IA Projects

General Guidelines:

- Teams composed from 5-6 students who are registered for the course.
- Each team's Application should satisfy its software requirements.
- Backend should be in (.Net Core Web API) and the Frontend should be in React.js.
- Every team member will be questioned regarding any frontend or backend related matter
- Any Code written without understanding will allow student to get Zero in project discussion.

1- EventMaster

A powerful event planning and management tool to help users organize conferences, workshops, and community events efficiently.

Actors (3 types of users):

- Admin
- Event Organizer
- Participant

Requirements:

- Different actors can log in/log out. A participant can view events without logging in but can’t do anything further unless they log in.
- Admin can manage (accept or reject) new registered accounts (Event Organizers).
- Admin can accept or reject events written by the event organizers. If the event post is accepted, the event can appear for any participant.
- Participant can view all posted events. Participant can search for event based on criteria such as location, and Date.
- Event includes: Event organizer name, Title, Description, Venue, Event date, Ticket Price, Number of tickets left, and Number of participants submitted.
- Participant can register for an event by paying for a ticket if there are available tickets to be bought. Also Participant can save important events.
- Event organizer can manage events (create, list, update, delete) (CRUD).
- Event organizer can upload attachments (e.g., event materials) and send updates to participants.
- Participant receive event notifications and updates. Participant can download event-related attachments.

Notes:

- **You must use real-time sockets as an example.**
- **DB design (Schema) is required**

2- RentMate

A property rental platform to connect landlords and tenants for rental agreements.

Actors (3 types of users):

- Admin
- Landlord
- Tenant

Requirements:

- Different actors can log in/log out. A Tenant can view properties without logging in but can’t do anything further unless they log in.
- Admin can manage (accept or reject) new registered accounts (Landlord).
- Admin can accept or reject posts written by the landlord. If the property post is accepted, the post can appear for any tenant.
- Tenant can view all posted properties. Tenant can search for properties based on criteria such as location, and price.
- Property post includes: Landlord name, Title, Description, Price, Location, images, Number of viewers (post readers), and rental status (available, rented).
- Tenant can apply for rentals. The applied proposal can contain required documents for rental approval. Also Tenant can save important posts.
- Landlord can manage properties (create, list, update, delete) (CRUD).
- Landlord can review proposals and accept or reject them. Post is removed if a proposal is accepted (rented).
- Landlord Communicate with tenants via messaging and comments

Notes:

- **You must use real-time sockets as an example.**
- **DB design (Schema) is required**

3- ArtGallery

An online exhibition and marketplace for artists to showcase and sell artwork.

Actors (3 types of users):

- Admin
- Artist
- Buyer

Requirements:

- Different actors can log in/log out. A Buyer can view artworks without logging in but can’t do anything further unless they log in.
- Admin can manage (accept or reject) new registered accounts (Artist).
- Admin can accept or reject posts written by the artist. If the artwork post is accepted, the post can appear for any buyer.
- Buyer can view all posted artworks. Buyer can filter for artworks based on criteria such as artist, and tags.
- Artwork post includes: Artist name, Title, Description, Initial Price (minimum price the seller is willing to accept), Auction Start Time, Auction End Time, Category (Portrait, Landscape), Tags (keywords to help buyers find the artwork, e.g., "modern, oil painting, vibrant"), Image.
- Buyer can start bid price in order to purchase artwork. Each new bid must be at least $10 higher than the last bid.
- Artist can manage artworks (create, list, update, delete) (CRUD).
- Artist can extends auction time. Artist can view winner bidder after auction end time.
- Any visitor can view bid history of specific artwork (list of previous bids with timestamps).

Notes:

- **You must use real-time sockets as an example.**
- **DB design (Schema) is required**

4- MarketPlace

An e-commerce platform for small vendors to list and sell their products online.

Actors (3 types of users):

- Admin
- Vendor
- Customer

Requirements:

- Different actors can log in/log out. A Customer can view products without logging in but can't do anything further unless they log in.
- Admin can manage (accept or reject) new registered accounts (Vendor).
- Admin can enable or disable some permissions (features) for vendor role or individual vendors.
- Admin can accept or reject posts written by the Vendor. If the product post is accepted, the post can appear for any customer. Admin can enable permission for specific vendors, so that no need to accept their posts.
- Customer can view all posted products. Customer can search for products based on criteria such as category, and price.
- Product post includes: Vendor name, Title, Description, Price, images, Number of available units, Number of viewers (post readers).
- Customer can purchase product as 1 unit or more. Customer can save important posts.
- Vendor can manage products (create, list, update, delete) (CRUD). Vendor can update number of units per product.
- Customer can't purchase products that are out of stock (zero units). Vendor can view history of the customers purchased specific product.

Notes:

- **You must use real-time sockets as an example.**
- **DB design (Schema) is required**

5- CarShare

A peer-to-peer car rental platform where car owners rent vehicles to verified users.

Actors (3 types of users):

- Admin
- Car Owner
- Renter

Requirements:

- Different actors can log in/log out. A Renter can view available cars without logging in but can't do anything further unless they log in.
- Admin can manage (accept or reject) new registered accounts (Car Owner).
- Admin can accept or reject car posts written by the car owners. If the post is accepted, the post can appear for any renter.
- Renter can view all available cars. Renter can search for cars based on criteria such as car type, and price.
- Car post includes: Owner name, Title, Description, Car Type, Brand, Model, Year, Transmission (Automatic / Manual), Location, Rental Status (Available / Rented) Availability Dates (Start/End), Rental Price.
- Renter can apply for car rentals, submitting license verification and proposal documents.
- Car Owner can manage car posts (create, list, update, delete) (CRUD). Car owner can't delete car post if it is rented.
- Car Owner can review proposals and accept or reject them. Car Post is marked as "Rented" until returned.
- Renters can rate the car and provide feedback. Only users who have rented the car can submit feedback, and it should be listed on the car post.

Notes:

- **You must use real-time sockets as an example.**
- **DB design (Schema) is required**

6- BookSwap

A book exchange platform where users lend and borrow books.

Actors (3 types of users):

- Admin
- Book Owner
- Reader

Requirements:

- Different actors can log in/log out. A Reader can view available books without logging in but can’t do anything further unless they log in.
- Admin can manage (accept or reject) new registered accounts (Book Owner).
- Admin can accept or reject book posts written by the book owners. If the post is accepted, the post can appear for any reader.
- Reader can view all available books. Reader can search for books based on criteria such as genre, and borrow price.
- Book post includes: Book Owner name, Title, Genre, ISBN, Language, Publication Date, Borrow Status (Available / Borrowed) Availability Dates (Start/End), Borrow Price, Cover Image.
- Reader can apply to borrow books, submitting requests.
- Book Owner can manage books posts (create, list, update, delete) (CRUD). Book owner can’t delete book post if it is borrowed.
- Book Owner can review requests and accept or reject them. Book Post is marked as “Borrowed” and removed from availability.
- Readers can like or dislike books. They can also write comments on a book post, with a single level of replies.

Notes:

- **You must use real-time sockets as an example.**
- **DB design (Schema) is required**