Designer Onboarding to OpenMRS 3

Introduction: What is OpenMRS 3?

Welcome to the OpenMRS 3 design world! OpenMRS 3 (aka 3.0, or 3.x) is a large project that:

- Substantially modernizes OpenMRS' frontend technology;
- redesigns the UX/UI leveraging Carbon Design; and,
- brings together many organizations to use both this technology and the shared design framework to share the burden of work and benefit from collaborative processes. So that good projects can achieve more good, together.

See the section below called "A Visual History of OpenMRS" to better understand why we arrived here.

Contents:

- Connect With Us
  - Online
  - Weekly 3.x Product Design Call
  - Weekly 3.x Squad Call (all-hands)
- Self-Onboarding Guide
- Design Pattern Documentation
- Styleguide & Components
- Design Principles of 3.x
- Videos
- Visual History of the OpenMRS Frontend
  - OpenMRS 1.x: (2004)
  - OpenMRS 2.x: (2011)
  - A Different RefApp: Bahmni (2016)
  - Turning Point: The Situation by 2020
  - OpenMRS 3.x: (2021)
    - Example: Before OpenMRS 3
    - Example: With OpenMRS 3

Quick Design Links
Styleguide & Components
lightbulb Design Patterns  video-filled How to Use Zeplin gallery Styleguide & Components

Sample Designs*

* Note: to see detailed CSS guidance at the Detailed links, you require an invite to our OpenMRS Zeplin project. Just let us know if you require an invite. In the meantime you can review all designs with this account:

u: openmrszeplinreview@gmail.com
p: zeplinreview123

Connect With Us

Online

- On Slack: #ux-design-advisory and #openmrs3
- On OpenMRS Talk (Talk is the OpenMRS community forum)
- Support:
  - #openmrs3-helpme slack channel
  - OpenMRS 3 Project Home Page
  - OpenMRS 3 Contacts: OpenMRS 3 Project Lead: Jonathan Dick OpenMRS Director of Product: Grace Potma
  - General questions or problems with Carbon Design: Carbon Issue and Question hub

Weekly 3.x Product Design Call

What: A weekly gathering of BAs, PMs, Designers and stakeholders co-designing 3.x workflows and features. We share emerging ideas, detailed requirements, discuss concerns, and review the latest design outputs and user testing results.

When: Mondays at 3pm UTC (See in your timezone) (GCal link here)

video-circle Zoom link: om.rs/zoommf edit Call Notes & Recordings

Weekly 3.x Squad Call (all-hands)

What: Our weekly 3.x initiative all-hands. Everyone shares their updates, and a short design showcase is shared to keep everyone current on the latest design thinking and user feedback.

Thursdays at 1pm UTC / 6:30pm IST / 4pm EAT / 3pm CEST / 9am EDT / 6am PDT (GCal link here)

video-circle Zoom link: om.rs/zoommf edit Call Notes & Recordings

Self-Onboarding Guide
1. **Videos!** Let's start with something fun. Check out the videos in the video section below to get a rapid intro to the what, where, and how of 3.x design.

2. **Review key design conventions and assets**
   a. **Design Patterns:** Read [this page on Design Patterns for OpenMRS 3](#). This will also explain our chosen Design System (Carbon) and show you where and how to access our Style Guide tool (Zeplin). Be sure to review [this full page of Design Pattern Documentation](#).
   b. **Zeplin Access:** Get access to the OpenMRS Zeplin projects by contacting grace@openmrs.org. Familiarize yourself with some of the existing features so you can see existing workflows that have already been User Tested and are being / have already been built into the Reference Application.
   c. **OpenMRS Styleguide:** Find the OpenMRS styleguide in Zeplin. It's important designers are familiar with this so you can see the existing OMRS Carbon Component inventory, and to be aware of where we have intentionally created custom components.
   d. **Carbon:** OpenMRS 3 uses the open-source project, *Carbon Design System*. If you're not already familiar with Carbon Design, check out their helpful *Carbon Onboarding for Designers*. You'll find things like the Carbon Design Kit, grid system, and guidance on how to use them.

3. **Demo:** Play with the Reference Application! Go to [https://o3.openmrs.org](https://o3.openmrs.org) u: admin p: Admin123
   a. Tips: Play with the sample patient chart for Agnes Testerson. Try following the steps taken in some of the OpenMRS 3.x demo videos (don’t forget to look at patterns like what is used in Patient Lists and Offline Mode)

4. **Existing Research:** Wondering how we’ve been doing UX Research thus far? See our existing research, process, and findings at [UX Research Questions We’ve Worked Through](#).

5. **Communication:** Join our UX communication slack channel: #ux-design-advisory

---

**Design Pattern Documentation**

Be sure to review [this full page of Design Pattern Documentation](#).

**Styleguide & Components**

Explore the growing OpenMRS 3 Styleguide and Component Library.

*Note: This will require an invitation to the OpenMRS Zeplin project. Contact grace@openmrs.org for access.*

**Design Stack**

*Note: so long as you can access the patterns in Zeplin, you should use the tools you are most comfortable with or have to access to.*

- Design Handover and Styleguide: Zeplin
- Designing: Sketch (but moving to Figma in 2022 once Carbon supports Figma)
- Clickable Mockups: Protopie (but probably anything you’re used to can work)
- Asynchronous Feedback: Loom Videos has been helpful ([free Chrome extension here](#))
Visual History of the OpenMRS Frontend

OpenMRS 1.x: (2004)

The main strategic focus of the OpenMRS project was our modular backend: the software platform, the data model, and modular add-on software that unlocked additional backend features. The frontend code provided simple role- and privileged-based control along with extension points and path-based space for modules to add their own frontend pages.

This visual shows the first reference UI that provided the very basic fundamentals for a system. Implementations needed to add or build modules to realize even basic clinical functions.

Some problems with the 1.x RefApp UI are:

- Based on Java Server Page (JSP) technology that was tricky for developers.
- Developing and debugging frontend pages is slow and cumbersome.
- Relies on server-based rendering of web pages, which is no longer best practice.

OpenMRS 2.x: (2011)

In 2011, the OpenMRS Reference Application ("RefApp") frontend was updated, along with the UI. Newer frontend approaches (like React and Angular) had appeared, but were not yet established as best practice.

The modular backend/platform continued to be the main strategic focus of the community. The primary focus of the 2.x frontend was to address key complaints from 1.x frontend development: primarily the ease of building new frontends for modules and development cycle time. Groovy Server Page (GSP) technology was introduced, allowing frontend pages to be scripted more easily, support debugging of frontend pages, and made for quicker development cycles.

2.x was successful in achieving shareable frontend elements across different organizations. However, some problems with the 2.x RefApp UI are:

- The Groovy Server Page (GSP) approach was custom-designed for OpenMRS, providing an OpenMRS-specific approach to web application development.
- Technology is now outdated. Difficult to find devs to work on; long dev onboarding.
- Still relies on server-based rendering of web pages, which is no longer best practice.
- UI uses an in-house style guide that requires maintenance. Not originally set up to be responsive (e.g. to work on tablets, phones).
Some implementers found it difficult to extend the Design to handle more detailed workflows and deeper EMR chart navigation.

A Different RefApp: Bahmni (2016)

After launching the 2.x RefApp, many in the OMRS community realized that a point-of-care EMR was needed, especially one that came out-of-the-box with an ecosystem of the critical tools needed in clinics and hospitals - Lab System, Pharmacy/Dispensing System, and RIS/PACS. A sub-faction of the community then created Bahmni, named after a very remote village in India in the middle of a tiger sanctuary. The name was a reminder of the extreme settings that the product needed to serve.

Bahmni was built on the OpenMRS platform, but added separate technical layers of abstraction and applied a different UI/UX. This means that frontend features developed for Bahmni cannot be used by 2.x RefApp users, and vice-versa.

Turning Point: The Situation by 2020

Despite sharing the same backend, many OpenMRS systems had a radically different frontend. The world had decided that frontend development was better done with client-side technologies and frameworks like React and Angular were well-established as best practice for web application development. As a result, many implementations looking to development new features, turned to these newer technologies and built in-house frontends on top of the OpenMRS platform.

This prevented collaboration on frontend features, and substantial duplicated effort around the world. These are all patient summary views in OpenMRS systems. (Click to zoom)

OpenMRS 3.x: (2021)

OpenMRS now recognizes a need to offer both a solid platform and a compelling frontend offering / modern user experience. In the OpenMRS 3 project, we are now working on a new UI/UX convention:

- Uses modern frontend technology e.g. JavaScript and client-side rendering; this has a larger and growing global dev talent pool.
- Modular frontend allows teams to plug-and-play with widgets created by others.
- Design and technology optimized for point of care, responsiveness, and patient care records that can be simple or very complex.
- Uses a 3rd party Design System maintained by another open source community.
- Avoiding/minimizing OpenMRS-specific approaches to development.

This time, we have put a lot of effort into the design, shareability, reusability, and extensibility of the User Interface.

Example: Before OpenMRS 3

These 2 organizations wanted to collaborate together on new feature. They couldn’t, because they didn’t share frontend technology, nor a common set of design principles. (Click to zoom)
Example: With OpenMRS 3

Now: With OpenMRS v3, these groups are actively pooling development and design time, since they can share frontend technology and consistent UX/UI in features. (Click to zoom)