CHAPTER 1
INTRODUCTION

OBJECTIVE

TO DESIGN DSATM PLACEMENT INFO :

The Purpose of the project is to build a web application program to reduce the manual work for managing student's Placement Records, Reviews, User Logins, Extra Information of the firms, etc.

1.1 WEB TECHNOLOGY AND ITS APPLICATION

A Web based application systems are designed to manage and store information that are used in real-time applications. By different groups of people such as, seals department, programmers or project managers will be let by project applications a controlled access to information and automated distribution of information. The objective for collaboration has been: getting thing done faster, cheaper and better by applying their common knowledge, bringing together a selection of resources and attainments in a project. Because valid collaboration with teams improves productivity, speeds up result-making and optimizes of making a right decisions, it also helps to intercept precious intellectual fortune and time. Web based applications can surprisingly increase performance, productivity and efficiency within an organization. Since web-based applications can be accessed through any web browser, no desktop installation or updates are required. Moreover, developers, who write great code while staying out of the way are able to use it along the distance, while they stay in geographically different place and collaboration between team still exists.

A **web application** (or **web app**) is application software that runs on a web server, unlike computer-based software programs that are run locally on the operating system (OS) of the device. Web applications are accessed by the user through a web browser with an active network connection.

These applications are programmed using a client–server modeled structure—the user ("*client*") is provided *services* through an *off-site server* that is hosted by a third-party. Examples of commonly-used web applications include: web-mail, online retail sales, online banking, and online auctions.

Some among them which are widely used are:-

1) Hospital web application

2) Restaurant web application

3) Salary web application

4) Whole sale web application

5) Bank web application, etc

Thus this project deals with a concept derived from WEB TECHNOLOGY, it is a unique attempt to aid the management of information of students in a university know as DSATM PLACEMENT INFO.

What is DSATM Placement Info ?

A DSATM Placement Info is automation of manual performance record management which enables the user to assess necessary data at any place and any time through internet. The user can access the student details once they run the application therefore the student details is appeared for the user where it shows important information and records in the college like which firm did he or she placed, firm review, interview process and his or her experience,

student batch, details of the student as well as firm, etc. the student can update his review for every firm which enables user to improve his performance in forthcoming semester. The faculty / admin can also make changes in records in case of any mistake immediately which eliminate the time consuming activities like registering a complaint and then faculty approving it then the administration making changes. The admin module allows admin to make changes in all aspects and modules of the application.

1.2 OVER VIEW OF SYSTEM

- **Firm**
  Create firm profiles with limited custom categories and fields including demographic data, name, phone, address and more, and share records with faculties, and administrators.
- **Student**
  Shows the results of every student who gets placed in the firm for a particular semester.
- **Users**
  Provides the User data of respective student. The user module enables the user to update information of any table and all details regarding the student academic result, co-curricular as well as extracurricular activities, and personal data, etc.

- **Admin**

  This feature includes the description of all actions that admin can perform with respect to the web application. He or She will be responsible person to look after any modification performed in the design system of the web application.

CHAPTER 2

REQUIREMENTS ANALYSIS

The requirement analysis specifies the requirements needed to develop a graphic project. In this phase, we collect the requirements needed for designing the project. The requirements collected are then analyzed and carried to the next phase.

2.1 SOFTWARE REQUIREMENTS:

1. Operating System: Mac OS or Windows 7or above
2. Runtime Environment and Library: Node.js
3. Front-end Development: Express (framework), HTML5, CSS3
4. Back-end Development: Mongo DB (No SQL)

2.2 HARDWARE REQUIREMENTS

1. Processor – Pentium IV or above
2. RAM – 2 GB or more
3. Hard disk – 3 GB or more

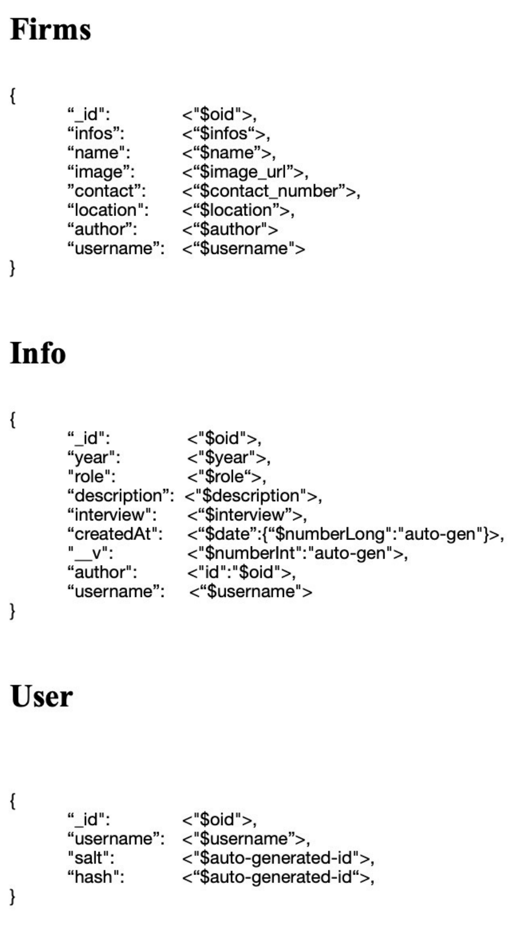
CHAPTER 3

DESIGN

3.1 MONGO DB Normal Data Models

An **Mongo DB – Normal Data model** (**Schema model**) describes inter-related things of interest in a specific domain of knowledge. An Mongo Db Schema model is composed of objects , array types (which classify the things of interest) and specifies relationships that can exist between instances of those objects types. Here data is stored in Jason format which are treated as specific object collections.

DSATM P I there are 3 collections as shown below :



Firms

```
{
        “_id":          <"$oid">,
        “infos”:        <“$infos“>,
        “name":         <“$name”>,
        “image”:        <“$image_url”>,
        ”contact”:      <“$contact_number”>,
        “location":     <“$location”>,
        “author”:       <“$author">
        “username”:     <“$username">
}
```

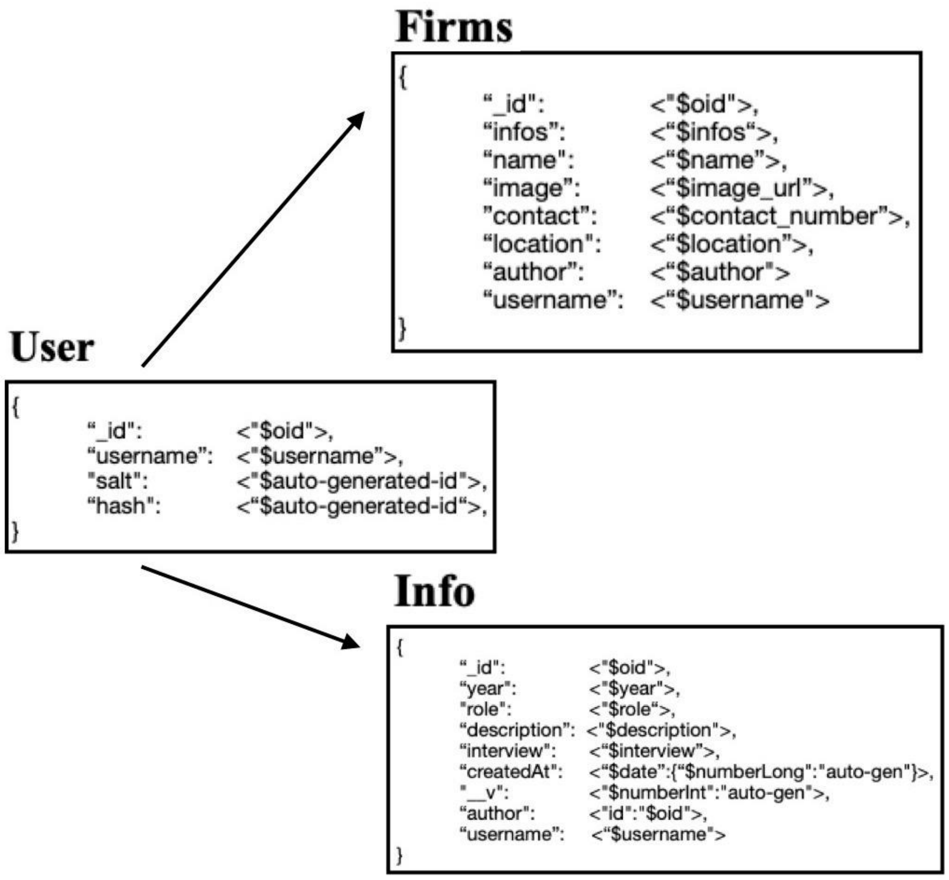
Info

```
{
        “_id":          <"$oid">,
        “year":         <"$year">,
        "role":         <"$role“>,
        “description”: <"$description">,
        “interview":    <“$interview”>,
        “createdAt":    <“$date”:{“$numberLong":"auto-gen"}>,
        "__v":          <"$numberInt":"auto-gen">,
        “author":       <"id":"$oid">,
        “username”:     <“$username">
}
```

User

```
{
        “_id":          <"$oid">,
        “username”:     <"$username”>,
        "salt":         <"$auto-generated-id">,
        “hash":         <“$auto-generated-id“>,
}
```

Figure 1: Collections diagram of DSATM P I



```
Firms
{
        "_id":          <"$oid">,
        "infos":        <"$infos">,
        "name":         <"$name">,
        "image":        <"$image_url">,
        "contact":      <"$contact_number">,
        "location":     <"$location">,
        "author":       <"$author">
        "username":     <"$username">
}
```

```
User
{
        "_id":          <"$oid">,
        "username":     <"$username">,
        "salt":         <"$auto-generated-id">,
        "hash":         <"$auto-generated-id">,
}
```

```
Info
{
        "_id":          <"$oid">,
        "year":         <"$year">,
        "role":         <"$role">,
        "description":  <"$description">,
        "interview":    <"$interview">,
        "createdAt":    <"$date":{"$numberLong":"auto-gen"}>,
        "__v":          <"$numberInt":"auto-gen">,
        "author":       <"id":"$oid">,
        "username":     <"$username">
}
```

Figure 2: Mongo DB Schema Design Diagram of DSATM P I

The above Normalized Data Model diagram describes one primary id of a user is related for all collections and their usage for referencing between documents.

In general, we use normalized data models:

- when embedding would result in duplication of data but would not provide sufficient read performance advantages to outweigh the implications of the duplication.
- to represent more complex many-to-many relationships.
- to model large hierarchical data sets.

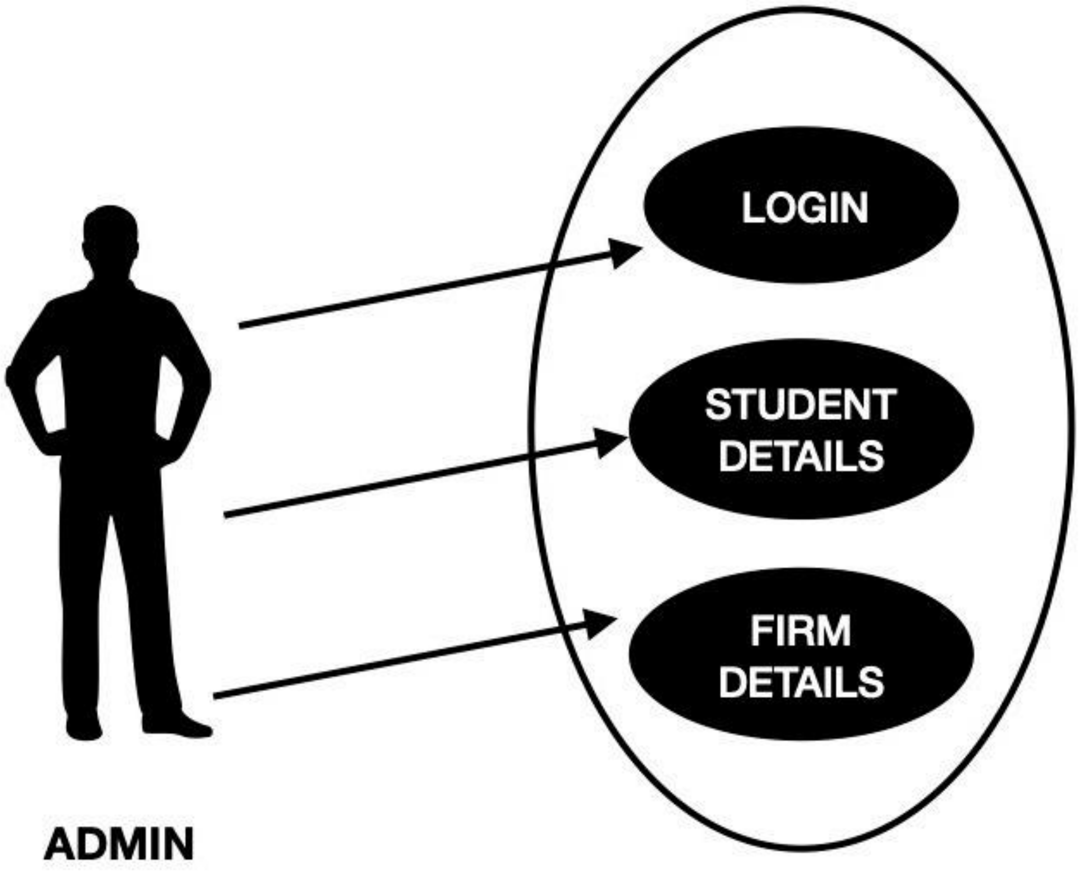


Figure 3: Use case diagram of DSATM P I

CHAPTER 4

IMPLEMENTATION

4.1 INTRODUCTION TO FRONT END TOOL

4.1.1 Express J S AND Node JS : The age of JavaScript is truly upon us. From its humble beginnings as a client-side scripting language, not only has it become completely ubiquitous on the client side, but its use as a server-side language has finally taken off too, thanks to Node. The promise of an all-JavaScript technology stack is clear: no more context switching! No longer do you have to switch mental gears from JavaScript to PHP, C#, Ruby, or Python (or any other server-side language). Furthermore, it empowers frontend engineers to make the jump to server-side programming. This is not to say that server-side programming is strictly about the language: there's still a lot to learn. With JavaScript, though, at least the language won't be a barrier.

EXPRESS : The Express website describes Express as "a minimal and flexible node.js web application framework, providing a robust set of features for building single and multipage and hybrid web applications." What does that really mean, though? Let's break that description down:

Minimal

This is one of the most appealing aspects of Express. Many times, framework developers forget that usually "less is more." The Express philosophy is to provide the *minimal* layer between your brain and the server. That doesn't mean that it's not robust, or that it doesn't have enough useful features. It means that it gets in your way less, allowing you full expression of your ideas, while at the same time providing something useful.

Flexible

Another key aspect of the Express philosophy is that Express is extensible. Express provides you a very minimal framework, and you can add in different parts of Express functionality as needed, replacing whatever doesn't meet your needs. This is a breath of fresh air. So many frameworks give you *everything*, leaving you with a bloated, mysterious, and complex project before you've even written a single line of code. Very often, the first task is to waste time carving off unneeded functionality, or replacing the functionality that doesn't meet requirements. Express takes the opposite approach, allowing you to add what you need when you need it.

Web application framework

Here's where semantics starts to get tricky. What's a web application? Does that mean you can't build a website or web pages with Express? No, a website *is* a web application, and a web page *is* a web application. But a web application can be more: it can provide functionality to *other* web applications (among other things). In general, "app" is used to signify something that has functionality: it's not just a static collection of content (though that is a very simple example of a web app). While there is currently a distinction between an "app" (something that runs natively on your device) and a "web page" (something that is served to your device over the network), that distinction is getting blurrier, thanks to projects like PhoneGap, as well as Microsoft's move to allow HTML5 applications on the desktop, as if they were native applications. It's easy to imagine that in a few years, there won't be a distinction between an app and a website.

Single-page web applications

Single-page web applications are a relatively new idea. Instead of a website requiring a network request every time the user navigates to a different page, a single-page web application downloads the entire site (or a good chunk of it) to the client's browser. After that initial download, navigation is faster because there is little or no communication with the server. Single-page application development is facilitated by the use of popular frameworks such as Angular or Ember, which Express is happy to serve up.

Multipage and hybrid web applications

Multipage web applications are a more traditional approach to websites. Each page on a website is provided by a separate request to the server. Just because this approach is more traditional does not mean it is not without merit or that single-page applications are somehow better. There are simply more options now, and you can decide what parts of your content should be delivered as a single-page app, and what parts should be delivered via individual requests. "Hybrid" describes sites that utilize both of these approaches.

NODE J S : Node.js is a platform built on Chrome's JavaScript runtime for easily building fast and scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

Node.js is an open source, cross-platform runtime environment for developing server-side and networking applications. Node.js applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux.

Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.

```
Node.js = Runtime Environment + JavaScript Library
```

Features of Node JS :

Following are some of the important features that make Node.js the first choice of software
architects.

 Asynchronous and Event Driven − All APIs of Node.js library are asynchronous, that is, non-blocking. It essentially means a Node.js based server never waits for an API to return data. The server moves to the next API after calling it and a notification mechanism of Events of Node.js helps the server to get a response from the previous API call.

 Very Fast − Being built on Google Chrome's V8 JavaScript Engine, Node.js library is very fast in code execution.

 Single Threaded but Highly Scalable − Node.js uses a single threaded model with event looping. Event mechanism helps the server to respond in a non-blocking way and makes the server highly scalable as opposed to traditional servers which create limited threads to handle requests. Node.js uses a single threaded program and the same program can provide service to a much larger number of requests than traditional servers like Apache HTTP Server.

 No Buffering − Node.js applications never buffer any data. These applications simply output the data in chunks.

 License − Node.js is released under the **MIT license**.

Where to use Node.js :

Following are the areas where Node.js is proving itself as a perfect technology partner.

● I/O bound Applications

● Data Streaming Applications

● Data Intensive Real-time Applications (DIRT)

● JSON APIs based Applications

● Single Page Applications

4.1.2 HTML 5 : HTML5 is a mark-up language used to create the web pages or the website. It is used to describe the structure of a website and present the information via internet. HTML5 is the most recent version of HTML and provides necessary tags and functionalities to render the intended information into the website. HTML5 is used to create the static pages of a website using the additional functionalities of the CSS (Cascading Style Sheets). Many new features have been added to the HTML5 compared to the previous versions of HTML which makes HTML5 more powerful and easy to use. One of the main advantages of HTML5 is that HTML5 allows developing applications that easily adapt to different resolutions, screen sizes, aspect ratios and guidelines. Excellent features such as GPS, camera and accelerometer in modern devices can be made use of excellently with HTML5. It has improved the mark-up available for documents and has introduced application programming interfaces (API) and Document Object Model (DOM).

4.1.3 JAVA SCRIPT : JavaScript often abbreviated as JS, is a programming language that conforms to the ECMA Script specification. JavaScript is high-level, often just-in-time compiled, and multiparadigm. It has curly-bracket syntax, dynamic typing, prototype-based object-orientation, and first-class functions. Alongside HTML and CSS, JavaScript is one of the core technologies of the World Wide Web. JavaScript enables interactive web pages and is an essential part of web applications. Most websites use it for client-side page behaviour, and all major web browsers have a dedicated JavaScript engine to execute it.

4.1.4 CSS3 : Cascading Style Sheets (CSS) is a style sheet language used for describing the look and formatting of a document written in a markup language. CSS3 is a latest standard of css earlier versions(CSS2). The main difference between css2 and css3 is follows −

- Media Queries
- Namespaces
- Selectors Level 3
- Color

CSS3 is collaboration of CSS2 specifications and new specifications, we can called this collaboration is **module**. Some of the modules are shown below −

- Selectors
- Box Model
- Backgrounds
- Image Values and Replaced Content
- Text Effects
- 2D Transformations
- 3D Transformations
- Animations
- Multiple Column Layout
- User Interface

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML. CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript. CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file which reduces complexity and repetition in the structural content as well as enabling the .css file to be cached to improve the page load speed between the pages that share the file and its formatting.

4.2 INTRODUCTION TO BACK END TOOL

4.2.1 MONGO D B : MongoDB is a document database designed for ease of development and scaling. The Manual introduces key concepts in MongoDB, presents the query language, and provides operational and administrative considerations and procedures as well as a comprehensive reference section. MongoDB offers both a *Community* and an *Enterprise* version of the database:

- MongoDB Community is the source available and free to use edition of MongoDB.
- MongoDB Enterprise is available as part of the MongoDB Enterprise Advanced subscription and includes comprehensive support for your MongoDB deployment. MongoDB Enterprise also adds enterprise-focused features such as LDAP and Kerberos support, on-disk encryption, and auditing.

Document Database :A record in MongoDB is a document, which is a data structure composed of field and value pairs. MongoDB documents are similar to JSON objects. The values of fields may include other documents, arrays, and arrays of documents.
The advantages of using documents are:

- Documents (i.e. objects) correspond to native data types in many programming languages.
- Embedded documents and arrays reduce need for expensive joins.
- Dynamic schema supports fluent polymorphism.

Collections/Views/On-Demand Materialized Views : MongoDB stores documents in collections. Collections are analogous to tables in relational databases.

In addition to collections, MongoDB supports:

- Read-only Views (Starting in MongoDB 3.4)
- On-Demand Materialized Views (Starting in MongoDB 4.2).

Key Features :

1. High Performance: MongoDB provides high performance data persistence. In particular, **Support for embedded data models reduces I/O activity on database system. Indexes support faster queries and can include keys from embedded documents and arrays.**

2. Rich Query Language : MongoDB supports a rich query language to support read and write operations (CRUD) as well as: Data Aggregation**,** Text Search **and** Geospatial Queries**.**

4.2.2 MONGO CLOUD (MONGO DB ATLAS) :

MongoDB Atlas is a fully-managed cloud database developed by the same people that build MongoDB. Atlas handles all the complexity of deploying, managing, and healing your deployments on the cloud service provider of your choice (AWS, Azure, and GCP). MongoDB Atlas is a fully-managed cloud database developed by the same people that build MongoDB. Atlas handles all the complexity of deploying, managing, and healing your deployments on the cloud service provider of your choice (AWS, Azure, and GCP).

Get Started with Atlas

Deploy a New Cluster	**Migrate an Existing Cluster**	**Connect to Your Atlas Cluster**
Create a cluster for free Load sample data into your cluster Deploy a sharded cluster Deploy a global cluster	Migrate from AWS Migrate from m Lab Migrate from Compose	Connect from your application Perform CRUD with the Atlas UI Connect from a desktop GUI

4.3 WTA CONCEPT USED

HTML (the Hypertext Markup Language) and CSS (Cascading Style Sheets) are two of the core technologies for building this Web pages. Along with graphics and scripting, HTML and CSS are the basis of building Web pages and Web Applications. In addition to the core web technologies are browsers, web servers, protocols, data formats, and APIs.

Information on the Web is stored in documents, using a language named HTML (Hyper Text Markup Language). Web clients interpret HTML and display the documents to a user. The protocol that governs the exchange of information between the Web server and Web client is named HTTP (Hyper Text Transfer Protocol).

Client-Side Scripting / Coding - Client-Side Scripting is the type of code that is executed or interpreted by browsers.

Client-Side Scripting is generally viewable by any visitor to a site (from the apply pass in "Home Page" to search for a pass).

Below are some common Client-Side Scripting technologies:

- HTML (Hyper Text Markup Language)
- CSS (Cascading Style Sheets)
- JavaScript
- Ajax (Asynchronous JavaScript and XML)
- jQuery (JavaScript Framework Library - commonly used in Ajax development)

Front-end web development, also known as client-side development is the practice of producing HTML, CSS and JavaScript for a website or Web Application so that a user can see and interact with them directly. In this project html, ajax, jQuery and JavaScript is used to develop the front end.

J Query aims to control HTML documents. It has a simple API to control events and project animations in browsers. Besides, jQuery is used to manipulate the DOM and also serves as a plug-in tool.

CSS interacts with HTML elements, components of the website.

- For communication with HTML, CSS uses selectors. A selector is the part of the CSS code that defines which HTML element will be affected by the CSS styling.
- The declaration contains properties and values that are used by the selector.
- Properties define font size, colour and margins. Values are the settings for these properties.

CSS are the basis of building Web pages and Web Applications. In addition to the core web technologies are browsers, web servers, protocols, data formats, and APIs.

Information on the Web is stored in documents, using a language named HTML (Hyper Text Markup Language). Web clients interpret HTML and display the documents to a user. The protocol that governs the exchange of information between the Web server and Web client is named HTTP (Hyper Text Transfer Protocol).

Client-Side Scripting / Coding - Client-Side Scripting is the type of code that is executed or interpreted by browsers.

Client-Side Scripting is generally viewable by any visitor to a site (from the apply pass in "Home Page" to search for a pass).

Below are some common Client-Side Scripting technologies:

- HTML (Hyper Text Markup Language)
- CSS (Cascading Style Sheets)
- JavaScript
- Ajax (Asynchronous JavaScript and XML)
- jQuery (JavaScript Framework Library - commonly used in Ajax development)

Front-end web development, also known as client-side development is the practice of producing HTML, CSS and JavaScript for a website or Web Application so that a user can see and interact with them directly. In this project html, ajax, jQuery and JavaScript is used to develop the front end.

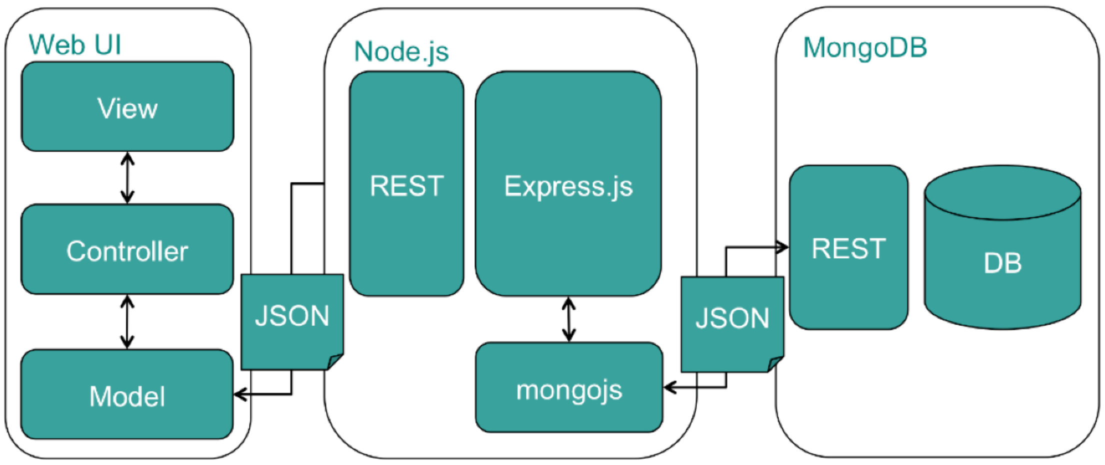
J Query aims to control HTML documents. It has a simple API to control events and project animations in browsers. Besides, jQuery is used to manipulate the DOM and also serves as a plug-in tool.

CSS interacts with HTML elements, components of the website.

- For communication with HTML, CSS uses selectors. A selector is the part of the CSS code that defines which HTML element will be affected by the CSS styling.
- The declaration contains properties and values that are used by the selector.
- Properties define font size, colour and margins. Values are the settings for these properties.

Below are the common Server-Side Scripting technologies: Below are the common Server-Side Scripting technologies:

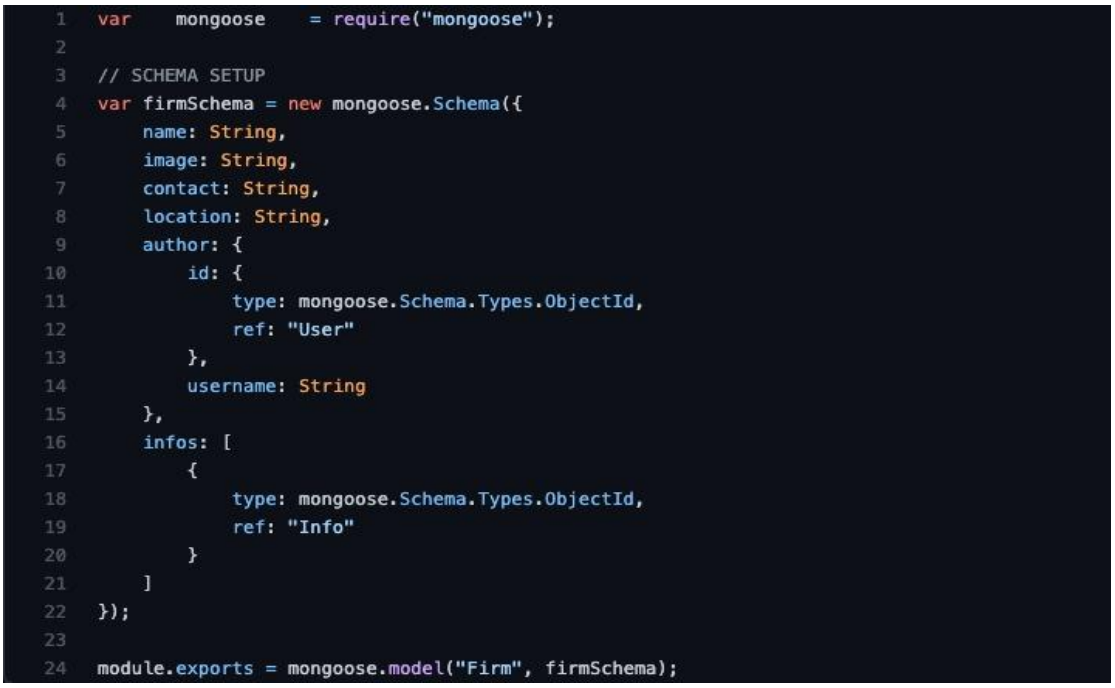
The official **MongoDB Node**. **js** driver allows **Node.js** applications to connect to **MongoDB** and work with data. The driver features an asynchronous API which allows you to access method return values through Promises or specify callbacks to access them when communicating with **MongoDB**.



4.4 MODULES

1. **Firm**

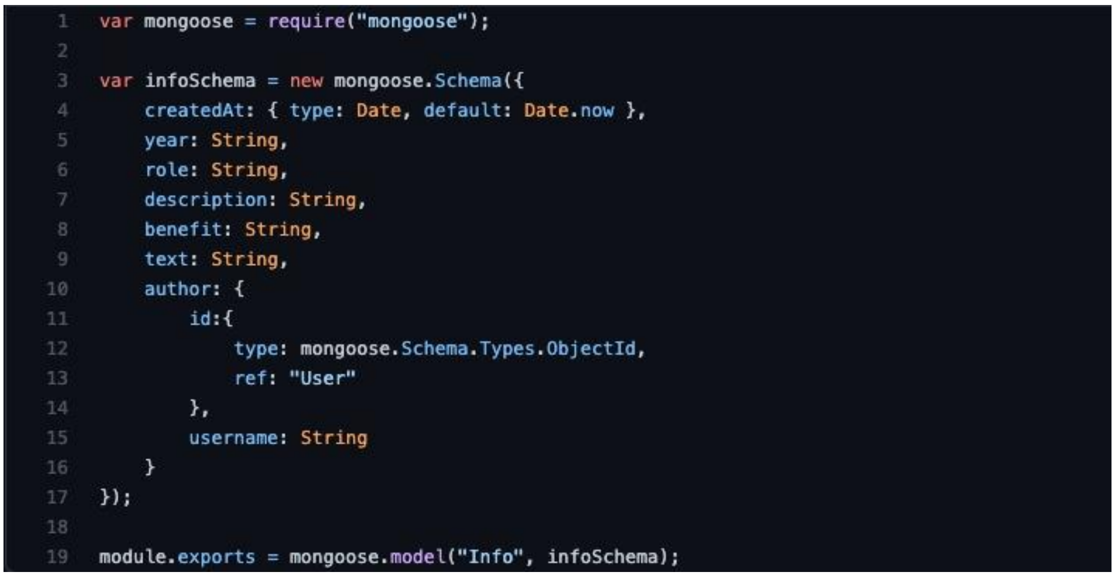
   A Firm page displays the list of firms that enters the campus drive pool and hire the student of DSATM.



```
var    mongoose    = require("mongoose");

// SCHEMA SETUP
var firmSchema = new mongoose.Schema({
    name: String,
    image: String,
    contact: String,
    location: String,
    author: {
        id: {
            type: mongoose.Schema.Types.ObjectId,
            ref: "User"
        },
        username: String
    },
    infos: [
        {
            type: mongoose.Schema.Types.ObjectId,
            ref: "Info"
        }
    ]
});

module.exports = mongoose.model("Firm", firmSchema);
```

2. **Info**

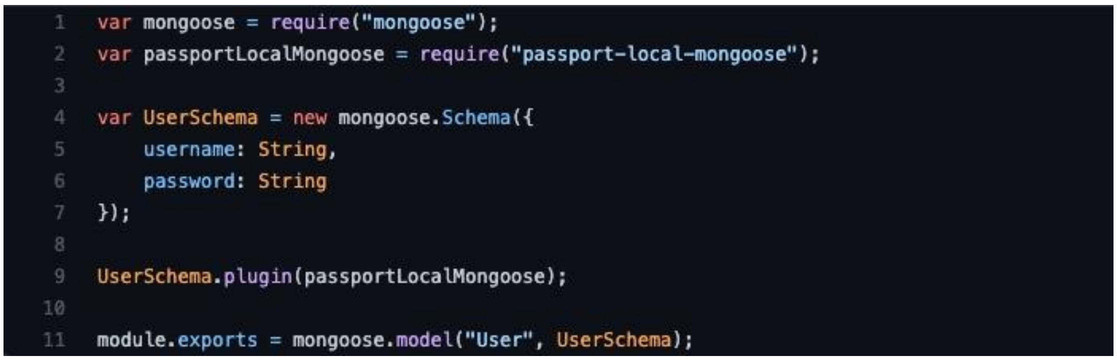
   A Info Page is where student enters the data and that gets stored under that particular firm he or she works.



```
var mongoose = require("mongoose");

var infoSchema = new mongoose.Schema({
    createdAt: { type: Date, default: Date.now },
    year: String,
    role: String,
    description: String,
    benefit: String,
    text: String,
    author: {
        id:{
            type: mongoose.Schema.Types.ObjectId,
            ref: "User"
        },
        username: String
    }
});

module.exports = mongoose.model("Info", infoSchema);
```

3. **User**

   A User Page is allows the user to signup or Login and access the user defined abstracted data.



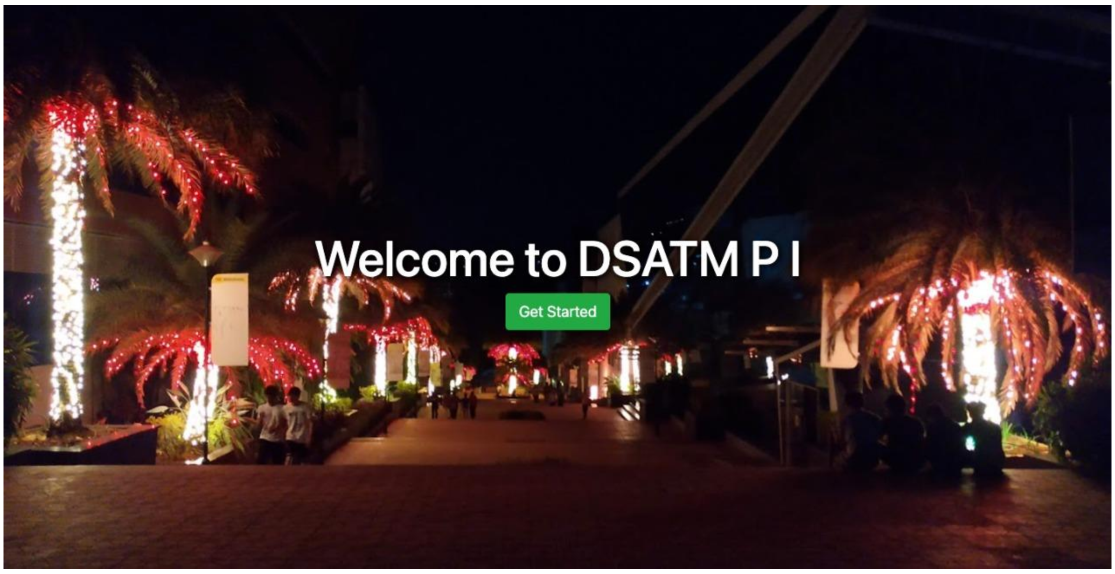
```
var mongoose = require("mongoose");
var passportLocalMongoose = require("passport-local-mongoose");

var UserSchema = new mongoose.Schema({
    username: String,
    password: String
});

UserSchema.plugin(passportLocalMongoose);

module.exports = mongoose.model("User", UserSchema);
```

CHAPTER 5

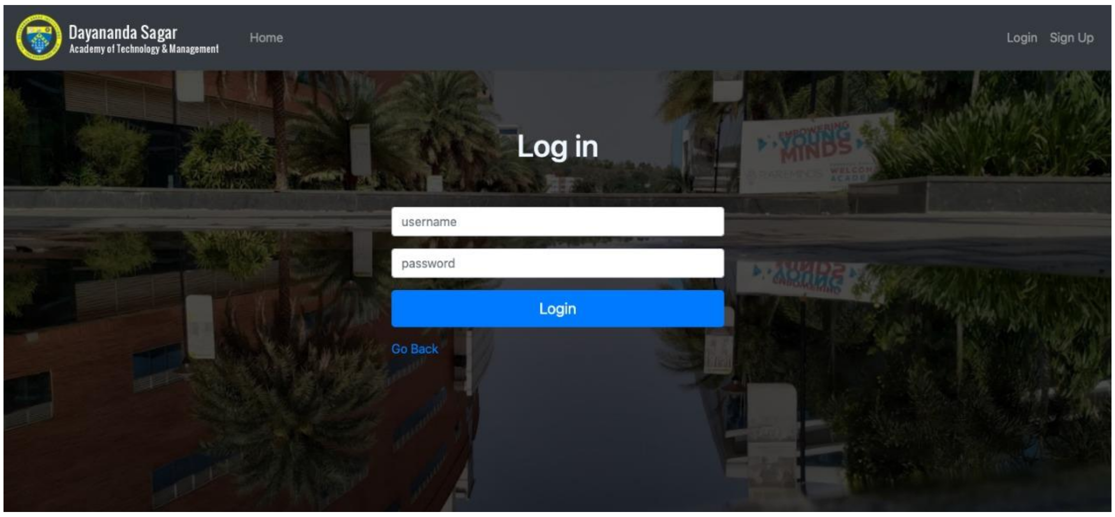
SNAPSHOTS

The user interface design was one of the core tasks of any application. The aim of UI is to provide a graphical-user-friendly interface to the end-users. UI makes it easy for the end users to access the application.

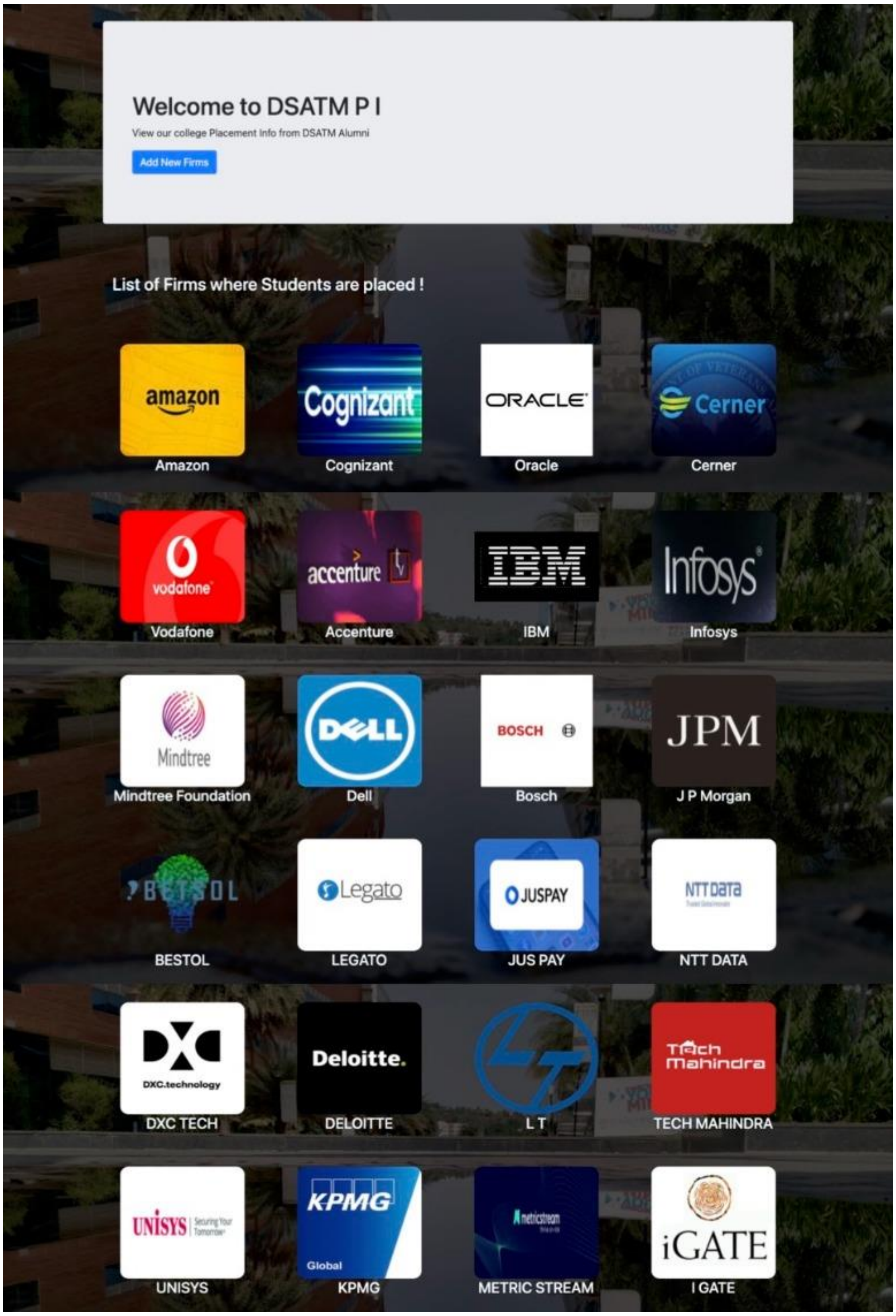
5.1 LANDING PAGE :



5.2 USER LOGIN AND SIGN UP :



5.3 FIRM / HOME PAGE :



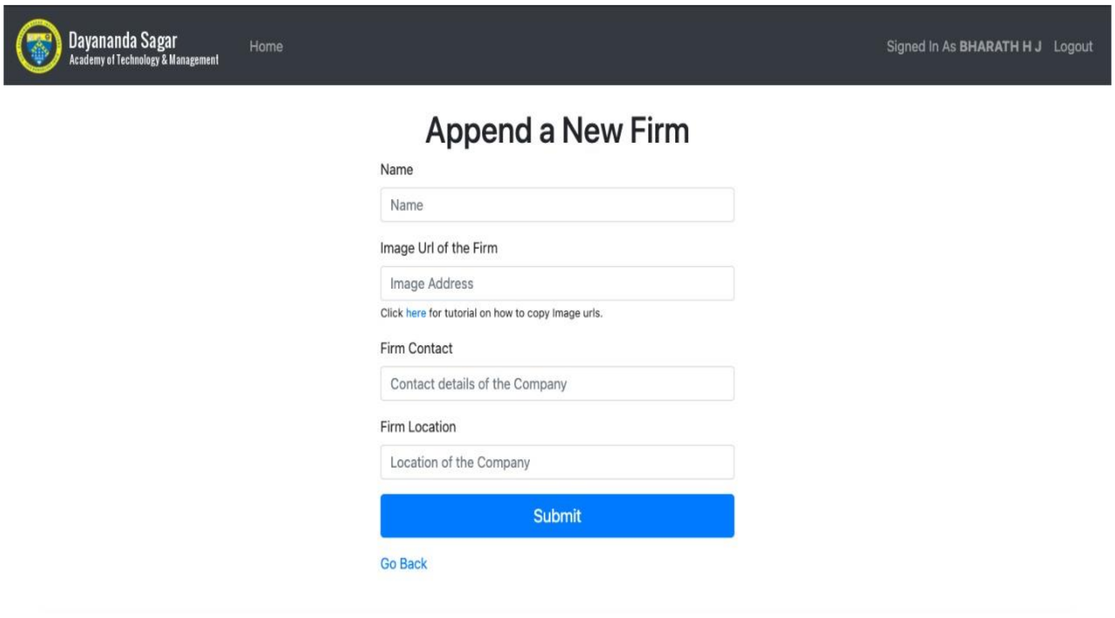
Welcome to DSATM P I

View our college Placement Info from DSATM Alumni

List of Firms where Students are placed !

Amazon
Cognizant
Oracle
Cerner
Vodafone
Accenture
IBM
Infosys
Mindtree Foundation
Dell
Bosch
J P Morgan
BESTOL
LEGATO
JUS PAY
NTT DATA
DXC TECH
DELOITTE
L T
TECH MAHINDRA
UNISYS
KPMG
METRIC STREAM
I GATE

5.4 ADD FIRM DETAILS :


Dayananda Sagar
Academy of Technology & Management
Home
Signed In As **BHARATH H J** Logout

Append a New Firm

Name
Name

Image Url of the Firm
Image Address
Click here for tutorial on how to copy image urls.

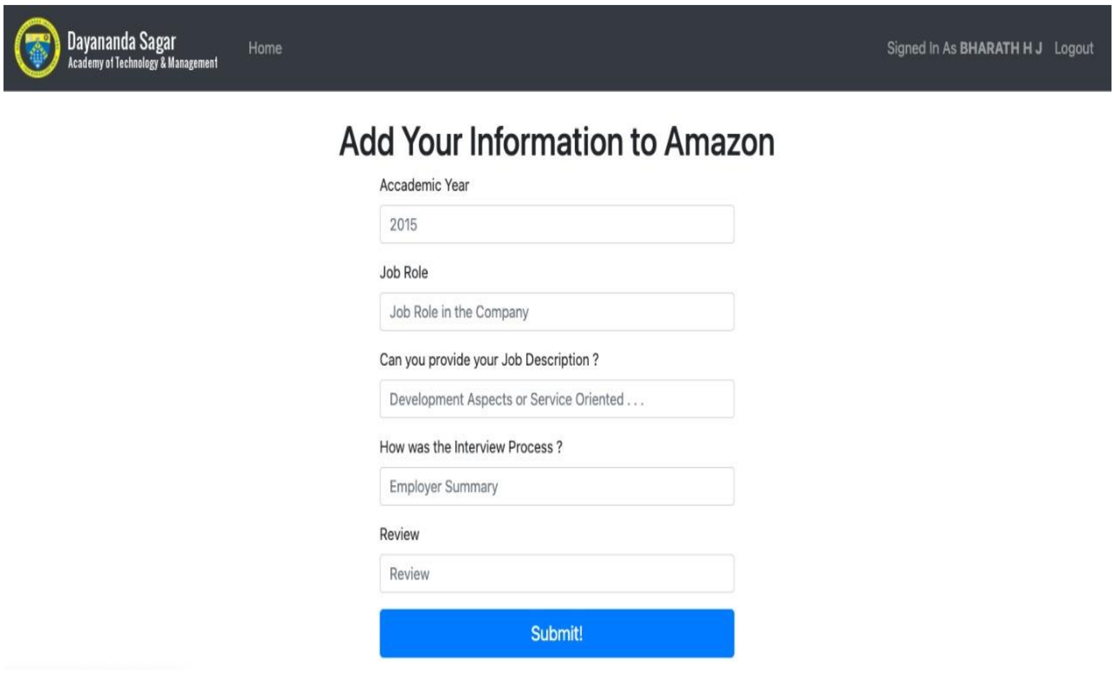
Firm Contact
Contact details of the Company

Firm Location
Location of the Company

Submit

Go Back

5.5 ADD USER DETAILS :


Dayananda Sagar
Academy of Technology & Management
Home
Signed In As **BHARATH H J** Logout

Add Your Information to Amazon

Accademic Year
2015

Job Role
Job Role in the Company

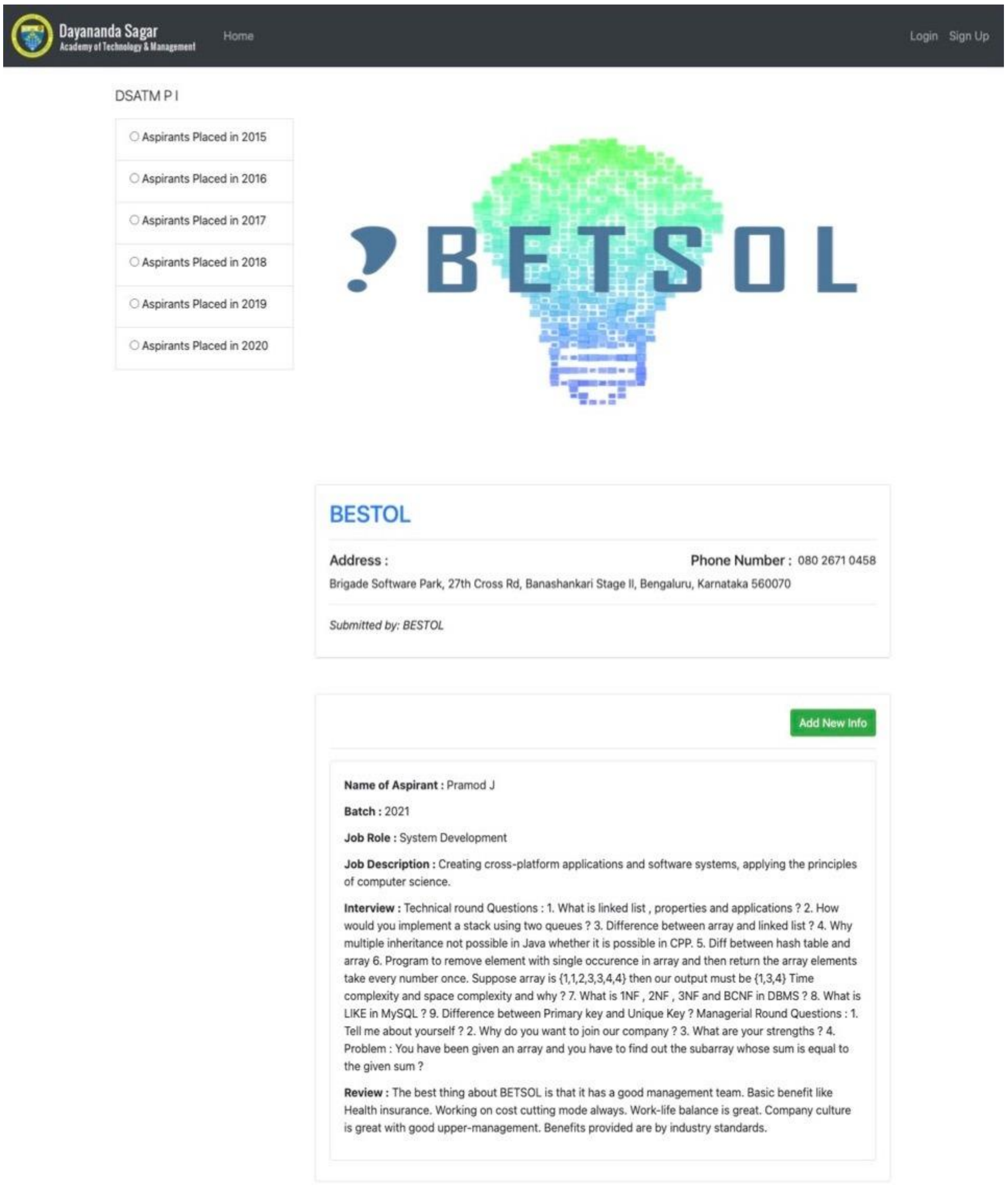
Can you provide your Job Description ?
Development Aspects or Service Oriented . . .

How was the Interview Process ?
Employer Summary

Review
Review

Submit!

5.6 INFO DISPLAY :



BESTOL

Address : Brigade Software Park, 27th Cross Rd, Banashankari Stage II, Bengaluru, Karnataka 560070

Phone Number : 080 2671 0458

Submitted by: BESTOL

Name of Aspirant : Pramod J

Batch : 2021

Job Role : System Development

Job Description : Creating cross-platform applications and software systems, applying the principles of computer science.

Interview : Technical round Questions : 1. What is linked list , properties and applications ? 2. How would you implement a stack using two queues ? 3. Difference between array and linked list ? 4. Why multiple inheritance not possible in Java whether it is possible in CPP. 5. Diff between hash table and array 6. Program to remove element with single occurence in array and then return the array elements take every number once. Suppose array is {1,1,2,3,3,4,4} then our output must be {1,3,4} Time complexity and space complexity and why ? 7. What is 1NF , 2NF , 3NF and BCNF in DBMS ? 8. What is LIKE in MySQL ? 9. Difference between Primary key and Unique Key ? Managerial Round Questions : 1. Tell me about yourself ? 2. Why do you want to join our company ? 3. What are your strengths ? 4. Problem : You have been given an array and you have to find out the subarray whose sum is equal to the given sum ?

Review : The best thing about BETSOL is that it has a good management team. Basic benefit like Health insurance. Working on cost cutting mode always. Work-life balance is great. Company culture is great with good upper-management. Benefits provided are by industry standards.

CONCLUSION AND FUTURE ENHANCEMENTS

CONCLUSION :

The backend-portal is enables its users to access, manage and update student's data effectively and efficiently. It allows for a centralized facility that can easily be modified and quickly shared among multiple users. The backend-portal eliminates the paper work which could lead to loss of data and data redundancy. It also allows the possibility of queries to obtain information for various surveys. Due to the many users reading and modifying student data in the department, Hence it is an ideal use for such a system.

FUTURE ENHANCEMENTS :

The following is just a sample of future opportunities that would help sustain the portal for undergraduates: -

1. By using artificial intelligence the web portal may track the usage of Wi-Fi based on the students registered device.

2. One can deploy this web application into mobile android application and be used in smaller devices like mobile phones, tablets and notepads.

3. In future web portal can be combined with the university internal and external web sites. So that all courses in the university will have single web app.

4. Students can directly fill a resume forum system will use artificial intelligence and sent the resume to companies as per student requirement and eligibility criteria.

REFERENCES

1) The complete reference -Udemy by Yelp Camp Project of Colt Steele

2) Project Deployed link: https://bharath-infotest.herokuapp.com/

3) Mongo DB : https://docs.mongodb.com/

4) Wikipedia:

- https://en.wikipedia.org/wiki/CSS#CSS_3

5). Books :

- Web Development with Node & Express – **Ethan Brown**
- Node J S – **Tutorials Point official documentation**
- Fundamentals of Web Development, 1stEdition, Pearson Education India - **Randy Connolly, Ricardo Hoar**