

مستندات مورد نیاز پروژه توسعه و تحویل نرم افزار / سایت

تهیه کننده : خشایار جام سحر



نسخه ۱,۰

۱۴۰۰/۰۳/۰۸

2021, May 29

فهرست عناوین

۲ لایسنس/مجوز
۳ مقدمه
۳ فاز تعریف:
۳ فاز توسعه Software development:
۴ فاز پشتیبانی Software maintenance :
۴ بیزینس کانسپت
۴ سازمان پروژه
۴ چک لیست امکانات، نیازمندیها و فرآیندها
۵ فاز طراحی نرم افزار
۹ محدودیتها
۹ فاز طراحی پایگاه داده
۱۰ متدولوژی انتخابی
۱۱ چرخه توسعه نرم افزار
۱۳ نظارت
۱۴ چرخه تست نرم افزار
۱۶ ابزارها و نحوه پیکربندی آنها
۱۷ راه کارهای جلوگیری از کاهش کارایی Performance سیستم
۱۷ امنیت
۱۷ فاز استقرار
۱۸ کیفیت ارائه سرویس SLA (Service-level agreement)
۱۸ تحویل دادنی ها
۱۹ پرداختها
۲۰ ریفرنسها

لایسنس/مجوز

این سند تحت لایسنس^۱ GPL V3 قرار داشته و از آدرس <https://github.com/jamsahar/Documents> قابل دانلود است.

^۱ - <https://www.gnu.org/licenses/gpl-3.0.html>

مقدمه

سند حاضر شامل کلیات مستندات لازم برای توسعه و نگهداری نرم افزار می باشد.
فعالیت های توسعه نرم افزار شامل موارد زیر می باشد:

- (۱) شناسایی نیاز
- (۲) برنامه ریزی توسعه نرم افزاری
- (۳) طراحی نرم افزار
- (۴) پیاده سازی، آزمایش و مستندسازی
- (۵) گسترش/توسعه و نگهداری نرم افزار

مهندسی نرم افزار به طور کلی به سه فاز^۲ تقسیم بندی می شود:

فاز تعریف:

بر چیهستی تاکید دارد. چه اطلاعاتی باید پردازش شود، کدام کارایی مطلوب است، چه رفتار های سیستمی قابل انتظار است، چه رابطه هایی را می توان برقرار کرد، چه محدودیت هایی وجود دارد و بطور کلی خواسته های کلیدی سیستم شناسایی می شود. سه کار عمده در این فاز شامل :

- (۱) مهندسی اطلاعات یا سیستم
- (۲) طرح ریزی پروژه نرم افزار
- (۳) تحلیل خواسته ها Software requirements

است.

فاز توسعه Software development :

بر چگونگی تاکید دارد. داده ها چه ساختاری داشته باشند، عملیات درون معماری چگونه پیاده سازی می شوند، جزئیات روال ها، ویژگی های واسط ها، زبان برنامه نویسی، نحوه آزمایش ها.
سه کار عمده در این فاز شامل :

- (۱) طراحی نرم افزار Software design
- (۲) تولید سورس کد Software development
- (۳) آزمایش نرم افزار Software testing

^۲ - مهندسی نرم افزار، نحوه نوشتن راه حل نیست، بلکه شناسایی آنچه باید در راه حل باشد است. نگاه کنید به لینکهای زیر:

- https://en.wikipedia.org/wiki/Software_engineering
- https://en.wikipedia.org/wiki/Software_architecture

بر تغییراتی تاکید دارد که با تصحیحات مورد نیاز در جهت تکامل محیط نرم افزار در ارتباط هستند. همچنین نیز تغییراتی که ناشی از تغییر خواسته های مشتریان/کاربران هستند.

بیزینس کانسپت

(۱) ایده اولیه و نحوه Promoting and Monetizing

(۲) بیزینس پلن

(۳) سرویس:

a. محرمانگی Confidentiality

b. دسترس پذیری Availability

c. یکپارچگی Integrity

(۴) سند نیازمندیها:

a. نیاز کاربران/مشتریان

b. نیازمندیهای تکنولوژی

(۵) انواع نقش ها و کاربران (Admin, Operator, Users, ...)

(۶) قرارداد(بندها، تبصره ها، پیوستها، فورس ماژور، جرائم و پناستی ها، ...)

سازمان پروژه

(۱) ترکیب نیروهای سازمان پروژه

(۲) زمان بندی(شکست) پروژه

(۳) نحوه مدیریت ریسک

چک لیست امکانات، نیازمندیها و فرآیندها

(۱) سند امکانات^۳ و نیازمندیها

a. کارکردی(چه کاری انجام شود(فعل)) Functional

b. غیر کارکردی(چگونه آن کار انجام شود(قید)) NonFunctional

i. تکنولوژی فنی

^۳ - A feature is a set of logically related requirements that allows the user to satisfy an objective. A feature tends to be a higher-level objective than a requirement.

- ۱. Responsive web applications
- ۲. Single-page web applications
- ۳. Progressive web applications
- ۴. **Real-time^۴ web applications**
- ۵. **Reactive^۵ web applications**
- ۶. Mashups and web services

- ii. اطلاعاتی
- iii. عملکردی/کارایی
- iv. محیطی (UI/UX)
- v. امنیتی
- vi. کیفی
- vii. مقیاس پذیری
- viii. درستی/یکپارچگی^۶ integrity

(۲) سند فرآیندهای کسب و کار

- a. ورودیها
- b. پروسه ، گردش کار
- c. خروجیها
- (۳) Workflow

فاز طراحی نرم افزار

(۱) سند استانداردهای پایه ای تولید و توسعه نرم افزار :

- a. استاندارد سند توصیف متدولوژی MDD
- b. استاندارد طرح مدیریت پروژه PMP
- c. استاندارد طرح تضمین کیفیت QAP

^۴ - A real-time web application delivers responses to events in a measurable and acceptable time period, depending on the nature of the event, by means of asynchronous, bidirectional communication between the client and the server. WebSocket is the core technology employed for developing real-time web applications. WebSocket provides a full-duplex and bidirectional communication protocol over a single TCP connection.

^۵ - **Asynchronous, Event-based, Non-blocking Architecture**

^۶ - integrity is a nonfunctional aspect of a system to be **safe, complete, consistent, correct, and free of corruption and errors.**

^۷ - مستندات کاملتر در سایت شرکت مهندسی نرم افزار گلستان <http://golsoft.com/pages/fa/fa-namatan.html> .نماتن .

ایضاً http://old.irannsr.org/web_directory/ کتابخانه اسناد سازمان نظام صنفی رایانه ای کشور

- d. استاندارد طرح مدیریت پیکربندی CMP
- e. استاندارد طرح تصدیق و صحه گذاری V&V
- f. استاندارد طرح آزمون نرم افزار
- g. استاندارد طرح انتقال و تحویل نرم افزار
- h. استاندارد طرح ضمانت نرم افزار
- i. استاندارد طرح نظارت

(۲) معماری سیستم:

- a. Blackboard
- b. Centralized, Decentralized(Distributed)
- c. Client-server (2-tier, 3-tier, n-tier, cloud computing exhibit this style)
- d. Component-based
- e. Data-centric
- f. Event-driven (or implicit invocation)
- g. Layered (or multilayered architecture)
- h. Microservices architecture
- i. Monolithic application
- j. Model-view-controller (MVC)
- k. Peer-to-peer (P2P)
- l. Pipes and filters
- m. Plug-ins
- n. Reactive architecture
- o. Representational state transfer (REST)
- p. Rule-based
- q. Service-oriented
- r. Shared nothing architecture
- s. Space-based architecture

(۳) انتخاب پلاتفرم، تکنولوژیها و امکانات مرتبط:

a. دسکتاپ

b. وب

c. موبایل

۸

(۴) ساخت بر پایه :

a. Framework

b. CMS

(۵) Prototyping

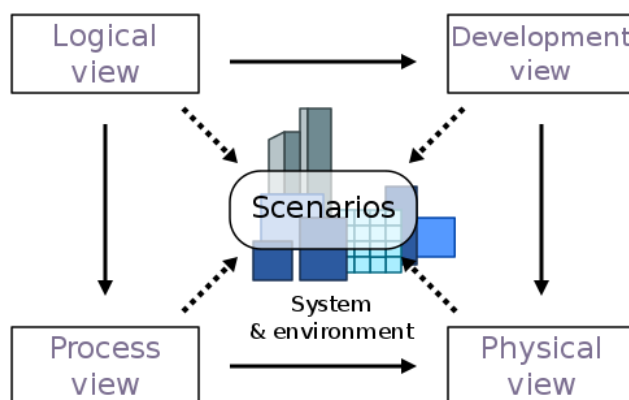
^۸ - Before starting a new project, there is a difficult decision on whether it will be based on a framework or on a CMS. When choosing to use a framework, you need to spend much time creating CMS features for the project. On the other hand, when choosing to use a CMS, it's more difficult to build custom application functionality. It is impossible or at least very hard to customize the core parts of the CMS.

https://symfony.com/doc/current/cmfd/quick_tour/the_big_picture.html

۶) نوع توسعه (مونولیتیک، مایکروسرویس، هاست+پلاگین)

۷) سند معماری ۱+۴

مدل ۱+۴ توسط فیلیپ کروچن برای "توصیف معماری سیستم‌های نرم‌افزاری" معرفی شد. این مدل مبتنی بر استفاده از چند view است. viewها برای توصیف سیستم از دید مصرف کنندگان مختلف و سرمایه‌گذاران نرم‌افزار است مانند کاربران نهایی، برنامه‌نویسان نرم‌افزار و مدیران پروژه. ۴ view در این مدل شامل مدل منطقی، توسعه، فرایند و فیزیکی می‌شود. هر یک از دیدگاه‌های این معماری با یکی یا چند نمودار uml در ارتباط است.



a. مدل منطقی Logical view : این دیدگاه نشان می‌دهد که عملکرد سیستم چگونه توسط طراحی داخلی فراهم می‌شود این دیدگاه ساختار ایستا و پویای داخل سیستم را مشخص می‌کند از جمله نمودارهایی با این دیدگاه در ارتباط اند می‌توان به **Class diagrams & Object diagrams** اشاره کرد.

b. مدل توسعه Development (implementation) View : دید توسعه برای تشریح سیستم از دید یک برنامه‌نویس است و درگیر مدیریت نرم‌افزار است. به این View همچنین Implementation View هم می‌گویند. در UML از نمودار **Component diagrams** برای توصیف اجزای سیستم استفاده می‌کند.

c. مدل فرآیند Process View : دید فرایند، درگیر وجهه پویای سیستم است. این دیدگاه امان‌های سیستم و ارتباط آن‌ها با هم و ترتیب انجام کارها بر اساس زمان را بررسی می‌کند از جمله نمودارهای با این بخش درگیرند می‌توان به **Sequence diagrams, Activity diagrams, State machine diagram** اشاره کرد.

d. مدل فیزیکی Physical (deployment) View : دید فیزیکی سیستم را از دید یک مهندس سیستم نمایش می‌دهد. این دید درگیر توپولوژی کامپوننت‌های نرم‌افزاری در لایه فیزیکی است، به‌علاوه ارتباطات فیزیکی بین این اجزا. در UML از نمودارهای **Deployment diagrams** برای نمایش لایه فیزیکی استفاده می‌شود.

e. سناریوها Use Case : این دیدگاه دید کاربران خارجی نسبت به نرم‌افزار را مورد بررسی قرار می‌دهد نمودار که در uml این دیدگاه را پوشش می‌دهد **Use case diagram** است.

۸) سند طراحی تفصیلی

^۹ - https://en.wikipedia.org/wiki/4+1_architectural_view_model

^۱ - <https://www.omg.org/spec/category/software-engineering/>
<https://www.omg.org/spec/category/modeling/>
<https://www.omg.org/spec/category/business-modeling/>

<https://www.omg.org/spec/SysML/1.6/PDF>

۹) سند معماری (اطلاعات) سازمانی

a. داده ها

b. اطلاعات

c. دانش

Data Validation (۱۰)

(۱۱) هماهنگی/تجمیع/انسجام با سایر سیستمهای سازمان

(۱۲) Business Analysis

(۱۳) System Analysis

(۱۴) Bottom-up و یا Top-Down

(۱۵) انتخاب نوع پایگاه داده SQL/NoSQL

a. Relational

b. Object Relational

c. Multi-Dimensional

d. NoSQL

i. Key-value store

ii. Document store

iii. Graph

iv. Object database

v. Tabular

vi. Tuple store

vii. Triple/quad store (RDF) database

viii. Hosted

ix. Multivalue databases

x. Multimodel database

xi. Wide Column Store

xii. Native multi-model database

(۱۶) انتخاب DB Engine

(۱۷) مرزها و محدودیتها

(۱۸) API

(۱۹) مکانیزم دریافت/ارسال داده از/به سایر سیستم ها OLTP/ETL/OLAP

(۲۰) لایه های نرم افزار:

a. Application layer

b. Execution layer

Semantic layer .c
Propagation layer .d
Consensus layer .e

محدودیتها

- **Technical Limitations**
 - Limited scalability
 - High costs
 - Lack of flexibility
 - Lack of privacy
 - The security model
 - Critical size
 - Hidden centrality
- **Nontechnical Limitations**
 - Lack of legal acceptance
 - Lack of user acceptance

فاز طراحی پایگاه داده

(۱) ابزار طراحی مدل ER

(۲) طراحی 3NF

(۳) لیست جداول

(۴) لیست ستونها/فیلدها

(۵) کاربرد هر یک از جداول و ستونها

(۶) ایندکسها (و انواع آن) و PK

a. روش به روزآوری ایندکسها

b. فواصل زمانی به روزآوری ایندکسها

c. ارائه ایندیکتوری که تریگر فعال سازی به روزآوری ایندکسها را فعال کند.

(۷) جامعیت ارجاعی و FK

(۸) لیست Viewها و نحوه عملکرد آن

(۹) لیست Procedureها و نحوه عملکرد آن

(۱۰) لیست Triggerها و نحوه عملکرد آن

(۱۱) لیست Functionها و نحوه عملکرد آن

(۱۲) لیست کاربران

(۱۳) لیست مجوزها و نحوه تخصیص آنها

Character set & Collation (۱۴)

متدولوژی انتخابی

(۱) سند توصیف استانداردهای متدولوژی

a. مدل آبشاری Waterfall Model

b. مدل تدریجی Incremental Model

c. مدل حلزونی/مارپیچ Spiral Model

d. RAD

e. مدل چابک Agile^۱

i. Extreme Programming

ii. Kanban

iii. Scrum

f. DevOps

Pool of Ideas	Feature Preparation		Feature Selected	User Story Identified	User Story Preparation		User Story Development		Feature Acceptance		Deployment	Delivered
Epic 431	3 - 10		2 - 5	30	15		15		8		5	Epic 294
Epic 478	In Progress	Ready	Epic 602		In Progress	Ready	In Progress	Ready (Done)	In Progress	Ready		Epic 386
Epic 562	Epic 444	Epic 662	Epic 302	Story 302-03 Story 302-01	Story 602-02 Story 602-03	Story 602-06 Story 602-04	Story 602-05 Story 602-01	Epic 401	Epic 609	Epic 694		Epic 419
Epic 439	Epic 589		Epic 335	Story 302-08 Story 302-02	Story 302-07 Story 302-08	Story 302-09	Story 302-04	Epic 468	Epic 577	Epic 276		Epic 388
Epic 329	Epic 651		Epic 512	Story 335-09 Story 335-08	Story 335-10 Story 335-01	Story 335-04 Story 335-02	Story 335-06 Story 335-07	Epic 362		Epic 339		Epic 287
Epic 287	Discarded			Story 335-03 Story 512-04 Story 512-05	Story 512-07 Story 512-06	Story 512-02 Story 512-01				Epic 521		Epic 274
Epic 606	Epic 511	Epic 213								Epic 582		
	Epic 221											

Policy

Business case showing value, cost of delay, size estimate and design outline.

Policy

Selection at Replenishment meeting chaired by Product Director.

Policy

Small, well-understood, testable, agreed with PD & Team

Policy

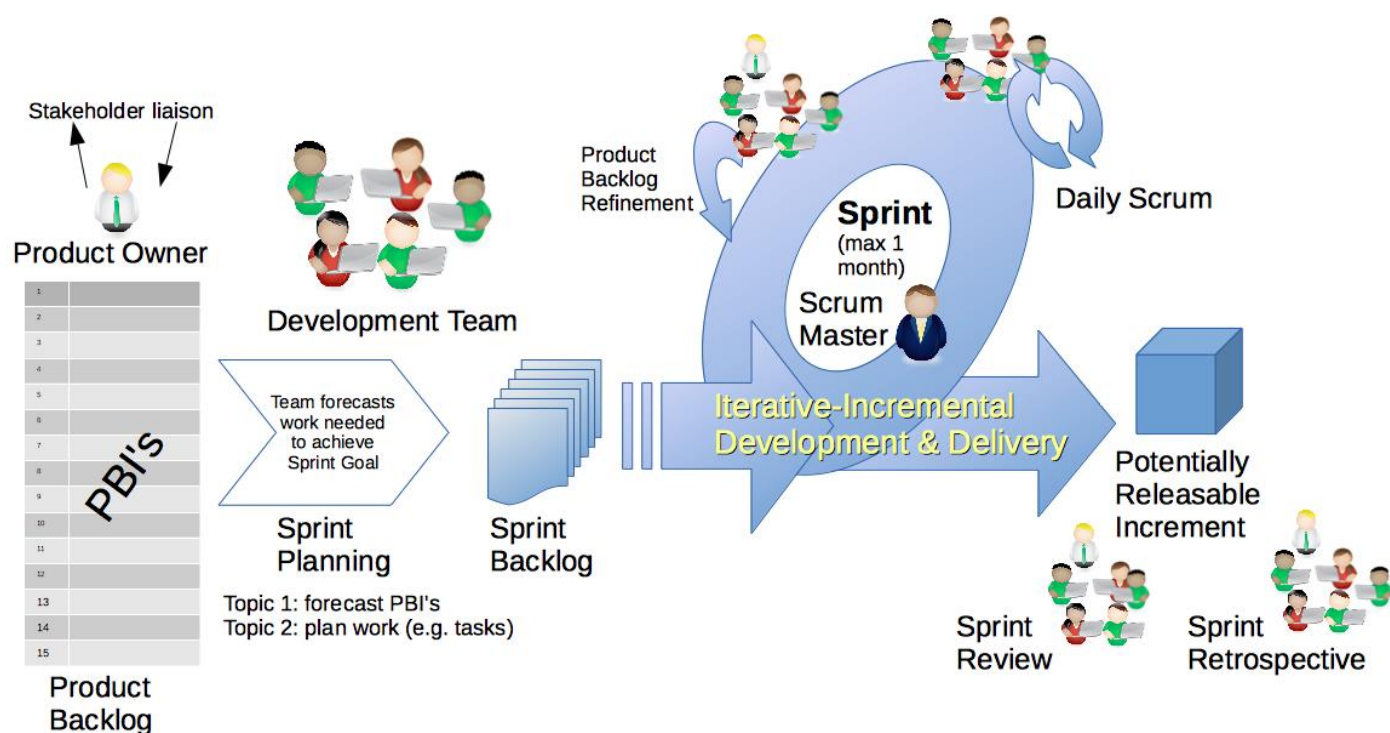
As per "Definition of Done" (see...)

Policy

Risk assessed per Continuous Deployment policy (see...

^۱ - https://en.wikipedia.org/wiki/Software_development_process

^۱ - <https://www.agilealliance.org>



چرخه توسعه نرم افزار

(۱) محل پیاده سازی منطق کسب و کار Business Logic:

a. MVC(Model)

b. Database (Stored Procedure)

(۲) مستندات فریم ورکها، میکروفریم ورک، **دلیل انتخاب** و نحوه مدیریت و به روز آوری آن

a. Backend

i. Synchronous I/O (PHP, Python, Java, .Net)

ii. Asynchronous I/O (NodeJS, reactPHP, ...)

b. Frontend

i. واکنشگرا Responsive

ii. پیش رونده Progressive

iii. Server side & Client side Template engine

iv. CSS Framework (Bootstrap, ...)

v. JavaScript Framework(React, Angular, VUE, ...)

c. Fullstack

(۳) مستندات اپلیکیشن، نحوه مدیریت و به روز آوری آن

(۴) مستندات تطبیق فرآیندها و نیازمندیها با ماجولهای طراحی شده

(۵) ساختار فایلها و دایرکتوری های ثابت و نقش هریک

- a *.php, *.html, *.css, *.js
- b *.jpg, *.png, *.webp
- c ...

(۶) ساختار فایلها و دایرکتوری هایی که رشد می یابند و نقش هریک

- a *.uploads
- b *.emails
- c *.logs
- d ...

(۷) ساختار فایلها و دایرکتوری های موقت و نقش هریک

(۸) مستندات Best Practice و استانداردهای مورد استفاده

- a Unicode/UTF-8
- b Autoload Standards\ (PSR-0, **PSR-4**)
- c Coding Standards (PSR-1, PSR-2, **PSR-12**)
- d HTML Forms Implementation
 - i Form implement in **php class**
 - ii Buttons implement in **Templates**
 - iii Multiple Buttons implement in **Controllers**
- e ...

(۹) تطبیق استانداردهای سورس کد با آنچه که پیاده سازی شده است

(۱۰) نحوه ارتباط با سایر نرم افزارها و مدیریت آن

(۱۱) Vendor Packages/Plugins/Library

(۱۲) Version Control

(۱۳) Unit tests code

(۱۴) Patterns

(۱۵) Micro Services

(۱۶) اجرای نسخه Prototype

(۱۷) UX/UI

a. یو آی یا UI مخفف عبارت User Interface Design به معنای طراحی رابط کاربری است. در واقع UI طراحی بخشی از وبسایت یا اپلیکیشن است که کاربر آن را مشاهده می کند و بیشتر به جنبه ی گرافیکی موضوع می پردازد. اینکه ظاهر هر بخش به چه صورت باشد تا جلوه ی مناسبی به وبسایت دهد را طراح بخش UI مشخص می کند. طراحی رابط کاربری به کمک گرافیکست های سایت و برنامه نویس Front-end پیاده سازی می شود. طراحی رابط کاربری مهم است زیرا کاربر مستقیماً با آن ارتباط دارد و معمولاً افراد به وبسایت هایی با UI ضعیف، کمتر اعتماد می کنند. مثال هایی از UI:

- i. ظاهر منو چگونه است؟
 - ii. رنگ دکمه‌ی جستجو با رنگ اصلی سایت هماهنگ باشد.
 - iii. باکس مربوط به بنرها تبلیغاتی چگونه طراحی شوند؟
- b. یو ایکس یا UX مخفف عبارت User Experience Design به معنای طراحی تجربه‌ی کاربری است. به احساسات و راحتی کاربر حین کار کردن با اجزای مختلف سایت گفته می‌شود و جوانب تعامل کاربر را با هر بخش در نظر می‌گیرد. اینکه وبسایت ما طوری طراحی شده باشد که هر بخش به راحتی در دسترس کاربران قرار بگیرد و برای کار کردن با هر قسمت آسودگی و راحتی کاربر در نظر گرفته شود اینها مسائلی از طراحی UX هستند. البته ناگفته نماند که این فقط جنبه‌ای از طراحی UX بود. جنبه‌ی دوم آن به هدایت کاربر مرتبط می‌شود. اینکه چگونه ما کاربران یک صفحه را به صفحه‌ی دلخواه خود هدایت کنیم. اینکه چگونه کاربر اهداف ما را در وبسایت دنبال کند. در واقع بعد دوم تجربه‌ی کاربری به مدیریت و هدایت کاربر از لحظه‌ی ورود به سایت تا لحظه‌ی خرید محصول (یا هر هدف دیگر) اشاره می‌کند. مثال‌هایی از UX:
- i. در منو چه گزینه‌هایی قرار داده شوند.
 - ii. باکس جستجوی کجای صفحه قرار بگیرد.
 - iii. بنرهای تبلیغاتی کجای صفحه سایت باشند.

SEF^۱ /SEO (۱۸)

Server Side Render a

Client Side Render b

MVC (۱۹)

Template Engine (۲۰)

License (۲۱)

محیط‌های استاندارد (۲۲)

Development a

Pre Production/Staging/Testing b

Production/Live c

نظارت

Trace ability (۱) قابلیت ردیابی

Correctness (۲) درستی

Validity (۳) اعتبار

Sufficiency (۴) کفایت

Consistency (۵) سازگاری

Uniformity (۶) یکنواختی

Feasibility (۷) امکان پذیری

^۱ - Search Engine Friendly & Search Engine Optimization

(۸) نگهداشت پذیری Maintainability

(۹) مقیاس پذیری Scalability

چرخه تست نرم افزار

نکته: تست فقط وجود خطا را نشان می دهد و نه عدم وجود آن را. پیدا نشدن خطا در تست به معنای بدون خطا بودن برنامه نیست.

(۱) تست سند گزارشات بیزینسی

(۲) تست آماره های اپلیکیشن/سایت

(۳) سطوح تست :

a. **سند تست واحد (Unit testing):** تست واحد یا **micro level** پایین ترین سطح تست است. هر کد تست واحد، یک قطعه کد یا یک تابع (متد) خاص را تست می کند. این تست نیاز به دانش در مورد طراحی و نحوه کارکرد داخلی تابع یا قطعه کد دارد و توسط برنامه نویس (و نه تست کننده) انجام می شود. این تست خود به ۲ بخش تقسیم می شود:

(۱) **test driven development (TDD)**

(۲) **behavior driven development (BDD)**

b. **سند تست یکپارچگی افزایشی:** تست یکپارچه سازی افزایشی با افزوده شدن قابلیت جدید به نرم افزار، مجدداً نرم افزار تست می شود. هدف این تست، بررسی درستی نرم افزار پس از افزوده شدن امکان جدید است.

c. **سند تست یکپارچگی (Integration Testing):** این تست بر روی بررسی ارتباط داده ها (Data Communication) در میان ماژول های مختلف متمرکز است. تست یکپارچه سازی برای تایید ماژول های نرم افزاری برای کار در یک پیکر واحد ضروریست .

^۱ - https://en.wikipedia.org/wiki/Software_testing

^۱ - Test Methods of a class

^۱ - Unit test included but not limited to below list:

- 1) **Overflows and underflows:** Make sure that your code doesn't allow numbers to become larger than the largest valid value or smaller than the smallest valid value. Either situation will cause an error.
- 2) **Valid return values:** Ensure that each function returns values that are valid for the caller. In some cases, the return value is calculated. Your tests should ensure that any calculated return values are always valid.
- 3) **Boundary conditions:** Always test that your code handles data that meets or exceeds expected limits.
- 4) **Iteration limits:** Test each looping structure to ensure that it doesn't iterate more times than you intend and burn up all your gas.
- 5) **Input and output data formats:** Test your code to make sure that it handles data provided or returned in unexpected formats.
- 6) **Input and output data validation:** Ensure that your code either sanitizes or rejects invalid characters or sequences of characters.

^۲ - Checks if Class A works with Class B and Ensure different parts of system work together.

^۲ - Integration test included but not limited to below list:

- 1) Wrong function name
- 2) Wrong number or format of input parameters
- 3) Out of range or bad input data
- 4) Input data containing boundary values
- 5) Wrong expected output parameters
- 6) Attempt to call a function that isn't visible
- 7) Smart contract function properly completed with return codes
- 8) Set a timeout for a function call that is too short

d. سند تست سیستم^۲ (System Testing) : به منظور بررسی عملکرد نرم افزار بر روی پلتفرم های مختلف انجام می شود.

- 1) Page testing
- 2) Cross-page testing
- 3) Logic testing
- 4) Linting (isn't about finding errors, but potential errors)
- 5) Link checking (making sure there are no broken links on your site)

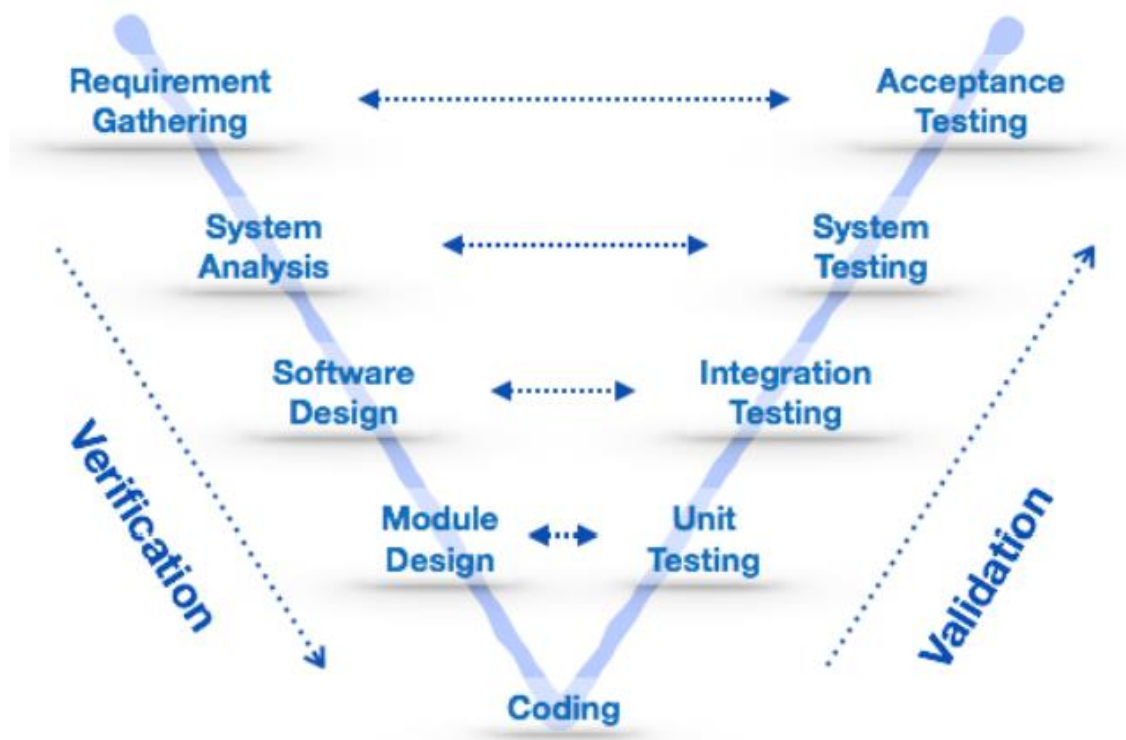
a. سند تست پذیرش^۳ (Acceptance Testing) : هدف از این آزمون اطمینان از این نکته است که سیستم در شرایط عملیاتی معمولی و با اطلاعات واقعی قادر به برآورده کردن نیازهای کاربران می باشد.

(۶) ابزارهای تست

(۷) آزمون استقلال سیستم از پلتفرم (Linux, Windows, Mac,...)

(۸) آزمون استقلال سیستم از مرورگر (Chrome, Firefox, Edge, IE, Safari,...)

(۹) آزمون اندازه های مختلف صفحه نمایش.



(۱۰) آزمون کارکردی (مشخصات پیش بینی شده در اهداف ورودی و خروجی کارکردی)

9) Reverse a transaction

^۲ - System testing ensures the whole system works as user expected before sending it to acceptance testing.

^۳ - When tests above are for developers at development stage. Acceptance tests are actually done by the users of the software. Users do not care about the internal details of the software. They only care how the software works.

- ۱۱) آزمون عملکرد (کارکردی با هزینه (مصرف زمان و منابع) قابل قبول. آستانه پذیرش کارایی سیستم در هر کارکرد باید با توافق کاربر نهایی تعیین گردد.)
- ۱۲) آزمون همسازی داده ها Data integrity (در صورت کار دائم سیستم، هیچ یک از Constraint ها نقض نشود.)
- ۱۳) آزمون چرخه کسب و کار Business cycle
- ۱۴) آزمون واسط کاربر GUI
 - a. Smoke testing
 - b. ...
- ۱۵) آزمون امنیت
- ۱۶) تست نفوذ Penetration test
- ۱۷) آزمون تحمل خرابی Fault-tolerance (Planned/Unplanned crash recovery)
- ۱۸) آزمون پیکربندی
- ۱۹) آزمون بازگشتی Regression (پس از هر بار ارائه یک نسخه جدید از سیستم)
 - a. تصحیحات انجام شده، منجر به رفع اشکالات قبلی یا بهبود کارایی سیستم شده است.
 - b. تصحیحات انجام شده، منجر به بروز اشکالات جدید در دامنه پوشش آزمونهای قبلی نشده است.
- ۲۰) آزمون تحمل بار Load (پایداری سیستم در ماکزیمم پیک کار به مدت ۷۲ ساعت)
- ۲۱) آزمون تنش Stress
- ۲۲) تست API طراحی شده برای نرم افزار.

۲

۴

ابزارها و نحوه پیکربندی آنها

- ۱) کانتینرها (Docker, ...)
- ۲) پایگاه داده MySQL, MariaDb, ...
- ۳) مفسر PHP, NodeJS, ...
- ۴) وب سرور Apache, Nginx, ...
- ۵) پلاتفرم
- ۶) فریم ورکها
- ۷) Template engines

(۸) ابزار مدیریت وابستگی ها (Composer ,NPM ,...)

a. Package/Plugins های مورد نیاز

b. 3rd Party Library

(۹) ابزارها و سیاستهای بکاپ گیری

راه کارهای جلوگیری از کاهش کارایی Performance سیستم

(۱) Load Balancer

(۲) Clustering

(۳) Cache

(۴) JIT

(۵) بهینه سازی کد

(۶) بهینه سازی دیتا بیس

(۷) بهینه سازی وب سرور

(۸) بهینه سازی مفسر/کامپایلر

(۹) بهینه سازی فریم ورک

(۱۰) بهینه سازی معماری و طراحی کل سیستم

امنیت

(۱) Authentication/Authorization

(۲) Encryption/Hash Algorithms

(۳) Penetration Test

(۴) SQL Injection

(۵) Hardening

a. Keep update

b. Hard password to guess

c. Backup

d.

(۶) Over Flow

(۷) PHP Sodium Extension + Hasher/Encryption Algorithms

فاز استقرار

(۱) سند انتقال و واگذاری نرم افزار

- (۲) سند نگهداری و به روز آوری (Update/Upgrade/Patch/BugFix/Refactoring) سیستم
- (۳) تفاوت‌های پیکربندی نسخه Development با نسخه Production
- (۴) تفاوت‌های پیکربندی Host, Localhost
- (۵) ابزارهای Upload/Download/Backup
- (۶) پیکربندی DNS
- (۷) پیکربندی eMail
- (۸) مانیتورینگ، NOC
- (۹) Log circulation و Logs

کیفیت ارائه سرویس (SLA (Service-level agreement

- (۱) Planned down time
- (۲) Unplanned down time
- (۳) Crash/Recovery Policy
- (۴) Restore & Resolution time
- (۵) Retention time
- (۶) Ticketing
- (۷) پروتکل ارتباط با مشتریان، نحوه ارجاع مشکلات به واحد فنی (L2,L3) و پروسه رفع آن

۲

۵

تحویل دادنی‌ها

- (۱) نسخه نهائی اجرایی نرم افزار (به صورت Optimum^۲)
- a. **تحویل آزمایشی^۷ (به صورت هفتگی)** (محیط آزمایشی، تست کارکردی و عملکردی با داده های تستی و واقعی)
- b. **تحویل موقت (فاز انتهایی پروژه)** (محیط عملیاتی، عملیاتی شدن سیستم و تست پایداری سیستم و بررسی باگهای احتمالی (سه الی شش ماه))
- c. **تحویل دائم (خاتمه پروژه)** (گذراندن کلیه تستهای آزمایشی و عملیاتی و پایدار شدن سیستم)
- (۲) پایگاه اطلاعاتی سیستم (اسکرپت ایجاد، Dump)
- (۳) راهنمای نصب و استقرار

^۲ - https://en.wikipedia.org/wiki/Outline_of_software_engineering See Deliverables

^۲ - نسخه ای کامل است که نتوان چیزی از آن کم کرد.

^۲ - Demo به صورت هفتگی

- (۴) راهنمای کاربران
- (۵) آموزش کاربران
- (۶) راهنمای عملیاتی سیستم
- (۷) طرح آزمون پذیرش
- (۸) کلیه اسناد توسعه نرم افزار

۲

۸

پرداختها

پروژه تحویل گیری پروژه های بزرگ معمولاً طولانی و بتدریج می باشد، که پیشنهاد میشود شامل ۱۵ زیر بخش لیست زیر باشد (تا از ریسکها و مشکلات سایر پروژه های مشابه دوری جست):

- (۱) پیش پرداخت اولیه
- (۲) پرداخت تحویل مستندات و سورس طراحی نیازمندیها Requirements.
- (۳) پرداخت تحویل مستندات و سورس سند طراحی تفصیلی و معماری.
- (۴) پرداخت تحویل مستندات و سورس کد طراحی دیداری Mockups.
- (۵) پرداخت تحویل مستندات و سورس کد نمونه اولیه.
- (۶) پرداخت تحویل مستندات و سورس کد طراحی و تست امکانات کارکردی اولیه/اجباری .
- (۷) پرداخت تحویل مستندات و سورس کد طراحی و تست امکانات کارکردی ثانویه و گزارشات.
- (۸) پرداخت تحویل مستندات و سورس کد طراحی API.
- (۹) پرداخت تحویل مستندات و سورس کد طراحی و تست امکانات عملکردی (از نظر Performance, ...).
- (۱۰) پرداخت تحویل مستندات و سورس کد طراحی تست کلی نرم افزار و تست نفوذ.

۱. Unit test

۲. Integration test

۳. System test

۴. Acceptance test

(۱۱) پرداخت تحویل مستندات و سورس طراحی نحوه استقرار، نگهداری و پشتیبانی سیستم روزانه/هفتگی/ماهانه/سالانه و سطح

کیفیت سرویس SLA.

(۱۲) پرداخت مستندات آموزشی کاربران.

(۱۳) پرداخت تحویل مستندات و سورس طراحی نحوه توسعه و نقشهای مورد نیاز.

(۱۴) پرداخت تحویل موقت.

(۱۵) پرداخت پس از دوره گارانتی و تحویل دائمی.

^۲ - اگر نرم افزار جهت فروش/صادرات باشد نیازی نیست، ولی چنانچه "خریدار/مشتري/کاربر نهائی/کارفرما/بهره بردار" خود بایستی ادامه توسعه و نگهداری را عهده دار باشد، ارائه کلیه مستندات توسعه نرم افزار توسط "پیمانکار/مجری"، ضروری خواهد بود.

رئفرنسها

https://en.wikipedia.org/wiki/List_of_software_development_philosophies
https://en.wikipedia.org/wiki/Agile_software_development
[https://en.wikipedia.org/wiki/Scrum_\(software_development\)](https://en.wikipedia.org/wiki/Scrum_(software_development))
https://en.wikipedia.org/wiki/Extreme_programming
<https://en.wikipedia.org/wiki/DevOps>
https://en.wikipedia.org/wiki/Programming_paradigm
<https://en.wikipedia.org/wiki/Anti-pattern>