can put particular and fiscal information at threat, opening themselves up for potentially dangerous attacks or indeed a stolen identity.

Word directors, still, can reduce the threat of falling victim to online traps. As the data associated with word directors is point-specific, your word director does not bus- colonize information when you visit an illegitimate point. This can stop you from entering a username and word into a phishing point, effectively precluding this usual form of cybercrime.

- **You do not have to remember all your watchwords.**

Still, you 're not alone, if you 're tired of trying to remember watchwords across in numerous spots. Still, a word director can help you circumvent this common burden by storing all your username and word combinations in a secure manner so that you do not have to calculate on memory alone.

When you visit spots or open programs that are stored in your word director, word information can bus- colonize. This makes it easy to log in anywhere without the need to remember watchwords yourself.

Indeed, if you go times between using a particular point, your word director will insure you always have access.

- **Sync across all your bias and operating systems.**

As numerous web druggies with multiple bias know, a word saved on your computer may not bus- colonize on your phone. Different word memory systems will keep your access to watchwords disintegrated, creating a logistical headache that can make using the same services across multiple bias unnecessarily frustrating.

Luckily, some word directors work across bias, furnishing a way to unify word use for a more effective approach to penetrating websites. Watchwords enforced across all your bias can be secured in one accessible position, simplifying the process of creating and using watchwords.

### **3. Types of Password Manager**

Word directors come in as many different shapes and sizes to best accommodate individual requirements. The three most common types are desktop, pall- grounded and single sign-on.

- A desktop word director is one of the oldest and most popular options. This kind of operation encrypts and stores watchwords directly on a stoner's machine, limiting the eventuality for breaches. Still, these products can only be used on a single machine a distinct limitation for those who use multiple computers and mobile bias.

- A cloud- based word director lives in the pall and can be penetrated from any device, anyhow of network or position. This makes all watchwords readily available from anywhere, offering an ease of use else unapproachable. Still, security is left in the hands of the word operation provider, which can increase the liability of implicit breaches.
- Single sign- on word directors are most common in a commercial setting. With this tool, all websites and accounts needed can be penetrated using a single sign-on as opposed to different usernames and watchwords for different services. While diversity is recommended for the sake of security, SSO products consolidate watchwords on platforms that are all related, like a variety of work spots hosted internally.

#### **4. How does a Password Manager Work?**

A word director is a third- party program that creates and manages watchwords on your behalf. Of remembering your own watchwords or using the same watchwords across multiple spots — a tactic not recommended due to the increased liability of unauthorized account access — a word director can do the arduous work for you.

To get started, you will need to choose your asked product, subscribe up, make any necessary payments, and download the software. This will bear adding an extension to any cybersurfs you use. However, you may also need to install a phone or tablet app on your mobile device if you would like to synchronise a pall- grounded word director across bias.

When you subscribe up for a new website, your word director will suggest a complex word that is hard to guess and store it for you so that you do not have to remember it. The coming time you visit the point, the program can automatically colonize your login information so that you do not have to manually enter long and complicated watchwords for every point you visit. However, this function can be impaired on a point-by-point base, if for some reason you do not want a word director to produce or store a word for you.

Still, this is accessible as well, if you need to pierce a full list of watchwords contained within the word director. This can be a helpful way of viewing watchwords to log in on a different device when using a desktop or cybersurfed-specific word director.

## Chapter 03:

### TOOLS & TECHNOLOGY USED

#### 3.1 REACT

React is a JavaScript library that aims to make the development of visual interfaces easy. Developed at Facebook and released to the world in 2013, it drives some of the most extensively used apps, powering Facebook and Instagram among in numerous other operations.

Its primary thing is to make it easy to reason about an interface and its state at any point in time. It does this by splitting the UI (User Interface) into an assemblage of elements. React makes numerous stuffs well, and its ecosystem is filled with great libraries and tools.

React has a veritably small API (Application Programming Interfaces). React makes it effortless to produce interactive UIs (User Interface). Design simple views for each state in your operation, and React will efficiently modernize and render just the right factors when your data changes.

Build encapsulated components that manage their own state, also compose them to make complex UIs. Since component logic is written in JavaScript of templates, you can fluently pass rich data through your app and keep state out of the DOM (Document Object Model).

The most common website armature moment is MVC (model view regulator). Reply is the view in the MVC design, whereas Flux or Redux is the armature. A ReactJS operation is made up of multitudinous factors, each being responsible for generating an applicable piece of HTML law. Factors are the foundation of all Reply programs. Complex operations can be developed using simple structure blocks by nesting these factors with other factors. ReactJS interconnects the DOM- grounded approach for colonizing data in the HTML DOM. Because the virtual DOM alters specific DOM factors rather than refreshing the full DOM, it is presto.

To produce a React app, we produce Reply factors that correspond to colourful aspects. The operation structure is made up of these factors, which are arranged into advanced- position factors. Consider a form containing the needed fields, markers, and buttons, among other effects. Each form element can be created as a React element, which we also combine to make the form element. The form rudiments would describe the structure and corridor of the form.

ReactJS' major purpose is to construct a Stoner Interface that aids program performance. It makes use of a virtual DOM (JavaScript objects), that puts up the app. In JavaScript, the virtual DOM is quicker than the traditional DOM. ReactJS can also be used on garçon and customer sides, as well as with other fabrics. Element and data structures are used to frame and make app conservation easier.

**React Components:** This section is one of the most important buildings of the React. In other words, we can say that every app you develop in React will be made up of pieces called components. The components make the task of building UI much easier. You can see the UI divided into many individual pieces called sections and work on them independently and integrate them all into the parent component which will be your final UI.

The Google Custom Search at the top can be seen as a component, the navigation bar can be viewed as a component, the sidebar is an individual component, a list of articles or posts is also a component and finally, we can cover it all. of these separate parts to make the parent component will be the final UI of the homepage. The parts in React return a piece of JSX code that tells what to give to the screen. At React, we have two types of components:

**Functional Components:** Active components are simply JavaScript functions. We can create an active component in React by typing a JavaScript function. These functions may or may not receive data as parameters, we will discuss this later in the lesson. The example below shows the active component in React:

```
const Democomponent = () =>
{
  return <h1> Welcome Message! </h1>;
}
```

**Class Components:** Class components are much more complex than functional components. Active components do not recognize other components in your system while class components may work alone. We can transfer data from part of the classroom to other parts of the classroom. We can use JavaScript ES6 classes to build class-based components in React. The example below shows a valid class-based component in React:

```
the Democomponent category extends React.Component
{
  give () {
    return <h1> Welcome Message! </h1>;
  }
}
```

The parts we created in the two examples above are equal, and we also mentioned the basic differences between the working part and the class part. We will learn about

additional classroom-based component structures in other subjects. In the meantime, keep in mind that we will only use the active component if we are certain that our component does not need to work or work with any other component. That is, these components do not require data from other components, but we can name multiple operating components under one operating component. We may also use class-based components for this purpose, but it is not recommended as using class-based components unnecessarily will make your application less efficient.

```
LIVE JSX EDITOR  JSX? RESULT
class HelloMessage extends React.Component {
  render() {
    return <div>Hello {this.props.name}</div>;
  }
}

root.render(<HelloMessage name="Taylor" />);

Hello Taylor
```

Fig 3: A simple react component

**Features of React:** ReactJS is rapidly becoming the most popular JavaScript framework among web developers. It is an important part of late ecology. The following are some of the key features of ReactJS ':

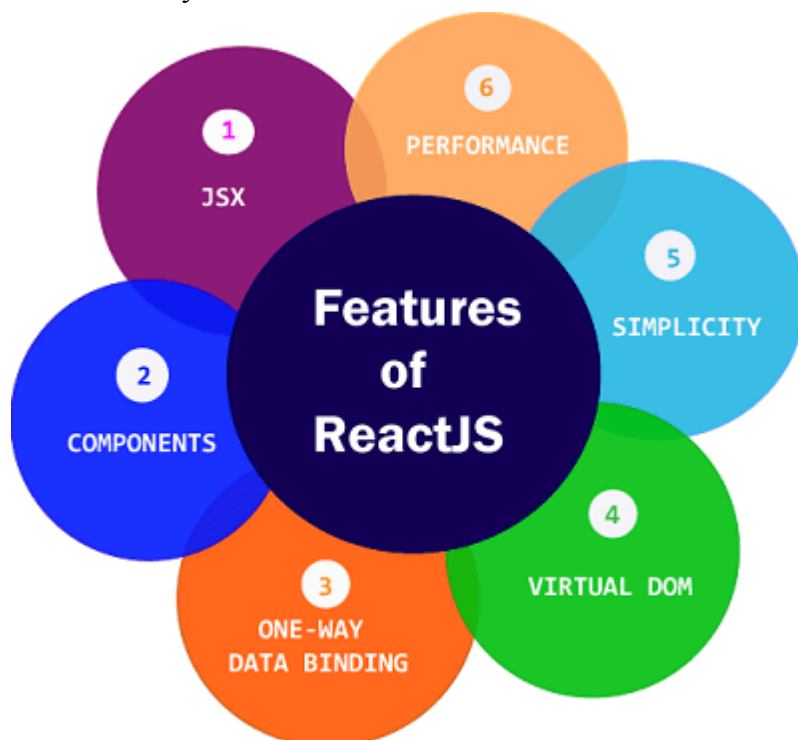


Figure 4: Features of ReactJS



- JSX

JavaScript XML is abbreviated as JSX. It is a JavaScript syntax extension. ReactJS uses XML or syntax like HTML. This syntax is converted to React Framework JavaScript phones. It upgrades ES6 to allow HTML-like text and JavaScript response code to work together. It is not necessary using JSX, however, is highly recommended in ReactJS.

- Components

The parts are in the heart of ReactJS. The ReactJS app contains a few components, each has its own logic and controls. These parts can be reused, making them easier to maintain the code is clean while working on major projects.

- One-way data binding

ReactJS is designed to handle one-way binding data or a single data flow. One way data binding allows you to have more control over your app. When data flow occurs on the contrary, additional features are needed. Because the parts are designed to be it cannot be changed, and the data contained cannot be changed, this is the case. Flux is a pattern that 16 resources in data unidirectionality. This increases efficiency by making the app more compact various.

- Virtual DOM

The first DOM object is represented by a visible DOM object. It works the same in a one-way data binding. The complete UI is redesigned to represent the visual DOM at any time any changes made to the web application. Then compare the differences between the old ones and new DOM presentations. After that, the true DOM will only update the items it contains has changed. These speeds up the application and eliminates memory pollution.

- Simplicity

ReactJS uses a JSX file, making the program easy to code and understand. We you already know that ReactJS is a component-based solution that allows you to reuse code like required. This makes it easy to use and understand.

- Performance

ReactJS Speed is immensely popular. This separates it from other structures at present available. This is because it controls the visual DOM. Document Object Model (DOM) is a cross-platform computer programming API for HTML, XML, and XHTML. DOM is based on memory only. As a result, we did not write directly to the DOM at the time to form part. Instead, we will create tangible objects that will be converted to DOM, which leads to smoother and faster performance.

## 3.2 SPRINGBOOT

Spring Boot is a project built on the framework of the Spring. It provides an easy and fast way to set up, configure, and use both simple and web-based applications. It is the Spring module that provides the RAD (Quick Application Development) feature in the Spring Framework. It is used to build an independent Spring-based application that you can simply use because it requires minimal Spring configuration.

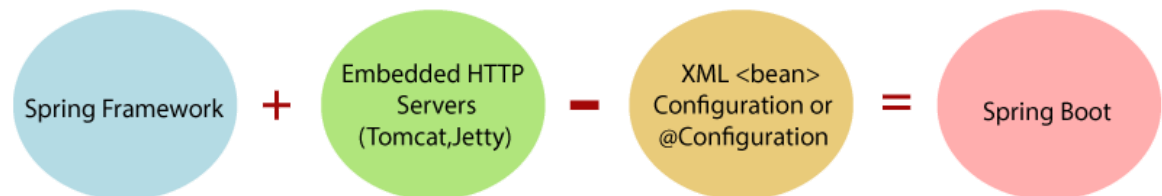


Fig 5: What is SpringBoot?

In short, Spring Boot is a combination of Spring Frame and Embedded Servers. In Spring Boot, there is no need for XML configuration (usage description). It uses a conventional over configuration software design paradigm which means it reduces the effort of the developer.

We may use Spring STS IDE or Spring Initializer to improve applications for Spring Boot Java.

Why should we use the Spring Boot Framework?

We should use the Spring Boot Framework because:

- The dependency injection method is used in the Spring Boot.
- Contains the ability to manage transaction on a powerful website.
- It makes it easy to integrate with other Java components such as JPA / Hibernate ORM, Struts, etc.
- Reduce the cost and development time of the app.
- Along with the Spring Boot Framework, many other Spring projects are helping to create programs that address the needs of modern business needs.

There are the following spring sister projects:

**Spring Data:** Make data access easier from NoSQL related websites.

**Spring Collection:** Provides powerful bulk processing.

**Spring safety:** It is a protective framework that provides strong security in operating systems.

**Spring Social:** Supports integration with social networking sites such as LinkedIn.

**Spring Integration:** It is the implementation of Enterprise Integration Patterns. Helps integrate with other business applications using lightweight messages and adapters.

### **Advantages of SpringBoot:**

- Creates independent Spring applications that can be run using Java-jar.
- It easily scans web applications with the help of various HTTP embedded servers such as Tomcat, Jetty, etc. We do not need to feed WAR files.
- It provides 'start-up' POMs to ease our Maven configuration.
- Supplies ready-to-produce features such as metrics, health tests, and external configurations.
- No need for XML configuration.
- Provides CLI tool for developing and evaluating the Spring Boot application.
- Supplies several plug-ins.
- It also reduces the writing of multiple boilerplate codes (code that should be inserted in multiple locations with little or no modification), XML configuration, and annotations.
- It increases productivity and reduces development time.

## **3.3 MYSQL**

**Database:** A database is a separate application that stores a collection of data. Each website has one or more different APIs for creating, accessing, managing, searching, and duplicating the data it owns.

Other types of data stores can also be used, such as files in the file system or large hash tables in memory but downloading and writing data would not have been faster and easier with those types of systems.

Nowadays, we use RDBMS to store and manage substantial amounts of data. This is called a relational database because all data is stored in separate tables and relationships are set up using key keys or other keys known as External Keys.

**MySQL Database:** MySQL is a fast, easy-to-use RDBMS for many small and large businesses. MySQL is developed, marketed, and supported by MySQL AB, a Swedish company. MySQL has become exceedingly popular for many good reasons -

- MySQL is issued under an open-source license. So, you have nothing to pay for it.
- MySQL is a powerful program. Holds a large subset of high-performance and powerful website packages.

- MySQL uses a standard type of known SQL data language.
- MySQL works on multiple operating systems and in many languages including PHP, PERL, C, C ++, JAVA, etc.
- MySQL works amazingly fast and works well even on large data sets.
- MySQL is very friendly to PHP, the most respected language in web development.
- MySQL supports a large, up to 50 million or more lines in the table. The default table file size limit is 4GB, but you can upgrade this (if your operating system is unable to carry it) to a theoretical limit of 8 million terabytes (TB).
- MySQL can be customized. The open-source GPL license allows programmers to customize MySQL software to fit their specific locations.

### 3.4 GITHUB

Software developers and engineers can use GitHub to build cloud-based cloud-based environments for free. You can transfer GitHub cache to your device, add and change files locally, and then "push" your changes back to the archive, where they will be visible to the public.

#### **GitHub Repository:**

The repository (sometimes abbreviated as "repo") is a site where all project files are stored. Each project has its own repository, which you can access using a specific URL.

#### **Forking a Repository**

When you launch a new project based on an existing project, this is known as "forking." This is an amazing tool that encourages the development of new applications and projects. If u find a project on GitHub that you would like to contribute, you can install it, make changes you want, and then republish it as a new repo. You can quickly add changes to your current fork if the original repository you have forged to create your new project is being updated.

#### **Git Commit**

You are ready to commit once you have staged the files you want to add. The message of commitment says it is important whether you are committed to using GitHub Desktop or the command line. It is short bind statements that describe your change should be used. Sending messages will take the lead to you about the history of your last place, so it should be informative. Next The message format can be used in the Commitment command line:

*git commit -m "example of git message"*

## Pull Requests

You have found the last place, made a positive change to the project, and are looking for the first devs be careful — even put it in the original project / repository. Create a drag request to achieves this. Real buffer writers can look at your work and decide whether Welcome to the official project. When you send a pull request, GitHub supplies an excellent communication channel between you and the main project manager.

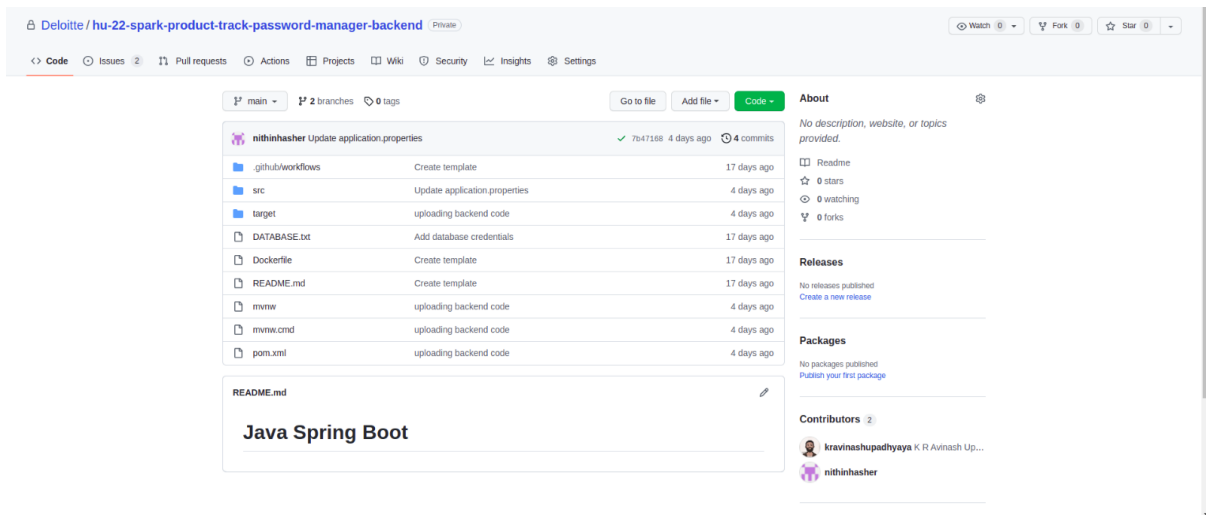


Fig 6: GitHub Repo of backend part of the project

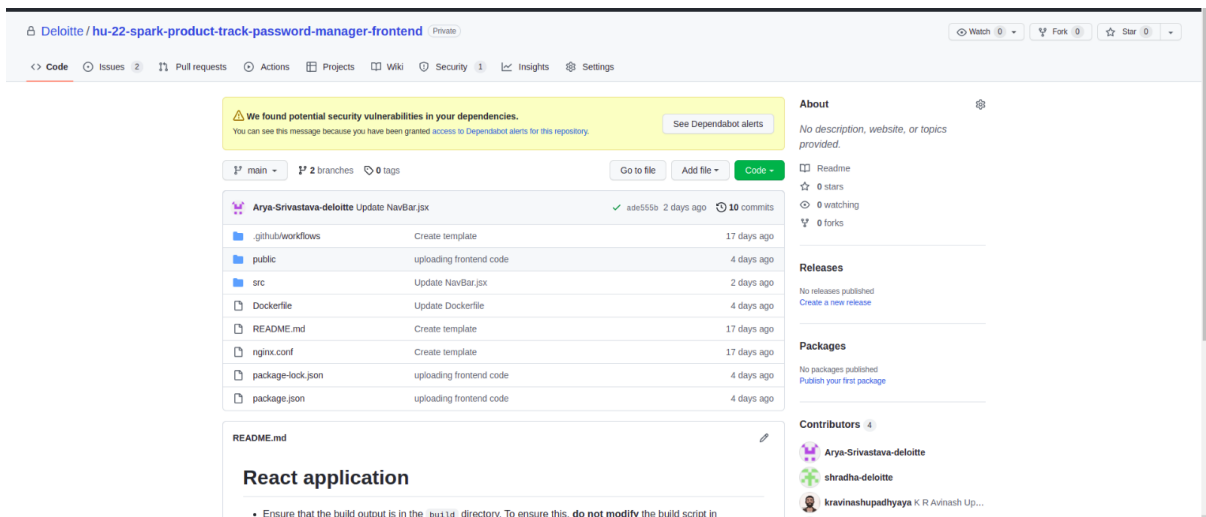


Fig 7: GitHub Repo of frontend part of the project

## 3.5 POSTMAN

Postman is a well-known API testing tool. APIs can be easily created, tested, shared, and written with the help of this tool. Postman Test Application Program Interface (API), a tool that can be used to create, test, develop, change, and compose APIs. It is a basic metaphor user interface for making and analysing HTTP requests and

responses. When using Post person for testing, you do not need to write any HTTP contact code. Instead, we use Postman to communicate with the API and create test suits called collections. All aspects an engineer may need to be included in this program. This tool can do GET, POST, PUT, and PATCH HTTP queries, as well as changing APIs.

Postman is built next to a set of powerful technologies that are extremely easy to use. Postman has become a useful tool for over 8 million users. The postman is used for the following reasons:

1. Accessibility- After installing Postman on the device, simply log in to the account you are going to use anywhere.
2. Use Groups - Users can create their own API call collections using Postman. Each the set can create multiple applications and subfolders. It will help the testing organization suites.
3. Test development - Every API application should include successful HTTP authentication response status in test sites.
4. Automatic Testing- Collection Runner or Newman can run tests repeatedly repetition or repetition, saving time for repeated testing.
5. Site Creation- The use of multiple sites reduces the frequency of testing because the same collection can be used in different settings.
6. Debugging- The postman console helps to fix errors effectively by tracking what data is available.
7. Collaboration- To improve file sharing, you can import and export collections places. You can also share collections via direct link.
8. Continuous integration- It can support continuous integration. It is as easy as typing a URL in your browser to send a request. With Postman, we can send requests to APIs. You can use the API request to access or retrieve data from the data source. An HTTP method is needed to send an API request. SEND, RECEIVE, REMOVE, PUT, AND PATCH is one of the most widely used methods.
  - GET: This HTTP method is used to access data in the API.
  - POST: This method transmits new data.
  - DELETE: This is used to remove or delete existing data.
  - PATCH: This method is used to update existing data.
  - PUT: This method is used to update existing data.

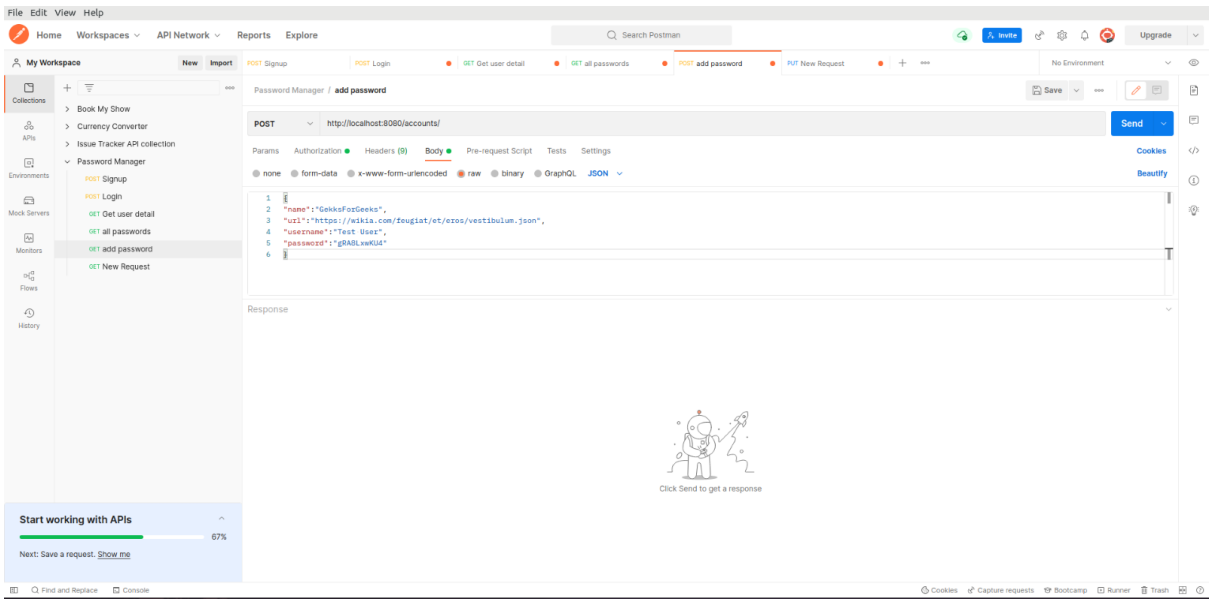


Figure 8: Postman Application

## Chapter 04:

# LIVE PROJECT

### 4.1 DESCRIPTION

The major task in this project is to create an app which is secure, easy to use and supply user the ease to autofill their passwords. As we are working on a full stack web application, my major task is to create the front end of the project. Front end part of the project included creating login and signup pages, one security questions page, dashboard, navbar page, option to add an account, info cards that will show the accounts user has saved, feature to update the credentials in the info card, also dropping the account information that user has stored in the info card.

### 4.2 CODE

```
1 import './App.css';
2 import SignUp from './components/SignUp/SignUp';
3 import { BrowserRouter, Route, Routes } from 'react-router-dom';
4 import LogIn from './components/LogIn/LogIn';
5 import Dashboard from './components/DashBoard/Dashboard';
6 import ForgotPassword from './components/ForgotPassword/ForgotPassword';
7 import Settings from './components/Settings/Settings';
8 import InfoCard from './components/InfoCard/InfoCard';
9 import EditPassword from './components/EditPassword/EditPassword';
10
11 function App() {
12   return (
13     <BrowserRouter>
14       <Routes>
15         <Route path="/" element={<SignUp />} />
16         <Route path="/login" element={<LogIn />} />
17         <Route path="/dashboard" element={<Dashboard />} />
18         <Route path="/forgotPassword" element={<ForgotPassword />} />
19         <Route path="/settings" element={<Settings />} />
20         <Route path="/card" element={<InfoCard />} />
21         <Route path="/editPassword" element={<EditPassword />} />
22         <Route
23           path="*"
24           element={
25             <main style={{ padding: "1rem" }}>
26               <h3>
27                 Sorry! We couldn't find that page. Try searching or go to{" "}
28                 <a href="/">Password Manager's home page</a>.
29               </h3>
30             </main>
31           }
32         />
33       </Routes>
34     </BrowserRouter>
35   );
```

Figure 9: App.jsx



```

17 function LogIn() {
18     const navigate = useNavigate();
19     const [username, setUsername] = useState("");
20     const [password, setPassword] = useState("");
21     const [validated, setValidated] = useState(false);
22     // Function to update username state with
23     const handleNameChange = (e) => {
24         console.log("this is event", e.target.value);
25         setUsername(e.target.value);
26     };
27     // Function to update password state with
28     const handlePasswordChange = (e) => {
29         setPassword(e.target.value);
30     };
31
32     const handleSubmit = async (event) => {
33         event.preventDefault();
34         setValidated(true);
35         console.log(event.target.value);
36         let user = {
37             username,
38             password,
39         };
40         console.log({ user });
41         if (event.currentTarget.checkValidity() === false) {
42             event.stopPropagation();
43         } else {
44             axios
45                 .post("http://localhost:8080/token", user)
46                 .then((response) => {
47                     if (response.status === 200) {
48                         localStorage.setItem("userToken", response.data);
49                         navigate("/dashboard");
50                     }
51                 });
52         }
53     };
54 }

```

Fig 10: Login.jsx

```

src > components > SecurityQues > SecurityQues.jsx > SecurityQues > handleSignup > form
54   return (
55     <div className="Background">
56       <Container className="d-flex vh-100">
57         <Row className="m-auto align-self-center" style={{ width: "28rem" }}>
58           <Col>
59             <Card className="my-3" id="security">
60               <Card.Header as="h5">Security Questions</Card.Header>
61               <Form noValidate validated={validated} onSubmit={handleSignup}>
62                 <Card.Body>
63                   <Form.Text>
64                     <a href="/">Back to Sign Up</a>
65                   </Form.Text>
66
67                   <Form.Group
68                     className="mt-4 mb-3"
69                     as={Row}
70                     controlId="question-1"
71                   >
72                     <Form.Label>
73                       What is the name of your favorite childhood friend?
74                     </Form.Label>
75                     <Col>
76                       <Form.Control
77                         type="text"
78                         value={ques1}
79                         required
80                         onChange={handleQues1Change}
81                       />
82                       <Form.Control.Feedback type="invalid">
83                         Missing answer
84                       </Form.Control.Feedback>
85                     </Col>
86                   </Form.Group>
87

```

Fig 11. Security Question component

```

src > components > SignUp > SignUp.jsx > SignUp > handleNext
30   const handleNext = async (event) => {
31     const form = event.currentTarget;
32     setValidated(true);
33     event.preventDefault();
34     if (form.checkValidity() === false) {
35       event.stopPropagation();
36     } else {
37       setLoginInfo({ username, password });
38       console.log({ loginInfo });
39       setSecQues(true);
40     }
41   };
42
43   const handleSubmit = async (question) => {
44     const userInfo = {
45       username,
46       password,
47       question,
48     };
49     const response = await axios.post(
50       `http://localhost:8080/user/credentials`,
51       userInfo
52     );
53     if (response.status === 200) {
54       navigate("/login");
55     }
56     console.log("successfully account created", response);
57   };
58
59   return (
60     <>
61       {secQues ? (
62         <SecurityQues handleSubmit={handleSubmit} />
63       ) : (
64         <div className="App">

```

Fig 12: Signup Component

```

src > components > InfoCard > InfoCard.jsx > InfoCard > accountData.map() callback
30   const userToken = localStorage.getItem("userToken");
31
32   const config = {
33     headers: {
34       "Access-Control-Allow-Origin": "*",
35       Authorization: `Bearer ${userToken}`,
36     },
37   };
38
39   useEffect(() => {
40     async function fetchPasswords() {
41       // console.log(userToken);
42       const response = await axios.get(
43         "http://localhost:8080/accounts",
44         config
45       );
46       setAccountData(response.data.data);
47     }
48     if (userToken) {
49       fetchPasswords();
50     }
51   }, [userToken]);
52
53   const getData = () => {
54     axios.get(`http://localhost:8080/accounts`, config).then((getData) => {
55       setAccountData(getData.data.data);
56     });
57   };
58
59   const handleOpenModal = (id) => {
60     handleEditChange(id);
61   };
62
63   const onDelete = async (id) => {
64     try {

```

Fig 13: Info Card Component

```
Dashboard.jsx M  NavBar.jsx X  InfoCard.jsx M  # LogIn.css M  SignUp.jsx M  #
src > components > NavBar > NavBar.jsx > NavBar
11 export default function NavBar({
12   show,
13   handleShow,
14   handleClose,
15   validated,
16   handleSubmit,
17   handleAppNameChange,
18   handleUrlChange,
19   username,
20   password,
21   handlePasswordChange,
22   handleNameChange,
23   url,
24   name,
25 }) {
26   const userToken = localStorage.getItem("userToken");
27   const navigate = useNavigate();
28   const [profileName, setProfileName] = useState();
29
30   useEffect(() => {
31     axios
32       .get(`http://localhost:8080/user/credential`, {
33         headers: {
34           "Access-Control-Allow-Origin": "*",
35           Authorization: `Bearer ${userToken}`,
36         },
37       })
38       .then((response) => {
39         //console.log(response.data);
40         setProfileName(response.data);
41       });
42   }, []);
43
44   const handleLogout = () => {
45     localStorage.removeItem("userToken");
```

Fig 14 NavBar.jsx

```
Dashboard.jsx M X  NavBar.jsx  InfoCard.jsx M  # LogIn.css M  SignUp.jsx M
src > components > DashBoard > Dashboard.jsx > Dashboard > handleEditChange
1  import axios from "axios";
2  import React, { useState } from "react";
3  import InfoCard from "../InfoCard/InfoCard";
4  import NavBar from "../Navbar/NavBar";
5
6  export default function Dashboard() {
7    const [accountId, setaccountId] = useState();
8    const [accountData, setAccountData] = useState([]);
9    const userToken = localStorage.getItem("userToken");
10   const [name, setName] = useState("");
11   const [url, setUrl] = useState("");
12   const [username, setUsername] = useState("");
13   const [password, setPassword] = useState("");
14   const [validated, setValidated] = useState(false);
15   const [show, setShow] = useState(false);
16   const [edit, setEdit] = useState(false);
17
18   const handleClose = () => {
19     setName("");
20     setPassword("");
21     setUrl("");
22     setUsername("");
23     setEdit(false);
24     setShow(false);
25   };
26   const handleShow = () => setShow(true);
27
28   const handleNameChange = (e) => {
29     setUsername(e.target.value);
30   };
31   const handleAppNameChange = (e) => {
32     setName(e.target.value);
33   };
34   const handlePasswordChange = (e) => {
35     setPassword(e.target.value);
```

Fig 15 Dashboard.jsx

## 4.3 RESULTS

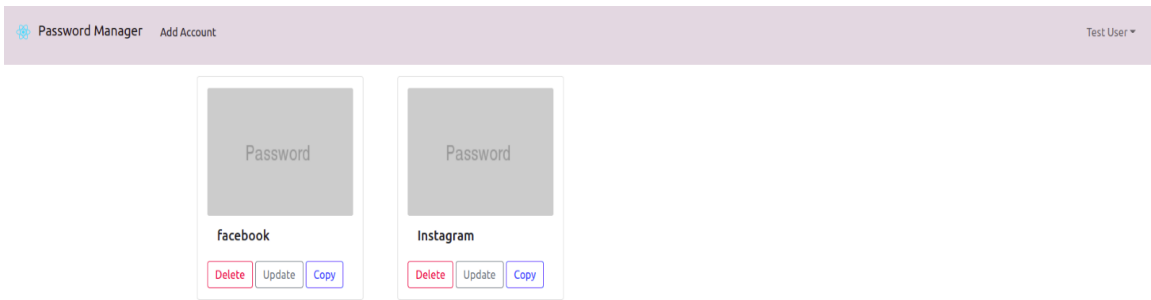


Fig 16 Dashboard Screen



Fig 17 Navbar

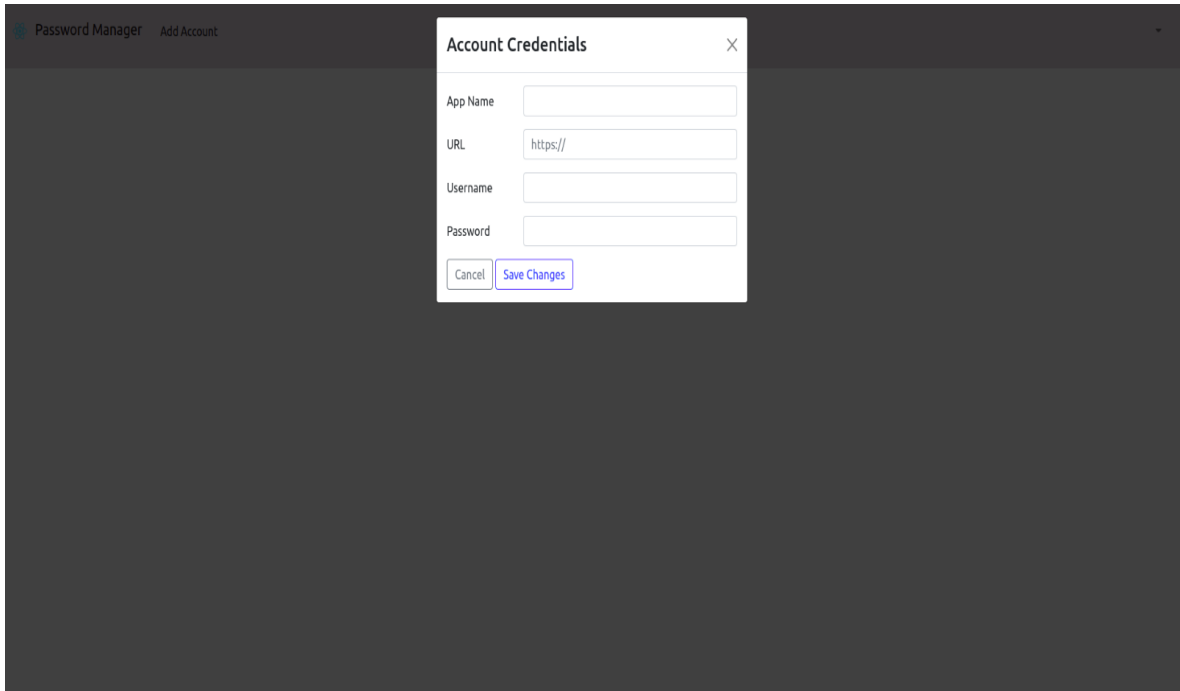


Fig 18 Add Account Modal

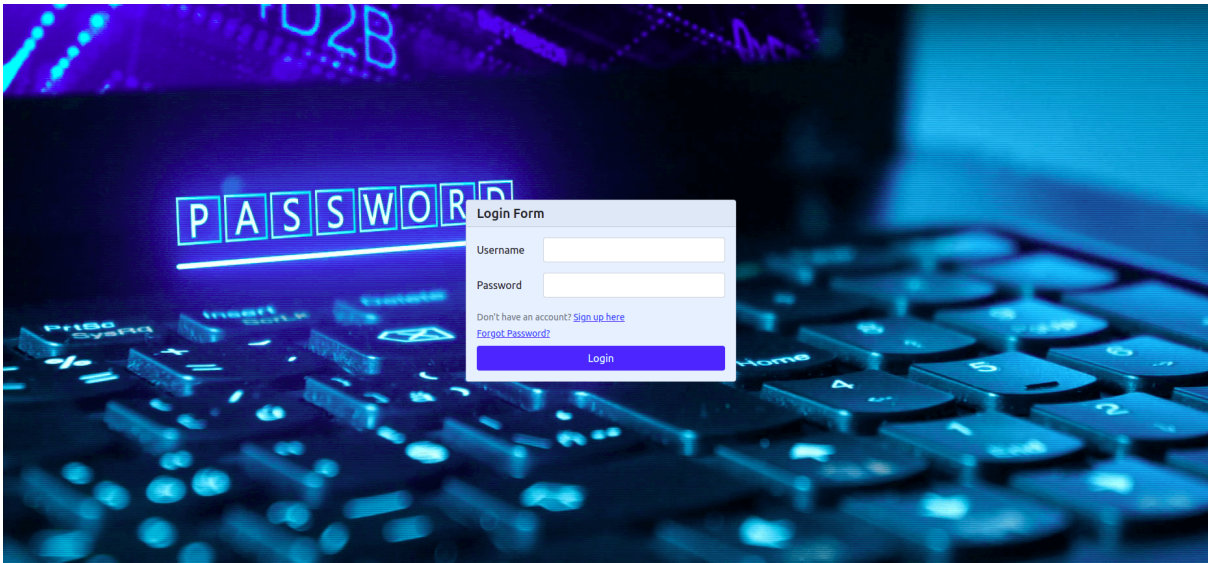


Fig 19 Login Screen



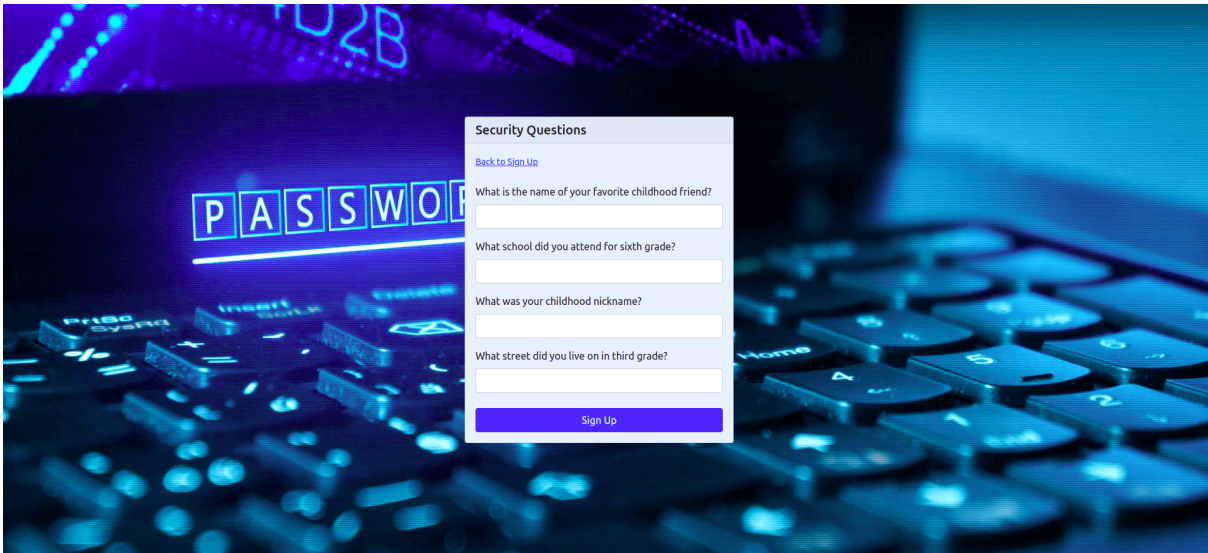


Fig 20 Security Questions Screen

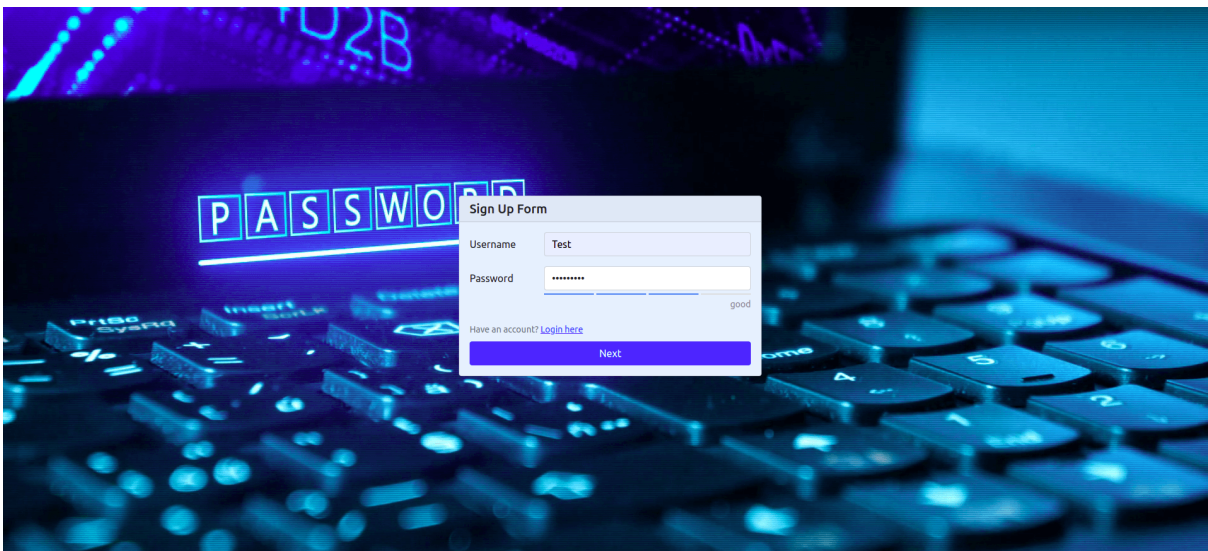


Fig 21 Signup Screen

## **Chapter 05:**

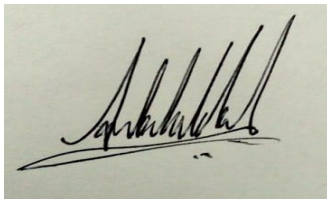
# **CONCLUSION**

### **5.1 CONCLUSION**

In conclusion, this internship has become a wonderful and rewarding experience. I can say that my stay with Hashedin by Deloitte was quite beneficial to me. The technical components of my work are not perfect and, given enough time, could be improved. As someone who had no prior experience with full-stack development, I think the effort I spent learning and understanding about was well spent, as it helped to construct a fully effective web service. Two of the most significant lessons I have learned are time management and self-motivation.

### **5.2 MENTOR'S REVIEW**

She started her training period in Hashedin University in which she worked on full-stack web development using ReactJS, SpringBoot, MySQL and along with this some devops concepts. Currently, she is working on a Password Manager project. In this project, she is working on the front-end using ReactJS. She has been able to successfully implement good designs and has actively participated in brainstorming and is an active team member.

A handwritten signature in black ink on a light-colored background. The signature is stylized and appears to read 'Ankit Santosh Bhimte'.

Ankit Santosh Bhimte  
Functional Analyst

## REFERENCES

1. <https://www.w3schools.com/html/default.asp>
2. <https://www.w3schools.com/css/default.asp>
3. <https://www.w3schools.com/js/default.asp>
4. <https://www.youtube.com/watch?v=ScsSCuHhOw0>
5. <https://www.npmjs.com/package/express>
6. <https://www.w3schools.com/REACT/DEFAULT.ASP>
7. <https://stackoverflow.com/>
8. <https://github.com/>

## Internship\_Final\_Year\_Report.pdf

### ORIGINALITY REPORT

<b>27%</b> SIMILARITY INDEX	<b>25%</b> INTERNET SOURCES	<b>2%</b> PUBLICATIONS	<b>18%</b> STUDENT PAPERS
--------------------------------	--------------------------------	---------------------------	------------------------------

### PRIMARY SOURCES

<b>1</b>	<b>www.00dv00.com</b> Internet Source	<b>5%</b>
<b>2</b>	<b>www.geeksforgeeks.org</b> Internet Source	<b>4%</b>
<b>3</b>	<b>Submitted to Jaypee University of Information Technology</b> Student Paper	<b>2%</b>
<b>4</b>	<b>www.coursehero.com</b> Internet Source	<b>2%</b>
<b>5</b>	<b>Submitted to Baze University</b> Student Paper	<b>2%</b>
<b>6</b>	<b>dev.to</b> Internet Source	<b>2%</b>
<b>7</b>	<b>Submitted to University of Sydney</b> Student Paper	<b>1%</b>
<b>8</b>	<b>www.linuxbymaster.in</b> Internet Source	<b>1%</b>
<b>9</b>	<b>Submitted to University of North Texas</b> Student Paper	<b>1%</b>

10	<a href="http://www.javatpoint.com">www.javatpoint.com</a> Internet Source	1 %
11	Submitted to National Institute of Technology, Kurukshetra Student Paper	1 %
12	Submitted to Higher Education Commission Pakistan Student Paper	1 %
13	Submitted to University of Wales Institute, Cardiff Student Paper	1 %
14	Submitted to Asia Pacific University College of Technology and Innovation (UCTI) Student Paper	1 %
15	<a href="http://www.studocu.com">www.studocu.com</a> Internet Source	<1 %
16	Submitted to CSU, San Jose State University Student Paper	<1 %
17	<a href="http://www.ir.juit.ac.in:8080">www.ir.juit.ac.in:8080</a> Internet Source	<1 %
18	Submitted to University of Wolverhampton Student Paper	<1 %
19	<a href="http://onlinelibrary.wiley.com">onlinelibrary.wiley.com</a> Internet Source	<1 %
20	<a href="http://belvg.com">belvg.com</a> Internet Source	