前端系分文档

作者：Rabbitzzc
原文链接：https://ld246.com/article/1611563351207
来源网站：链滴

最近在看一些行业内文章的时候，了解到一个概念，也就是标题 - 前端系分文档。虽然在美团能到很多基本的方法论和提升自身基本功与软技能，但这是我工作一年多第一次听到这个词语，很快就这个概念也吸引了。
<blockquote>
所以也尝试百度了相关概念以及结合自身角度去理解。
</blockquote>
<h4 id="WHY">WHY</h4>
相信很多朋友跟我类似，在工作刚开始的时候，需求评审的过程很有可能都是走个形式，投入度不高，一般是等设计稿出来，产品文档终稿出来，接口文档出来以后才回去关注整个需求，然后做一接口评审，没问题以后就开始开发上线了。这个流程看起来是没啥问题的，但是随着需求的复杂性以迭代频次越来越高，作为项目的执行者，开发扮演的角色也是越来越重要的，业务理解、排期、接口理性、技术深度、更高产出、Roadmap 等都需要前端去把控，挑战也是越来越大的，这时候还是跟前一样悠闲等待终稿再去开发是行不通的（受制于业务的复杂度，一般前端主要是实现，逻辑上还是后端，理论上前端不需要完全关注业务）。
这个时候，就是需要系分文档的。
<h4 id="WHAT">WHAT</h4>
前端系分文档类似于一份项目管理文档，介绍项目的所有信息。美团内部是可以 wiki 平台的，似于语雀之类，我可以针对于某个项目创建一个个人需求的目录文档。
系分文档的好处是：

让前端更了解需求，提前发现需求不合理的地方
做好依赖链接的整理，避免四处翻找资料
更细的拆分模块，合理评估工作时间。
更了解业务方的目标和相关方的计划
画出交互流程图，加深对交互流程的掌握
找出技术难点，提前暴露风险
做好上线的准备，避免出现意外
方便事后复盘

<blockquote>
核心观点：技术服务于业务，当业务理解不断加深，业务全貌越来越了解的时候，就能发现当前务的痛点，然后用技术手段去解决，或者用新的业务理解去重构之前业务。这都是很有意义的。
</blockquote>
<h4 id="TEMPLATE">TEMPLATE</h4>
系分文档的模板我这里主要是看了三个文档，如下面参考文档，主要是如下分类：

需求背景
包括项目提供的业务价值、解决的痛点、业务目标、目标人群等（尽量自我理解并输出）
需求目标
定性、定量
项目文档
需求地址（Aone）、视觉稿地址、交互稿地址、后端系分文档地址、迭代地址、发布计划文档选测分文档、开发环境地址、测试环境地址、Git 地址
项目人员
项目各角色接口人，包括 BD、PD、前端、服务端、测试、交互、视觉、其他合作方等
排期计划
关键节点：需求评审、交互/视觉评审、前期分析、前端系分 & 评审、后端系分 & 审、测试系分 & 评审、研发、联调、测试、预发、发布。
任务拆分
任务拆分必填，模块拆分选填 -- 将整体需求拆分成细粒度的任务，并说明优先级、依赖关系、作量
按照内部的工作，可以建立一个表格，然后拆分各种功能模块，并设定预估工作时间，优先级，责人等等


```
<table>
<thead>
<tr>
<th>功能模块</th>
<th>功能描述</th>
<th>系分</th>
<th>开发</th>
<th>自测</th>
<th>联调</th>
<th>提测</th>
<th>埋点验收</th>
<th>视觉验收</th>
<th>合计</th>
</tr>
</thead>
<tbody>
<tr>
<td>xxxx 模块</td>
<td>``</td>
<td>0.5H</td>
<td>1H</td>
<td>1H</td>
<td>1H</td>
<td>1H</td>
<td>``</td>
<td>``</td>
<td>1H</td>
</tr>
<tr>
<td>xxxx 组件</td>
<td>``</td>
<td>``</td>
<td>``</td>
<td>``</td>
<td>``</td>
<td>``</td>
<td>``</td>
<td>``</td>
<td>``</td>
</tr>
<tr>
<td>xxxx 功能</td>
<td>``</td>
<td>``</td>
<td>``</td>
<td>``</td>
<td>``</td>
<td>``</td>
<td>``</td>
<td>``</td>
<td>``</td>
</tr>
<tr>
<td>总计</td>
```

<td>35H ≈ 7D</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
<ol start="7">
技术目标【可选】
技术目标：对于本次需求，在业务目标之外要实现的技术目标【可选，尽量满足】
性能指标（首屏性能低于 800 ms）
框架层面（B 端沉淀并统一到小程序，某某模块组件沉淀到资产库并发布 beta 版本）
问题分析（或者说是问题记录与分析）
可选软件工程中的需求分析部分，这里主要是从需求开始到详细设计之前的分析工作，对详细设部分起到辅助作用（比如说交互、视觉、技术方案、风险等维度，比如交互需要确认的问题，素材何提供，方案选型待确认，接口问题如何记录）

<blockquote>
交互待确认问题

:white_check_mark:xxxx，要怎么处理？
结论：xxxxx
:black_square_button:XXX，素材提供（标题、描述、图片、动画）

</blockquote>
<ol start="9">

详细设计

技术设计：流程图、时序图、技术难点、特殊动效设计、接口、测试重点（一般时序图很少，业前端项目）
每个功能点需要阐述的内容：具体实现、异常处理、改动影响业务范围、需要注意的潜在风险点需要清楚说明需求所涉及需求的具体设计实现，建议细化到功能点）
描述方式：如果需要可放置相关部分的设计图、交互流程图、数据结构、API 接口定义，必要时贴出你的代码。此处包括但不限于以上表述方式，重点是将问题陈述清楚。
关于一些交互以及流程，产品也会给出，如果自己能够重新整理一份也还是不错的，尽量在大需或者必要的需求中去整理，小需求的话，容易将个人时间填的满满的

风险

需求相关的一切可能产生的风险点，例如：兼容、稳定性、数据安全、资损等，均需在此处进行估说明

旧版本影响评估

上下游业务链路兼容性评估
可能造成的舆情评估
法务、合规、隐私办和数据安全
稳定性评估
资损评估
性能
安全
发布风险评估
依赖三方的流程风险
初次接触的技术风险

<ol start="11">

埋点监控

性能上报、事件埋点（埋点的详细信息等等）、错误监控（新增啥监控，监控看板地址等）

自测/测试说明

自测用例（我一般也会使用一个表格列出所有模块下所有功能点的测试用例，通过标绿，不通过标红）
产品测试信息（提交的一些 bug 等修复情况）

发布 checklist

发布过程需要做的一些事情，一般很少需要，但是可以自己尝试弄一个模板，每次走一遍流程即。
是否存在灰度
测试环境是否已经测试
所有 bug 是否已经全不修复
CR 是否通过
预上线环境是否通过
是否需要审批发布
故障应急方案与原则等等

其他

变更记录啊，验收注意点等等（一般我在工作一年多基本上没有遇到过）

总结复盘、感想

我在了解系分文档概念之前，也是写过相关文档的，只不过没有系分文档这么全，主要是需求背、目标的理解，基本信息，以及 roadmap 吧。顺便给自己留个 TODO，后续在一些需求上，去尝试用该模板，然后记录一下整体感受。
<h4 id="Reference">Reference</h4>

https://g.yuque.com/kenan-5xhg/io2v7a/agp0vl
http://hpoenixf.com/posts/20802/
https://www.yuque.com/anzhi/as3di/yk06p9
