# BENJI HAIR - 고객 경험 관리 시스템 (Japan Edition) 결과 보고서



| **항목** | **내용** |
| --- | --- |
| **프로젝트명** | BENJI HAIR - Client Experience Management System |
| **타겟 시장** | 일본 (Japan Edition) - 다국어 지원 포함 |
| **플랫폼** | Web Application (Mobile-First SPA) |
| **주요 기술** | React, Firebase, Tailwind CSS, GitHub Actions |
| **배포 상태** | Firebase Hosting (CI/CD 자동화 적용) |

## 프로젝트 개요 및 목적

### 1.1. 기획 배경

미용실 방문 고객의 대기 시간을 효율적으로 활용하고, 디자이너에게 정확한 시술 정보를 전달하기 위해 **비대면 사전 상담 시스템**을 기획했습니다. 특히 일본 현지 미용실의 니즈를 반영하여 섬세한 상담 옵션과 다국어(일/한/영) 기능을 탑재했습니다.

### 1.2. 핵심 목표

* **고객 경험(CX) 향상**: 종이 차트 대신 모바일로 간편하게 자신의 스타일과 모발 상태를 입력.
* **업무 효율화**: 디자이너가 실시간으로 고객 정보를 확인하고 시술 전 'Critical Point(주의사항)'를 미리 인지.
* **운영 자동화**: 별도의 백엔드 구축 없이 Firebase를 활용하여 데이터 저장 및 인증 처리.

## 기술 아키텍처 (Technical Architecture)

### 2.1. 기술 스택 (Tech Stack)

* **Frontend**: React.js (v19), Create-React-App
* **Styling**: Tailwind CSS (CDN 방식 적용 - 빠른 프로토타이핑 및 커스텀 용이)
* **Icons**: Lucide-React (직관적인 UI 아이콘)
* **Backend (Serverless)**:
  + **Firebase Authentication**: 익명 로그인(Anonymous Auth)을 통한 진입 장벽 제거.
  + **Firebase Firestore**: NoSQL 기반의 실시간 데이터베이스 (상담 내역 동기화).
* **DevOps**:
  + **GitHub**: 소스 코드 버전 관리.
  + **GitHub Actions**: CI/CD 파이프라인 구축 (Push 시 자동 빌드 및 배포).
  + **Firebase Hosting**: 글로벌 CDN을 통한 정적 웹 호스팅.

### 2.2. 데이터 흐름도

1. **Client**: 고객이 모바일 웹에서 상담지 작성 → Firestore consultations 컬렉션에 저장.
2. **Database**: Firestore가 데이터 변경을 감지(OnSnapshot).
3. **Admin**: 관리자 대시보드에 실시간으로 새로운 고객 카드가 팝업.

## 3. 주요 기능 명세 (Feature Specifications)

### 3.1. 고객용 (Client View) - 4단계 상담 프로세스

코드를 분석한 결과, 고객 여정은 다음 4단계로 구성되어 있습니다.

1. **기본 정보 (Step 0)**
   * 이름, 전화번호 입력 (유효성 검사 포함).
   * 개인정보 수집 동의 (필수 체크).
2. **모발 상태 진단 (Step 1)**
   * 머리 길이, 두피 상태, 모발 고민(중복 선택), 최근 시술 이력.
   * 시술 이력 '있음' 선택 시, 구체적인 시기/종류 입력창이 동적으로 등장.
3. **상세 선호도 (Step 2)**
   * 샴푸 마사지 강도 (약하게/보통/강하게/알아서).
   * 방문 빈도 및 홈케어 습관 조사.
   * **동적 입력 폼 (Dynamic Inputs)**: 사용 중인 제품이나 복용 약물을 자유롭게 추가/삭제 가능.
4. **최종 확인 및 제출 (Step 3)**
   * 작성된 내용을 요약 카드로 보여주고 제출 시 Firestore로 전송.

### 3.2. 관리자용 (Admin Dashboard)

* **보안 접근**: 설정된 비밀번호('1234')를 통해서만 대시보드 진입 가능.
* **실시간 대기 리스트 (Real-time Queue)**:
  + onSnapshot 리스너를 사용하여 새로고침 없이 최신 접수 현황 확인.
  + 최신순 정렬 및 상태 아이콘 표시.
* **상세 전자 차트**:
  + 고객 선택 시 우측 패널에 상세 정보 표시.
* **스마트 경고 시스템 (Critical Alert Logic)**:
  + 코드 로직: scalpType이 '건조/지성'이거나 hairConditions에 '탈모/손상'이 포함된 경우.
  + UI 반응: 관리자 화면 상단에 빨간색 경고 박스를 자동으로 띄워 디자이너가 주의하도록 유도.
* **상태 관리**: 상담 대기중 🟢 → 시술 중 🔵 → 시술 완료 ⚪ 상태 변경 가능.

### 3.3. 글로벌 로컬라이제이션 (i18n)

* TRANSLATIONS 객체를 통해 **일본어(기본), 한국어, 영어** 3개 국어를 완벽 지원.
* 버튼 하나로 즉시 언어 전환 가능 (React State 기반).

## 배포 자동화 시스템 (CI/CD Pipeline)

이 프로젝트의 가장 큰 기술적 성과는 **완전 자동화된 배포 파이프라인 구축**입니다.

### 4.1. 워크플로우 구조 (.github/workflows)

1. **Trigger**: main 브랜치에 코드가 푸시(Push)되면 작동.
2. **Build Environment**: Ubuntu 최신 환경 + Node.js v20.
3. **Process**:
   * npm install: 의존성 패키지 설치.
   * npm run build: React 프로젝트를 정적 파일로 변환 (CI=false 옵션으로 경고 무시).
4. **Deploy**: Firebase Service Account 키를 사용하여 build 폴더 내용을 Firebase Hosting 서버로 전송.

### 4.2. 도입 효과

* 개발자가 매번 수동으로 배포할 필요 없음.
* 코드 수정 후 git push 명령어 하나로 2~3분 내에 실서버 반영 완료.

## 트러블 슈팅 (Troubleshooting Log)

프로젝트 진행 중 발생했던 주요 기술적 이슈와 해결 과정입니다.

| **구분** | **문제 현상 (Issue)** | **원인 분석 (Cause)** | **해결 방법 (Solution)** |
| --- | --- | --- | --- |
| **Build** | GitHub Actions 배포 실패 (X 표시) | React 프로젝트인데 단순 HTML로 인식하여 빌드 스크립트가 누락됨. | YAML 파일에 npm run build 단계를 명시적으로 추가. |
| **Lint** | 빌드 중단 (Treating warnings as errors) | React CI 환경은 사소한 경고도 에러로 처리함. | 빌드 명령어에 CI=false 환경 변수를 추가하여 경고 무시. |
| **Compat** | Firebase CLI incompatible 에러 | 최신 Firebase 도구(v15+)는 Node.js 18 버전을 지원하지 않음. | GitHub Actions의 Node 버전을 **v18 → v20**으로 업그레이드. |
| **Syntax** | YAML 파일 파싱 에러 | 설정 파일 복사 과정에서 줄바꿈 깨짐 및 한글 주석 오류. | 주석을 제거하고 들여쓰기를 표준화한 코드로 교체. |
| **Auth** | Firebase API Key 노출 우려 | 클라이언트 사이드 코드에 키가 포함됨. | Firebase 보안 규칙(Security Rules)을 통해 도메인 제한을 걸어 안전성 확보. |

## 결론 및 기대 효과

### 6.1. 프로젝트 결과

* **완성도**: 기획했던 클라이언트/관리자 기능 100% 구현 완료.
* **안정성**: GitHub Actions를 통한 안정적인 배포 파이프라인 확보.
* **확장성**: 다국어 구조 설계를 통해 향후 글로벌 지점 확장 용이.

### 6.2. 향후 개선 계획 (To-Do)

* **디자이너별 계정 분리**: 현재 단일 비밀번호 방식에서 개별 ID 로그인 방식으로 고도화.
* **시술 사진 기록**: Firestore Storage를 연동하여 시술 전/후 사진 저장 기능 추가.
* **통계 페이지**: 월간 방문자 수 및 주요 시술 유형 통계 그래프 도입.

작성일: 2024년 12월 23일

작성자: Project Owner