Product Development Artifacts

Product Description

This a narrative that describes the product, why it should be built, and who will use it. Start it with the product name, and the one sentence description of the product. From there, provide a narrative answering the questions described above.

Feature List

Refine the product description into specific features. Describe each feature and how it will be used by or support the product users. This section should describe everything you want your product to do in the Minimum Viable Product (MVP).

Constraints List

List all known functional, and non-functional constraints. Anything that has to be in order for the product to be accepted by the users. This can include anything from user interfaces to platforms. This involves thinking about the product from both a technical perspective, and a user perspective. Having a well-thought-out user avatar is very useful at this stage.

Module List

Categorize the features of the product into modules. The categories are a logical grouping of functionality. It doesn't have to represent sub-systems or even a technical collection of code, but it can. The module list should take constraints into consideration, including platform constraints. The module list should include relationships with other modules. The nature of the relationships should be described. That includes whether the module consumes data from, provides data to, or both, to the related module.

Function List

Decompose the module list into specific functionality. This should serve as the bridge from the high-level design to the detailed design. The function list should be mappable directly to code. It is often defined by the navigation architecture for products with a GUI interface, or command structure for products with a CLI interface, or the API interface for system level functionality. This list will likely evolve as the design works from high to low level.

Data Model

Describe the data repository in detail including all entities, attributes, and relationships. This is applicable for both structured and unstructured data. It should include all data that needs to persist in the system, including configuration files and cookies if used.

Interface Design

Describe each interface in detail. For human-computer interfaces, this means a low-fidelity prototype, or wireframe. For system-to-system interfaces, this means a detailed API design.

Task List

Decompose all the work needed to build the system into time-boxed tasks. The time-box should be whatever time you've selected for sprints. The suggested length for this class is one week. Each task must represent a chunk of working, testable, code. Each task must include an acceptance test. The acceptance test describes the circumstances that must exist for the task to be marked as completed and accepted.

Test repository

Build a repository that includes a description of all tests performed at all levels of the system. This includes everything from acceptance tests through release tests. The full testing hierarchy. This repository will provide a critical roll in both development and maintenance.