**Deliverables**

**System Requirement Specification Document**  
This documentation deliverable provides a detailed description of the software system, as well as a detailed list of features (both critical and non-critical), planned interfaces, and other requirements necessary for project success. It is deliverable one to two weeks after project startup.

**System Design Document**  
This document provides a more in-depth look at the systems architectures of the TapWater project. It includes the rationale for the architectural choice of each system of the software, including platforms and subsystems. This document is deliverable roughly three to four weeks into the project.

**Internal & External Software Documentations**These are two separate technical documentation deliverables. The Internal Software Documentation deliverable consists of a detail list of all classes (objects) and methods (functions) of the software system. The External Software Documentation deliverable provides a less technical, more functional look at the software system’s function (method) capabilities and the overall structure of the code. These deliverables are available roughly six to eight weeks after the project start date.

**Testing Documentation**  
The Testing Documentation deliverable provides proof of iterative use-case testing, following the IEEE 829 format. It includes testing of all features and cases covered in the System Requirement Specification document. This deliverable will be available after roughly ten weeks.

**Risks**

In any developmental undertaking there is a certain amount of risk. Risks associated with this software development project can be broken down into several categories including time, resources, and liability.

The TapWater software development project has very low overall risk associated with it. The core application features themselves are simple, and provide a foundation upon which further features could be added iteratively without much time constraint. Even the time spent developing the base application is, in this instance, well-spent because the skills, experience, and knowledge gained will undoubtedly be useful in other projects for the developers.

Further, because the TapWater project is an independent undertaking, there is very little other resource or financial risk. In this case, the primary resource risk is time, which as shown above can be mitigated. The risk for delay in the original schedule are also mitigated in that, being an independent project, the communication-loop is relatively swift.

Additional risks for the project (though very small) include the risk of requirement inflation and some liability issues. The first of these – requirement inflation – can be combated by adhering to strict guidelines on what is and is not a “necessary” or “core” feature for the initial development cycle. The second – liability – comes from the fact that TapWater is, essentially, a health app. Therefore, the project team must be very careful and realistic about any promises or advertisements made either within the app or during presentations. A legal disclaimer is also a good idea.