

QA/Test Strategy + Test Evidence

مقدمه ۴/۵/۱

به عنوان یک سرویس حساس پزشکی `nobatnou.ir` سیستم نوبت‌دهی نیازمند سطح بالایی از قابلیت اطمینان، در دسترس بودن و امنیت است. تضمین و اجرای منظم انواع (QA Strategy) کیفیت از طریق استراتژی جامع تست تست‌ها انجام می‌شود.

اهداف ۴/۵/۲ QA

- کارکرد صحیح: **همه سناریوهای کاربری طبق نیازمندی‌ها اجرا شوند** -
- قابلیت اطمینان: **سیستم در شرایط واقعی پایدار باشد** -
- امنیت: **داده‌های کاربران و فرآیندهای حساس محافظت شوند** -
- تجربه کاربری: **رابط کاربری روان، بدون باگ و قابل درک باشد** -
- قابلیت نگهداری: **کد تمیز، قابل تست و مستند باشد** -

****انواع تست‌های اجرایی ۴/۵/۳*****

**** (Unit Testing) تست واحد ۱.*****

- هدف: بررسی عملکرد هر تابع یا متد به صورت مستقل
- (برای جاوا اسکریپت) Jest + (برای PHP) PHPUnit: چارچوب
- و ۷۰٪ جاوا اسکریپت PHP پوشش هدف: حداقل ۸۰٪ کد
- موارد تست شده
- `login_user_ajax`، `add_nobat_ajax` توابع -
`register_user_ajax`
- اعتبارسنجی شماره تلفن (۱۱ رقم، فقط عدد)
- (`position`) محاسبه موقعیت نوبت
- هش رمز عبور و احراز هویت
- Xdebug + PHPUnit coverage report: ابزار اندازه‌گیری

**** (Integration Testing) تست یکپارچه‌سازی ۲.*****

- (AJAX دیتابیس، سشن،) هدف: تست تعامل بین کامپوننت‌ها
- موارد تست

ثبت نوبت → ذخیره در دیتابیس → نمایش در لیست نوبت‌ها -
ورود کاربر → ایجاد سشن → دسترسی به داشبورد -
لغو نوبت → به‌روزرسانی وضعیت در دیتابیس -
mock session و (MySQL) روش: ** استفاده از دیتابیس تست ** -

*** **۳. تست end-to-end (E2E)***

هدف: ** شبیه‌سازی رفتار واقعی کاربر از آغاز تا پایان ** -
Cypress یا Playwright: ** چارچوب ** -
** سناریوهای کلیدی ** -
1. کاربر جدید ثبت‌نام می‌کند → وارد می‌شود → نوبت جدید می‌گیرد.
2. کاربر نوبت خود را پیگیری می‌کند (وضعیت و زمان تخمینی).
3. (pending → in-progress → completed) ادمین وضعیت نوبت را تغییر می‌دهد.
4. خود را لغو می‌کند pending کاربر نوبت.
با داده‌های ساختگی staging محیط: ** محیط ** -

*** **۴. تست امنیتی (Security Testing)***

هدف: ** شناسایی آسیب‌پذیری‌های امنیتی ** -

**** موارد بررسی **** -

- استفاده شده اند `wpdb->prepare` ها با query همه SQL: تزریق -
- پالایش `esc_html` و `sanitize_text_field` خروجی ها با XSS: -
می شوند.
- (wp_nonce نیاز به افزودن) بررسی توکن غیر فعال: CSRF -
- ها محدود شده (بررسی سشن) Ajax احراز هویت: دسترسی به -
- هش می شود `wp_hash_password` داده های حساس: رمز عبور با -
- (اسکن خودکار) + بررسی دستی کد OWASP ZAP **** ابزار **** -

**** (Regression Testing) تست رگرسیون ۵ **** #####

- هدف: **** اطمینان از عدم شکست قابلیت های موجود پس از تغییرات **** -
- ** روش **** -

- commit (با GitHub Actions) اجرای خودکار تست های واحد و یکپارچه در هر -

(release) قبل از هر انتشار E2E اجرای دستی سناریو های -

های اصلی (ثبت نام، ورود، نوبت گیری، مدیریت feature پوشش: **** همه **** -
(نوبت)

**** (Load Testing) تست بار ۶ **** #####

- .هدف: ** بررسی عملکرد تحت بار کاربری زیاد ** -
- .سناریو: ** ۱۰۰ کاربر همزمان درخواست نوبت ارسال می‌کنند ** -
- .k6 یا Apache JMeter **: ابزار -
- .** معیارها ** -
- .زمان پاسخگویی زیر ۲ ثانیه -
- .نرخ خطا زیر ۱٪ -
- .حداقل ۵۰ (concurrent users) توان پردازش همزمان -

- و تضمین کیفیت release فرآیند ۴/۵/۴ ** ###
1. جدا ایجاد branch جدید را در feature توسعه: ** توسعه‌دهنده ** می‌کند.
 2. GitHub Actions تست واحد/یکپارچه: ** اجرای خودکار با **.
 3. حداقل یک توسعه‌دهنده دیگر کد را ** (Code Review) مرور کد ** بررسی می‌کند.
 4. سناریوهای کلیدی را اجرا می‌کند QA staging: ** تست دستی در **.
 5. تست‌های موجود suite تست رگرسیون: ** اجرای کامل **.
 6. تست امنیت: ** اسکن خودکار + بررسی دستی آسیب‌پذیری‌ها **.

7. تصویب انتشار: **مدیر فنی تأیید نهایی می‌کند**.
8. آماده rollback با deployment: **production: انتشار به**.
9. Sentry, مانیتورینگ: **رصد خطاها و عملکرد با ابزارهایی مانند** Google Analytics.

*** (Test Evidence) مستندات تست ۴/۵/۵ ***

*** گزارش اجرای تست (نمونه) ***

تست نوع	تاریخ اجرا	تعداد تست‌ها	گذرنده	شکست خورده	پوشش کد
-----	-----	-----	-----	-----	-----
۲	۴۰	۴۲	۱۵/۰۲/۱۴۰۳	(PHP) واحد	۷۸%
۰	۱۸	۱۸	۱۵/۰۲/۱۴۰۳	(JS) واحد	۶۵%
-	یکپارچه‌سازی	۱۶/۰۲/۱۴۰۳	۱۲	۱۱	۱

E2E	۱۷/۰۲/۱۴۰۳	۸	۸	۰	
				-	
۱	۴	۵	۱۸/۰۲/۱۴۰۳	امنیت	-

نمونه تست واحد (PHPUnit)

```php

```
class NobatAjaxTest extends WP_UnitTestCase {
```

```
 public function
```

```
test_add_nobat_with_valid_phone() {
```

```
 $_POST['patient_name'] = 'علی محمدی';
```

```
 $_POST['phone_number'] = '09121234567';
```

```
 ob_start();
```

```
 add_nobat_ajax();
```

```
 $output = ob_get_clean();
```

```
 $this->assertStringContainsString('نوبت ثبت شد',
```

```
 $output);
```

```
}
```

```
public function
test_add_nobat_with_invalid_phone() {
 $_POST['patient_name'] = 'علی محمدی';
 $_POST['phone_number'] = '123'; // اشتباه
```

```
 ob_start();
```

```
 add_nobat_ajax();
```

```
 $output = ob_get_clean();
```

```
$this->assertStringContainsString('شماره تلفن باید ۱۱ رقم
'باشد', $output);
```

```
}
```

```
}
```

```
...
```

```
** (Coverage Report) گزارش پوشش کد **
```



! [تصویر گزارش پوشش کد] (<https://nobatnou.ir/coverage-report-14030215.png>)

\*(لینک نمونه به گزارش واقعی)\*

---

\*\*\* (Bug Fix Policy) سیاست رفع باگ ۴/۵/۶ \*\*\*

۱. طبقه‌بندی باگ‌ها

- **Critical:** سیستم از کار افتاده (مثلاً ثبت نوبت امکان‌پذیر نیست)  
زمان رفع: حداکثر ۲ ساعت
- **High:** عملکرد اصلی مختل شده (مثلاً محاسبه زمان تخمینی اشتباه)  
زمان رفع: ۲۴ ساعت
- **Medium:** مشکل در بخش غیرحساس (مثلاً نمایش نادرست در مرورگر خاص)  
زمان رفع: ۷ روز
- **Low:** مشکل ظاهری جزئی  
زمان رفع: در انتشار بعدی

۲. فرآیند رفع باگ

- ثبت می‌شود GitHub Issues گزارش باگ در -
- برچسب‌گذاری و اختصاص به توسعه‌دهنده -
- (`fix/bug-description`) برای رفع branch ایجاد -
- نوشتن تست برای باگ (حتی المقدور) -
- رفع کد → اجرای تست‌ها → مرور کد -
- در صورت فوریت patch انتشار -

### 3. \*\*پیگیری\*\*

قابل رصد (GitHub Projects مثلاً) پروژه board همه باگ‌ها در هستند.

---

### \*\*\*نتیجه‌گیری ۴/۵/۷\*\*

بر پایه \*\*تست خودکار\*\*، \*\*مرور `nobatnou.ir` در QA استراتژی جدید تنها پس از feature کد\*\* و \*\*مانیتورینگ مستمر\*\* استوار است. هر گذراندن مراحل فوق اجازه انتشار می‌یابد. این فرآیند از شکست سیستم در تولید جلوگیری کرده و هزینه نگهداری را کاهش می‌دهد.