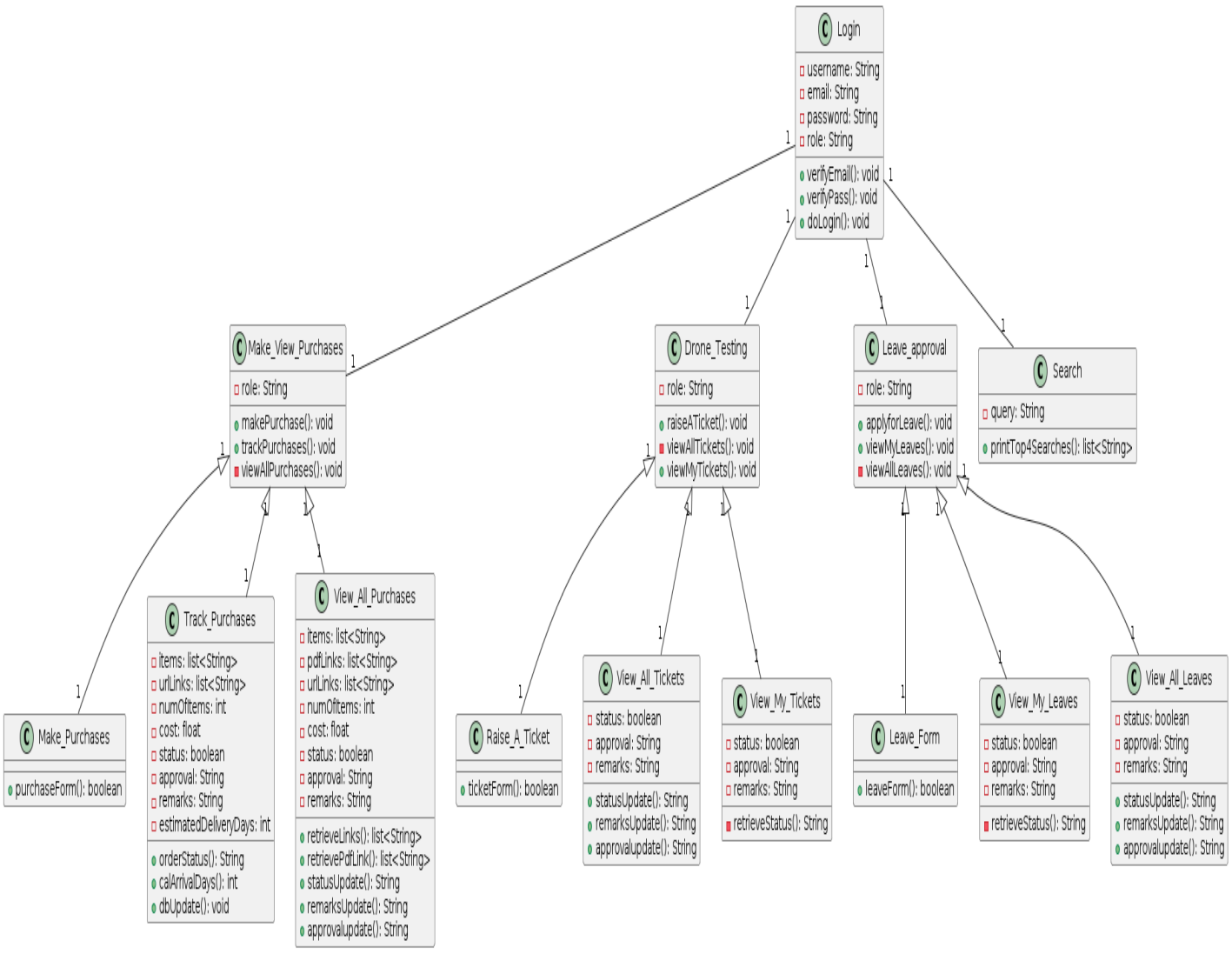
Product Design

Team **31 (Arka Organization Management)**

Members **Aditya Garg, Agrim Mittal, Aniket Bansal, Atidipt Ashnin, Shivam Singh**

Design Model

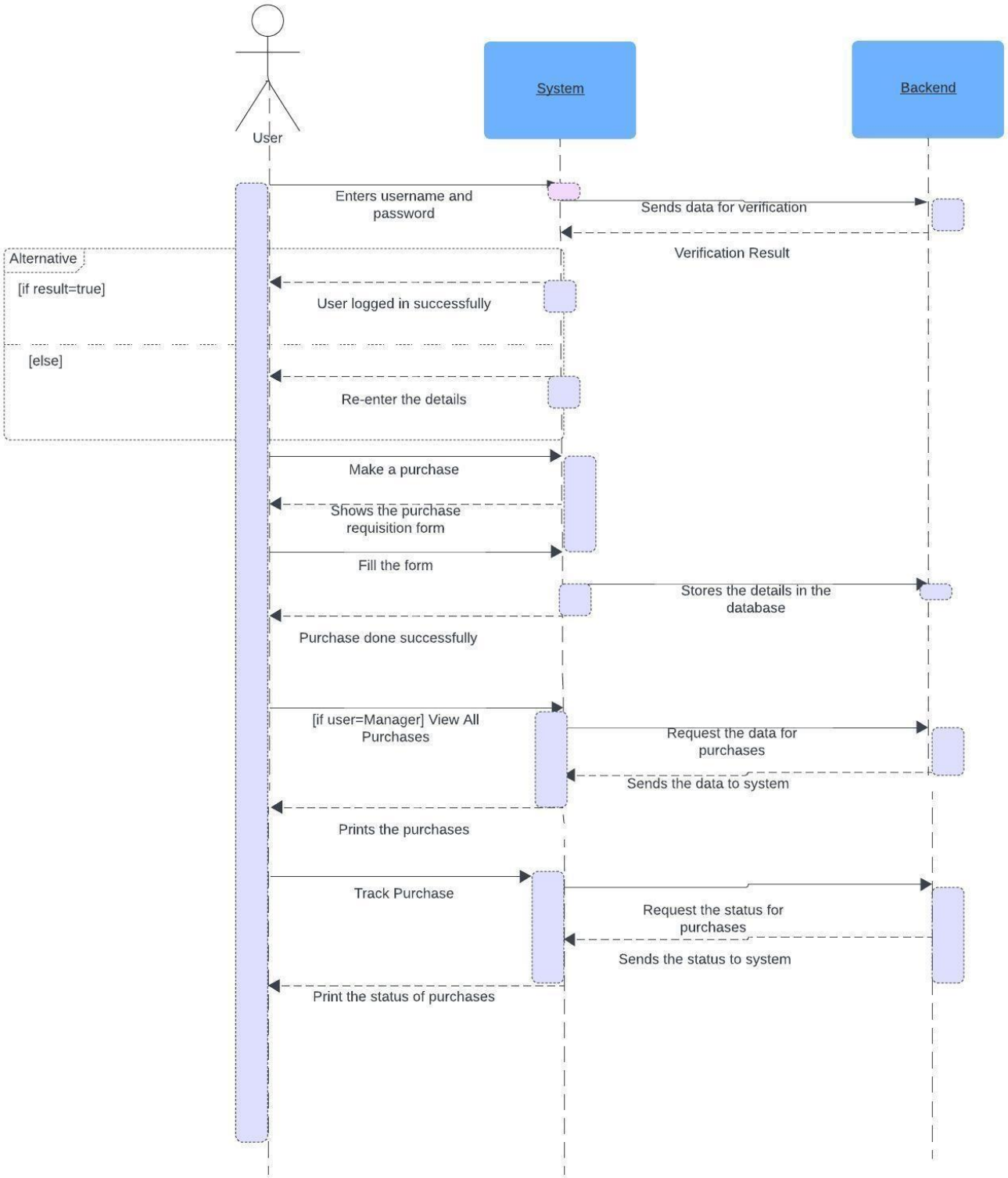


Login	Class state • Maintains information about the User credentials that include the Username, E-mail address, Password, and Role in the company. Class behavior • Validates and confirms if the data entered corresponds to a registered user. • If the user credentials match with the database, the page redirects to the Home page of the user. • If the credentials entered are incorrect, the system raises an alert and sends error/warning message.
Make and view Purchases	Class state • Used to redirect to the page which allows making purchase, tracking purchases and viewing all purchases. Class behavior • Allows filling of purchase form with the required constraints. • Allows viewing the products orders and all the products ordered depending on user role
View All Purchases	Class state • Maintains records of purchases made by users using the system. Class behavior • View all purchases in a easy to use tabular format. • The Manager has the option to Approve the purchase request, change Status of the purchase, and leave Remarks.
Track purchases	Class state • Stores data about all the purchases made by the currently logged-in user. Class behavior • View the status of the purchases.

Make Purchase	Class state • Temporarily stores the data filled in by the user in the form. Class behavior • Sends all data filled to the database on clicking the Submit button along with the username of the User. • In case any required field has been left unfilled then returns error message and does not send details. • Gives an alert message when the form is submitted successfully
Drone testing	Class state • Used for redirection to the page which leads to raising ticket for drone issues, viewing all raised tickets, and viewing tickets raised by the user. Class behavior • Allows filling of ticket raising form with the required constraints. • Allows viewing the tickets raised depending on user role.
Raise A Ticket	Class state • Temporarily stores the data filled in by the user in the form. Class behavior • Sends all data filled to the database on clicking the Submit button along with the username of the User. • In case any required field has been left unfilled then returns error message and does not send details. • Gives an alert message when the form is submitted successfully.
View All Tickets	Class state • Maintains records of tickets raised made by users using thesystem. Class behavior • View all tickets in a easy to use tabular format. • The Manager has the option to change status of raised ticket and update remarks.
View My Tickets	Class state • Stores data about all the tickets raised by the currently logged-in user. Class behavior • View the status of the tickets raised.

Search	Class state • Takes input query from the user and returns search results by API call. Class behavior • View the top 4 products of the search along with its rating and reviews.
Leave Approval	Class state • Used for redirection to the page which leads to leave apply page, viewing all applied leaves, and viewing leaves applied by a particular user. Class behavior • Allows filling of leave approval form with the required constraints. • Allows viewing the leaves applied depending on user role.
Leave Form	Class state • Temporarily stores the data filled in by the user in the form. Class behavior • Sends all data filled to the database on clicking the Submit button along with the username of the User. • In case any required field has been left unfilled then returns error message and does not send details. • Gives an alert message when the form is submitted successfully.
View All Leaves	Class state • Maintains records of leaves applied by users using the system. Class behavior • View all leaves applied in a easy to use tabular format. • The Manager has the option to change status of applied leave and update remarks.
View My Leaves	Class state • Stores data about all the leaves applied by the currently logged-in user. Class behavior • View the status of the leaves applied.

Sequence Diagram(s)



Design Rationale

- **React for User Interface:**
  - React.js was preferred for frontend development as it provides essential features like routing, middleware support, and template engines.
  - React's component-based architecture promotes modular development, making it easier to manage complex UIs.
  - It's diverse libraries and tools helps to create interactive and responsive user interfaces with ease.

- **Incorrect Login Detection Failure:**
  - The system matches the string message returned on entering the login details to decide if the user should be allowed access to the system or not.
  - The unregistered users were also allowed access as the string matching wasn't working as desired.
  - This was changed to check the status returned. Status: 201 was used for correct users and Status: 210 was used for incorrect users.
  - Checking the status avoided any design errors and hence the login system worked as desired.

- **Using MongoDB for database management**
  - MongoDB is a NoSQL database, which means it does not require a predefined schema like traditional relational databases.
  - MongoDB is designed to scale out horizontally, making it suitable for handling large volumes of data and high throughput applications.

- **Data retrieval through Form**
  - Data in the Radio Button format needed to be retrieved.
  - We changed the radio buttons to a drop-down format. This simplified the data retrieval process.
  - Doing this was also a benefit from the design perspective as it shortened the form which otherwise was unnecessarily long.

- **Date display format**
  - Earlier the date type was used to both retrieve the current data and storing it in the database.
  - The date format during extraction was: "DD-MM-YYYY" while in the database it was stored in: "MM-DD-YYYY". This led to an erroneous date being displayed e.g. If the date is 11 March 2024, it was displayed as 03 November 2024.
  - The date was then stored in a string format. The day, month and year were extracted from the Date() function and were concatenated into a string.