Задача. Небольшой прототипчик для системы электронного документооборота.

Предварительные замечания.

Подразумевается, что разработка будет вестись в несколько этапов. На каждом этапе требования дополняются или меняются, что-то может быть выброшено.

Основное требование каждого этапа - чтобы работало (выполнялись затребованные функции), не нужно следить за качеством кодом - как хочется, так и делаешь. Уже по ходу работы будут появлятся замечания, и по ним код нужно будет менять.

Следует использовать общедоступную систему контроля версия. Предлагаю остановиться на bitbucket.org и git репозитарии. Дизайн приложения вылизывать не нужно, но сильно страшным делать не надо - в общем чтобы более менее было приличным. За образец рекомендую взять дизайн админки Apache Tomcat - строго, просто и функционально.

Требования к этапу №1

1. Окружение: Apache tomcat 7, java 6 (именно 6, чтобы не видеть и не использовать фичи 7 или 8 java). Tomcat 8.5 по этой причине не подходит - ему нужна вроде java 8, git, IDE - Idea или Eclipse, Postgres SQL
2. Описание приложения: одна страничка, на которой находится таблица из списка доступных для закачки документов (doc, xlsx, pdf). Поля таблицы: название файла, размер файла, дата загрузки, версия + 1 колонка без заголовка. В последней колонке размещаются 2 кнопки - удалить и обновить. По кнопке удалить - просто удаляется документ, по кнопке обновить - появляется диалог для загрузки документа. Если выбираешь обновить - то версия документа инкрементится. Да, имя файла меняется на новое. На странице также есть кнопка Добавить. Выбираешь добавить - появляется диалог на добавление файла. Версия добавленного файла автоматом становится = 1.
3. Архитектура. Это JSP приложение с одной страничкой - index.jsp. Зааплоуженый документ физически хранится на диске (не в базе), а такие вещи как название файла, версия и размер хранятся в базе данных. На данном этапе не использовать никакие сторонние библиотеки. Обращение к базе идет только по JDBC, страница не должна содержать никаких левых скриптов.

Еще раз - делать как угодно и в плане разработки и в плане логики приложения, то есть, если требования в каком-то пункте кажутся недостаточными - делаешь как считаешь нужным - только потом надо попытаться объяснить почему так.