**JOB PORTAL WEBSITE**

**Index**

1. Introduction

1.1. Job Portal Website Description

1.2. Job Portal Website Key Features

1.3. Basic Steps to connect with Job Portal Website

2. Target Audience of the document

2.1. Product flow diagram of Job Portal Website

2.4. Updated Tabular form

2.5. Milestones Planned- Phase 3

3. Purpose of the document

3.1. Architecture block diagram of Job Portal Website

4.Tools Usability

4.1. Front-end Tools:

4.1.1. Figma UI Designing Tool

4.1.2. Visual Studio Code

4.1.3. Node.js and React

4.2. Back-end Tools:

4.2.1. PostgreSQL

4.2.2. Postman

4.2.3. Python- Django

5. Tools Features

5.1. Front-end Tools:

5.1.1. Figma UI Designing Tool

5.1.2. Visual Studio Code

5.1.3. Node.js and React

5.2. Back-end Tools:

5.2.1. PostgreSQL

5.2.2. Postman

5.2.3. Python- Django

6. Tools Setup process

6.1. Device Firmware

6.1.1. Binwalk

6.1.2. Firmwalker

6.2. Android tools

6.2.1. Static Analyis

6.2.2. Dynamic Analysis

6.2.3. Mara Framework

6.3. Network Scan Tools

6.3.1. Nmap

6.3.2. NetCreds/Password Extractor

6.3.3. Pcap Analyser:

6.4. Web Application

6.4.1. SSLScan

6.4.2. Wapiti

6.4.3. Whatweb

6.4.4. Nikto

6.4.5. Dirbuster

6.5. Thick Client Application

6.5.1. Uncompyle6

***1. Introduction***

**Job Portal Website:**

Job Portal Website is designed for every applicant’s easy access to apply for the desired job. It includes their personal information as well as education details of the applicant. This can be implemented for LTTS also as it be easier to filter the applicant’s based on their interests filled.

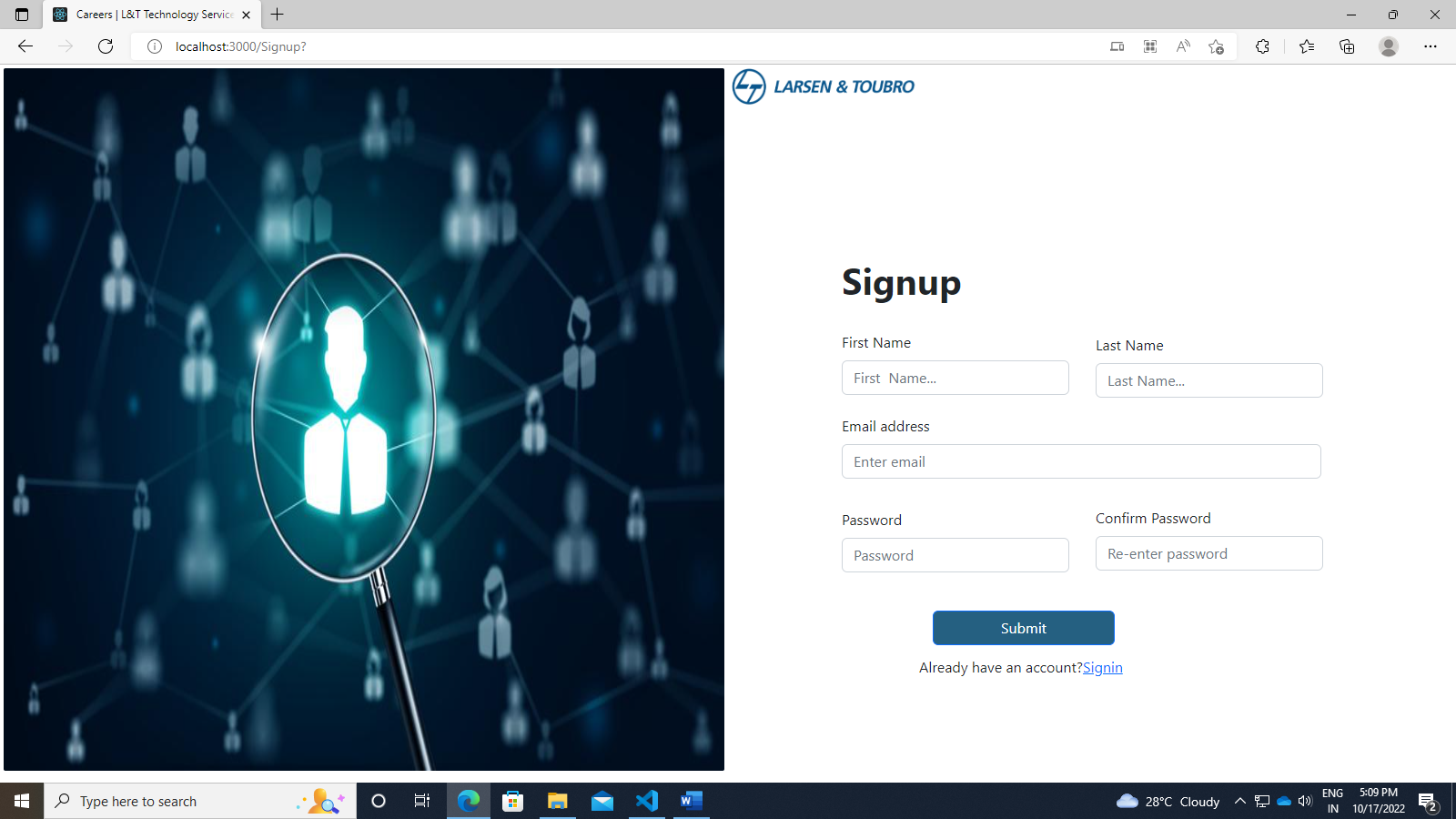
In Job Portal Website the key features are as follows:

* Clear and exact information collection
* Simple moves to change or correct the information if they are given wrong
* Preview of the pages to ensure the information provided is correct
* A strong database support to fetch the loaded information

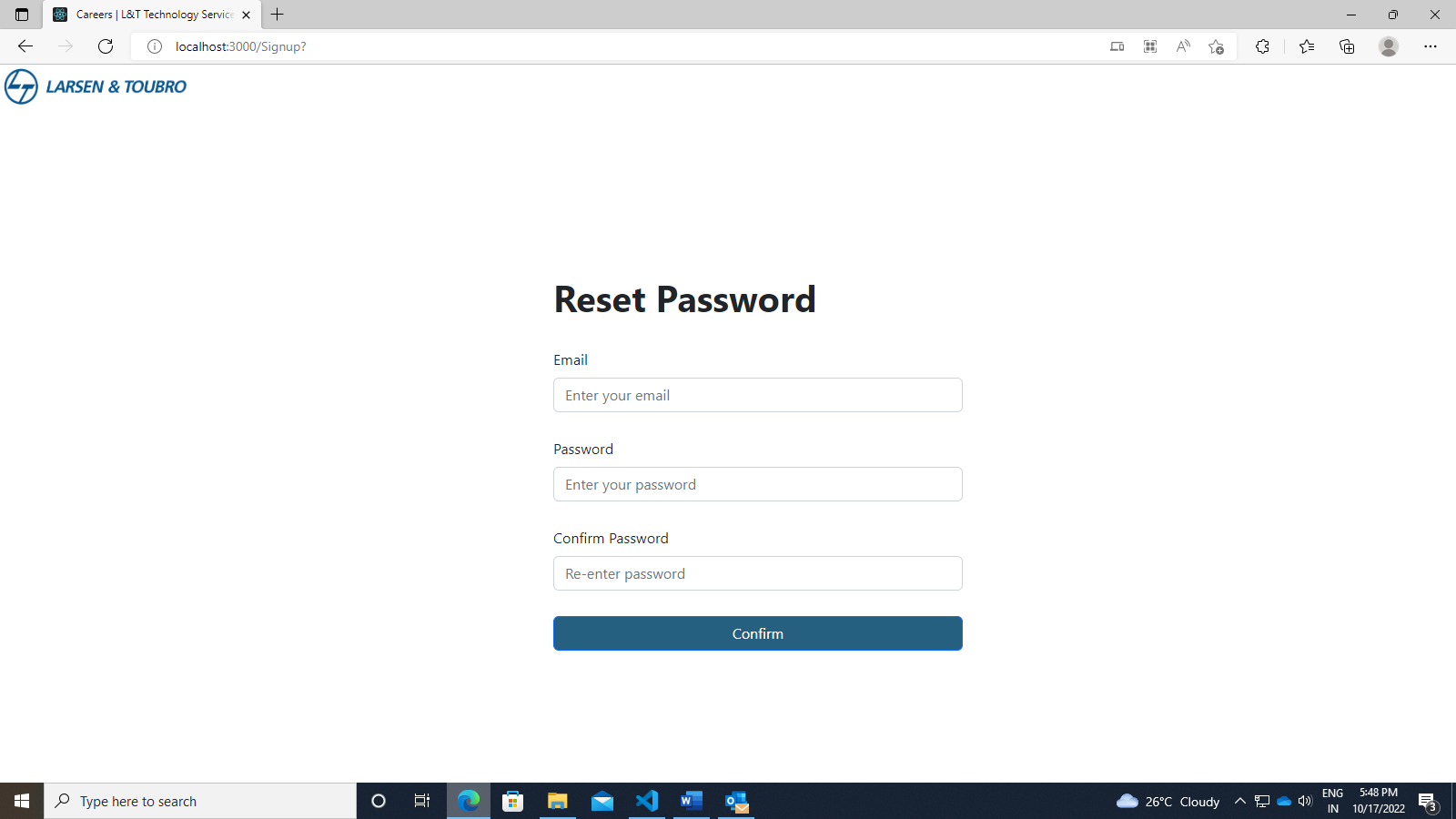
**Basic Steps to connect with Job Portal Website:**

The Job Portal Website needs an account to be created for user and using the same to login.

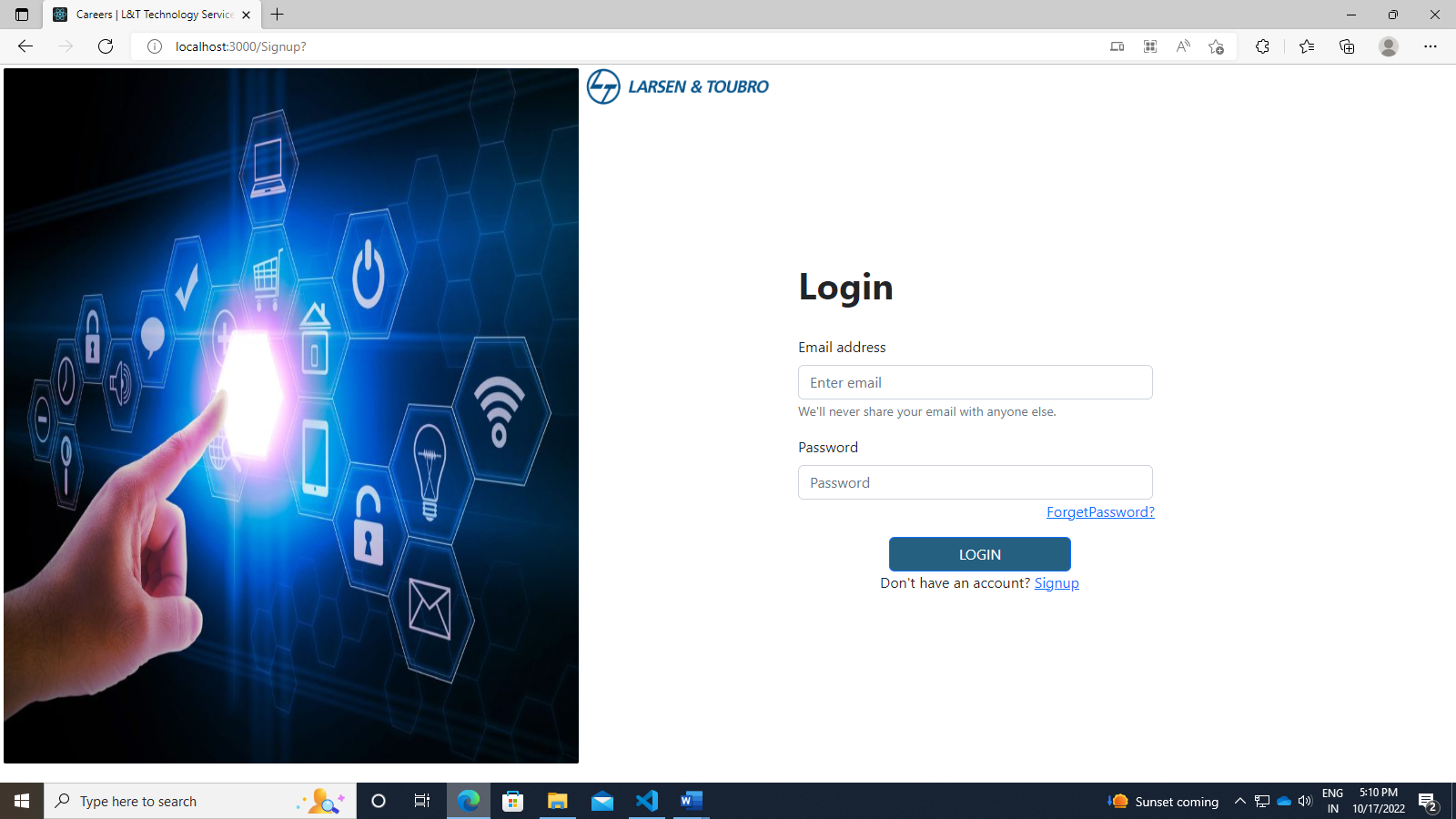
Step 1: To create an account, fill the details and click on Signup.



Step 2: The Forgot Password option is used to retrieve the account.



Step 3: Provide the respective credentials and Login.



***2. Target Audience of the Document***

The work of the user is just to give the correct information and recheck if the given information is correct. It would be comfortable for the users to provide the details as the exact information required is prescribed. The details approved by them is easily storable and can be fetched from the industry side for short listing.

2.1. Product Flow Diagram of Job Portal Website:

User

Create Account

Login

Sign -Up

Confirm

Preview Page

Job Details

Personal Details

***3. Purpose of the Document***

The purpose of the document is to get the exact and required personal and educational information of the user who is in for applying a job. The Job portal is developed in the motive of giving easy and secure means of applying to the preferred job and also the details can be fetched from the industry side using the easy database storing methods.

Architecture Block Diagram of Job Portal Website:

Create Account

User

Login

Sign -Up

Preview Page

Job Details

Personal Details

Internship Details

Upload Resumes

Previous Job Details

Graduation Details

Personal Details

Job Details

Location Details

Contact Details

User’s Details

***4. Tools Usability***

In Job Portal Website, we are using open source tools (OST) as following

4.1. Front-end Tools:

4.1.1. Figma UI Designing Tool

4.1.2. Visual Studio Code

4.1.3. Node.js and React

4.2. Back-end Tools:

4.2.1. PostgreSQL

4.2.2. Postman

4.2.3. Python- Django

**4.1. Front-end Tools:**

The part of a website that the user interacts with directly is termed the front end. It is also referred to as the ‘client side’ of the application. It includes everything that users experience directly: text colors and styles, images, graphs and tables, buttons, colors, and a navigation menu. HTML, CSS, and JavaScript are the languages used for Front End development. The structure, design, behavior, and content of everything seen on browser screens when websites, web applications, or mobile apps are opened up, is implemented by Front End developers.

**4.1.1. Figma UI Designing Tool:**

Figma is a collaborative web application interface designing tool that helps in designing websites and applications.

**4.1.2. Visual Studio Code:**

Visual Studio (VS)Code is a source code editor made by Microsoft with the Electronic framework for Windows, Linux and MacOS. This helps in running code, debugging, code refactoring and embedded git.

**4.1.3. Node.js and React:**

Node.js in an open-source, cross-platform, backend JavaScript runtime environment that runs on a JavaScript engine and executes the JavaScript code outside a web browser, which was designed to build scalable network applications.

React is an open-source front-end JavaScript library for building user interfaces based on UI Components.

**4.2. Back-end Tools:**

The backend is the server-side of the website. It stores and arranges data, and also makes sure everything on the client-side of the website works fine. It is the part of the website that you cannot see and interact with. It is the portion of software that does not come in direct contact with the users. The parts and characteristics developed by backend designers are indirectly accessed by users through a front-end application. Activities, like writing APIs, creating libraries, and working with system components without user interfaces or even systems of scientific programming, are also included in the backend.

**4.2.1. PostgreSQL:**

PostgreSQL is an open-source relational Database management system emphasizing extensibility and SQL Compliance. It is used as primary data storage or data warehouse for web applications.

**4.2.2. Postman:**

Postman is an API platform for designing, building, testing and iterating the API’s. It has the ability to make various types of HTTP requests (GET, POST, PUT and PATCH).

**4.2.3. Python-Django:**

Django is an open-source, Python-based web framework that follows the model-template-views architectural pattern. It enables rapid development of maintainable websites and securing it.

***5. Tools Features:***

In this category the features of the tools used are briefed as follows

**5.1. Front-end Tool Features:**

**5.1.1. Figma UI Designing Tool:**

* The feature set of Figma focuses on user interface and user experience design, with an emphasis on real-time collaboration, utilizing a variety of vector graphics editor and prototyping tools.
* The Figma mobile app for Android and iOS allows viewing and interacting with Figma prototypes in real-time on mobile and tablet devices.
* Figma is a cloud-based design tool that is similar to Sketch in functionality and features, but with big differences that make Figma better for team collaboration.
* Figma also shares live embed code snippets to paste an iFrame in third-party tools.
* Figma is a web-based graphics editing and user interface design app. All kinds of graphic design work from wireframing websites, designing mobile app interfaces, prototyping designs, crafting social media posts, and everything in between.
* Figma is different from other graphics editing tools. Mainly because it works directly on your browser.

**5.1.2. Visual Studio Code:**

Visual Studio (VS) Code supports multiple programming languages. This also means it easily detects if there’s any fault or cross-language reference, it’ll be able to detect it easily. It can detect if any snippet of code is left incomplete. Also, common variable syntaxes and variable declarations are made automatically.

* ***Cross-Platform Support:*** Traditionally, editors used to support either Windows or Linux or Mac Systems. But Visual Studio Code is cross-platform. Therefore, it can work on all three platforms. Also, the code works on all three platforms; else, the open-source and proprietary software codes used to be different.
* ***Extensions and Support:*** Usually supports all the programming languages but, if the user/programmer wants to use the programming language which is not supported then, user/programmer can download the extension and use it.
* ***Repository:*** With the ever-increasing demand for the code, secure and timely storage is equally important. It is connected with Git or can be connected with any other repository for pulling or saving the instances.
* ***Hierarchy Structure:*** The code files are located in files and folders. The required code files also have some files, which may be required for other complex projects. These files can be deleted as per convenience.
* ***Improving Code:*** Some code snippets can be declared a bit differently, which might help the user in the code. This function prompts the user, wherever necessary, to change it to the suggested option.
* ***Terminal Support:*** Many of the times, the user needs to start from the root of the directory to start with a particular action, in-built terminal or console provides user support to not to switch in-between two screens for the same.
* ***Multi-Projects:*** Multiple projects containing multiple files/folders can be opened simultaneously. These projects/folders might or might not be related to each other.

**5.1.3. Node.js and React:**

**Node.js:**

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

* ***Asynchronous and Event Driven*** − All APIs of Node.js library is 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.
* ***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.
* 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.

**React:**

React is a library for building composable user interfaces. It encourages the creation of reusable UI components, which present data that changes over time. React can also render on the server using Node, and it can power native apps using React Native. React implements one-way reactive data flow, which reduces the boilerplate and is easier to reason about than traditional data binding.

* ***JSX*** − JSX is JavaScript syntax extension. It isn't necessary to use JSX in React development, but it is recommended.
* ***Components*** − React is all about components. Everything must be thought as a component. This will help user maintain the code when working on larger scale projects.
* ***Unidirectional data flow and Flux*** − React implements one-way data flow which makes it easy to reason about your app.
* ***License*** − React is licensed under the Facebook Inc. Documentation is licensed under CC BY 4.0.

**5.2. Back-end Tool Features:**

**5.2.1. PostgreSQL:**

PostgreSQL is an advanced, enterprise class open-source relational database that supports both SQL (relational) and JSON (non-relational) querying. It is a highly stable database management system, backed by more than 20 years of community development which has contributed to its high levels of resilience, integrity, and correctness. PostgreSQL is used as the primary data store or data warehouse for many web, mobile, geospatial, and analytics applications.

* Exclusion Constraints
* Explicit Locks, Advisory Locks
* Concurrency, Performance
* Table partitioning
* All transaction isolation levels defined in the SQL standard, including Serializable
* Just-in-time (JIT) compilation of expressions
* Reliability, Disaster Recovery
* Write-ahead Logging (WAL)
* Replication:
* Asynchronous, Synchronous, Logical Point-in-time-recovery (PITR), active standbys
* Tablespaces
* Security
* Authentication: GSSAPI, SSPI, LDAP, SCRAM-SHA-256, Certificate, and more
* Robust access-control system
* Column and row-level security
* Multi-factor authentication with certificates and an additional method
* Extensibility
* Stored functions and procedures
* SQL/JSON path expressions
* Customizable storage interface for tables
* Many extensions that provide additional functionality, including PostGIS
* Case-insensitive and accent-insensitive collations
* Full-text search

**5.2.2. Postman:**

Postman is an API (application programming interface) development tool which helps to build, test and modify APIs. Almost any functionality that could be needed by any developer is encapsulated in this tool. It has the ability to make various types of HTTP requests (GET, POST, PUT, PATCH), saving environments for later use, converting the API to code for various languages like JavaScript, Python).

* ***Accessibility***- One can use it anywhere after installing Postman into the device by simply logging in to the account.
* ***Use Collections***-Postman allows users to build collections for their API-calls. Every set can create multiple requests and subfolders. It will help to organize the test suites.
* ***Test development***- To test checkpoints, verification of successful HTTP response status shall be added to every API- calls.
* ***Automation Testing***-Tests can be performed in several repetitions or iterations by using the Collection Runner or Newman, which saves time for repeated tests.
* ***Creating Environments***- The design of multiple environments results in less replication of tests as one can use the same collection but for a different setting.
* ***Debugging***- To effectively debug the tests, the postman console helps to track what data is being retrieved.
* ***Collaboration***- To enhance the sharing of files the collections and environments can be imported or exported. A direct connection to share the collections can also be used.

**5.2.3. Python-Django:**

Django is a web application framework written in Python programming language. It is based on MVT (Model View Template) design pattern. The Django is very demanding due to its rapid development feature. It takes less time to build application after collecting client requirement. Using Django, web applications can be built in very less time. Django is designed in such a manner that it handles much of configure things automatically, which application development only is focused.

* ***Rapid Development*** - Django was designed with the intention to make a framework which takes less time to build web application. The project implementation phase is a very time taken but Django creates it rapidly.
* ***Secure*** - Django takes security seriously and helps developers to avoid many common security mistakes, such as SQL injection, cross-site scripting, cross-site request forgery etc. Its user authentication system provides a secure way to manage user accounts and passwords.
* ***Scalable*** - Django is scalable in nature and has ability to quickly and flexibly switch from small to large scale application project.
* ***Fully loaded*** - Django includes various helping task modules and libraries which can be used to handle common Web development tasks. Django takes care of user authentication, content administration, site maps, RSS feeds etc.
* ***Versatile*** - Django is versatile in nature which allows it to build applications for different-different domains. Now a days, Companies are using Django to build various types of applications like: content management systems, social networks sites or scientific computing platforms etc.
* ***Open Source*** - Django is an open-source web application framework. It is publicly available without cost. It can be downloaded with source code from the public repository. Open source reduces the total cost of the application development.
* ***Vast and Supported Community*** - Django is a one of the most popular webs frame work. It has widely supportive community and channels to share and connect.

***6.Tools Setup Process (Developer):***

In Job Portal Website, each tool requires a separate installation and has separate process to proceed with.

To Know about each tool in the Job portal website, let’s get into the following open-source network Interface tools (OSNIT) and the idea about the setup process of each tool is briefed hereby.

**6.1. Front-end Tools Setup Process:**

**6.1.1. Figma UI Designing Tool:**

***STAGE 1: SKETCHES -*** This is the quickest, lowest cost and lowest commitment way to kick off your designs. It gets the high-level, top of mind ideas out of your head and onto your paper or screen. Generally, these are roughly captured ideas and if they look pretty bad then you're probably on the right track. It's something you should be able to do in a very short amount of time on paper, a whiteboard or using a wireframing app.

Contents:

* + Outlined boxes
  + Place holder boxes to represent images, text and graphics
  + Can be done in low, mid-or hi fidelity depending on the project needs at this

Purpose:

* + To start off the conversion about design needs and layout
  + Helps to clarify and define features needed and change ideas quickly
  + Helps visualize how screens might look on different device sizes and in different contexts

***STAGE 2: Wireframes or Gray boxing -*** Wireframes are created to show what things look like before visual design principles have been applied. It's where you can start to see the layout and elements taking shape.

Contents:

* + Filler content
  + Actual text
  + Placeholder images or stock images
  + Usually, can be done in your design software using an existing wireframing UI kit or done from scratch

Purpose:

* + Visualize the general layout
  + Establish correct proportions
  + Build trust with stakeholders and help them see the idea more fleshed out
  + Fast and cheap way to create initial ideas or to low-fidelity prototype

***STAGE 3 - Component design -*** Dynamic user interfaces (unlike static website landing pages or marketing sites) require you to think through states and conditionals. It also requires you to be consistent, apply good usability principles and decide the best way to layout and organize and disclose information and actions. This is where thinking in components and atomic elements and database views can really help.

Contents:

* UI kits
* Symbols (buttons, badges)
* Element states (hover state, clicked state)
* Components (like tables, lists, cards, forms)
* Actual text

Purpose:

* + Discover and address usability issues,
  + Uncover accessibility and responsive problems
  + Translate ideas into consistent elements that can be shared with engineers and more easily and accurately discussed and shared across teams.

***STAGE 4 - User Flows and Task flows -*** This is about understanding the user's mental models and your system's model and the orchestration of the pathways and responses that your interface will provide.

Contents:

* + Wireframes
  + Flow charts
  + Wire flows
  + Arrows
  + Conditionals

Purpose:

* + To show navigation routes
  + To check for missing states and information
  + To visualize entry, exist and decision points for your user/customer journey

***STAGE 5– Prototype -*** This tie everything together and shows how the app is expected to look and behave.

Contents:

* + All screens hi-fidelity
  + Target points to mimic what it will be like when the user interacts with the real thing

Purpose:

* Links together all of the screen and flows in a simulated environment
* Ensures that everything looks good, work well and flows as intended before it goes to production.
* This can be considered the pre-code or no-code version of your product.

**6.1.2. Visual Studio Code and node.js:**