Project Title: **Sneaky Shoes E-Commerce**

Group Name: **UNT-DEV**

Team Members List:

* GIRISH VARDHAN GAVIREDDY
* RAHEEM ALI
* SAI KIRAN REDDY SOMA
* LETHASWI THOKALA
* PRATHYUSHA REDDY PIPPIRADA
* LIKHITHA REDDY SAMA
* SAI VENKATARAMA INNAMURI

Project description**:**

Our Project is to develop an E-Commerce website specifically designed for shoes. The global sneakers market was valued at UDS 78.59 billion in 2021 and is expected to register a compound annual growth rate (CAGR) of 5.2% from 2022 to 2030. The proliferation of online platforms across the globe act as one of the major factors driving the growth of the industry. To grab some of the market share our team is going to create an E-commerce website that is specially designed for footwear market using the latest technologies.

Programing Languages used for this project are:

* JavaScript
* React
* Mangodb
* Express etc

Here is a list of Features that we will implement in our website:

**USERS:**

Users will be able to create an account on site using their email address, choosing a username and password. The User collection stores all the data related to the users.

All the users will have the timestamp of Joining the site. The timestamp will be an hidden attribute attached to the user.

|  |
| --- |
| {  "\_id": "636321e276d849ce760ca1c1",  "username": "john",  "email": "johnfloyd@gmail.com",  "password": "bbf13ae4db87d475ca0ee5f97e397248....hashed\_password",  "timestamp\_joined": "2022-11-05T21:57:54.917Z"  } |

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Description** |
| \_id | String | A Globally unique value that represent the user |
| Username | String | A Single Unique Name for the User |
| e-mail | String | A Unique Email for the User |
| Password | String | A hashed password User to login |
| Time\_statmp | String | The time recorded when User joined the website |

**PROFILE**

A collection called "Profile" is used to store a user's profile or other personal information. The user may at any moment amend this user data. Except for \_id and user id, all fields are optional, and it is up to the user how much information they choose to share with the world. When a person registers for an account on the website, there will already be a default profile image there. The User collection stated above is referenced by the user id.

|  |
| --- |
| {  "\_id": "234321e276d849ce760ca2r4",  "user\_id":"636321e276d849ce760ca1c1",  "first\_name": "John",  "last\_name": "Floyd",  "Age": "22",  "Gender": "Male",  "size": "7.2",  "profile\_picture": "/static/uploads/profile\_picture/profile\_picture.jpg",  "website\_link": "www.Johnfloyd.com",  "Address\_line\_1": "9855 Ranch Valley",  "Address\_line\_2": "Apt #1",  "City": "Irving, City",  "State": "Texas",  "Country": "USA",  "pincode": "75063",  "Bio": "You are young and life is long"  } |

|  |  |  |
| --- | --- | --- |
| Name | Type | Description |
| \_id | String |  |
| User\_id | String | ID of user whom the profile belongs |
| First\_name | String | First Name of the User |
| Last\_name | String | Last Name of the User |
| Age | String | Age of the User |
| Profile\_picture | String | URL for the Profile picture of the user |
| Address\_line\_1 | String | Address Line 1 for the User |
| Address\_line\_2 | String | Address Line 2 for the User |
| City | String | City name where the user lives |
| State | String | State name where the user lives |
| Pin\_code | String | Pin co |

**SEARCH BOX AND FILTER OPTION**

An eCommerce website's search box and filter options aid consumers in finding the products they're looking for quickly. The search function of the website will then return a list of relevant products once customers enter keywords or phrases connected to the product they are looking for in the search box. This makes it easier for consumers to find what they're looking for even if they're unsure of which category it falls under or if the website has a lot of different products.

Contrarily, the filter options enable users to specify precise criteria for their search results, such as a particular price range, color, brand, or size. Users can more precisely find what they're looking for by using this to filter their search results. In comparison to painstakingly browsing through pages of search results, consumers can quickly and efficiently limit the search results to only see products that suit their precise needs.

In conclusion, the search bar and filter choices on an eCommerce website are crucial resources for assisting consumers in swiftly and simply locating the products they're looking for.

***Technical Details:***

* Node.js and Express.js, a well-liked framework for creating online applications, will be used to develop the database.
* The data will be stored in a NoSQL database, such as MongoDB, which is well-suited for dynamic and semi-structured data.
* The database will use RESTful API for communication between the front-end and back-end.

**ADD TO CART**

In order to store the things that a customer has added to their cart, the "Add to Cart" feature would need building a shopping cart object in the database. Information about the product, including the product ID, name, price, and quantity, would be included in the shopping cart object.

Here's a sample code snippet in Node.js that demonstrates how the "Add to Cart" feature could be implemented:

|  |
| --- |
| // Defining the shopping cart object  let shoppingCart = {};  // Function to add an item to the cart  function addToCart(productId, productName, productPrice, quantity) {  if (shoppingCart[productId]) {  // If the item is already in the cart, update the quantity  shoppingCart[productId].quantity += quantity;  } else {  // If the item is not in the cart, add it  shoppingCart[productId] = {  name: productName,  price: productPrice,  quantity: quantity  };  }  }  // Example usage  addToCart(101, "Shoes Adidas", 1022, 1);  addToCart(102, "Shoes Nike", 1999, 2);  console.log(shoppingCart); |

**SHIPPING DETAILS**

A Sneaky Shoes eCommerce website's shipping details include specifics about how to deliver online-purchased goods to the intended address of the consumer. The following components are frequently found in this data:

* Shipping address: the address where the product will be delivered, which may be different from the billing address.
* Shipping method: the type of delivery service offered by the seller, such as standard ground shipping, expedited shipping, or international shipping.
* Shipping cost: the cost of delivering the product to the customer, which may be based on the weight of the item, shipping method, and destination. Etc.

**PAYMENT METHOD**

The many ways that customers can pay for the things they buy online are referred to as payment options on an eCommerce website. Typical forms of payment include:

* Credit/debit cards are accepted as forms of payment by customers, including Visa, Mastercard, American Express, and others.
* Customers can make cash payments when a product is delivered to their door using the cash on delivery (COD) method.

The chosen payment method may be determined by the buyer's preferences, the seller's policies, and the acceptance of the chosen payment method in the buyer's nation. To meet the needs and interests of various customers, Sneaky Shoes e-commerce websites will provide an alternatives of payments system.

**ORDER INVOICE AND EMAIL INVOICE**

A customer's purchase is described in full in an order invoice, which also includes the products they ordered, their prices, quantities, and delivery information. An electronic order invoice that is issued to the customer by email is known as an email invoice.

Here is an example of a database proposal for storing order invoices in JavaScript using MongoDB:

|  |
| --- |
| const Invoices = new mongoose.Schema({  invoiceID: {  type: Number,  required: true,  unique: true  },  orderID: {  type: Number,  required: true,  ref: 'Orders'  },  customerID: {  type: Number,  required: true,  ref: 'Customers'  },  invoiceDate: {  type: Date,  required: true,  default: Date.now  },  totalAmount: {  type: Number,  required: true  }  });  module.exports = mongoose.model('Invoices', Invoices); |

**EMAIL NOTIFICATION OF ORDER**

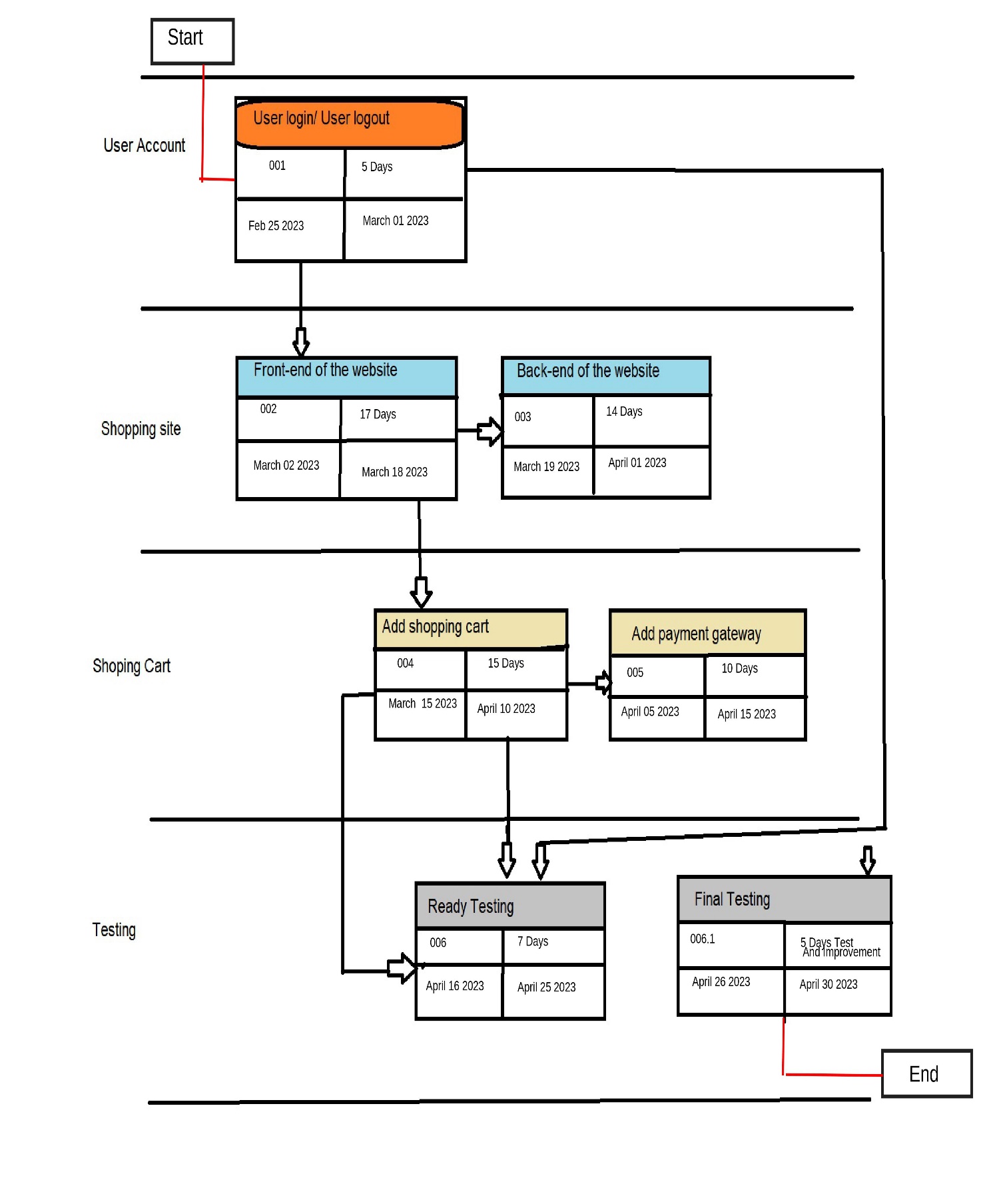
An automated email will be sent to a consumer to confirm the receipt of their transaction. The order number, date, items ordered, and total amount are frequently included. A consumer will receive an email notification informing them of the status of their order and giving them a record of their transaction. These notifications are a crucial tool for customer care and communication in e-commerce and we will issue this automatically after an order is placed.

|  |
| --- |
| const Notifications = new mongodb({  orderID: {  type: Number,  required: true,  ref: 'Orders'  },  customerID: {  type: Number,  required: true,  ref: 'Customers'  },  email: {  type: String,  required: true  },  subject: {  type: String,  required: true  },  message: {  type: String,  required: true  },  status: {  type: String,  required: true,  default: 'pending'  },  date: {  type: Date,  required: true,  default: Date.now  }  });  module.exports = mongoose.model('Notifications', Notifications); |

Project Timeline:

Gantt Chat:

PERT Chat:



Risk Management

An essential component of operating an eCommerce website is risk management. It includes locating, evaluating, and prioritizing potential threats to the website, then taking action to reduce or eliminate those threats. Typical danger in eCommerce is ***Verified Users.***

1. *Why do we need to* ***MONITOR*** *verified User?*

* Trust and security
* Better customer experience
* Preventing malicious activity

1. *reevaluation of the risks as the semester*

* *Manual User Testing*
* *Make application strong to be safe and secure*
* *Making sure the UI be friendly for our users.*

1. *The solution for this is as follows:*

* Email verification
* Phone verification
* Social media verification
* CAPTCHA
* Manual review

It's crucial to strike a balance between user verification and upholding a smooth and user-friendly experience. The needs and level of risk that the eCommerce website is exposed to will determine how well this balance is struck.

Team Members Role:

* Team Leader - Girish Vardhan Gavireddy
* Project Manager – Raheem Ali
* UI/UX Design - Prathyusha Reddy Pippirada
* Front-end Development (Team-A) - Sai Venkatarama Innamuri, Likhitha Reddy Sama, Prathyusha Reddy Pippirada, Raheem Ali
* Back-end Development (Team-B) - Girish Vardhan Gavireddy, Raheem Ali, Sai Kiran Reddy Soma, Lethaswi Thokala.

All the Team members will work on both front-end and back-end of the website. The team above is created with the experienced knowledge before. But Each member will work on different feature of the website.

**Member Contribution Table**

|  |  |
| --- | --- |
| **Name** | **Contribution** |
| RAHEEM ALI | Sample coding, Features descriptions with Details, Risk Management |
| GIRISH VARDHAN GAVIREDDY | Project features and Details |
| SAI KIRAN REDDY SOMA | Project PPT and Email Notification Feature |
| LETHASWI THOKALA | Solutions for Risk Management and Environment Specification |
| PRATHYUSHA REDDY PIPPIRADA | Search box and filter options with their technical Details |
| LIKHITHA REDDY SAMA | Information Gathering about the content and programming environment and User Profile Feature idea |
| SAI VENKATARAMA INNAMURI | Gantt and PERT chart and Monitoring Risk that develop during development |