**GitAlert Internship Report**

SUBMITTED IN PARTIAL FULFILLMENT FOR THE REQUIREMENT OF THE AWARD OF DEGREE OF

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE**



Submitted by

Rhythm Garg (2100290120141)

**Session 2023-24**

DEPARTMENT OF COMPUTER SCIENCE

**KIET GROUP OF INSTITUTIONS, GHAZIABAD**

**(Affiliated to Dr. A. P. J. Abdul Kalam Technical University, Lucknow, U.P., India)**



*CERTIFICATE*

Un

This is to certify that the “**Internship report”** submitted by **RHYTHM GARG** is work done by him and submitted during 2023 – 2024 academic year, in partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE,** at **KIET GROUP OF INSTITUTIONS, GHAZIABAD.**

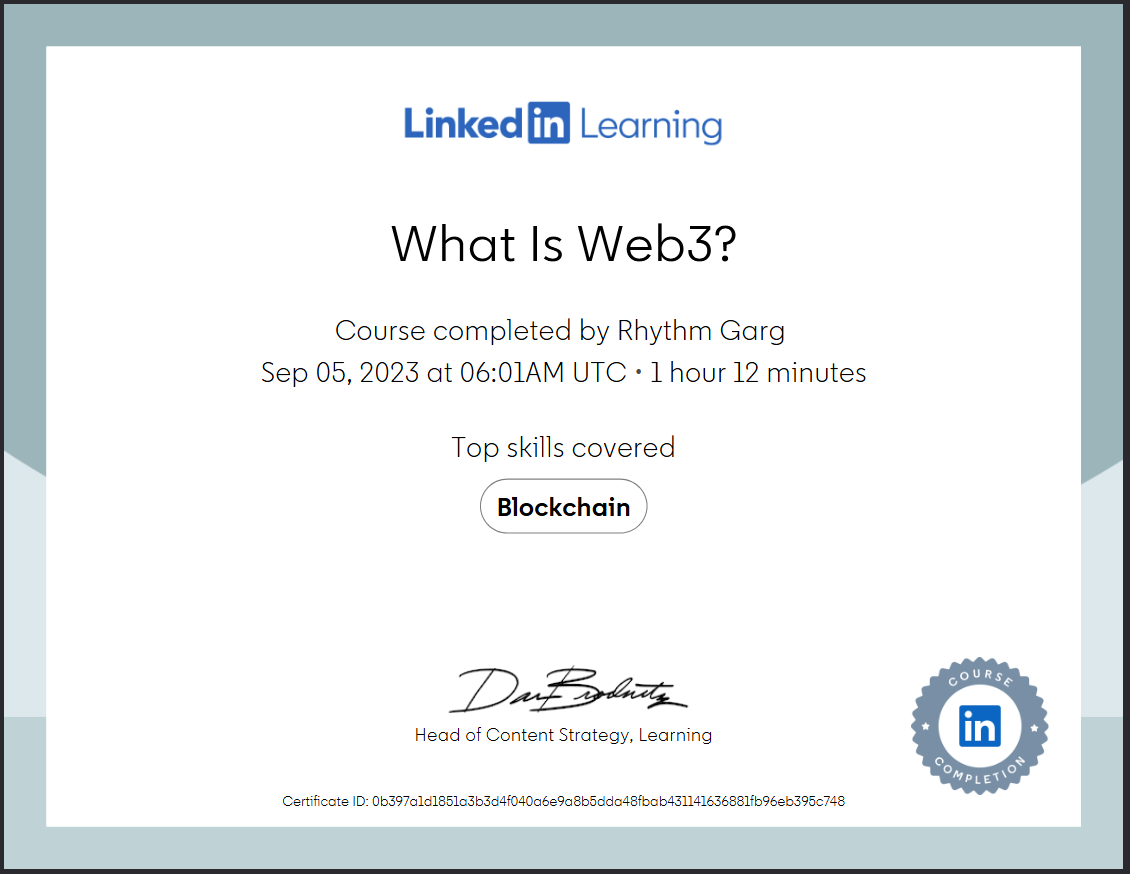
**College Internship Coordinator: Mr Rahul**

**Department Internship Coordinator:**   **Mr Harshvardhan**

**Head of the Department of CS**

**Dr Ajay Kumar Shrivastava**









# ACKNOWLEDGEMENT

I am highly indebted to Director **Dr. Amik Garg**, for the facilities provided to accomplish this internship.

I would like to thank my Head of the Department **Dr. Ajay Srivastava** for his constructive criticism throughout my internship.

I would like to thank **Mr. Harsh Vardhan** internship coordinator Department of CS for their support and advices to get and complete internship in above said organization.

I am extremely great full to my department staff members and friends who helped me in successful completion of this project.

**Rhythm Garg**

**(2100290120141)**

# ABSTRACT

This internship report documents my experience and insights gained during my tenure as a Frontend Developer Intern at INVSTO from Aug 2022 to Feb 2023. The report provides an overview of the projects I worked on, the skills I acquired, and the challenges I encountered while contributing to the development of web applications.

Throughout the internship, I actively participated in various phases of the software development life cycle, including requirements analysis, design, implementation, testing, and deployment. I collaborated closely with cross-functional teams, including designers, backend developers, and product managers, to deliver responsive and user-friendly web solutions.

**Organisation Information:**

In the rapidly evolving landscape of finance and investment, where technology is reshaping traditional practices, INVSTO stands at the forefront as a dynamic and innovative fintech trading company. Established with a vision to democratize financial markets and empower individuals with cutting-edge tools, INVSTO is dedicated to providing seamless access to global financial markets, transforming the way people invest and trade.

**Programs and opportunities:**

At INVSTO, our mission is clear: to revolutionize the world of finance by making it more inclusive, transparent, and efficient. We believe that everyone, regardless of their background or experience, should have the opportunity to participate in the financial markets and secure their financial future. Through our state-of-the-art technology and user-centric approach, we are breaking down barriers and enabling individuals to take control of their financial destiny.

**Methodologies:**

We follow a structured methodology for our projects which starts from designing the solution to the implementation phase. Well planned Project reduces the time to deliver the project and any additional ad-hoc costs to our clients, hence we dedicate the majority of our time understanding our clients business and gathering requirements. This ground up approach helps us deliver not only the solution to our clients but also add value to your investments.

**Key parts of the report:**

Under each division we further provide specific industry solutions on focused domains with cutting edge technologies.

**Benefits of the Company/Institution through our report:**

Under each division we further provide specific industry solution on focused domains with cutting edge technologies. We emphasise on building relationships with our clients by delivering projects on time and within budget.

**INDEX**

## S.no CONTENTS Page no

1. Introduction……………………………………………………………………....1
   1. Modules……………………………………………………………………….2
2. Analysis…………………………………………………………………………....3
3. Software requirements specifications ……………………………………………..4
4. Technology…………………………………………………………………………5
   1. ASP.NET……………………………………………………………………….5
   2. ADP.NET ……………………………………………………………………....6
   3. C#.NET………………………………………………………………………….7
   4. JAVA…………………………………………………………………………….7
   5. SQL Data Base………………………………………………………………..…8
5. Coding……………………………………………………………………………….10
6. Screenshots...................................................................................................................11
7. Conclusion…………………………………………………………………………….16
8. Bibilography…………………………………………………………………………..17

## Learning Objectives/Internship Objectives

* Internships are generally thought of to be reserved for college students looking to gain experience in a particular field. However, a wide array of people can benefit from Training Internships in order to receive real world experience and develop their skills.

* An objective for this position should emphasize the skills you already possess in the area and your interest in learning more

* Internships are utilized in a number of different career fields, including architecture, engineering, healthcare, economics, advertising and many more.

* Some internship is used to allow individuals to perform scientific research while others are specifically designed to allow people to gain first-hand experience working.

* Utilizing internships is a great way to build your resume and develop skills that can be emphasized in your resume for future jobs. When you are applying for a Training Internship, make sure to highlight any special skills or talents that can make you stand apart from the rest of the applicants so that you have an improved chance of landing the position.

**WEEKLY OVERVIEW OF INTERNSHIP ACTIVITIES**

|  |  |  |  |
| --- | --- | --- | --- |
| **1**  **st**    **WEEK** | **DATE** | **DAY** | **NAME OF THE TOPIC/MODULE COMPLETED** |
| 8/05/17 | Monday | Introduction of .NET frame work |
| 9/05/17 | Tuesday | Features of .NET { IDE,CLR,CTS} |
| 10/05/17 | Wednesday | Introduction of C# .NET |
| 11/05/17 | Thursday | Introduction of console application |
| 12/05/17 | Friday | Continuing console applications |
| 13/05/17 | Saturday | Understanding different types of class and collection |

|  |  |  |  |
| --- | --- | --- | --- |
| **2**  **nd**    **WEEK** | **DATE** | **DAY** | **NAME OF THE TOPIC/MODULE COMPLETED** |
| 15/05/17 | Monday | Introduction to windows application |
| 16/05/17 | Tuesday | Understanding controls like [label, button] etc., |
| 17/05/17 | Wednesday | Continuing understanding controls |
| 18/05/17 | Thursday | Introduction to windows services |
| 19/05/17 | Friday | Continuing windows services |
| 20/05/17 | Saturday | Introduction to class libraries |

|  |  |  |  |
| --- | --- | --- | --- |
| **3**  **rd**    **WEEK** | **DATE** | **DAY** | **NAME OF THE TOPIC/MODULE COMPLETED** |
| 22/05/17 | Monday | Introduction .NET REMOTING |
| 23/05/17 | Tuesday | Introduction to web programming |
| 24/05/17 | Wednesday | Understanding what is Internet website, web request and web response |
| 25/05/17 | Thursday | Understanding client side web technologies VS server side web technologies |
| 26/05/17 | Friday | Introduction to HTML |
| 27/05/17 | Saturday | HTML continued |

|  |  |  |  |
| --- | --- | --- | --- |
| **4**  **th**    **WEEK** | **DATE** | **DAY** | **NAME OF THE TOPIC/MODULE COMPLETED** |
| 29/05/17 | Monday | Understanding classic ASP VS ASP.NET |
| 30/05/17 | Tuesday | Introduction to ASP.NET web server controls |
| 31/05/17 | Wednesday | Working with different web server controls |
| 01/06/17 | Thursday | Standard controls , validation controls |
| 02/06/17 | Friday | Introduction to data control in depth |
| 03/06/17 | Saturday | Project session |

|  |  |  |  |
| --- | --- | --- | --- |
| **5**  **th**    **WEEK** | **DATE** | **DAY** | **NAME OF THE TOPIC/MODULE COMPLETED** |
| 05/06/17 | Monday | Design& Analysis |
| 06/06/17 | Tuesday | Coding |
| 07/06/17 | Wednesday | Testing |

## 1. INTRODUCTION

In the ever-evolving world of finance, success in trading and investment relies heavily on well-defined strategies. These strategies serve as roadmaps, guiding individuals and institutions toward their financial goals, whether that's wealth accumulation, risk management, or portfolio diversification. Trading and investment strategies encompass a broad spectrum of approaches tailored to different objectives, risk tolerances, and time horizons.

The Power of Strategies: Trading and Investment Strategies are the cornerstones of informed decision-making in financial markets. They provide structure, discipline, and a systematic approach to navigating the complexities of investing and trading. Whether you are a long-term investor seeking to build wealth gradually or a day trader looking for short-term gains, strategies are the key to achieving your desired outcomes.

Key Components of Trading and Investment Strategies:

* Goals and Objectives: Clear and well-defined financial goals and objectives are the starting points of any strategy. Are you aiming for capital preservation, income generation, or aggressive growth? Your goals will shape your strategy.
* Risk Tolerance: Understanding your risk tolerance is crucial. Some strategies are conservative and focus on minimizing risk, while others embrace risk for the potential of higher returns. Your risk appetite will determine your strategy's level of risk.
* Asset Allocation: Asset allocation is the process of distributing your investments across different asset classes, such as stocks, bonds, real estate, and commodities. The allocation should align with your objectives and risk tolerance.
* Time Horizon: Your time horizon, whether short-term, medium-term, or long-term, will influence the choice of assets and the frequency of trading decisions within your strategy.
* Research and Analysis: Effective strategies involve thorough research and analysis of assets, markets, and economic conditions. Fundamental analysis, technical analysis, and market sentiment all play roles in strategy development.
* Execution Plan: Your strategy should outline how you plan to execute trades and investments. This includes entry and exit points, stop-loss levels, and profit-taking strategies.

Types of Strategies: Trading and Investment Strategies are diverse and tailored to specific needs:

* Buy and Hold: A long-term strategy focused on buying assets and holding them for an extended period, often with minimal trading activity.
* Day Trading: A short-term strategy where traders buy and sell assets within the same trading day, aiming to profit from price fluctuations.
* Value Investing: A strategy that seeks undervalued assets with long-term growth potential, often associated with fundamental analysis.
* Technical Analysis: A strategy based on chart patterns, indicators, and historical price data to forecast future price movements.
* Income Investing: Focused on generating regular income through dividends, interest, or rental income from investments.
* Diversification: A risk management strategy that spreads investments across various asset classes to reduce overall risk.

**1.1Module Description**:

Shark Sigma

======

1. User registers the site.
2. Products will be showed
3. If user selected the product and then save 4) User selected product is send to the Order.

5) If user wants to buy the product they can also buy.

Trading Sim

=======

1. send product details
2. send payment verification
3. Store buying detail
4. Store line items using join product and order

## 2. SYSTEM ANALYSIS

**2.1 Requirement Analysis Existing System:**

In an Existing we address these challenges and present an approach to efficient, incremental consolidation of data-intensive flows. Following common practice, our method iterates over information requirements to create the final design. we show how to efficiently accommodate a new information requirement to an existing design and also, how to update a design in lieu of an evolving information requirement. The final design satisfying all requirements comprises a multi-flow. As ‘coal’ is formed after the process and extreme compaction of layers of partially decomposed materials1, Co Al processes individual data flows and incrementally consolidates them into a unified multi-flow.

**Proposed System**

Following the previously proposed set of flow transformations in the context of ETL processes in Co Al we extend this set considering also the associative property of n-array operations (e.g., Join) and thus rely on the following four flow transformations used for reordering the operations. Swap Applied to a pair of adjacent unary operations, it interchanges the order of these operations. Distribute/Factorize. Applied on a unary operation over an adjacent n-array operation, it respectively distributes the unary operation over the adjacent nary operation or factorizes several unary operations over the adjacent narray operation. Merge/Split. Applied on a set of adjacent unary operations, it respectively merges several operations into a single unary operation or splits a unary operation into several unary operations. Re-associate. Applied on a pair of mutually associative n-array operations, it interchanges the order in which these operations are executed.

# 3. SOFTWARE REQUIREMENTS SPECIFICATIONS

**3.1 System configurations**

The software requirement specification can produce at the culmination of the analysis task. The function and performance allocated to software as part of system engineering are refined by established a complete information description, a detailed functional description, a representation of system behavior, and indication of performance and design constrain, appropriate validate criteria, and other information pertinent to requirements.

**Software Requirements**:

* Operating system : Windows 7 Ultimate.
* Coding Language : MVC 4 Razor
* Front-End : Visual Studio 2012 Professional.
* Data Base : SQL Server 2008.

**Hardware Requirement**:

* System : Pentium IV 2.4 GHz.
* Hard Disk : 1TB.
* Ram : 4GB.

## 4. TECHNOLOGY

**4.1 JavaScript**

JavaScript is a widely used programming language that is primarily known for its role in web development. Here's a brief overview of JavaScript:

1. Purpose and Usage:

* JavaScript is a versatile and dynamically typed scripting language used primarily for building interactive and dynamic web applications.
* It allows you to add interactivity to web pages, manipulate HTML and CSS, and create client-side logic.
* JavaScript can also be used on the server-side (Node.js) for building scalable and efficient web server applications.

2. Key Features:

* Dynamic: JavaScript variables can change their data types during runtime.
* Weakly Typed: You don't need to declare data types explicitly; JavaScript infers them.
* Event-Driven: JavaScript is often used for responding to user interactions like clicks and form submissions.
* Asynchronous: JavaScript supports asynchronous programming through callbacks, Promises, and async/await, allowing non-blocking operations.

3. Core Components:

* Variables: Declare variables using var, let, or const.
* Functions: Create reusable blocks of code with functions.
* Objects: Use objects to group related data and functions (methods).
* Arrays: Store and manipulate collections of data with arrays.
* Control Structures: Employ if statements, loops (for, while), and switch statements to control program flow.

4. Libraries and Frameworks:

* JavaScript has a rich ecosystem of libraries and frameworks like jQuery, React, Angular, and Vue.js, which simplify and streamline web development tasks.

5. DOM Manipulation:

* The Document Object Model (DOM) represents the structure of a web page, and JavaScript is commonly used to interact with and manipulate the DOM to create dynamic web applications.

6. Asynchronous Operations:

* JavaScript uses callbacks, Promises, and async/await to manage asynchronous operations like API requests, file handling, and timers.

7. Browser Compatibility:

* JavaScript code runs on web browsers, and it's essential to consider cross-browser compatibility when developing web applications.

8. Debugging:

* Debugging tools and browser developer consoles are used to identify and fix errors in JavaScript code.

9. Security:

* JavaScript runs on the client-side, which means it's accessible to users. Developers need to be cautious about security vulnerabilities like Cross-Site Scripting (XSS) attacks.

10. ES6 and Beyond:

* ECMAScript 6 (ES6), also known as ES2015, introduced significant improvements to the language, including arrow functions, classes, destructuring, and more. Subsequent versions have continued to enhance JavaScript's capabilities.

11. Popular Use Cases:

* Building interactive web applications
* Developing mobile apps using frameworks like React Native
* Creating server-side applications with Node.js
* Building browser extensions
* Data visualization and charting
* Game development with libraries like Phaser

JavaScript is a foundational technology for modern web development, and its versatility makes it a crucial skill for web developers and programmers. It continues to evolve and remains an integral part of the web development stack.

**4.2 ReactJS**

React, also known as React.js or ReactJS, is a popular JavaScript library for building user interfaces (UIs) and front-end web applications. Developed and maintained by Facebook (and a community of open-source contributors), React has gained widespread adoption in the web development community due to its component-based architecture and efficient rendering.

Here are some key aspects of React:

1. Component-Based Architecture:

* React is centered around the concept of reusable UI components. A component is a self-contained unit of a user interface that can be composed and reused throughout an application.
* Components can have their own state and lifecycle methods, making it easier to manage UI updates.

2. Virtual DOM:

* React introduces the Virtual DOM, an in-memory representation of the actual DOM (Document Object Model). When a component's state changes, React updates the Virtual DOM first, then efficiently updates the real DOM, minimizing browser reflows and enhancing performance.

3. Declarative Syntax:

* React uses a declarative approach to building UIs, where developers describe what the UI should look like based on the application's state. This leads to more predictable and maintainable code.

4. JSX (JavaScript XML):

* JSX is an extension to JavaScript that allows developers to write HTML-like code within their JavaScript files. It's used to define the structure of React components.
* JSX is transpiled into regular JavaScript by tools like Babel before being executed in the browser.

5. Unidirectional Data Flow:

* React enforces a one-way data flow, meaning data flows in a single direction—from parent components to child components. This helps maintain data consistency and simplifies debugging.

6. React Router:

* React Router is a popular library for adding client-side routing to React applications. It allows developers to create single-page applications (SPAs) with different views or "pages."

7. State Management:

* While React provides a built-in way to manage local component state, for global state management, developers often use external libraries like Redux or MobX to handle complex state management scenarios.

8. Hooks:

* Introduced in React 16.8, hooks are functions that allow developers to use state and other React features in functional components. They simplify state management and lifecycle management in functional components.

9. Large Ecosystem:

* React has a vast ecosystem of libraries and tools, including libraries for form handling (Formik), component styling (Styled-components), and state management (Redux, MobX).

10. Server-Side Rendering (SSR):

* React can be used for server-side rendering, which improves initial page load performance and SEO by rendering the initial HTML on the server.

11. Mobile Development:

* React Native is a framework that allows developers to use React to build native mobile applications for iOS and Android platforms.

React has become a cornerstone of modern web development, enabling developers to create highly interactive and performant user interfaces. Its popularity continues to grow, and it's widely used in both small-scale projects and large-scale applications.

**4.3 Django**

Django is a high-level, open-source web framework written in Python. It was created to simplify and accelerate the development of web applications by providing a robust and pragmatic set of tools and conventions. Django follows the "batteries-included" philosophy, meaning it includes a wide range of built-in features and libraries for common web development tasks.

Here are some key aspects of Django:

1. Model-View-Controller (MVC) Architecture:

* Django follows the Model-View-Controller (MVC) architectural pattern, but it uses slightly different terminology: Model-View-Template (MVT).
* Models define the data structure and interact with the database.
* Views handle the logic for processing requests and rendering responses.
* Templates define the presentation layer, allowing developers to separate HTML from Python code.

2. Object-Relational Mapping (ORM):

* Django provides an ORM that allows developers to define database models using Python classes. These models are then translated into database tables and queries automatically.
* The ORM supports various database backends, including PostgreSQL, MySQL, SQLite, and Oracle.

3. Admin Interface:

* Django includes an admin interface that can be used out of the box to manage application data. Developers can easily create custom admin interfaces for their models.

4. URL Routing:

* Django's URL routing system allows developers to define URL patterns and associate them with specific views.
* URL patterns can include dynamic parts that capture values from the URL and pass them to view functions.

5. Security Features:

* Django includes built-in security features to protect against common web vulnerabilities, such as cross-site scripting (XSS), cross-site request forgery (CSRF), and SQL injection.
* It also provides tools for user authentication and authorization.

6. Form Handling:

* Django simplifies form handling with its form library. Developers can define form structures, validate user input, and handle form submissions easily.

7. Middleware:

* Middleware components can be used to process requests and responses globally. For example, middleware can handle authentication, compression, and more.

8. Template Engine:

* Django's template engine allows developers to create HTML templates with placeholders for dynamic content. These templates are then rendered with data from the views.

9. Internationalization and Localization:

* Django supports internationalization and localization, making it possible to create applications that work in multiple languages and regions.

10. Testing Framework:

* Django provides a comprehensive testing framework for writing unit tests, integration tests, and functional tests.

11. Third-Party Packages:

* The Django ecosystem includes a vast array of third-party packages and extensions that can be easily integrated to extend its functionality.

12. Scalability:

* While Django is well-suited for building small to medium-sized applications, it can also scale to handle large and complex projects with proper architecture and optimization.

Django is widely used in web development because of its efficiency, security, and flexibility. It's particularly popular for building content management systems (CMS), e-commerce platforms, social networks, and other data-driven web applications. Django's extensive documentation and strong community support make it an excellent choice for both beginners and experienced developers.

**4.4 SQL**

SQL, or Structured Query Language, is a domain-specific programming language used for managing and manipulating relational databases. SQL is essential for working with databases, as it allows you to perform various operations such as creating, querying, updating, and deleting data. Here are some fundamental aspects of SQL:

1. Database Management Systems (DBMS):

* SQL is used in conjunction with Database Management Systems like MySQL, PostgreSQL, Oracle, Microsoft SQL Server, SQLite, and many others.
* Each DBMS may have its own dialect of SQL, but the core concepts are largely consistent.

2. Data Manipulation:

* SQL provides a set of commands for performing operations on data stored in a relational database. These operations include:
  + SELECT: Retrieving data from one or more tables.
  + INSERT: Adding new records to a table.
  + UPDATE: Modifying existing records in a table.
  + DELETE: Removing records from a table.

3. Data Definition:

* SQL allows you to define the structure of your database, including:
  + CREATE TABLE: Defining tables and their columns.
  + ALTER TABLE: Modifying the structure of existing tables.
  + DROP TABLE: Deleting tables.
  + CREATE INDEX: Creating indexes for faster data retrieval.

4. Data Querying:

* SQL is commonly used to query databases to retrieve specific data. Queries can be simple or complex and can involve filtering, sorting, grouping, and joining multiple tables.

5. Data Constraints:

* SQL enables you to impose constraints on data integrity, such as:
  + PRIMARY KEY: Ensuring unique values in a column.
  + FOREIGN KEY: Establishing relationships between tables.
  + UNIQUE: Enforcing uniqueness across table rows.
  + CHECK: Specifying conditions that data must meet.

6. Aggregation and Grouping:

* SQL provides functions like COUNT, SUM, AVG, MIN, and MAX for aggregating and summarizing data.
* You can use GROUP BY to group data by one or more columns.

7. Joins:

* SQL allows you to combine data from multiple tables using JOIN operations. Common types of joins include INNER JOIN, LEFT JOIN, RIGHT JOIN, and FULL JOIN.

8. Transactions:

* SQL supports transactions, which are sequences of one or more SQL statements treated as a single unit. Transactions ensure data consistency by allowing either all the statements in a transaction to succeed or none at all.

9. Views:

* SQL allows you to create views, which are virtual tables based on the result of a query. Views simplify complex queries and provide an abstracted layer of data access.

10. Stored Procedures and Functions: - SQL lets you define stored procedures and functions, which are reusable blocks of code that can be called to perform specific tasks or calculations.

SQL is a fundamental skill for database administrators, data analysts, and software developers who work with relational databases. It's a powerful tool for managing and extracting insights from data, making it an essential part of the data-driven world of modern software development and data management.

**4.5 MySQL**

MySQL is an open-source relational database management system (RDBMS) that is widely used for managing and storing structured data. It is one of the most popular database systems globally and is known for its performance, reliability, and ease of use. Here are some key aspects of MySQL:

1. Relational Database Management System (RDBMS):

* MySQL follows the principles of a relational database, where data is organized into tables with rows and columns.
* It allows you to establish relationships between tables, enforce data integrity, and perform complex queries using Structured Query Language (SQL).

2. Open Source:

* MySQL is released under an open-source license, which means it is freely available for use, modification, and distribution.
* It is actively developed and maintained by the open-source community.

3. Cross-Platform:

* MySQL is available on various operating systems, including Windows, Linux, macOS, and more.
* It can be used with a wide range of programming languages and development environments.

4. Data Types:

* MySQL supports a variety of data types, including integers, floating-point numbers, strings, dates, and more.
* It also allows users to define custom data types and storage engines.

5. SQL Support:

* MySQL fully supports SQL, which is used to create, retrieve, update, and delete data in the database.
* It also supports advanced SQL features like subqueries, joins, views, and stored procedures.

6. Performance and Scalability:

* MySQL is known for its excellent performance and can handle high traffic and large datasets efficiently.
* It offers features like indexing, caching, and replication to optimize performance and scalability.

7. Security:

* MySQL provides robust security features, including user authentication, access control, and data encryption.
* It supports various authentication methods, including password-based and certificate-based authentication.

8. High Availability and Replication:

* MySQL supports features like replication and clustering to ensure high availability and data redundancy.
* Replication allows you to create multiple copies (replicas) of a database to distribute the workload and ensure fault tolerance.

9. Storage Engines:

* MySQL supports multiple storage engines, including InnoDB (the default), MyISAM, and others. Each engine has unique characteristics and is suited to different use cases.

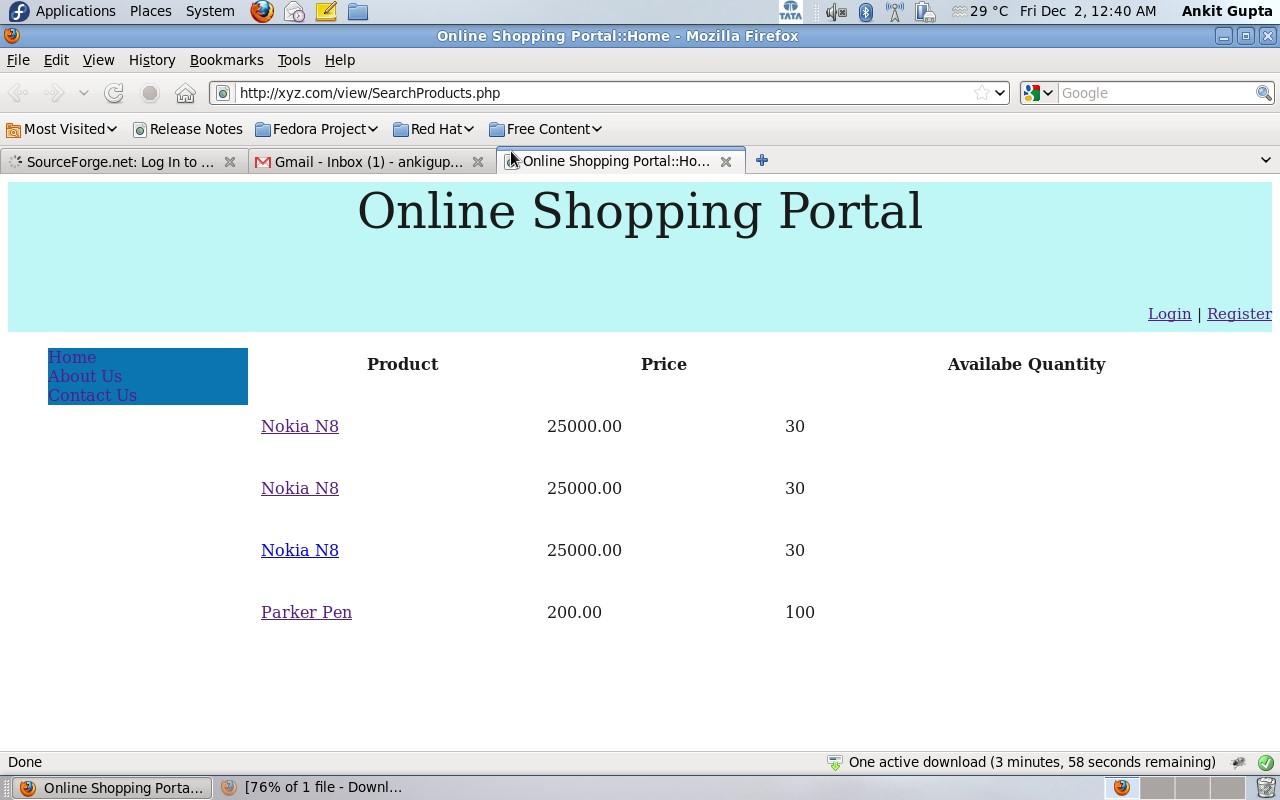
10. Community and Enterprise Editions: - MySQL is available in two editions: the community edition (free and open-source) and the enterprise edition (commercial with additional features and support).

11. Tools and Ecosystem: - MySQL has a rich ecosystem of tools and software that complement its functionality, including graphical user interfaces (GUIs), connectors for various programming languages, and monitoring tools.

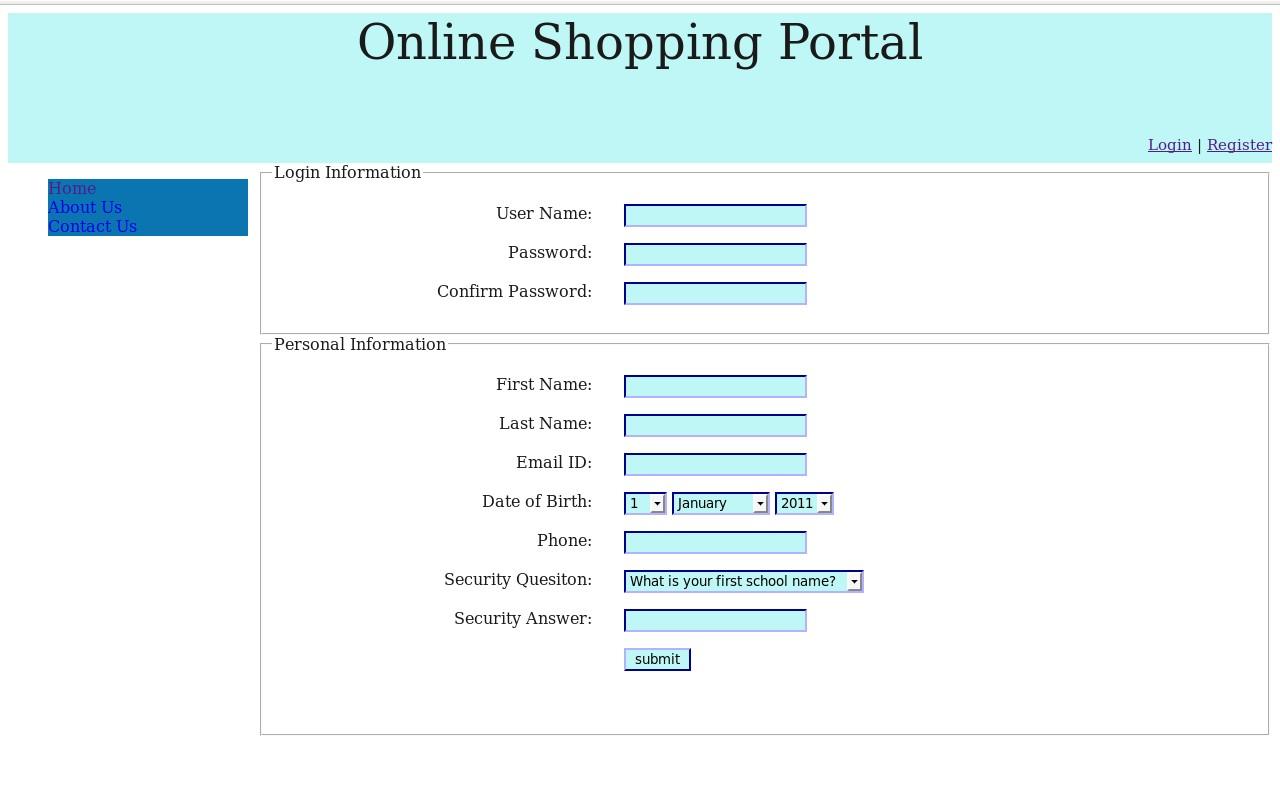
MySQL is widely used in web applications, content management systems (CMS), e-commerce platforms, and various other software applications that require robust and efficient data storage and retrieval. Its combination of performance, reliability, and cost-effectiveness has made it a popular choice for both small-scale projects and large-scale enterprise applications.

## 6. SCREENSHOTS

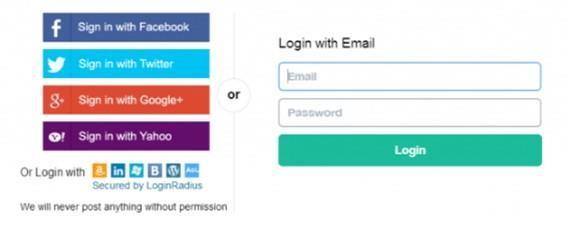
**Home page:**

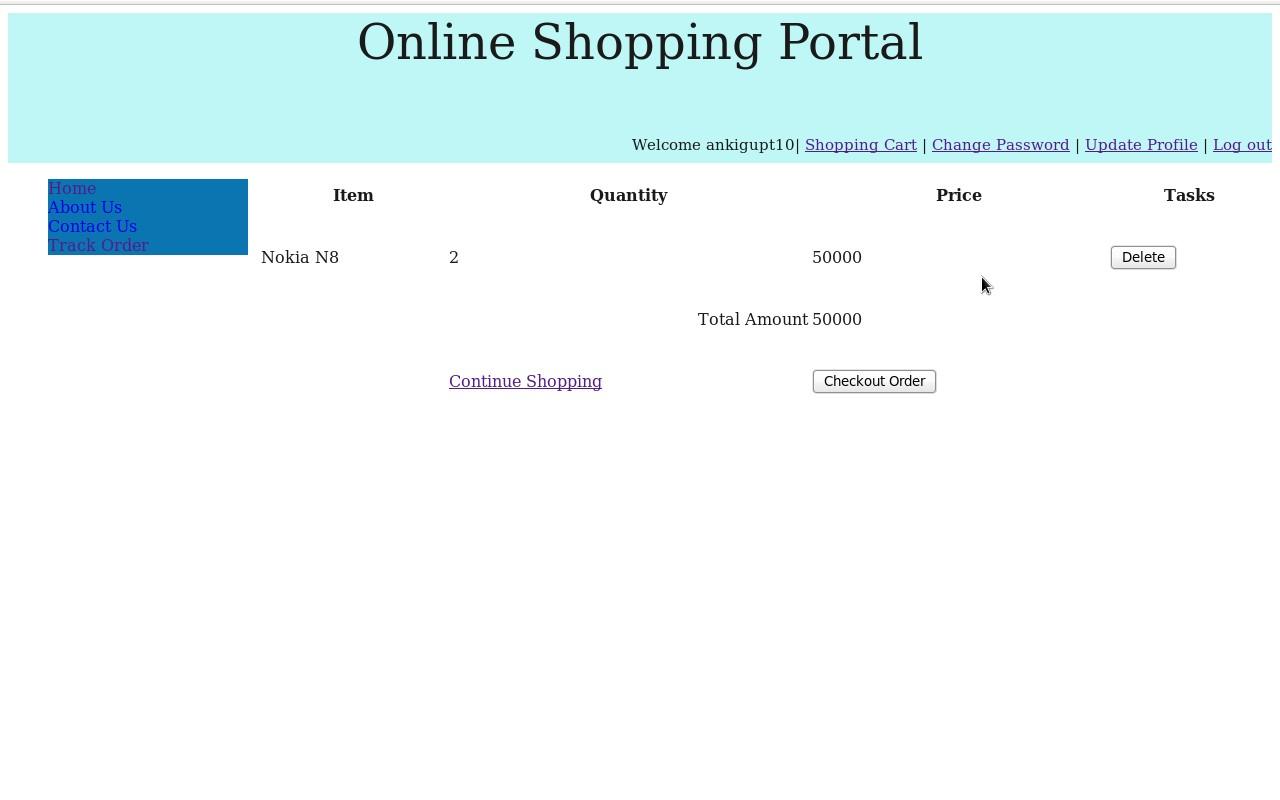


**Registration page:**



**Login Page:**





### 7. CONCLUSION

We have presented Co Al, our approach to facilitate the incremental consolidation of data-intensive Flows. Co Al starts from data Flows that satisfy single information requirements. Iteratively, Co Al Identifies different possibilities for integrating new data Flows into the existing multi- Flow, focusing on the maximal data Flow reuse. Finally, Co Al suggests a unified data Flow design evaluating it with the user-specified cost model. We have developed a prototype that implements the complete functionality of Co Al. We used it to evaluate the efficiency, scalability, and the quality of the output solutions of our approach, reporting the improvement of the overall execution time as well as other benifits of integrated multi- Flows. The final goal of our overall work is to provide an end-to-end platform for self-managing the complete lifecycle of BI solutions, from information requirements to deployment and execution of data-intensive Flow

### 8. BIBLOGRAPHY

The following books are referred during the analysis and execution phase of the project

1. M. Lenzerini, “Data integration: A theoretical perspective,” in PODS, 2002, pp. 233– 246.
2. D. Caruso, “Bringing Agility to Business Intelligence,” February 2011, Information Management,http://www.information-management.com/infodirect/2009191/business intelligence metadata analytics ETL data management-10019747-1.html.
3. R. Hughes, Agile Data Warehousing: Delivering world-class business intelligence systems using Scrum and XP. IUniverse, 2008.
4. Y. Chen, S. Alspaugh, and R. Katz, “Interactive analytical processing in big data systems: A cross-industry study of map reduce workloads,” Proceedings of the VLDB Endowment, vol. 5, no. 12, pp. 1802–1813, 2012.

**WEBLINKS:**

* 1. [www.c#tutorial.com -](http://www.c/#tutorial.com) covering all the most important C# concepts. This tutorial is primarily for new users.

* 1. [.www.DotnetSpider.com -](http://www.dotnetspider.com/) what is the .NET all about? For sample projects.