# Stage 2

This stage includes your first prototype.

It must build using Maven

| Version | Date | Comment |
|---|---|---|
| **Current Version (v. 2)** | **Jan 20, 2016 15:26** | **Marjorie Elizabeth Osborn Locke** |
| v. 1 | Jan 20, 2016 14:37 | **Marjorie Elizabeth Osborn Locke** |

## Confluence Wiki Pages

The documents posted or linked to on the “Wiki” portion of your Team’s private Confluence Space must be up to date.   Any suggested modifications from the grading feedback from Stage 1 must be implemented.

Your wiki at the deadline for this stage must include:

- Home Page
- Personnel Profile Page
- Software Design Page (See note below)
- UI Design Page
- Project Plan Page

Each page should have a title and be clear and organized. There is an editor built right into Confluence that you can use like a rich text editor to create your report.  Text should be added directly to the page by using the Confluence editor, rather than including files created in Word or otherwise.

### Software Design Page

This page will outline the design of your project, and will consist at this stage of a UML class diagram.

★ This must reflect the current state of your code.  You must generate the UML diagram from the code using a tool like

- Object Aid http://www.objectaid.com/

## JIRA Board and Git/Bitbucket Use

You will be graded on how well you are using your JIRA Board and how you are using your repository.

Any suggestions from your Stage 1 feedback should be implemented.

The tasks and boards should be up to date, and those that are complete should be marked as complete.

You should be using your repo with a brash-per-task model.  There should be a branch for each user story, which is then pulled back into dev (or other).  You can have other branches to help organize your epics (for example).  The branch names should be labeled with the unique task ID from JIRA.

## Prototype

The first prototype is due at this stage. While only a third of the user stories are marked for completion, there must be a GUI interface working for them, there must be a test mode, the Bitbucket repository must be organized and build automation must be setup for use with Maven.

## 1/3 of User Stories

To count the percentage completion, we will use the story points assigned to your user stories. The total you have completed divided by the total number in the project must be at least 0.333.

Please consider this as a *minimum* to have done by this deadline. Your team should already be working on having at least the core functionality done by this point to be in good standing to complete and test the project in time for stage3.

- Label the stories you would like to have considered for grading by using a Version for Stage2.
- We will test your prototype with your acceptance criteria (as listed in the user story) as well as our own.
- You will not be graded on GUI design at this stage, though we will give you feedback on it.

## Test Mode

You MUST be able to run the prototype without access to the Fitbit API. You should implement the switch between test mode and normal mode by reading in a flag from the command line.

```
#normally:

java -jar jarname.jar


#test mode example:

java -jar jarname.jar test
```

## Documentation

Your code must be documented using JavaDoc-style comments for all classes and their methods. You must generate the JavaDoc html files for all members (not just public ones), add and commit them for your tagged submission.

## Build Automation

It must be possible to both compile and package the prototype into an executable JAR file using the Maven command mvn package.

- You must create a pom.xml file and submit this.

## Bitbucket Submission

You will submit your prototype through Bitbucket, as you have done with your lab. Issue a pull request to the main branch.

Your repo must be organized into “src” and “doc” folders. The “src” directory must contain the source code (in the correct subdirectory as per Maven specifications), and the “doc” directory must contain the JavaDoc html generated for the code submitted.

## README.md

You must modify the README.md in the repository to give full instructions on how to clone your repository and build your prototype. This includes giving instructions on mvn package. You should also give instructions on how to run the program in normal and testing mode.