**Note:**

* **Along with every project a reference site is provided; students have to go through the reference site and make the clone along with add-on functionalities; students are allowed to clone similar kinds of websites.**

### **1. Task Management System**

**Website Reference:** Todoist (<https://todoist.com>)  
**Project Overview:**Clone the Todoist website to build a task management application that manages tasks by status (To‑Do, In Progress, Completed), priority, and deadlines. Student teams are encouraged to enhance the clone by adding extra functionalities according to project requirements.  
**Core Features (Flask Implementation):**

* User authentication (login/signup/logout)
* Create, edit, and delete tasks
* Categorize tasks by status and priority
* Filtering, sorting, and searching tasks
* Dashboard view for task progress  
  **Bonus Features:**
* Pagination for task lists
* Infinite scrolling on the dashboard  
  **UI/UX Considerations:**
* Responsive design with Bootstrap
* Clear visual indicators for task status
* Dark/light mode toggle  
  **Technologies:** Flask, SQLite, HTML, CSS, Bootstrap, Jinja Templating, VS Code

### **2. Blog Platform**

**Website Reference:** Medium (<https://medium.com>)  
**Project Overview:**Clone the Medium website to create a multi‑user blogging platform where users can create, edit, and delete posts, and readers can comment and like posts. Student teams are encouraged to add extra functionalities as per their project requirements.  
**Core Features (Flask Implementation):**

* User authentication and profile management
* Create, edit, and delete blog posts
* Commenting and liking system
* Filtering and searching posts by categories or tags  
  **Bonus Features:**
* Pagination for post listings
* Infinite scrolling for comment threads  
  **UI/UX Considerations:**
* Clean, modern layout with intuitive navigation
* SEO‑friendly structure
* Simple text editor for post‑creation  
  **Technologies:** Flask, SQLite, Bootstrap, VS Code

### **3. E-commerce Website**

**Website Reference:** Amazon (<https://www.amazon.com>)  
**Project Overview:**Clone the Amazon website to build an online store where users browse products, manage a shopping cart, and simulate a checkout process. Student teams are encouraged to incorporate additional functionalities as per their project requirements.  
**Core Features (Flask Implementation):**

* User authentication (customer and admin roles)
* Product catalogue with categories
* Shopping cart management and simulated checkout
* Filtering, sorting, and searching products  
  **Bonus Features:**
* Pagination for product listings
* Infinite scrolling for product feeds
* Basic cart and payment simulation  
  **UI/UX Considerations:**
* User‑friendly product display with high‑quality images
* Secure, clear checkout process
* Responsive design for all devices  
  **Technologies:** Flask, SQLite, Bootstrap, VS Code

### **4. Job Portal**

**Website Reference:** Indeed (<https://www.indeed.com>)  
**Project Overview:**Clone the Indeed website to build a job portal where employers post jobs and job seekers apply, including features like resume upload. Student teams are encouraged to extend the system with additional functionalities as required.  
**Core Features (Flask Implementation):**

* Dual authentication for employers and job seekers
* CRUD for job postings and applications
* Filtering, sorting, and searching jobs by category or location
* Email notifications for applications  
  **Bonus Features:**
* Pagination for job listings
* Infinite scrolling for application history  
  **UI/UX Considerations:**
* Clear job search and application forms
* Intuitive employer dashboard
* Responsive design  
  **Technologies:** Flask, SQLite, Bootstrap, VS Code

### **5. Inventory Management System**

**Website Reference:** Zoho Inventory (https://www.zoho.com/inventory/)  
**Project Overview:**Clone the Zoho Inventory website to build a system that tracks stock, suppliers, and orders in a warehouse or business setting. Student teams are encouraged to add extra functionalities based on their project requirements.  
**Core Features (Flask Implementation):**

* User authentication with role‑based access (Admin, Staff)
* CRUD for inventory items and supplier data
* Stock management with low‑stock alerts
* Filtering, sorting, and searching inventory  
  **Bonus Features:**
* Pagination for inventory lists
* Infinite scrolling for dynamic stock updates  
  **UI/UX Considerations:**
* Clear, colour‑coded dashboard
* Intuitive forms for product management
* Responsive design  
  **Technologies:** Flask, SQLite, Bootstrap, VS Code

### **6. Online Course Platform**

**Website Reference:** Udemy (<https://www.udemy.com>)  
**Project Overview:**Clone the Udemy website to develop a learning management system where instructors create courses and students enrol to access lessons and quizzes. Student teams are encouraged to add further functionalities as needed.  
**Core Features (Flask Implementation):**

* Separate authentication for instructors and students
* CRUD for courses, lessons, and quizzes
* Filtering and searching courses by category or instructor
* Dashboard to track course progress  
  **Bonus Features:**
* Pagination for course listings
* Infinite scrolling for course content feeds  
  **UI/UX Considerations:**
* Organised, clean layout with easy navigation
* Accessible design for content consumption
* Responsive design  
  **Technologies:** Flask, SQLite, Bootstrap, VS Code

### **7. Event Management System**

**Website Reference:** Eventbrite (<https://www.eventbrite.com>)  
**Project Overview:**Clone the Eventbrite website to create a web platform for event organizers to create events and for users to register or purchase tickets. Student teams are encouraged to extend the application with additional functionalities as per requirements.  
**Core Features (Flask Implementation):**

* User authentication for organizers and attendees
* CRUD for event creation and management
* Filtering, sorting, and searching events by date or category
* Registration and ticket booking process  
  **Bonus Features:**
* Pagination for event listings
* Infinite scrolling for event feeds  
  **UI/UX Considerations:**
* Visually appealing event cards
* Simple, secure registration process
* Responsive design  
  **Technologies:** Flask, SQLite, Bootstrap, VS Code

### **8. Hotel Booking System**

**Website Reference:** Booking.com (<https://www.booking.com>)  
**Project Overview:**Clone the Booking.com website to build a reservation system for hotels where customers search for rooms and manage bookings. Student teams can enhance the system with additional features as per requirements.  
**Core Features (Flask Implementation):**

* User authentication for customers and hotel admins
* CRUD for room listings and bookings
* Filtering, sorting, and searching for hotels/rooms
* Booking history and cancellation management  
  **Bonus Features:**
* Pagination for room listings
* Infinite scrolling for hotel search results  
  **UI/UX Considerations:**
* Elegant, user‑friendly search interface
* Clear room details and pricing
* Responsive design  
  **Technologies:** Flask, SQLite, Bootstrap, VS Code

### **9. Personal Finance Tracker**

**Website Reference:** Mint (<https://www.mint.com>)  
**Project Overview:**Clone the Mint website to build an application that logs and monitors income, expenses, and budgets. Student teams are encouraged to include additional functionalities as needed.  
**Core Features (Flask Implementation):**

* User authentication
* CRUD for financial transactions
* Filtering and searching by date, category, or amount
* Dashboard with spending trends and visualizations  
  **Bonus Features:**
* Pagination for transaction history
* Infinite scrolling for logs  
  **UI/UX Considerations:**
* Clear, intuitive charts and forms
* Responsive design for desktop and mobile
* Simple data entry  
  **Technologies:** Flask, SQLite, Bootstrap, VS Code

### **10. Recipe Sharing Platform**

**Website Reference:** Allrecipes (<https://www.allrecipes.com>)  
**Project Overview:**Clone the Allrecipes website to develop a community website for sharing, rating, and commenting on recipes. Student teams may extend the functionality according to their project requirements.  
**Core Features (Flask Implementation):**

* User authentication and profile management
* CRUD for recipes with images, ingredients, and instructions
* Filtering, sorting, and searching by cuisine or ingredient
* Commenting and rating system  
  **Bonus Features:**
* Pagination for recipe lists
* Infinite scrolling for comment sections  
  **UI/UX Considerations:**
* Visually appealing recipe cards
* Intuitive recipe submission forms
* Responsive layout  
  **Technologies:** Flask, SQLite, Bootstrap, VS Code

### **11. Quiz Application**

**Website Reference:** Kahoot (<https://kahoot.com>)  
**Project Overview:**Clone the Kahoot website to develop a quiz application where admins can create quizzes and users can attempt them to test their knowledge. Student teams may add further functionalities as per project requirements.  
**Core Features (Flask Implementation):**

* User authentication for quiz takers and admins
* CRUD for quizzes, questions, and answers
* Filtering and searching quizzes by category or difficulty
* Timer‑based quiz functionality  
  **Bonus Features:**
* Pagination for quiz lists
* Infinite scrolling for question banks  
  **UI/UX Considerations:**
* Distraction‑free quiz interface
* Clear feedback on quiz results
* Mobile‑friendly design  
  **Technologies:** Flask, SQLite, Bootstrap, VS Code

### **12. Movie Database**

**Website Reference:** IMDb (<https://www.imdb.com>)  
**Project Overview:**Clone the IMDb website to build a searchable database of movies where users can view details, rate, and review films. Student teams are encouraged to add additional features as needed.  
**Core Features (Flask Implementation):**

* User authentication for reviewers and admins
* CRUD for movies, reviews, and ratings
* Filtering, sorting, and searching by genre, release date, etc.
* Detailed movie information display  
  **Bonus Features:**
* Pagination for movie listings
* Infinite scrolling for review sections  
  **UI/UX Considerations:**
* Visually appealing movie cards
* Clear forms for reviews
* Responsive design  
  **Technologies:** Flask, SQLite, Bootstrap, VS Code

### **13. Pet Adoption System**

**Website Reference:** Petfinder (<https://www.petfinder.com>)  
**Project Overview:**Clone the Petfinder website to build a system for animal shelters to list pets available for adoption and for users to submit adoption requests. Student teams may add extra functionalities based on requirements.  
**Core Features (Flask Implementation):**

* User authentication (Shelter Admin and Adopter roles)
* CRUD for pet profiles with images and details
* Filtering, sorting, and searching pets by type, age, breed
* Adoption request submission  
  **Bonus Features:**
* Pagination for pet listings
* Infinite scrolling for pet feeds  
  **UI/UX Considerations:**
* Visually appealing pet cards
* Clear and intuitive request forms
* Responsive design  
  **Technologies:** Flask, SQLite, Bootstrap, VS Code

### **14. Healthcare Appointment Booking System**

**Website Reference:** ZocDoc (<https://www.zocdoc.com>)  
**Project Overview:**Clone the ZocDoc website to build a healthcare appointment booking system where patients can search for doctors, view available appointment slots, and book appointments, while doctors manage their profiles and schedules. Student teams may enhance the clone with additional features according to requirements.  
**Core Features (Flask Implementation):**

* User Authentication: Separate login/signup processes for patients, doctors, and administrators
* Appointment Management: Patients can book, cancel, or reschedule appointments; doctors can manage their availability and appointment slots
* CRUD Operations: Create, edit, and delete doctor profiles and appointment details
* Filtering, Sorting, and Searching: Users can filter and search for doctors by specialty, location, or available times, with sorting options for appointment dates and doctor ratings  
  **Bonus Features:**
* Pagination for doctor listings and appointment histories
* Infinite scrolling on doctor search results or appointment lists  
  **UI/UX Considerations:**
* Responsive design with Bootstrap
* Professional layout with clear forms and dashboards
* Intuitive navigation for patients and doctors  
  **Technologies:** Flask, SQLite, HTML, CSS, Bootstrap, Jinja Templating, VS Code

### **15. Grocery Delivery System**

**Website Reference:** Instacart (<https://www.instacart.com>)  
**Project Overview:**Clone the Instacart website to create an online platform for ordering groceries with product browsing, cart management, and simulated order processing. Student teams are encouraged to extend the system with extra functionalities per their requirements.  
**Core Features (Flask Implementation):**

* User authentication (Customer and Admin roles)
* CRUD for grocery products and order management
* Filtering, sorting, and searching products
* Shopping cart and checkout simulation  
  **Bonus Features:**
* Pagination for product listings
* Infinite scrolling in product views  
  **UI/UX Considerations:**
* Clear product images and descriptions
* Secure, intuitive checkout process
* Responsive design  
  **Technologies:** Flask, SQLite, Bootstrap, VS Code

### **16. Employee Leave Management System**

**Website Reference:** BambooHR (<https://www.bamboohr.com>)  
**Project Overview:**Clone the BambooHR website to build a system where employees can apply for leave and managers can review and process these requests. Student teams are encouraged to add extra features according to their project requirements.  
**Core Features (Flask Implementation):**

* Role‑based authentication (Employee, Manager)
* CRUD for leave requests with status tracking
* Filtering, sorting, and searching leave applications
* Manager dashboard for approval/rejection  
  **Bonus Features:**
* Pagination for leave listings
* Infinite scrolling for leave history  
  **UI/UX Considerations:**
* Clear and accessible forms
* Visual indicators for leave status
* Responsive layout  
  **Technologies:** Flask, SQLite, Bootstrap, VS Code

### **17. Online Food Ordering System**

**Website Reference:** UberEats (<https://www.ubereats.com>)  
**Project Overview:**Clone the UberEats website to create a web‑based food ordering system for a college canteen, where users can browse the menu, place orders, and track their order status. Student teams are encouraged to include extra functionalities based on requirements.  
**Core Features (Flask Implementation):**

* User authentication (Students, Canteen Staff, Admin)
* CRUD operations for menu items and orders
* Order tracking (e.g., Pending, Preparing, Ready, Delivered)
* Payment integration (Cash on Delivery/Online Payment)  
  **Bonus Features:**
* Order history for students
* Real‑time notifications for order updates
* Discounts and coupon management  
  **UI/UX Considerations:**
* Easy‑to‑navigate menu display
* Clear order summary before checkout
* Mobile‑friendly and responsive design  
  **Technologies:** Flask, SQLAlchemy (SQLite/PostgreSQL), Jinja2, Bootstrap, VS Code

### **18. Lost and Found System**

**Website Reference:** LostAndFound.com (<https://lostandfound.com>)  
**Project Overview:**Clone the LostAndFound.com website to develop a system for reporting and finding lost items on campus. Student teams are encouraged to incorporate additional functionalities as per their requirements.  
**Core Features (Flask Implementation):**

* User authentication (Students, Faculty, Admin)
* CRUD operations for lost and found item listings
* Search and filter functionality (by category, date, location)
* Contact system for claim verification  
  **Bonus Features:**
* Image upload for lost and found items
* Status tracking (Lost, Found, Returned)
* Email notifications for matched items  
  **UI/UX Considerations:**
* Simple reporting forms
* Visual gallery view for found items
* Mobile‑responsive design  
  **Technologies:** Flask, SQLAlchemy (SQLite/PostgreSQL), Jinja2, Bootstrap, VS Code

### **19. Online Bookstore**

**Website Reference:** Barnes & Noble (<https://www.barnesandnoble.com>)  
**Project Overview:**Clone the Barnes & Noble website to build an online bookstore where users can browse books by category, add them to their cart, and place orders. Student teams are encouraged to enhance the clone with extra features as needed.  
**Core Features (Flask Implementation):**

* User authentication (Customers, Admin)
* CRUD operations for books and orders
* Shopping cart and checkout system
* Order tracking (e.g., Pending, Shipped, Delivered)  
  **Bonus Features:**
* Search and filter books by category, author, or price
* Payment integration
* Book rating and review system  
  **UI/UX Considerations:**
* Organised book displays with images and descriptions
* User‑friendly checkout process
* Mobile‑responsive design  
  **Technologies:** Flask, SQLAlchemy (SQLite/PostgreSQL), Jinja2, Bootstrap, VS Code

### **20. Hostel Management System**

**Website Reference:** Hostelworld (<https://www.hostelworld.com>)  
**Project Overview:**Clone the Hostelworld website to build a system for managing hostel accommodations, where students can book rooms, check availability, and manage their stay. Student teams are encouraged to add further functionalities as per requirements.  
**Core Features (Flask Implementation):**

* User authentication (Students, Admin, Staff)
* CRUD operations for hostel rooms, bookings, and student details
* Room availability checking and booking system
* Rent payment tracking and status updates  
  **Bonus Features:**
* Notification system for rent due dates and booking confirmations
* Search and filter for available rooms  
  **UI/UX Considerations:**
* Intuitive booking interface with room selection feature
* Visual indicators for room occupancy status
* Mobile‑responsive design  
  **Technologies:** Flask, SQLAlchemy (SQLite/PostgreSQL), Jinja2, Bootstrap, VS Code

### **21. Blood Donation Management System**

**Website Reference:** American Red Cross Blood (<https://www.redcrossblood.org>)  
**Project Overview:**Clone the American Red Cross Blood website to build a system that connects blood donors with those in need. Student teams are encouraged to further enhance the system with additional features as per requirements.  
**Core Features (Flask Implementation):**

* User authentication (Donors, Recipients, Admin)
* CRUD operations for donor registration and blood requests
* Blood type tracking and availability management
* Notification system for matching blood donors  
  **Bonus Features:**
* Location‑based donor search
* Emergency request prioritization
* Donation history tracking  
  **UI/UX Considerations:**
* Simple registration and request forms
* Clear indicators of available blood types
* Mobile‑friendly design  
  **Technologies:** Flask, SQLAlchemy (SQLite/PostgreSQL), Jinja2, Bootstrap, VS Code

### **22. E-commerce Website (Nykaa Clone)**

**Website Reference:** [Nykaa](https://www.nykaa.com)

#### **Project Overview:**

Clone the **Nykaa** website to build a full-fledged **e-commerce application** where users can browse, search, and purchase products. Student teams are encouraged to extend the clone with additional functionalities as per their requirements.

#### **Core Features (Flask Implementation):**

* **User Authentication:** Sign up, login, logout, and password management
* **Product Management:** CRUD operations for adding, editing, and deleting products
* **Shopping Cart & Checkout:** Add/remove items, update quantity, and process checkout
* **Filtering & Searching:** Products can be searched and filtered by category, brand, or price
* **Order Management:** Users can view past orders and track delivery status
* **Dashboard:** Admin panel for managing users, products, and orders

#### **Bonus Features:**

* **Wishlist & Favorites:** Users can save items for later
* **Ratings & Reviews:** Customers can leave feedback and rate products
* **Discounts & Offers:** Apply promo codes or special discounts
* **Pagination & Infinite Scrolling:** Browse products smoothly
* **Recommendation System:** Suggest products based on user preferences

#### **UI/UX Considerations:**

* **Elegant & Modern UI:** Aesthetic, user-friendly design inspired by **Nykaa**
* **Mobile Responsiveness:** Optimized for both desktop and mobile users
* **Smooth Navigation:** Intuitive product browsing and easy checkout process

#### **Technologies:**

* **Backend:** Flask, SQLite (Django will be used as the syllabus will progress)
* **Frontend:** Bootstrap, JavaScript, HTML, CSS
* **Development Tools:** VS Code