Student name: Sokha Ordom

1. **What have you done for the project last week?**
    - Last week, I worked on setting up the basic structure of the Java project using JavaFX. I also implemented the user authentication system, allowing users to register and log in. Additionally, I created the event listing page where users can view available events.
2. **What will you do this week?**
    - This week, I plan to work on the ticket ordering functionality. I will implement the backend logic for selecting tickets, adding them to the cart, and processing the order. I will also start integrating a payment gateway (e.g., Stripe) for handling payments.]
3. **Any problem related to project development?**
    - I faced some challenges with integrating the payment gateway due to API documentation issues. I will need to spend more time understanding the payment gateway's API to ensure smooth integration.

Student name: Chheng Rayuth

1. **What have you done for the project last week?**
    - Last week, I worked on designing the database schema for the project. I created tables for users, events, tickets, and orders using MySQL. I also implemented the backend logic for fetching event details from the database and displaying them on the frontend.
2. **What will you do this week?**
    - This week, I plan to work on the frontend design for the ticket ordering system. I will create the user interface for selecting tickets, viewing the cart, and completing the purchase. I will also work on improving the UI/UX of the event listing page.
3. **Any problem related to project development?**
    - I had some issues with database optimization, especially when fetching large amounts of event data. I will need to look into indexing and query optimization to improve performance.

Task 2: Technologies Used For Development

Programming language: Java
Framework: JavaFX GUI, SceneBuilder
Frontend: JavaFX for GUI
Database: Mysql (for storing user, event and ticket data)
Version Control: Github
Testing: Junit

Task 5: Write Test Plan

For your **Event Ticket Ordering System**, you can create a test plan to ensure the application works as expected. Here's an example:

Test Case 1: User Authentication

- **Input**: Valid username and password.
- **Expected Output**: User successfully logs in and is redirected to the event listing page.
- **Input**: Invalid username or password.
- **Expected Output**: Error message displayed (e.g., "Invalid credentials").

Test Case 2: Ticket Ordering

- **Input**: User selects 2 tickets for an event and proceeds to checkout.
- **Expected Output**: Tickets are added to the cart, and the user can proceed to payment.
- **Input**: User selects more tickets than available.
- **Expected Output**: Error message displayed (e.g., "Not enough tickets available").

Test Case 3: Payment Integration

- **Input**: User enters valid payment details (e.g., credit card information).
- **Expected Output**: Payment is processed successfully, and the user receives a confirmation.
- **Input**: User enters invalid payment details.
- **Expected Output**: Error message displayed (e.g., "Payment failed").

Test Case 4: Event Listing

- **Input**: User searches for an event by name.
- **Expected Output**: Relevant events are displayed in the search results.
- **Input**: User filters events by date.
- **Expected Output**: Events are filtered and displayed based on the selected date.

Test Case 5: Database Performance

- **Input**: Fetching 1000 events from the database.
- **Expected Output**: Events are retrieved within a reasonable time (e.g., less than 2 seconds).