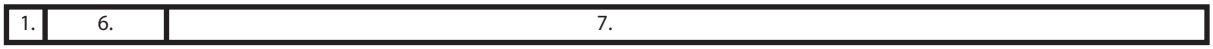
REQUIREMENT PILOT DEVELOPMENT

Many projects begin with workshops where management, architects and UX analysts to define price, scope and a few interactions, followed by a more or less agile implementation phase. The project is usually driven from the backend, resulting in a frontend publishing backend data, or driven from UX, resulting in organically complex backend requirements. During implementation, cost and time are increased by numerous change requests, and the most aggressive pivots are postponed to later versions, which are slowed down by legacy requirements.
To avoid this waterfall requirement process, a light-weight requirement pilot application can be used for defining requirements in an agile and including manner. It becomes a glue between contributors, a driver of innovation, a technical and visual blueprint, a tool for expectation management and a complexity link between vision- and achitecture abstractions, keeping them grounded.
It includes all user interactions with a system, and is used for identifying conflicts and synergies between various usage scenarios.
It is used for embracing pivots and scope creep before deciding a scope roadmap, thus minimizing costly change requests and misunderstandings during implementation, as well as maximizing system rubustness. Early, frequent demos for all contributors and stakeholders for opinions and feasibility validation break silos and promote collaboration.
Start with obvious features, and add more complex interactions when the need and inspiration appears. Select scope, architecture and determine implementation price only after the pilot is satisfactory. Benefits include better scope definition, scope roadmap, completeness, robustness, pivoting prior to massive resource expenditure, mobile design and early demonstrations.
A preliminary execution of phases 1.-3. prior to involving a client may minimize some IP rights and reuse issues.
Relative resources required per phase, without requirement pilots :



Relative resources required per phase, using requirement pilots:

1. Identify

Determine price for phases1-5. May be fixed price or time-and-materials cost. Identify as many user- and API interactions with the system as possible (customers, support, administration, analysis, security etc.)

2. Sketches

Create paper wireframes of all user interaction screens. The goal is to identify information transfer requirements, so avoid pixel-perfect UI work.
Summarize data transfers with technical systems, and how they restrict a system data model.

3. Static interfaces

Create static text or navigatable images. A pilot client platform is chosen for maximum development- and pivot speed, complexity control, distribution and mobile interaction. iOS has integrated storyboarding, refactoring- and animation support and capabilities like Apple Watch, AR, VR, AI and video.

4. Client sim

Define and simulate a data model and backend API on the client, to find conflicts and possibilities.

5. Backend sim

Implement a pilot backend API, to locate multi-user conflicts and synergies. Use limited data transfer complexity and no security.

Continuous, cheap pivots and scope creep for robustness

6. Determine price for phases 6-7. May be fixed price or time-and-materials cost.
Determine scope limitations, stages, timeline, milestones, project structure, production platforms, architecture, integration, devops, security, outsourcing, operations, maintenance, support, UI, AI

7. Implementation

Use production platforms. All information transfer requirements are defined by the pilot backend API.
Production backend may use the pilot client. Production client may use the pilot backend.