**Инструкции за успешно представяне и защита на дипломния проект**

*„Уеб платформа за продажба на касови апарати“*

**1. Предварителна подготовка (преди изпита)**

| **Задача** | **Проверка** |
| --- | --- |
| **XAMPP / Apache / MySQL** са стартирани | ![✓] |
| Папка **cash-register-site** е в htdocs |  |
| SQL-файлът е импортиран (виж “cash\_registers” в phpMyAdmin) |  |
| Файлът **tailwind.config.js** има правилните content пътища |  |
| В frontend са инсталирани dependency-тата → npm install |  |
| npm run dev стартира без грешки на [**http://localhost:5173**](http://localhost:5173) |  |
| Приложението показва поне 1 тестов продукт и 1 консуматив |  |

**2. Сценарий за демонстрация (15–20 мин)**

| **Етап** | **Какво показваш** | **Защо е важно** |
| --- | --- | --- |
| **1. Навигация** | Начална страница → меню → “Продукти” | SPA-навигация с React Router |
| **2. Филтри** | Филтрирай по цена, производител, функция | Доказваш динамично търсене |
| **3. Админ панел** | Влизаш c admin / admin → форма *Добави продукт* | Ролево управление |
| **4. Редакция / изтриване** | „✏️ Редактирай“ и „🗑 Изтрий“ | CRUD + защита (само admin) |
| **5. Консумативи + отстъпка** | Добави ≥ 10 консуматива, презареди → цена в червено (-10 %) | Логика за лоялни клиенти |
| **6. Количка** | Като логнат: добави продукт → *Количка* → избор на плащане Покажи IBAN / Crypto адрес | State management + плащания |
| **7. Поръчка** | Изпрати поръчка → phpMyAdmin: таблиците orders, order\_items | Backend транзакция |
| **8. Сервиз** | Бутон *Сервиз* → изпрати заявка → виж service\_requests | Допълнителна функционалност |
| **9. Контакт форма** | Изпрати съобщение → виж messages | Валидации + CORS + Session |
| **10. Гост потребител** | Logout → опитай да добавиш в количка → получаваш предупреждение | Guard логика |

**3. Ключови технологични акценти, които да споменеш**

* **REACT + Vite + Tailwind**: бърз билд, utility-first стилизиране.
* **PHP 8.2 + MySQL**: PDO, prepared statements, bcrypt хеширане.
* **Role-based Access**: requireLogin() и requireAdmin() в middleware.
* **CORS / proxy**: Vite проксира /api → избягва CORS проблеми.
* **Сесии**: credentials: "include" – еднакъв origin благодарение на proxy.
* **Транзакции** в addOrder.php (BEGIN / COMMIT).
* **SQL enum-и** – гарантират “твърдите” стойности за brand, size, function.
* **Responsive дизайн** – Tailwind grid + flex.

**4. Често задавани въпроси (гответе отговори)**

| **Въпрос на комисията** | **Какво отговаряш** |
| --- | --- |
| *Защо избра React, а не Vanilla JS?* | Компонентен модел, бърза разработка, React Router, useState/useEffect. |
| *Как защитавате базата от SQL инжекции?* | Prepared statements ($conn->prepare) и автоматично ескейпване от PDO. |
| *Как се гарантира, че само админ може да редактира?* | requireAdmin() проверява $\_SESSION['user']['role'], иначе 403. |
| *Къде се пазят сесиите?* | На сървъра (PHP), браузърът държи само session cookie. |
| *Как би скалирал проекта?* | Отделяне на API-то (REST), JWT вместо PHP сесии, CI/CD, Docker. |

**5. План Б (ако нещо се „счупи“)**

| **Проблем** | **Бърз фикс** |
| --- | --- |
| **Vite не стартира** | Изтрий node\_modules, npm install, после npm run dev. |
| **Apache порт 80 зает** | Смени в httpd.conf на 8080 + актуализирай target в vite.config.js. |
| **MySQL грешка “table exists”** | Стартирай SQL скрипта – той вече съдържа DROP DATABASE IF EXISTS. |
| **Tailwind класове без стил** | npx tailwindcss -i ./src/index.css -o ./dist.css –watch или CDN fallback. |

**6. Финални съвети**

* Тествай пълния сценарий **поне два пъти** преди защитата.
* Носи **флашка или Git repo** – ако компютърът е различен, можеш бързо да клонираш.
* Поддържай екрана чист: само браузър, редактор и phpMyAdmin таб.
* Говори уверено, показвай кода **само при въпрос** – фокусът е върху работещата система.
* Принтирай синтезирана „архитектурна схема“ (PDF или слайд) – клиент ↔ API ↔ БД.

Успех на защитата!