Visit Scheduling or Queueing

Tasks To Be Completed Prior To Design

Type of Project (Spike, Sprint, Epic): Sprint

Goals

Goals: *Must Have, Should Have, Could Have and Won't Have*

Must Have means we must have this as part of the solution. It is a requirement or we should rethink doing this work. **This is what we will deliver before the end of the work period** this also doubles as our DOD "Definition Of Done"

Should have means we would be embarrassed not to have it. We could technically produce the solution without it but people would be surprised.

Could have means anything else we could do. But this does not always mean low priority. But our selection of which could haves to include or exclude (or deliver later). Also is considered low hanging fruit (if we are in this code already for *x* reason wecan deliver *y* with little or no extra effort) or can be items or ideas for this module that can be packaged for a later spike or sprint.

Won't Have means this will not happen. It might be technically impossible, a taboo for our organization or out of scope. It does not mean "might happen or would be nice".

- **Must Have:**
  - **Create the module(s) in Github (*
    Error rendering macro 'jira'

    Unable to locate Jira server for this macro. It may be due to Application Link configuration.
)*
  - *Load modules into CI (*
    Unable to locate Jira server for this macro. It may be due to Application Link configuration.
)
  - Get Yonatan Grinberg and Adam Lauz to the point where they can develop on their own
  - Lay groundwork for future, bigger, sprints
- **Should Have:**
  - Unable to locate Jira server for this macro. It may be due to Application Link configuration.
  - Unable to locate Jira server for this macro. It may be due to Application Link configuration.
  - Unable to locate Jira server for this macro. It may be due to Application Link configuration.
  - Unable to locate Jira server for this macro. It may be due to Application Link configuration.
  - Unable to locate Jira server for this macro. It may be due to Application Link configuration.
  - Unable to locate Jira server for this macro. It may be due to Application Link configuration.
  - Unable to locate Jira server for this macro. It may be due to Application Link configuration.
- **Could Have:**
  - Unable to locate Jira server for this macro. It may be due to Application Link configuration.
  - Unable to locate Jira server for this macro. It may be due to Application Link configuration.

- Unable to locate Jira server for this macro. It may be due to Application Link configuration.
- Unable to locate Jira server for this macro. It may be due to Application Link configuration.

- **Won't have:**
  - **The whole module or queuing completed**

*

*

Quick overview Notes

General Info

Topic: Visit Scheduling or Queueing

Lead (Product Owner): Tobin

Known Developers Assigned (amount of effort in hrs dedicated to this work if possible): Yonatan Grinberg, Adam Lauz, Daniel Kayiwa

Date: Start: November 22nd - December 6

How to Participate

This first sprint for the Appointment module will not be advertised to the whole community however those interested in working with us on future sprints should contact us!

Add your name to the list on this wiki page (with any comments about your availability). If you want to join after the sprint has started just join the IRC channel mentioned above and say hello.

The general process:

1. New to OpenMRS sprints? Want help getting started? Join the IRC channel and say "???": I'd like to participate in the sprint!"
2. Pick a ticket from the available tickets in the top-left of the sprint dashboard page. (listed below)
   - Make sure it does not depend on a ticket that is incomplete.
3. If you have any questions about the ticket, ask on the group chat
4. Do the ticket. See our HOWTO for git. **Sprint specific git HOWTO for devs with push rights: whatever works for you :-)** If you don't like pull requests, don't send them. Commit and push directly to the main repo. If you do like pull requests, fork the main repo and send pull requests, but merge them right after. My favorite way is to work on the main repo, but create local branches (without pushing them to the main repo). Merge branches locally to the master and push to the main repo.
5. Join the daily scrum to share your updates

Design

There should be multiple design meetings. The first starts off with a small group working with the leader to determine the high level scope of the project and then to break down into smaller pieces to eventually place as tickets. Once those meetings have happened use of the community design meeting time should be used to refine the tickets and if possible assign them to who will be doing the work.

Looking at creating 3 parts (Possibly 2/3 different sprints)

1. Structure issues w/OpenMRS and Pair program

1 Core Dev in a complementary timezone

2. Get more people involved with core features

This is where the community gets involved

3. Once in place some feedback from users to make changes.

Looking to see if we an build this incrementally

1. A separation of API and UI

Design Call 11/7 Proposal

1. Start small (example Create and appointment) Layout the design for that. Concern is that if it's too basic will it get used? Discussion on this is needed.
    1. Creating a table in DB
    2. basic UI
2. Get the developers on some specialize bugs to fix
3. Happy with the design for the 1st feature
    1. Possibly Daniel could schedule an hour a day to work with them setting up the project with their first feature.
        1. Talk to Daniel about scheduling (Sunday through Thursday is the workweek)
4. How do we guide tomorrow call correctly?
    1. Goal What's the data model needed?
        1. what are the items that needed in the DB (start, finish, date, time, location, what needs to be recorded)
    2. Some basic introductions
    3. When we get an idea of the smaller pieces figured out then how to create tickets
        1. We need to frame tickets as user stories and then have the developers build the actual tickets for code
5. Look at existing modules to see if we can reuse (module ID: appointment) need to look at this source code to see if it has potential

Risks (these should be resolved prior to or during the design meetings):

Enter anything that is still questionable or worrisome that has not been answered: (open issues, potential concerns) Once a decision is made make sure that a summarized answer is included.

Example: Q. How will we handle issues with conflicting userID's? A. All ID's will have an added identifier that is unique.

1. Developers need more experience with the OpenMRS framework/models
2.

Effort Accounted For: (At the end of the design call all tickets should have an estimated effort and that total should be balanced against the known available effort of the developers assigned) (Yes/No)

If not in balance in favor of success why and the action plan for remaining items

via IRC on the #openmrs channel on freenode.

Use this channel for **ALL** debugging and random questions having to do with the sprint. Please avoid direct messaging to personal contacts. If you have a question, someone else most likely does too, and our geographically distributed community benefitsfrom public group discussion.

Sprint break down: https://www.dropbox.com/s/cldus4rbu4mp777/User%20Stories%20v2.pdf

Kickoff Meeting

Kickoff meeting Date: TBD

(Meeting setup with known developers after the final design meeting, but prior to the start date):

Kickoff Meeting Checklist

1. Do we know where we are going? (Yes/No)
    1. Do we know the problem we are solving (*yes/no*).
    2. Do we have a complete backlog of items to complete this work? (yes/no)
    3. Do we know our scope and priorities? (yes/no)
    4. Have we defined success? (yes/no)
2. Do we know how to get there?
    1. Do we have any unknowns to be decided during the sprint? *If Yes what are they?*
    2. Do we know who is doing what on our project? (yes/no)
    3. Do we have a test to complete prior to completion beyond our normal submit process? (Yes/*No).*
    4. Do we have a high level architecture that is understood by the whole team? (This page fully completed will accomplish this) (Yes /No)
    5. Do we know the biggest constraint that is likely to inhibit our success? *(Yes and no) If no what is it?*
    6. Is this a part of a larger story or epic? (Yes/No) If Yes please link
3. Are we set up to succeed?
    1. Do we have the right people? *(yes*)
    2. Have we cleared the decks of all other distractions?

During Project Notes

Unknown macro: {iframe}

Post Project (Retrospective)

Did we complete tickets to a 100% DOD? (Yes/No) if no, have those tickets been assessed and placed for future work if needed?

What did we do well?

What could we do better?

What should we not do again?

Resources

Sprint Dashboard: https://tickets.openmrs.org/secure/Dashboard.jspa?selectPageId=12355

Source code at: https://github.com/openmrs/openmrs-module-appointment

Kickoff meeting recording: ??