**Codebase & Engineering Quality Pack**

## **۱. ریپو و ساختار**

### **ریپو اصلی:**

🔗 `https://github.com/nobatnou/nobatnou-core`

### **ساختار ساده:**

```
frontend/          # React + TypeScript
backend/           # Node.js + Express
shared/            # کد مشترک
infrastructure/    # Docker, k8s
docs/              # مستندات
```

## **۲. استانداردها**

### **کانونشن کد:**

- متغیرها: `camelCase`
- کامپوننت‌ها: `PascalCase`
- فایل‌ها: `kebab-case`

**:ابزارها** ###

- **ESLint:** برای linting

- **Prettier:** برای formatting

- **TypeScript:** برای type checking

### **Commit Convention:**

```

feat: ویژگی جدید

fix: رفع باگ

docs: مستندات

chore: تغییرات ساختاری

```

## **۳. تست‌ها**

### **انواع تست:**

۱. **Unit Tests:** (Jest) - ۷۰٪ coverage

۲. **Integration Tests:** (Supertest) - APIها

۳. **E2E Tests:** (Cypress) - گردش کار اصلی

### **سناریوهای تست E2E:**

- ثبت نوبت → نمایش وضعیت

- تغییر وضعیت نوبت

درج اورژانس -

## **۴. CI/CD Pipeline**

### **GitHub Actions:**

```yaml
name: CI/CD
on: [push, pull_request]

jobs:
test:
steps:
- lint
- type-check
- unit-tests
- integration-tests
- build

deploy:
needs: test
if: branch == main
steps:
- deploy to staging
```

- smoke tests

- deploy to production

```

### **Quality Gates:**

- Test coverage > ۷۰٪

- Zero lint errors

- All tests pass

- Build successful

## **۵. مدیریت نسخه**

### **برنچینگ:**

- `main` - production

- `develop` - staging

- `feature/*` - ویژگی جدید

- `hotfix/*` - رفع فوری

### **Versioning:**

`MAJOR.MINOR.PATCH`

`مثال: `۱.۰.۰

## **۶. Technical Debt مدیریت
```

### **ثبت Debt:**

```typescript
// TODO: Refactor - Priority: Medium - Est: 2 days
function complexFunction() { ... }
```

### **Debt Board:**

| موضوع | اولویت | وضعیت |
|---|---|---|
| جداکردن منطق اعتبارسنجی | P1 | در حال انجام |
| بهبود پیغام خطاها | P2 | برنامه‌ریزی شده |
| بازسازی سرویس نوبت | P1 | بک‌لاگ |

## **۷. مستندات**

### **مستندات ضروری:**

۱. **README.md:** راه‌اندازی سریع

۲. **API Docs:** Swagger

۳. **Component Docs:** کامنت در کد

۴. **ADRs:** تصمیمات معماری

### **کامنت‌گذاری:**

```typescript
/**
ایجاد نوبت جدید *
اطلاعات نوبت @param data *
نوبت ایجادشده @returns *
*/
function createAppointment(data) { ... }
```

## ۸.** انتقال دانش**

### **Knowledge Map:**

| سیستم | متخصص اصلی | متخصص پشتیبان |
|---|---|---|
| Appointment Service | سارا | علی |
| Frontend | محمدرضا | سارا |
| DevOps | علی | محمدرضا |

### **جلوگیری از وابستگی به فرد:**

۱. **Pair Programming:** ویژگی‌های حیاتی

۲. **مستندات:** کافی و به‌روز

۳. **Code Reviews:** اجباری

۴. **Shared Ownership:** مالکیت مشترک

**:برنامه انتقال اضطراری** ###

برای انتقال notice هفته ۲ -

جلسات انتقال دانش -

مستندسازی فشرده -

تست انتقال -

**مانیتورینگ .۹** ##

### **Health Checks:**

```javascript
app.get('/health', (req, res) => {
res.json({
status: 'healthy',
timestamp: new Date(),
services: {
database: 'up',
redis: 'up'
}
});
});
```

**:لاگ‌گیری** ###

error.log خطاها در فایل -

combined.log فعالیت‌ها در -

ساختاریافته و قابل جستجو -

**امنیت .۱۰** ##

**:اسکن خودکار** ###

build در هر `npm audit` -

هفتگی dependencies بررسی -

packages آپدیت منظم -

### **Secrets Management:**

GitHub Secrets استفاده از -

credentials کردن commit عدم -

منظم Rotation -

**کیفیت مهندسی .۱۱** ##

**:اصول اصلی** ###

قابل نگهداشت:** کد تمیز و ساختاریافته** .۱

قابل تست:** تست‌های کامل و خودکار** .۲

modular قابل توسعه:** معماری** .۳

۴. **قابل اعتماد:** پایدار CI/CD

### **معیارهای موفقیت:**

- **Test coverage:** > ۷۰٪
- **Deployment success:** > ۹۵٪
- **Time to fix bugs:** < ۴ ساعت
- **Bus factor:** > ۲ برای سیستم‌های حیاتی

### **بررسی‌های منظم:**

- **بررسی کیفیت کد:** هفتگی
- **اولویت‌بندی technical debt:** ماهانه
- **بازنگری معماری:** فصلی

## **۱۲. جمع‌بندی**

### **چرا این کیفیت مهم است:**

۱. **تیم سریع‌تر کار می‌کند:** سرعت توسعه
۲. **سیستم کمتر crash می‌کند:** پایداری
۳. **هزینه نگهداری کمتر می‌شود:** نگهداشت
۴. **رشد تیم و محصول آسان‌تر است:** مقیاس‌پذیری

### **چیزی که ما نیستیم:**

- کمال‌گرا در نوشتن کد

دارای کد تمیز در حد کتاب -

تیم بزرگی با فرآیندهای پیچیده -

**:چیزی که هستیم** ###

تیم کوچک با فرآیندهای ساده -

تمرکز بر حل مسئله -

تحویل پایدار و قابل اعتماد -

قابلیت نگهداشت بالا -

---

نکته پایانی:** کیفیت مهندسی ما ابزاری برای رسیدن به اهداف کسب‌وکار است، نه هدف به خودی خود. **
.همه تصمیمات بر اساس هزینه-فایده گرفته می‌شوند