MDE2 Projekt

To Do:

- □

Sprint Timeframe

-

1. 09.09 – 24.09
   - Goal: Complete planning & Organizing as SCRUM Team
2. 24.09 – 07.10
   - Goal: Finalize design decisions
3. 07.10 – 21.10
   - Goal Implementation of Alpha-Version
4. 21.10 – 11.11
   - Goal: Implementation of Beta-Version
5. 11.11 – 02.12
   - Goal: Finalize Testprotocols
6. 02.12 – 16.01
   - Goal: Release Version

Sprint 1

Task 1: Organize as Scrum Team

Project Owner: Michael Zechmeister
Scrum Master: Sebastian Schedl
Developer: Jakob Eder, Ronald Riegler

Task 2: Derive Product Backlog from MDE1

- Define User Stories and SCRUM Use Cases
- Finalize functional and non-functional Requirements
    - define Important data
    - data can be imported from example file that comply with ELGA Guidlines
    - data can be acquired by the user
    - data is saved to a database
    - generate the EP based on ELGA standards, EU standards or a custom format
    - the EP will be updated and regenerated when accessed
    - presenting the EP on a website using a Java dynamic web project, ensuring a short overview on the patients device and potentially
    - optional: providing a Quick Response (QR) code
- Identify/formulate Tasks from Requirements
    - Task 1: Group Decision on Important Data
    - Open interoperable project and implement git
    - Task 2: Creation of Example Dataset
    - Task 3: Java Program for Data Import and Extraction
    - Task 4: Creation of Summarized JSON for Client
    - Task 5: Database Setup
    - Task 6: Implementation of Data Saving Functionality (JSON Generation)
    - Task 7: Socket Connection Setup Between Client and Backend
    - Task 8: Sending FHIR Resources from Backend to Client
    - Task 9: Web Development with HTML and CSS
    - Task 10: Frontend Programming for Receiving EP
    - Task 11: Client-Side Functionality for EP Creation from FHIR Resources
    - Task 12: User Data Entry Functionality Implementation
    - Task 13: PDF EP Stylesheet Development
    - Task 14: Login Functionality Implementation (Patient + Optional Caretaker)

- Decide on technology to use for implementation
    - siehe Paper (Intellij, Java Dyn Web Project, SqlLite)
- Estimate time & resources needed to complete

Task 3: Organize & Document Tasks

- Prioritize Tasks
- Document meaning of priority level
- Be aware of any dependencies between tasks
- Plan Tasks such that each can be completed in a single Sprint

Task 4: Pre-plan Sprints

- 1 Sprint = 2 Weeks
- Draft Sprint Backlogs according to timeframe
- Plan daily or regular team meetings within Sprints

User Stories

Medical Data Import from ELGA

- **As a user**, I want the system to automatically import my medical data from ELGA, so that I don’t have to manually enter my medical history.

User Input for Additional Data

- **As a user**, I want to manually add or modify specific medical information that ELGA might not cover (e.g., allergies, recent surgeries), so that my DEPS reflects all necessary emergency data.

User Authentication and Access Control

- **As a user**, I want to log in to the DEPS web application using secure authentication, so that I can securely access my health data.

DEPS Generation

- **As a user**, I want to generate a DEPS report after providing input and having my ELGA data updated, so that I can have an emergency health summary available.

DEPS Versions for Different User Roles

- **As a user**, I want to generate different versions of DEPS (e.g., basic, detailed) depending on the user role (first responders, emergency doctors), so that the information is tailored to their needs.

Synchronization and Data Update

- **As a user**, I want the system to periodically synchronize my data with ELGA, so that any new medical information is reflected in my DEPS.

Use Cases

1. Use Case: User Authentication and Access

- **Actors**: User (patient), Medical Professional
- **Description**: A user or medical professional logs into the DEPS web application to securely access medical data.
- **Basic Flow**:
    1. The user accesses the web application.
    2. The system prompts for authentication (username/password or alternative method).
    3. The user enters credentials and submits.
    4. The system verifies the credentials.

5. Upon successful authentication, the user gains access to the application.

2. Use Case: Import Medical Data from ELGA

- **Actors**: System, User
- **Description**: The system imports medical data from ELGA to populate the DEPS with the user's medical history.
- **Basic Flow**:
    1. The user logs into the application.
    2. The system connects to the ELGA system.
    3. The system retrieves medical data (using CDA/FHIR standards).
    4. The system parses and stores the relevant medical information.

3. Use Case: Generate a DEPS

- **Actors**: User, Medical Professional
- **Description**: The user or medical professional generates a DEPS containing relevant medical information for emergencies.
- **Basic Flow**:
    1. The user or medical professional selects the option to generate a DEPS.
    2. The system checks for any new data from ELGA and user input.
    3. The system processes and formats the medical data into a DEPS.
    4. The DEPS is displayed or made available for download.

4. Use Case: Provide Additional Medical Information

- **Actors**: User
- **Description**: The user manually adds or modifies medical information that might not be available through ELGA.
- **Basic Flow**:
    1. The user navigates to the input section for additional medical information.
    2. The user enters additional information (e.g., allergies, recent medical history, medications).
    3. The system validates the input (e.g., using medical standards like FHIR).
    4. The system stores the additional data.

5. Use Case: Synchronize and Update Medical Data

- **Actors**: System, User
- **Description**: The system periodically or manually synchronizes the user’s medical data with ELGA to ensure the DEPS contains the latest information.
- **Basic Flow**:
    1. The user requests to synchronize their data or the system automatically schedules synchronization.
    2. The system connects to ELGA.
    3. The system retrieves updated medical data from ELGA.

4. The system parses, stores, updates, and refreshes the DEPS with the latest medical information.

6. Use Case: Generate Different Versions of DEPS for Different Roles

- **Actors**: System, Medical Professional, First Responder
- **Description**: The system generates customized versions of DEPS based on the role of the user (e.g., first responder or medical professional).
- **Basic Flow**:
  1. The medical professional or first responder logs into the system.
  2. The system identifies the user's role.
  3. Based on the role, the system generates a DEPS tailored to that role (e.g., basic for first responders, detailed for doctors).
  4. The DEPS is presented according to the user's permissions and role.

Sprint 2

. 24.09 – 07.10 • Goal: Finalize design decisions