Intermediates

Brooke Lampe

Task 3

Challenge 1	OpenMRS documentation was sometimes outdated, missing, and/or incomplete, so it was difficult to use OpenMRS services.
Resolution 1	*We examined relevant documentation and code, checked OpenMRS Talk for useful information, and consulted with our team members. If the challenge persisted, we asked a question on OpenMRS talk.*
Challenge 2	The architecture of OpenMRS was difficult to comprehend.
Resolution 2	*We looked up the architecture diagram of OpenMRS, which can be found on the OpenMRS Wiki, under Getting Started as a Developer --> Technical Overview. We also developed our own diagram of OpenMRS as it pertained to our module.*
Challenge 3	We found learning OpenMRS to be overwhelming because of its size.
Resolution 3	*We started with an abstracted overview of OpenMRS to understand its purpose and how it worked. Later, we focused on sections of relevant code and documentation. We avoided spending a lot of time learning information that was irrelevant to the module we were developing or maintaining.*
Challenge 4	Setting up OpenMRS for module development and maintenance is difficult because the documentation is vague and the process is complicated.
Resolution 4	*Our resolution involved trial and error. My team and I did not know which type of module was needed, so I downloaded several. Ultimately, we realized we needed our own server to run the module. We looked up and installed server software we found online, and we used it to run the module.*
Challenge 5	I found it difficult to remember all the steps to add our module to OpenMRS and run it, and the documentation did not help.
Resolution 5	*I recorded the steps to build, package, and run the module in my notes.*

Task 4
Setting up a development environment
Learning how to use the development environment
Familiarizing with OpenMRS overview and purpose
Familiarizing with OpenMRS tools (GitHub, Wiki, JIRA, Talk, etc.)
Familiarizing with OpenMRS architecture
Familiarizing with OpenMRS code
Learning how to contribute to OpenMRS
Learning common errors
Learning OpenMRS etiquette
Learning how to obtain help

Task 5

RELATIVE IMPORTANCE

1. Familiarizing with OpenMRS architecture
    a. This step is the most important step because it is very difficult to learn without help.
    b. The documentation for the architecture is difficult to locate and does not provide all of the important information.
    c. Understanding OpenMRS architecture is critical to building a module because the module will depend on a lot of the services supplied by the architecture.
    d. See *Challenge 2.*
2. Setting up the development environment
    a. This step is important because modules cannot be created or maintained without a development environment, and they cannot be tested without a working installation of OpenMRS.
    b. Though OpenMRS provides a lot of documentation for this step, the documentation can be vague and the process is complicated. See *Challenge 1.*
    c. This was a challenge my team experienced and ultimately solved through trial and error. We did not initially realize we needed our own server software, and OpenMRS did not provide support for finding and installing server software. See *Challenge 4.*
3. Learning how to use the development environment
    a. New developers cannot effectively contribute to OpenMRS without using the development environment, so this step is critical.
    b. It is lower on the list because the previous two steps are prerequisites.
4. Familiarizing with OpenMRS code
    a. OpenMRS has an extremely large codebase, so it is important for developers to learn techniques to understand the code and find what they need without being overwhelmed.
    b. See *Challenge 3.*
5. Familiarizing with OpenMRS tools (GitHub, Wiki, JIRA, Talk, etc.)
    a. This step is critical because the ability to use OpenMRS tools is a prerequisite for much of the onboarding process.
    b. However, the OpenMRS tools are fairly intuitive, and new developers should be able to understand the tools without much help.
6. Learning how to contribute to OpenMRS
    a. New developers join OpenMRS because they want to contribute, which gives this step its value.
    b. However, new developers must first familiarize with OpenMRS before concerning themselves with writing code and contributing.

7. Learning OpenMRS etiquette
    a. Learning OpenMRS etiquette is important when new developers ask questions, seek help, and start to contribute
    b. It is lower on this list because etiquette comes after general familiarization in the onboarding process.
8. Familiarizing with OpenMRS overview and purpose
    a. The overview and purpose of OpenMRS are critical to the onboarding process because it is hard to understand architecture or write code without knowing what the software is intended to do.
    b. However, locating and understanding the overview and purpose of OpenMRS is trivial and should not be a major focus of the onboarding process.
9. Learning common errors
    a. Learning common errors is important, but a number of the errors cannot be understood until other parts of the process are understood.
    b. In addition, developers can safely learn some common errors by making them.
10. Learning how to obtain help
    a. Learning how to obtain help is least important on this list because OpenMRS provides a lot of information on how to seek help.
    b. Furthermore, familiarization with the tools and etiquette of OpenMRS should give new developers an idea of how to obtain help.

Task 6

DEPENDENCIES

The development environment must be set up (2) before the developer can learn how to use it (3).

When familiarizing with OpenMRS, a new developer should first familiarize with OpenMRS architecture (1) in order to then understand OpenMRS code (4).

Learning OpenMRS etiquette (7) overlaps with learning how to contribute (6) and learning how to obtain help (10) because there is etiquette involved both in contributing and asking for help.

Familiarizing with OpenMRS architecture (1) would be difficult if developers did not understand OpenMRS purpose and what is expected of the software (8).

Learning common errors (9) requires developers to first understand OpenMRS (8), its architecture (1), and its etiquette (7).

Learning OpenMRS tools (5) is a prerequisite for familiarizing with OpenMRS architecture (1) and setting up a development environment (2), since both activities require use of the OpenMRS Wiki. Familiarizing with OpenMRS code (4) and learning how to obtain help (10) also depend on learning OpenMRS tools, since these activities utilize GitHub, OpenMRS Talk, and OpenMRS JIRA.

STEP	*DEPENDENCY*
Familiarizing with OpenMRS architecture (1)	*Familiarizing with OpenMRS overview and purpose (8)*
Setting up the development environment (2)	*Familiarizing with OpenMRS tools (5)*
Learning how to use a development environment (3)	*Setting up the development environment (2)*
Familiarizing with OpenMRS architecture (1)	*Familiarizing with OpenMRS tools (5)*
Familiarizing with OpenMRS code (4)	*Familiarizing with OpenMRS architecture (1)*
Familiarizing with OpenMRS code (4)	*Familiarizing with OpenMRS tools (5)*
Learning how to contribute (6)	*Learning OpenMRS etiquette (7)*
Learning how to obtain help (10)	*Learning OpenMRS etiquette (7)*
Learning how to obtain help (10)	*Familiarizing with OpenMRS tools (5)*
Learning common errors (9)	*Familiarizing with OpenMRS overview and purpose (8)*
Learning common errors (9)	*Familiarizing with OpenMRS architecture (1)*
Learning common errors (9)	*Learning OpenMRS etiquette (7)*

Task 7

GROUPING

Overview

- Familiarizing with OpenMRS overview and purpose
- Familiarizing with OpenMRS tools (GitHub, Wiki, JIRA, Talk, etc.)

Software

- Familiarizing with OpenMRS architecture
- Familiarizing with OpenMRS code

Development

- Setting up a development environment
- Learning how to use the development environment
- Familiarizing with OpenMRS tools (GitHub, Wiki, JIRA, Talk, etc.)

Contributing

- Learning how to contribute to OpenMRS
- Learning OpenMRS etiquette

Guidance

- Learning common errors
- Learning how to obtain help
- Learning OpenMRS etiquette
- Familiarizing with OpenMRS tools (GitHub, Wiki, JIRA, Talk, etc.)

Task 8

HIERARCHY

- OpenMRS Onboarding
  - Familiarizing with OpenMRS
    - Familiarizing with OpenMRS overview and purpose
    - Familiarizing with OpenMRS tools (GitHub, Wiki, JIRA, Talk, etc.)
    - Familiarizing with OpenMRS architecture
    - Familiarizing with OpenMRS code
  - Preparing to Contribute
    - Setting up a development environment
    - Learning how to use the development environment
    - Learning OpenMRS etiquette
    - Learning how to contribute to OpenMRS
  - Troubleshooting
    - Learning common errors
    - Learning how to obtain help

Task 9

Sequence

1. Familiarizing with OpenMRS overview and purpose
2. Familiarizing with OpenMRS tools (GitHub, Wiki, JIRA, Talk, etc.)
3. Familiarizing with OpenMRS architecture
4. Familiarizing with OpenMRS code
5. Setting up a development environment
6. Learning how to use the development environment
7. Learning OpenMRS etiquette
8. Learning how to contribute to OpenMRS
9. Learning common errors
10. Learning how to obtain help

Task 10

I will keep the sequence I created in Task 9 in order to prevent issues with dependencies and guarantee a logical flow of information.
I will keep the hierarchy I created in Task 8 because it fits well with the sequence and the information.
I will keep parts of the grouping from Task 7 so that related topics are more likely to be together. I cannot keep all aspects of the grouping because certain information fits in more than one group and would be repeated.

Task 11

OPENMRS ONBOARDING

1. Familiarizing with OpenMRS
    - A. Familiarizing with OpenMRS overview and purpose
        - i. Familiarizing with OpenMRS overview on the home page
        - ii. Familiarizing with OpenMRS purpose on the home page
    - B. Familiarizing with OpenMRS tools
        - i. Familiarizing with GitHub
            - a. GitHub maintains source code
        - ii. Familiarizing with OpenMRS Wiki
            - a. OpenMRS Wiki contains documentation
        - iii. Familiarizing with OpenMRS JIRA
            - a. OpenMRS JIRA tracks issues
        - iv. Familiarizing with OpenMRS Talk
            - a. OpenMRS Talk offers question and answer services
    - C. Familiarizing OpenMRS architecture
        - i. Familiarizing with OpenMRS architecture diagram
        - ii. Familiarizing with OpenMRS architecture documentation
    - D. Familiarizing with OpenMRS code
        - i. Selecting an OpenMRS module
            - a. Locate the module's source code on GitHub
            - b. Locate the README.md file
            - c. Read the README.md file
            - d. Locate the module's OpenMRS Wiki page
            - e. Browse the documentation
            - f. Locate the module's OpenMRS JIRA page
            - g. Selecting a JIRA ticket
                - a. Inspect the ticket
                - b. Locate relevant code
                - c. Inspect the code
2. Preparing to Contribute
    - A. Setting up a development environment
        - i. Find OpenMRS Wiki --> Developer Guide --> Getting Started as a Developer
        - ii. Follow instructions
        - iii. Find OpenMRS Wiki --> Developer Guide --> OpenMRS SDK
        - iv. Follow instructions
    - B. Learn how to use the development environment
        - i. Follow instructions on the OpenMRS Wiki
    - C. Learn OpenMRS conventions and etiquette

        i. Find the module conventions in OpenMRS Wiki --> Developer Guide --> For Module Developers --> Module Conventions
        ii. Read the module conventions
        iii. Browse OpenMRS Talk
    D. Learn how to contribute to OpenMRS
        i. Find the Contributing.md file in the OpenMRS core module
        ii. Read the Contributing.md file

3. Troubleshooting
    A. Learning common errors
        i. Find OpenMRS Wiki --> Troubleshooting
        ii. Browse the Troubleshooting page
        iii. Browse OpenMRS Talk, particularly "Ask OpenMRS" and "Implementing" categories
    B. Learning how to obtain help
        i. Explore OpenMRS community Help Desk
        ii. Review OpenMRS Talk

Task 12

OPENMRS ONBOARDING

1. Familiarizing with OpenMRS
    a. Explore OpenMRS on the OpenMRS home page
        i. Explore OpenMRS overview
        ii. Explore OpenMRS purpose
    b. Familiarize with OpenMRS tools
        i. Familiarize with GitHub
            1. GitHub maintains source code
        ii. Familiarize with OpenMRS Wiki
            1. OpenMRS Wiki contains documentation
        iii. Familiarize with OpenMRS JIRA
            1. OpenMRS JIRA tracks issues
        iv. Familiarize with OpenMRS Talk
            1. OpenMRS Talk offers question and answer services
    c. Examine OpenMRS architecture
        i. Examine OpenMRS architecture diagram
        ii. Examine OpenMRS architecture documentation
2. Investigating OpenMRS modules
    a. Select an OpenMRS module
        i. Locate the module's source code on GitHub
        ii. Read the README.md file
        iii. Locate the module's Wiki page
        iv. Browse the documentation
        v. Locate the module's JIRA page
        vi. Select a JIRA ticket
            1. Inspect the ticket
            2. Locate relevant code
            3. Inspect the code
3. Preparing to Contribute
    a. Set up a development environment
        i. Follow instructions in OpenMRS Wiki --> Developer Guide --> Getting Started as a Developer
        ii. Follow instructions in OpenMRS Wiki --> Developer Guide --> OpenMRS SDK
    b. Learn how to use the development environment
        i. Follow instructions on the OpenMRS Wiki
    c. Learn OpenMRS conventions and etiquette
        i. Read module conventions in OpenMRS Wiki --> Developer Guide --> For Module Developers --> Module Conventions
        ii. Browse OpenMRS Talk

    d. Learn how to contribute to OpenMRS
        i. Read the Contributing.md file in the OpenMRS core module
4. Troubleshooting
    a. Learn common errors
        i. Browse OpenMRS Wiki --> Troubleshooting
        ii. Browse OpenMRS Talk, particularly “Ask OpenMRS” and “Implementing” categories
    b. Learn how to obtain help
        i. Explore OpenMRS community Help Desk
        ii. Review OpenMRS Talk

Task 13

CONTRAST

- Each main section of the onboarding process will look noticeably different from the remaining sections. This contrast will distinguish the main steps in the process and provide a sense of sequence.
- Title text will be distinguished from subtitle text, and subtitle text will be distinguished from body text. This contrast will improve readability and indicate hierarchy.

REPETITION

- Each main section will have a similar layout of title, subtitle, and tasks. This repetition will keep the information from becoming overwhelming and demonstrate grouping.
- Elements of different sections that cover the same topics will be more similar to each other than elements of the same section that cover very different topics. This repetition will express grouping by topic.

ALIGNMENT

- Each section will have its title aligned with its subtitles and its tasks. This alignment will improve the readability of the visualization and indicate hierarchy and sequence.
- All of the sections will be aligned horizontally in a row from left to right. This alignment will demonstrate the sequence of sections.

PROXIMITY

- Elements of the same section will be closely grouped and kept apart from elements of other sections. This proximity will help separate steps and provide sequencing.
- Sections that are more closely related or are nearer to each other in sequence will be kept closer together than those that are not. This proximity will demonstrate grouping and sequence.

Task 14

CONTRAST

- Each main section of the onboarding process will look noticeably different from the remaining sections. This contrast will distinguish the main steps in the process and provide a sense of sequence.
    - Each section will utilize one main color, and each section will have a different main color compared to the remaining sections.
- Title text will be distinguished from subtitle text, and subtitle text will be distinguished from body text. This contrast will improve readability and indicate hierarchy.
    - Title text and text boxes will be larger in size than subtitle text and text boxes, and subtitle text and text boxes will be larger in size than body text and text boxes.

REPETITION

- Each main section will have a similar layout of title, subtitle, and tasks. This repetition will keep the information from becoming overwhelming and demonstrate grouping.
    - The title text of each section will be the same size, as will the subtitle text and body text. Section elements will be positioned and oriented similarly in all sections.
- Elements of different sections that cover the same topics will be more similar to each other than elements of the same section that cover very different topics. This repetition will express grouping by topic.
    - Each topic will be represented by a different color. Text pertaining to a specific topic, regardless of its section, will be displayed in that color.

ALIGNMENT

- Each section will have its title aligned with its subtitles and its tasks. This alignment will improve the readability of the visualization and indicate hierarchy and sequence.
    - The section title will be positioned directly above the section subtitles, and the body text will be positioned directly beneath the subtitle text.
- All of the sections will be aligned horizontally in a row from left to right. This alignment will demonstrate the sequence of sections.
    - Each element of each section will be horizontally aligned with the corresponding elements of the remaining sections. Each section will also be horizontally separated from the remaining sections.

PROXIMITY

- Elements of the same section will be closely grouped and kept apart from elements of other sections. This proximity will help separate steps and provide sequencing.
  - Elements of the same section will be positioned near each other, inside the same shape. A subtitle will be positioned nearer to its body text than to the body text of a different subtitle.
- Sections that are more closely related or are nearer to each other in sequence will be kept closer together than those that are not. This proximity will demonstrate grouping and sequence.
  - Sections will be positioned in order.

Task 15

- Each section will utilize one main color, and each section will have a different main color compared to the remaining sections.
- Title text and text boxes will be larger in size than subtitle text and text boxes, and subtitle text and text boxes will be larger in size than body text and text boxes.
- The title text of each section will be the same size, as will the subtitle text and body text. Section elements will be positioned and oriented similarly in all sections.
- The section title will be positioned directly above the section subtitles, and the body text will be positioned directly beneath the subtitle text.
- Each element of each section will be horizontally aligned with the corresponding elements of the remaining sections. Each section will also be horizontally separated from the remaining sections.
- Elements of the same section will be positioned near each other, inside the same shape. A subtitle will be positioned nearer to its body text than to the body text of a different subtitle.
- Sections will be positioned in order.

Task 18

USE CASES

1. New developers will refer to the visualization when joining the OpenMRS project.
2. New developers will refer to the visualization when setting up a development environment.
3. New developers will refer to the visualization when developing a module.
4. New developers will refer to the visualization when maintaining a module.
5. New developers will refer to the visualization when they encounter errors.
6. New developers will refer to the visualization when they need help.
7. New developers will refer to the visualization when looking for documentation.
8. New developers will refer to the visualization when preparing to contribute code to the OpenMRS repository.

Task 19

SCENARIOS

1. Joe is a new developer who wants to join and contribute to the OpenMRS project for his class. He is used to building programs from scratch and has never joined a project before. He does not know where to begin.
2. Anna is a new developer who researched and presented on OpenMRS but has not previously contributed. She wants to set up a development environment and start working with code.
3. James has been maintaining an OpenMRS module, but now he wants to create his own. He is planning to build a module that tracks activity levels in clinics.
4. Jenna designed a module to track activity levels in clinics. Now, she wants to help maintain OpenMRS by fixing a bug in the Provider Management module.
5. Nate has been maintaining the Provider Management module and has observed that the module crashes when someone attempts to add two providers with the same identifier.
6. Kate receives an error message, "You are not allowed to create polls" whenever she tries to create a post on OpenMRS Talk.
7. Grant is developing a module. He wants to access information in the OpenMRS database from his module, but he is not sure how to do it. He is searching for information about the process.
8. Gloria is a relatively new developer. She wants to submit a pull request for a bug she fixed in the Provider Management module, but she cannot remember the proper steps.

Task 20

QUESTIONS

1. Joe is a new developer who wants to join and contribute to the OpenMRS project for his class. He is used to building programs from scratch and has never joined a project before. He does not know where to begin.
    a. What is OpenMRS?
    b. What are OpenMRS modules?
    c. How can I create a module?
    d. How should I ask for help?
    e. What rules or guidelines do I need to know about OpenMRS?
2. Anna is a new developer who researched and presented on OpenMRS but has not previously contributed. She wants to set up a development environment and start working with code.
    a. How do I run OpenMRS?
    b. How can I obtain an OpenMRS account?
    c. I installed OpenMRS, but it will not run. What should I do to troubleshoot?
    d. I still cannot run OpenMRS. Where can I ask for help?
    e. People have tried to help me, but I still cannot run OpenMRS. Now what should I do?
3. James has been maintaining an OpenMRS module, but now he wants to create his own. He is planning to build a module that tracks activity levels in clinics.
    a. How do I create a module?
    b. What conventions are modules expected to follow?
    c. How can I add a page about my module to the OpenMRS Wiki?
    d. How do I run my module in OpenMRS?
    e. What level of testing is expected of an OpenMRS module?
4. Jenna designed a module to track activity levels in clinics. Now, she wants to help maintain OpenMRS by fixing a bug in the Provider Management module.
    a. How do I claim a ticket?
    b. What are the expectations of a bug fix?
    c. How should I ask for clarification on the details of the bug?
    d. What are the rules for submitting a pull request?
    e. What if my pull request is denied?
5. Nate has been maintaining the Provider Management module and has observed that the module crashes when someone attempts to add two providers with the same identifier.
    a. Where should I report a bug?
    b. What conventions should I follow when reporting a bug?

   c. How will I know if the bug has been reported already?
   d. How much detail is expected?
   e. Can I claim my own ticket?
6. Kate receives an error message, “You are not allowed to create polls” whenever she tries to create a post on OpenMRS Talk.
   a. How can I ask for help?
   b. Should I post to OpenMRS Talk or the community Help Desk?
   c. What information should I include when I post my question?
   d. Do I have to use professional writing in my post?
   e. How can I add information after posting my question?
7. Grant is developing a module. He wants to access information in the OpenMRS database from his module, but he is not sure how to do it. He is searching for information about the process.
   a. Where should I look for the documentation?
   b. What should I do if I cannot find the documentation?
   c. How can I fix outdated documentation?
   d. How can I obtain editing access to the OpenMRS Wiki?
   e. Do I need approval before I edit the OpenMRS Wiki?
8. Gloria is a relatively new developer. She wants to submit a pull request for a bug she fixed in the Provider Management module, but she cannot remember the proper steps.
   a. How do I create a poll request?
   b. What conventions must I follow when creating a pull request?
   c. How do I squash my commits?
   d. How can I modify my pull request in response to comments?
   e. What should I do if my pull request is denied?