BÁO CÁO: SO SÁNH CÁC MÔ HÌNH SDLC & THIẾT KẾ PROJECT "Bán đồng hồ online"

1. Mục tiêu  
- Hiểu và so sánh các mô hình SDLC phổ biến.  
- Thực hành: viết báo cáo mô hình (Waterfall, Agile, Spiral, V-Model, DevOps) và triển khai project "Bán đồng hồ online".  
- Deliverables: Báo cáo .docx, source code + database.

2. Tổng quan các mô hình SDLC  
a) Waterfall  
- Đặc điểm: tuần tự (Yêu cầu → Thiết kế → Triển khai → Kiểm thử → Bảo trì).  
- Ưu: dễ quản lý, phù hợp dự án yêu cầu rõ ràng, tài liệu.  
- Nhược: khó thay đổi khi yêu cầu biến động, phát hiện lỗi muộn.

b) Agile (ví dụ: Scrum)  
- Đặc điểm: lặp ngắn (sprint), phản hồi liên tục, ưu tiên làm sớm tính năng giá trị.  
- Ưu: thích nghi tốt với thay đổi, giao tiếp liên tục, giao hàng sớm.  
- Nhược: cần kỷ luật, quản lý product backlog; tài liệu có thể ít.

c) Spiral  
- Đặc điểm: kết hợp lặp và phân tích rủi ro theo từng vòng.  
- Ưu: quản lý rủi ro tốt cho dự án lớn/khó.  
- Nhược: phức tạp, chi phí cao.

d) V-Model  
- Đặc điểm: mở rộng Waterfall, gắn chặt giai đoạn kiểm thử tương ứng với mỗi giai đoạn thiết kế.  
- Ưu: kiểm thử rõ ràng, phù hợp yêu cầu ổn định.  
- Nhược: thiếu linh hoạt.

e) DevOps (một văn hóa + thực hành)  
- Đặc điểm: tích hợp phát triển và vận hành, CI/CD, tự động hoá.  
- Ưu: triển khai nhanh, feedback vòng lặp ngắn từ vận hành.  
- Nhược: cần đầu tư hạ tầng và thay đổi tổ chức.

3. So sánh ngắn (khi nào dùng model nào)  
- Yêu cầu rất rõ ràng, ít thay đổi → Waterfall/V-Model.  
- Yêu cầu thay đổi thường xuyên, cần giao sớm → Agile.  
- Dự án lớn, rủi ro cao → Spiral.  
- Muốn giao liên tục và vận hành hiệu quả → Agile + DevOps.

4. Đề xuất cho Project "Bán đồng hồ online"  
- Quy mô học tập / prototyping: dùng Agile (sprint 1-2 tuần). Lý do: dễ thay đổi UI/field, nhận phản hồi nhanh.  
- Tech stack đề xuất: Node.js + Express (REST API), SQLite (nhanh, file-based cho học), React (nếu cần giao diện).  
- CI/CD đơn giản: script npm + GitHub Actions (nếu triển khai).  
- Deliverables & milestones:  
 \* Sprint 1: DB schema + API CRUD sản phẩm, tạo README, báo cáo giữa kỳ.  
 \* Sprint 2: UI minimal (frontend), authentication cơ bản, validation.  
 \* Sprint 3: Export/Import data, tạo báo cáo (docx), hoàn thiện.

5. Thiết kế cơ sở dữ liệu  
- Bảng chính: WATCH (watchID, watchName, price, unitInStock, description, status, imageUrl, brandID, categoryID)  
- Các bảng phụ: CATEGORY, BRAND, PROMOTION, PROMO\_item, PURCHASEORDER, PO\_item, SUPPLIER, EMPLOYEE, PRICEHISTORY, GOODSRECEIPT, GR\_item, SALESORDER, SO\_item, INVOICE, RETURNSLIP, RL\_item, WARRANTYCARD, CUSTOMER.

6. Hướng dẫn cài đặt & sử dụng (kèm source code)  
- Backend: Node.js + Express, SQLite.  
- Commands:  
 \* cd product\_management\_project  
 \* npm install  
 \* npm run init-db # tạo sqlite DB và seed mẫu  
 \* npm start

7. API (tóm tắt)  
- GET /api/products → list products  
- GET /api/products/:id → product detail  
- POST /api/products → create product  
- PUT /api/products/:id → update product  
- DELETE /api/products/:id → delete product

8. Kết luận  
- Khuyến nghị phương pháp: Áp dụng Agile (Scrum) với hướng DevOps nhẹ cho CI/CD. Sử dụng thiết kế DB đã nộp làm cơ sở, triển khai nhanh bằng Node.js + SQLite để demo.