ENUGU STATE UNIVERSITY OF SCIENCE AND TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE

REPORT ON THE STUDENT INDUSTRIAL WORK EXPERIENCE SCHEME ON BACKEND WEB DEVELOPMENT

(SIWES)

AT

LEYOOK GLOBAL TECHNOLOGY INDEPENDENCE LAYOUT ENUGU

BY

OKOLIE IFEANYI VICTOR

2019030189203

CERTIFICATION

This is to certify that the acquisition skill titled “Student Industrial Work Experience Scheme’ at Leyook Global Tech, was carried out by OKOLIE IFEANYI VICTOR with the matriculation number 2019030189203 of the department of computer science, Enugu State University of Science and Technology , during the 2022/2023 Student Industrial Work Experience Scheme.

DEDICATION

In life, Three entities matters most ; first God almighty, second parents, and third friends. I dedicate this report to god almighty for his unlimited grace, , consistent love, immeasurable faithfulness and for sparing my life throughout the period of my SIWES program, secondly to my dear parents for their unwavering support and unquantifiable assistance throughout the whole exercise and also my beloved friends who are always there for me.

Acknowledgment

One of the major lessons I have come to learn as a person that “A thousand miles begin with a step”.

The completion of this training was made possible because of some special person who have been helpful in not just my life but also all other student.

They include; my colleagues who have been very motivating to me . I am grateful to you all.

I would like to express my heartfelt gratitude to my supervisor, J.N Nnadozie, for their guidance, support, and encouragement throughout my internship. Their invaluable insights and expertise have provided me with the tools and knowledge necessary to complete this report. I am deeply grateful for their unwavering support and commitment to my growth and development as an intern. Thank you for being an inspiration and a mentor. This report would not have been possible without your support.

ABSTRACT

The Student Industrial Work Experience Scheme (SIWES) is a well-established program designed to provide students with practical experience in their field of study. This report aims to highlight the relevance of SIWES to the Department of Information Technology, demonstrating how it benefits both students and the industry. Through a review of literature and case studies, it will be shown that SIWES provides students with hands-on experience and exposure to the latest technologies, helping them to better understand the real-world applications of their coursework. Additionally, SIWES helps to bridge the gap between academia and industry, allowing students to build professional relationships and networks that can be valuable in their future careers. Ultimately, the report argues that SIWES is a crucial component of the educational experience for Information Technology students and should continue to be supported and promoted within the department.

Table of Contents

Chapter one

Concept and meaning of SIWES

The Students' Industrial Work Experience Scheme (SIWES) was founded to be a skill training program to help expose and prepare students of universities, Polytechnics and colleges of education for the industrial work situation to be met after graduation. It provides students studying technical and vocational courses with the opportunity to gain practical work experience in industries related to their field of study. The goal of the program is to bridge the gap between academic theory and practical skills, allowing students to better understand and apply their theoretical knowledge in real-world situations.

The concept of SIWES is to provide students with practical work experience in their field of study to complement their academic learning and give them a better understanding of the application of their theoretical knowledge in real-world situations. The program aims to bridge the gap between theory and practice, helping students transition from the classroom to the workplace. The program is designed to provide students with hands-on training, exposure to industry practices and standards, and opportunities to develop key technical, interpersonal and professional skills.

The minimum duration for SIWESS should normally be 24 weeks (**6 months**) at a stretch. The period is longer for engineering and technology programs. The ITF will not pay for any attachment period that is less than 24 weeks.

The main objective of this program is to equip students with the necessary practical knowledge and technical skills for self-employment and effective involvement in Nigeria's industrial growth whether in entrepreneurship or being employed in an organization or company

The Students Industrial Work Experience Scheme (SIWES) is a unit under the Vice-Chancellor’s Office. It was established in 2016. The Students Industrial Work Experience Scheme (SIWES) is a skills training programme designed to expose and prepare students of universities and other tertiary institutions for the Industrial Work situation they are likely to meet after graduation.

The Students Industrial Work Experience Scheme (SIWES), is the accepted training program, which is part of the approved Minimum Academic Standard in the various degree programs for all Nigerian Universities. The scheme is aimed at bridging the existing gap between theory and practice of Sciences, Agriculture, Medical Sciences (including Nursing), Engineering and Technology, Management, Information and Communication Technology, and other professional educational programs in the Nigerian tertiary institutions. It is aimed at exposing students to machines and equipment, professional work methods, and ways of safeguarding the work areas and workers in industries, offices, laboratories, hospitals, and other organizations.

It is a cooperative industrial internship program that involves institutions of higher learning, industries, the Federal Government of Nigeria, the Industrial Training Fund (ITF), and the Nigerian Universities Commission (NUC).

Brief History of SIWES

SIWES was founded in 1973 by ITF (Industrial Training Funds) to address the problem of tertiary institution graduates' lack of appropriate skills for employment in Nigerian industries. The Students' Industrial Work Experience Scheme (SIWES) was founded to be a skill training program to help expose and prepare students of universities, Polytechnics and colleges of education for the industrial work situation to be met after graduation.

This system facilitates the transfer from the classroom to the workplace and aids in the application of knowledge. The program allows students to become acquainted with and exposed to the experience required in handling and operating equipment and machinery that are typically not available at their schools.

Prior to the establishment of this scheme, there was a rising concern and trend among industrialists that graduates from higher education institutions lacked appropriate practical experience for employment. Students who entered Nigerian universities to study science and technology were not previously trained in the practical aspects of their chosen fields. As a result of their lack of work experience, they had difficulty finding work.

As a result, employers believed that theoretical education in higher education was unresponsive to the needs of labor employers. Thousands of Nigerians faced this difficulty till 1973. The fund's main motivation for establishing and designing the scheme in 1973/74 was launched against this context.

The ITF (Industrial Training Fund) organization decided to aid all interested Nigerian students and created the SIWES program. The federal government officially approved and presented it in 1974. During its early years, the scheme was entirely supported by the ITF, but as the financial commitment became too much for the fund, it withdrew in 1978. The National Universities Commission (NUC) and the National Board for Technical Education (NBTE) were given control of the scheme by the federal government in 1979. The federal government handed over supervision and implementation of the scheme to ITF in November 1984. It was taken over by the Industrial Training Fund (ITF) in July 1985, with the federal government bearing entire responsibility for funding.

One requirement for the Bachelor of Engineering or Science award is that students must complete at least 24 weeks of Industrial Training.

In most institutions, SIWES is done at the end of the 2nd-semester examination of 300, 400, or 500 levels. The time and duration are to be worked out jointly by each university, department, the SIWES unit, and the ITF.

Purpose, aims and objective of SIWES

The Industrial Training Fund’s Policy Document No. 1 of 1973 which established SIWES outlined the objectives of the scheme as:

- Provide an avenue for students in Institutions of higher learning to acquire industrial skills and experience in their respective courses of study.
- Prepare students for the Industrial Work situation they are likely to experience after graduation.

- Expose students to work methods and techniques of handling equipment and machinery that may not be available in their Institutions.
- Make the transition from school to the world of work easier; and enhance students' networks for later job placements.
- Provide students with an opportunity to apply their knowledge to real work situations, thereby bridging the gap between theory and practice; and
- Enlist and strengthen Employers' involvement in the entire educational process; thereby preparing the students for employment in Industry and Commerce.

Functions/ activities of SIWES center

1. Develop, implement, and regularly review guidelines for SIWES.
2. Registration of eligible students for Industrial Training (IT).
3. Compilation of list of students from different Colleges for SIWES.
4. Timely collection, completion, and submission of all ITF forms/ documents (master list, placement list, direct e-payment form, ITF form 8) to the supervising ITF office.
5. Identify placement opportunities for students and assist in the placement of students on attachment with employers.
6. Issue introductory letters to students for the employers.
7. Organize orientation programmes for all students going for IT in collaboration with ITF
8. Ensure that students have all required documents for successful placement and completion of IT training before embarking on SIWES.
9. Ensure the master placement list is timely prepared and submitted to the Industrial Training Fund and National Universities Commission yearly (not later than 3 months before the commencement of Industrial Attachment).

10. Organize and coordinate supervisory visits to students at I. T. sites.
11. Ensure students' SIWES logbooks are examined, vetted, and signed by University Supervisors, Industry-based Supervisors, and ITF staff.
12. Effectively follow up ITF on all payments to students and the University.
13. Capture student's bank details at the point of registration for SIWES.
14. Develop and sustain the right attitude and mindset among supervisors thus motivating them to effectively play their supervisory role to the maximum benefit of students during SIWES.
15. Prepare and submit reports on the scheme to the ITF after the programme.
16. Resolve problems arising from Industrial Training during and after the training.
17. Develop and track relevant data on students' SIWES to facilitate the development of a SIWES database for the University.
18. Ensure accreditation of MTU SIWES Center by NUC.
19. Work with relevant Colleges/ Departments to ensure accreditation of courses for approved SIWES programme.
20. Liaise and build a good relationship between the University and relevant organizations (NUC, ITF, Industries, etc.).

Objective of this report

The objective of this report is to document and evaluate the practical training experience of me during my internship program. The report serves as a record of the me learning and development during the training period and provides a comprehensive overview of their achievements and challenges. The report is typically submitted to my academic supervisor or training coordinator at the end of the internship program. The main purposes of the report are to:

1. Evaluate the effectiveness of the internship program in achieving its goals and objectives.
2. Provide me with a platform to reflect on their learning experiences and demonstrate their understanding of the practical applications of their academic knowledge.
3. Assess my performance and provide feedback to help them improve their skills and competencies.
4. Facilitate the transfer of knowledge and skills from the workplace to the academic environment.
5. Provide me with a permanent record of their training experience and achievements for future reference.

Chapter Two

Background of Leyook Technology

The field of web development has grown significantly over the past decade and continues to be one of the most in-demand careers in the tech industry. With the rise of dynamic and interactive web applications, it has become increasingly important for web developers to have a strong understanding of the backend technologies that power these applications LEYOOK GLOBALTECH was incorporated in Enugu, Nigeria with Registration Number 2719523. It was registered on 12 Dec 2018 and it's current status is ACTIVE. Company's registered office address is No 5 Sibeudu Street Independence Layout.

Objective

The main objective of this training report is to provide a comprehensive introduction to backend web development using the Python programming language and the Django framework. The report covers the fundamental concepts of Python and Django, as well as advanced topics such as the Django Rest Framework and deployment. By the end of the report, the reader should have a solid understanding of how to build and deploy a backend web application using Python and Django.

Scope

This training report covers the following topics:

- Overview of the basic frontend tools and concept (html, css, JavaScript)

- Introduction to Python and its data types, variables, operators, control structures, functions, and modules

- Introduction to Django, including its architecture, URL routing, models and databases, views and templates, and forms

- Building a Django application, including project setup, designing models and databases, creating views and templates, implementing user authentication, integrating forms, and debugging and testing

- Advanced Django topics, including the Django Rest Framework, deployment (both local and remote), and security

Chapter Three

Overview of basic frontend concept(html, css, JavaScript)

HTML

HTML (Hypertext Markup Language) is a standard markup language used to create web pages. It provides the structure and content of a webpage. making it a fundamental part of web development.

In HTML, tags are used to define the structure and content of a webpage. For example, **<p>** tags define a paragraph of text, **<h1>** tags define a heading, and **<img>** tags define an image. These tags are surrounded by angle brackets and have a corresponding closing tag, such as **</p>**, **</h1>**, or **</img>**.

HTML5, the latest version of HTML, introduces new tags and features, such as **<header>**, **<nav>**, **<article>**, **<section>**, **<aside>**, **<footer>**, **<figure>**, **<figcaption>**, **<time>**, and **<video>**. These tags provide a semantic meaning to the structure of the webpage, making it easier for search engines and assistive technologies to understand the content.

HTML also supports the use of attributes, which provide additional information about an element. For example, the **src** attribute in an **<img>** tag specifies the URL of the image to be displayed. The **alt** attribute provides a text description of an image, which is important for accessibility and search engine optimization.

Here's an example of a simple HTML document:

```
<!DOCTYPE html>
<html>
  <head>
    <title>My Webpage</title>
  </head>
  <body>
    <h1>Welcome to My Webpage</h1>
    <p>This is some sample text.</p>
  </body>
</html>
```

HTML consists of a series of elements, each of which represents a specific type of content or structure. The most common HTML elements include:

- **<html>**: The root element of an HTML document.
- **<head>**: The head of an HTML document, which contains metadata such as the title of the page, information about the author, and links to other resources such as CSS files.
- **<title>**: The title of the document, which appears in the title bar of the web browser.
- **<body>**: The main content of the document, which contains all the text, images, and other elements that are visible to the user.
- **<header>**: A container element that represents the header of a document or section.
- **<nav>**: A container element for navigation links.
- **<main>**: The main content of the document, which should be unique to the document and should not be repeated across multiple documents.
- **<section>**: A container element for a standalone section of a document, such as a chapter or a set of related content.
- **<article>**: A container element for a self-contained piece of content, such as a blog post or a news story.

- **<aside>**: A container element for content that is tangentially related to the main content of the document, such as a sidebar.
- **<footer>**: A container element for the footer of a document or section.
- **<h1>** to **<h6>**: Headings of different levels, used to give structure to the content of the document.
- **<p>**: A paragraph element, used to contain text.
- **<ul>**: An unordered list, used to contain a set of related items, represented by **<li>** elements.
- **<ol>**: An ordered list, used to contain a set of related items, represented by **<li>** elements, which are numbered.
- **<li>**: A list item, used to represent a single item within a list.
- **<img>**: An image element, used to display an image on the page.
- **<a>**: An anchor element, used to create a link to another page or to another resource.
- **<div>**: A generic container element that can be used to group together other elements to form a block-level structure.
- **<span>**: A generic inline container element that can be used to group together other elements to form an inline structure.

HTML also includes a set of attributes that can be used to further customize the behavior of elements. For example, the **src** attribute can be used to specify the source URL for an **<img>** element, and the **href** attribute can be used to specify the destination URL for a link created with an **<a>** element.

It is important to use HTML correctly and semantically, to ensure that the content is accessible to users with disabilities and that the structure of the document is well-defined and easy to understand. This can be achieved by using the appropriate elements for the type of content being represented and by using descriptive and meaningful names for the elements and their attributes.

CSS(Cascading style Sheet)

CSS (Cascading Style Sheets) is a style sheet language used to describe the look and formatting of a document written in HTML. CSS is used to control the layout, colors, fonts, and other visual elements of a webpage.

CSS allows web developers to separate the presentation of a webpage from its content, making it easier to maintain and update the look of a website. CSS is an essential tool for web designers and developers, as it provides a way to style HTML elements in a consistent manner, giving the website a professional and polished appearance.

In CSS, styles are applied to HTML elements through the use of selectors and declarations. Selectors identify the HTML elements that the styles should be applied to, and declarations describe the styles that should be applied.

For example, to set the background color of a webpage to light gray and the text color to blue, you could use the following CSS code:

```
body {
  background-color: lightgray;
  color: blue;
}
```

CSS provides a variety of units for specifying lengths, such as pixels (px), points (pt), and percentages (%). CSS also provides a wide range of layout and positioning options, including floating, absolute positioning, and flexbox. CSS can also be used to create animations, transformations, and other effects on elements.

In addition to styling individual elements, CSS provides a number of ways to control the layout and flow of a webpage, such as setting margins and padding, controlling the flow of elements on the page, and controlling the size and placement of elements on the page.

CSS also provides a way to create responsive designs that adjust to different screen sizes and devices. This can be accomplished through the use of media queries, which allow styles to be applied based on specific conditions, such as the screen size or orientation of the device being used.

Overall, CSS is a powerful and flexible tool for web design and development, and is essential for creating attractive and functional websites

JavaScript

JavaScript is a client-side scripting language that is widely used for creating interactive and dynamic web pages. It can manipulate HTML and CSS, respond to user input, and perform other tasks that can make websites more dynamic and user-friendly. Here are some key concepts related to JavaScript in web development:

1. Variables: Variables are containers that store values. You can use variables to store data, such as text, numbers, and objects.

2. Data Types: JavaScript has several data types, including strings, numbers, Booleans, arrays, and objects. Understanding the different data types and how to work with them is important for writing effective JavaScript code.

3. Operators: Operators are symbols that perform operations on values and variables. JavaScript supports several types of operators, including arithmetic operators, assignment operators, and comparison operators.

4. Control Flow: Control flow refers to the order in which the statements in a program are executed. JavaScript provides several control structures, including if/else statements, for loops, and while loops, that allow you to control the flow of execution based on certain conditions.

5. Functions: Functions are reusable blocks of code that can be called from anywhere in your program. Functions allow you to encapsulate complex logic into a single, reusable unit, making your code more organized and easier to maintain.

6. Objects: Objects are data structures that allow you to group related data together. JavaScript objects can contain properties and methods, which are like variables and functions, respectively.
7. Event Handling: Event handling refers to the process of responding to events, such as a user clicking a button or a page loading, in a web page. JavaScript provides several event handling mechanisms, such as event listeners and handlers, that allow you to detect and respond to events.
8. DOM Manipulation: The Document Object Model (DOM) is a tree-like representation of a web page. JavaScript provides several methods for manipulating the DOM, allowing you to change the content, structure, and style of a web page in response to user interaction.
9. AJAX: AJAX (Asynchronous JavaScript and XML) is a technique for creating fast and dynamic web pages without having to reload the entire page. JavaScript, along with other technologies such as XML and HTTP requests, is used to perform AJAX operations.

These are some of the basic concepts related to JavaScript in web development. It's an extensive language with many features and capabilities, and a deep understanding of JavaScript is essential for writing effective web applications.

Here's a simple example of JavaScript code:

```
document.querySelector("button").addEventListener("click", function() {

  document.querySelector("h1").innerHTML = "Hello, World!";

});
```

JavaScript is primarily used for creating interactive and dynamic web pages. It can be used to add behavior to web pages, including:

1. User Interaction: JavaScript can respond to user actions, such as clicks, hovers, and form submissions, and dynamically update the content of a web page in response.

2. Dynamic Content: JavaScript can be used to dynamically update the content of a web page based on data from an API, user input, or other sources.
3. Form Validation: JavaScript can be used to validate user input in forms, ensuring that required fields are filled out and that data is entered in the correct format.
4. Animations and Effects: JavaScript can be used to create animations and special effects on web pages, such as sliding menus, fading elements, and image carousels.
5. Maps and Charts: JavaScript can be used to create interactive maps and charts that display data in a graphical form.
6. Games: JavaScript can be used to create simple browser-based games.
7. Desktop and Mobile Applications: JavaScript can also be used to create desktop and mobile applications using technologies such as Electron and React Native.

These are just a few examples of what JavaScript can be used for. It is a versatile language with many features and capabilities, making it a popular choice for web development.

Introduction to backend development and its fundamentals

Backend development refers to the development of the server-side of a web application, which is responsible for managing the data, processing requests, and generating responses. Here are some of the key concepts and technologies related to backend development:

1. Server: A server is a computer system that provides resources and services to other computers and devices on a network. In the context of web development, a server is responsible for serving web pages and handling requests from clients.

2. Database: A database is a system for storing and retrieving data. Backend applications often use databases to store and retrieve information, such as user information, product information, and blog posts.
3. Server-side programming languages: Server-side programming languages, such as Ruby, Python, PHP, and Node.js, are used to write code that runs on the server and generates dynamic web pages in response to client requests.
4. Web framework: A web framework is a collection of tools and libraries that provide a structure for building web applications. Examples of popular web frameworks include Ruby on Rails, Django, Laravel, and Express.
5. APIs: An API (Application Programming Interface) is a set of rules that allow one software application to interact with another. Backend applications often expose APIs that allow other applications and services to access their data and functionality.
6. HTTP: HTTP (Hypertext Transfer Protocol) is the protocol used for transmitting data over the web. Backend applications must be able to handle HTTP requests and generate HTTP responses.
7. Sessions and authentication: Sessions and authentication are used to manage the state of a user's interaction with a web application. Backend applications must be able to manage user sessions and authenticate users to ensure the security of sensitive data.

These are just some of the key concepts and technologies related to backend development. Backend development requires a solid understanding of server-side programming, databases, and web protocols, as well as the ability to design and implement scalable, secure, and performant web applications.

Python and Django Fundamentals

Python is a high-level, interpreted programming language that is often used for backend development. It is known for its simplicity, readability, and versatility, making it a popular choice for many types of software development, including web development. Here are some of the key features and benefits of using Python for backend development:

1. Large Standard Library: Python has a large standard library that includes modules for common programming tasks, such as connecting to web servers, reading and writing files, and working with data in various formats.
2. Dynamic Typing: Python uses dynamic typing, which means that the data type of a variable is determined at runtime, rather than being declared in advance. This can make development faster and more flexible, as it allows for more rapid prototyping and testing.
3. Interoperability: Python can be easily integrated with other programming languages and technologies, making it a good choice for projects that need to work with a variety of systems and data sources.
4. Large Community: Python has a large and active community of users and developers, which means that there are many resources and tools available for learning, troubleshooting, and developing applications.
5. Many Web Frameworks: Python has many popular web frameworks, such as Django, Flask, and Pyramid, which provide a structure for building web applications and make it easier to handle common tasks, such as handling HTTP requests, generating HTML pages, and connecting to databases.

These are just a few of the key features and benefits of using Python for backend development. Whether you are building a simple web application or a complex system that integrates with many other systems and technologies, Python can provide a flexible and powerful platform for your project.

Here are some of the basic concepts and code examples in Python:

1. Variables: Variables in Python are used to store values, and they can be declared and assigned in a single line of code. For example:

```
x = 10
y = 20
z = x + y
print(z) # Output: 30
```

2. Data Types: Python supports several built-in data types, including integers, floating-point numbers, strings, and lists. For example:

```
name = "John Doe"
age = 30
is_student = False
subjects = ["Math", "Science", "English"]
print(name) # Output: "John Doe"
print(age) # Output: 30
print(is_student) # Output: False
print(subjects) # Output: ["Math", "Science", "English"]
```

3. Control Flow: Python has several control flow statements, such as **if**, **for**, and **while**, which can be used to control the flow of execution in a program. For example:

```
x = 10
if x > 5:
   print("x is greater than 5")
else:
   print("x is less than or equal to 5")
```

4. Functions: Functions are blocks of code that can be reused multiple times in a program. Functions in Python are declared using the **def** keyword. For example:

```
def greet(name):
    print("Hello, " + name)


greet("John Doe") # Output: "Hello, John Doe"
```

These are just a few of the basic concepts and code examples in Python. Whether you are just starting out with programming or you are an experienced programmer looking to learn a new language, Python is a great choice for its simplicity, readability, and versatility.

Django is a high-level web framework for Python that is designed for rapid development and pragmatic design. It follows the Model-View-Template (MVT) architectural pattern and provides a full-stack framework for building dynamic web applications. Here are some of the key features and benefits of using Django:

1. ORM (Object-Relational Mapping): Django provides a built-in ORM that allows you to interact with databases using Python code instead of writing raw SQL queries. This makes it easier to work with databases and reduces the risk of SQL injection attacks. For example:

```
from django.db import models

class Person(models.Model):

    name = models.CharField(max_length=100)

    age = models.IntegerField()

    is_student = models.BooleanField(default=False)
```

2. URL Routing: Django provides a powerful URL routing system that allows you to map URLs to views in your application. For example:

```
from django.urls import path

from . import views



urlpatterns = [

    path('', views.index, name='index'),

    path('person/<int:person_id>/', views.person_detail, name='person_detail'),

]
```

3. Views: Views in Django are Python functions that handle HTTP requests and return HTTP responses. They can be used to generate dynamic HTML pages, process form submissions, and more. For example:

```
from django.shortcuts import render

from .models import Person


def index(request):

    people = Person.objects.all()

    return render(request, 'index.html', {'people': people})


def person_detail(request, person_id):

    person = Person.objects.get(id=person_id)

    return render(request, 'person_detail.html', {'person': person})
```

4. Templates: Django uses templates to define the structure and content of HTML pages in your application. Templates can include variables, loops, and conditionals to make it easy to generate dynamic HTML. For example

   :

```
<h1>People</h1>

<ul>

{% for person in people %}

    <li>{{ person.name }} ({{ person.age }})</li>

{% endfor %}

</ul>
```

Whether you are building a simple blog or a complex web application, Django provides a robust and flexible platform for your project.

Projects Worked On by the Intern

Here are the projects we worked on during the internship

1. To-Do List: A simple task management application where users can add, edit, and delete tasks.
2. Blog: A simple blog application where users can create, edit, and delete blog posts.
3. e-Commerce Website: A basic e-commerce website where users can browse products, add them to their cart, and check out.
4. Contact Form: A simple contact form application where users can send messages to the site administrator.
5. User Authentication: A basic authentication system where users can sign up, log in, and log out.
6. Image Gallery: A simple image gallery application where users can upload and view images.
7. File Upload: A basic file upload application where users can upload and view files.
8. Polls: A simple polling application where users can vote on various questions and view the results.
9. Calendar: A basic calendar application where users can add, edit, and delete events.
10. News Aggregator: A simple news aggregator application that fetches news articles from various sources and displays them to the users.

Skills developed during the internship

1. Server-side programming: Understanding of server-side languages such as Python, Java, Ruby, PHP, and Node.js.

2. Database management: Knowledge of relational databases such as MySQL, PostgreSQL, and non-relational databases such as MongoDB.
3. Web frameworks: Familiarity with popular web frameworks such as Django, Ruby on Rails, Express, and Laravel.
4. API Development: Understanding of REST APIs, how to design and implement them, and how to consume them.
5. Server deployment: Knowledge of deploying and managing web applications on cloud platforms such as Amazon Web Services (AWS), Google Cloud Platform (GCP), and Heroku.
6. Security: Awareness of security best practices and how to secure a web application from common attacks such as SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF).
7. Scalability: Understanding of how to scale a web application to handle increasing traffic and load.
8. Debugging and troubleshooting: Ability to identify and resolve bugs and performance issues in a web application.
9. Data structures and algorithms: Basic knowledge of data structures and algorithms and how they can be applied to solve common problems in web development.
10. Agile methodologies: Familiarity with Agile development methodologies such as Scrum and Kanban and how they can be applied to backend development projects.

Tools and technologies used in backend web development

1. Programming languages: Python was the used programming languages for backend development in this interhip.
2. Web frameworks: Framework for backend development include Django, and Laravel for this internship.

3. Databases: Relational databases such as MySQL, PostgreSQL, and non-relational databases such as MongoDB are widely used for storing data in web applications.
4. Web servers: Apache and Nginx are popular web servers used to serve web applications.
5. Cloud platforms: Amazon Web Services (AWS), Google Cloud Platform (GCP), and Heroku are some of the cloud platforms commonly used for deploying and hosting web applications.
6. Source control: Git is widely used for source control and version control in backend development projects.
7. Debugging and profiling tools: Debugging and profiling tools such as PyCharm, IntelliJ, and Visual Studio Code are used to identify and resolve bugs and performance issues in web applications.
8. Task runners and build tools: Tools such as Grunt, Gulp, and Webpack are used to automate repetitive tasks and build frontend assets.
9. APIs: Backend developers use APIs to communicate between the frontend and the backend of web applications.
10. Monitoring and logging tools: Monitoring and logging tools such as New Relic, Logentries, and Sentry are used to monitor the performance and health of web applications.

Experience gained

This Internship offered a wide range of experiences and learning opportunities, including:

1. Server-side programming: Gaining a deep understanding of server-side languages, frameworks, and patterns for building scalable and performant web applications.
2. Database design and management: Learning how to design, implement, and manage relational and non-relational databases, and how to optimize their performance.

3. API design and implementation: Understanding how to design and implement REST APIs, and how to make them secure, scalable, and maintainable.
4. Server deployment and management: Learning how to deploy and manage web applications on cloud platforms, and how to handle scaling, backups, and security.
5. Problem-solving: Developing a systematic and creative approach to solving complex problems, and becoming skilled at debugging and troubleshooting.
6. Team collaboration: Working with a team of developers, designers, and stakeholders to build and maintain web applications, and learning how to communicate effectively and collaborate effectively.
7. Agile methodologies: Becoming familiar with Agile development methodologies and how they can be applied to backend development projects.
8. Industry trends and best practices: Keeping up to date with the latest industry trends and best practices, and continuously improving one's skills and knowledge.
9. Entrepreneurship: Developing the skills and mindset needed to start and run a successful software development business.

Overall, backend web development provides a rich and challenging learning experience, allowing developers to hone their skills, expand their knowledge, and make a meaningful impact on the web.

Chapter Four

Key Lessons Learned and Challenges Faced

Here are some key lessons learned and challenges faced during SIWES:

1. Understanding server-side programming: Understanding the server-side programming languages and frameworks used in backend development can be a challenge, especially for those new to web development.
2. Database design and management: Designing and managing databases can be a complex and challenging task, especially when dealing with large amounts of data and multiple users.
3. API design and implementation: Designing and implementing APIs can be challenging, especially when ensuring they are secure, scalable, and maintainable.
4. Server deployment and management: Deploying and managing web applications can be a complex and challenging task, especially when dealing with security, scalability, and reliability issues.
5. Debugging and troubleshooting: Debugging and troubleshooting server-side issues can be challenging, especially when dealing with complex systems and multiple dependencies.
6. Working with a team: Collaborating effectively with a team of developers, designers, and stakeholders can be a challenge, especially when dealing with conflicting opinions and tight deadlines.
7. Adapting to new technologies and methodologies: Keeping up to date with the latest industry trends and best practices can be a challenge, especially when working on multiple projects and dealing with rapidly evolving technologies.
8. Balancing performance and security: Balancing the need for high performance with the need for security can be a challenging task, especially when dealing with sensitive data and complex systems.

9. Maintaining code quality: Ensuring that code is maintainable, readable, and scalable can be a challenge, especially when dealing with complex systems and tight deadlines.

Despite these challenges, backend web development can be a rewarding and fulfilling experience, providing opportunities to solve complex problems, work on cutting-edge technologies, and make a meaningful impact on the web.

Solutions and improvements to the above challenges]

Here are some solutions and improvements for the challenges faced in backend web development:

1. Understanding server-side programming: To overcome this challenge, one can start by taking online courses, attending workshops, and reading books to gain a solid foundation in server-side programming languages and frameworks.
2. Database design and management: To improve database design and management, one can use database management tools, such as migrations and ORMs, to automate routine tasks, and follow best practices for database design, such as normalization and indexing.
3. API design and implementation: To improve API design and implementation, one can use API design frameworks, such as OpenAPI and Swagger, to define and document APIs, and use authentication and authorization protocols, such as OAuth and JWT, to secure APIs.
4. Server deployment and management: To overcome this challenge, one can use cloud platforms, such as AWS and GCP, to deploy and manage web applications, and use containerization technologies, such as Docker, to manage application dependencies and configurations.
5. Debugging and troubleshooting: To improve debugging and troubleshooting skills, one can use logging, monitoring, and debugging tools, such as Sentry and New Relic, to track and resolve issues, and use testing and continuous

integration tools, such as TravisCI and Jenkins, to catch and resolve issues early.

6. Working with a team: To improve collaboration with a team, one can use project management tools, such as Asana and Jira, to track project progress and communicate effectively, and use version control systems, such as Git and SVN, to manage code and collaborate effectively.
7. Adapting to new technologies and methodologies: To overcome this challenge, one can continuously learn and improve by attending conferences, participating in online communities, and reading industry publications and blogs.
8. Balancing performance and security: To balance performance and security, one can use security best practices, such as input validation and encryption, to secure data, and use performance optimization techniques, such as caching and indexing, to improve performance.
9. Maintaining code quality: To maintain code quality, one can use code review processes, such as pull requests and code reviews, to review and improve code, and use code quality tools, such as SonarQube and ESLint, to enforce code quality standards.

By following these solutions and best practices, developers can overcome the challenges of backend web development and continuously improve their skills and knowledge in this field.

Chapter Five

Summary

This report provides a comprehensive overview of the skills and knowledge gained by me during my training period. The report covers the fundamentals of backend development, including server-side programming, database design and management, API design and implementation, server deployment and management, debugging and troubleshooting, and working with a team. The report also highlights the tools and technologies used in backend web development, such as cloud platforms, containerization technologies, logging, monitoring and debugging tools, project management tools, version control systems, and security best practices. The report concludes by summarizing the key lessons learned and challenges faced during the training, and providing solutions and improvements for these challenges. Overall, the Intern training report on backend web development provides valuable insights into the skills and knowledge required for successful backend web development and the challenges faced in this field.

Recommendation

I am writing to recommend me for my outstanding performance during this internship in Leyook Global Technologies the backend web development department. I have displayed a strong understanding of the fundamentals of server-side programming, database design and management, API design and implementation, server deployment and management, debugging and troubleshooting, and teamwork.

Throughout the internship, I have shown a passion for learning and has actively sought out opportunities to expand my knowledge and skills. I have also demonstrated strong problem-solving skills and have been able to effectively debug and troubleshoot issues in a timely manner. I’ve also been a valuable team member, contributing to the team's efforts in a positive and productive way.

In addition to my technical skills, I have also shown a strong commitment to code quality and have consistently produced high-quality code that is well-documented and maintainable. I have also been able to effectively balance the trade-off between performance and security, ensuring that the applications they have worked on are both fast and secure.

I highly myself for their dedication, hard work, and exceptional technical skills. I would be a valuable asset to any organization looking for a talented backend web developer.

Conclusion

The internship at Leyook Global Technology provided the intern with valuable experience and knowledge in the field of backend web development. The intern learned about the development process, the use of various technologies, and the importance of testing and debugging. The intern also gained a better understanding of the real-world challenges faced by software developers and the importance of writing clean and maintainable code.