Tools and Frameworks:

1. Web Framework: Since you have a two-week timeline, I recommend using a Python web framework for rapid development. Flask is a lightweight and easy-to-learn choice.
2. Front-End: For the front end of your dashboard, you can start with HTML, CSS, and JavaScript. Considering your limited experience with JavaScript, we'll keep the front end simple.

Steps to Create the Dashboard:

1. Set Up Flask: Install Flask using pip, and create a Flask application. This will serve as your web server.
2. Database Integration: Connect your Flask app to your MySQL database using SQLAlchemy (an ORM for Python). You'll use SQLAlchemy to retrieve data from the database.
3. HTML Templates: Create HTML templates for your dashboard using Flask's template engine (Jinja2). This is where your data will be displayed.
4. Route Creation: Define routes in Flask to render the HTML templates and pass data from the database to the templates.
5. Display Data: In your HTML templates, use Jinja2 to dynamically display data retrieved from the database. You can create tables, charts, or other visualizations as per your preference.
6. Web Interface: Users can access your dashboard through a web browser. Flask will handle user requests and serve the appropriate HTML templates.

Data Flow:

1. Your sensors (RFID, DHT11) collect data and send it to your Arduino.
2. As you mentioned, Your Arduino sends this data in JSON format to your laptop using PySerial.
3. Python script on your laptop reads this JSON data and stores it in the MySQL database using the MySQL connector.
4. When a user accesses the dashboard through a web browser, Flask routes handle the request.
5. Flask retrieves relevant data from the MySQL database using SQLAlchemy.
6. The retrieved data is rendered in the HTML templates, creating a user-friendly dashboard.

DETAILED SCHEDULE

Week 1:

Day 1: Project Kickoff and Planning (What, How, When)

- Define project objectives and success criteria.
- Create a project plan, including a Gantt chart for scheduling.
- Identify roles and responsibilities within the team.

Day 2-3: Architecture and Design (What, How, When)

- Plan the architecture of the system.
- Create an end-to-end diagram illustrating data flow through the system.

- Design the database schema.
- Decide on technologies and tools to be used.

Day 4-5: Sensor Integration and Atmega328p Programming (What, How, When)

- Set up the Atmega328p microcontroller for sensor data collection.
- Integrate DHT11 and RFID sensors with the microcontroller.
- Develop code for the microcontroller to collect and transmit data.

Day 6-7: Database Setup and Initial Backend Development (What, How, When)

- Set up a local MySQL database for data storage.
- Develop scripts for parsing and storing sensor data in the database.
- Begin designing the backend for the Flask-based web dashboard.

Week 2:

Day 8-10: Web Dashboard Development (Front-End) (What, How, When)

- Continue working on the web dashboard.
- Implement real-time data visualization features using Flask, HTML, CSS, and JavaScript.
- Focus on front-end development for user-friendly interface design.

Day 11-12: User Stories Implementation and Backend (What, How, When)

- Implement the user stories, including sensor data visualization, historical data access, and data export.
- Simultaneously, continue backend development to ensure data retrieval from the MySQL database.

Day 13: Testing and Quality Assurance (What, How, When)

- Conduct comprehensive testing, including unit testing and system testing for both front-end and back-end.
- Identify and address any bugs or issues.
- Refine the user interface for a polished look.

Day 14: Documentation and Final Testing (What, How, When)

- Create project documentation, including user manuals.
- Perform a final round of testing to ensure all features work correctly.
- Prepare for project submission.