Project Plan

UNIFIED HEALTHCARE MANAGEMENT SYSTEM (UHMS)

RuMED

Date	**:**	**07/03/2024**
Version	**:**	**1.0**
State	**:**	**State**
Author	**:**	**Karunadika Ruchira**

Version history

Version	Date	Author(s)	Changes	State
1.0	07/03/2024	Karunadika Ruchira		

Distribution

Version	Date	Receivers

Contents

1. Project assignment

1.1 Context

The project is developed for RuMED, a company focused on modernizing healthcare through technology. RuMED aims to bridge the gap between healthcare providers and patients by offering digital solutions that streamline health management processes. The context involves a healthcare industry that is increasingly seeking efficient, secure, and user-friendly ways to manage patient care, appointments, and medical records. By introducing the Unified Healthcare Management System (UHMS), RuMED addresses the need for a cohesive platform that enhances accessibility, reduces administrative burdens, and improves overall patient outcomes in a fast-paced digital world.

1.2 Goal of the project

The goal of this project is to improve healthcare management. It aims to make accessing and managing medical information easier for everyone involved in healthcare. The new system allows patients to view their health records online and book visits effortlessly. Doctors can quickly check patient history, update records, and set up care plans. Administrators will efficiently run the appointment schedule.

This project enhances healthcare by making it faster and more reliable. It reduces waiting times and errors. By improving patient care and saving healthcare workers' time, the project adds significant value. The ICT product, UHMS, opens new possibilities for smart health data handling and supports rapid decision-making. This system marks a move towards modern, efficient healthcare.

1.3 Scope and preconditions

Inside scope:	Outside scope:
1 Digital Health Records Management	1 Selecting Nearest Physician
2 Appointment Scheduling System	2 Insurance Processing
3 Health Alerts and Recommendations	3 Selling medicines

The project will utilize a Microsoft SQL Server database, reflecting a technology choice already made. This choice supports robust data management and scalability. Web and desktop applications will be developed using frameworks compatible with this database.

1.4 Strategy

The project will use an agile approach, specifically scrum. This choice suits the project's need for flexibility and regular feedback. Agile allows for quick adjustments based on user input and testing. It supports continuous improvement and adapts well to changes. Scrum, with its sprints and reviews, ensures the project stays on track and meets user needs effectively.

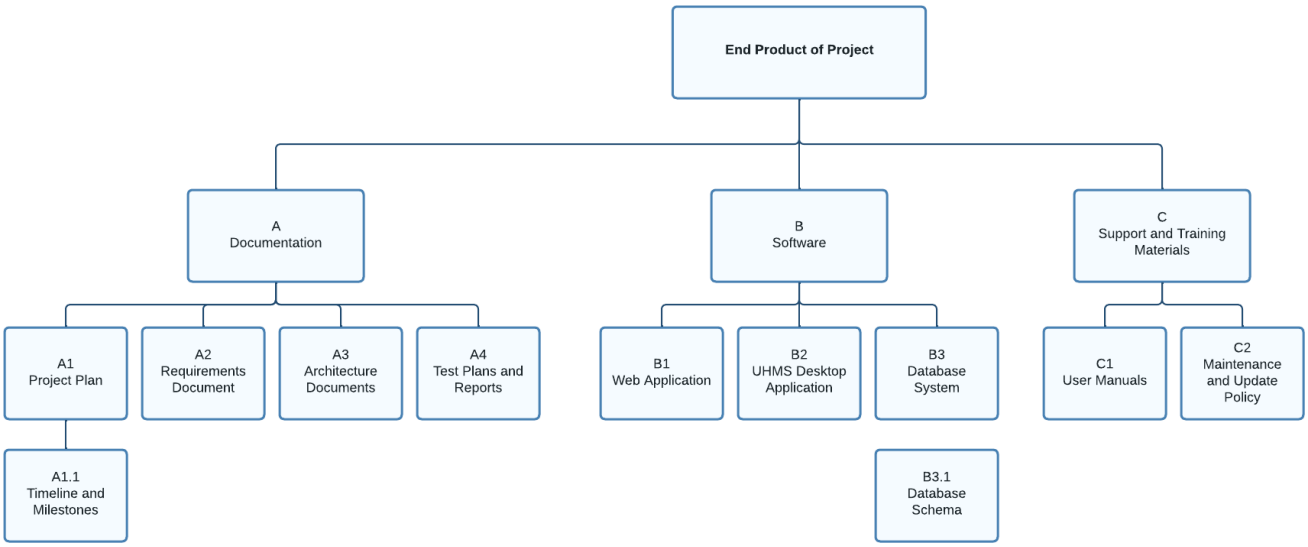
1.5 Research questions

Questions	Approach/Methodology	Dot Framework Strategy
How can we ensure data security and patient privacy in UHMS?	Conduct a literature review on best practices for data security in healthcare applications. Implement workshops with IT security experts to identify specific threats and solutions. Use lab testing to evaluate encryption methods and access control mechanisms.	Emphasize technology and process dots, focusing on secure software development practices and user authentication protocols.
What features are most needed by patients and healthcare providers in a healthcare management system?	Use surveys and interviews with potential users to gather requirements. Analyze feedback to prioritize features. During realization phases, prototype testing sessions will be conducted to refine these features based on direct user feedback.	Highlight user and function dots, with iterative design and development processes to ensure user needs are met.
How can UHMS integrate seamlessly with existing healthcare IT ecosystems?	Research existing healthcare IT infrastructures within target organizations through case studies. Develop a technical feasibility study to explore integration possibilities. Pilot integration modules in lab environments to test compatibility and interoperability.	Focus on the technology and environment dots, aiming for high compatibility and minimal disruption to current operations.

1.6 End products

- Project Plan
- Requirements Document
- Architecture Documents
- Research Reports
- User Interface Designs
- Database Schema
- Test Plans and Reports
- User Manuals
- Maintenance and Update Policy

Project Breakdown Structure (PBS)



2. Project organisation

2.1 Stakeholders and team members

Name	Role and functions	Availability
Chua Jessie Email: j.chua@fontys.nl	*Product Owner*	*Monday & Wednesday 09:00 – 14:00*
Makoveeva Olga o.makoveeva@fontys.nl	*Contact Person*	*Monday & Thursday 13:00 – 16:00*
Rabeling Bart b.rabeling@fontys.nl	*IT/Technical Liaison*	*Monday – Friday 09:00 – 16:00*

2.2 Communication

Weekly team meetings – In-person or Zoom meetings
(Bi-)weekly client meetings – In-person

3. Activities and time plan

3.1 Phases of the project

1. Initiation Phase

- **Problem Analysis:** Identify and analyze the current challenges in healthcare management that the UHMS will address. Gather initial requirements through stakeholder interviews and existing system evaluations.
- **Project Kickoff:** Formal project launch with stakeholders to agree on project goals, roles, and expectations.

2. Planning Phase

- **Sprint Planning:** Break down the project into sprints, each with specific deliverables and tasks aligned with the overall project goals.
- **Resource Allocation:** Determine the resources required for each sprint, including personnel, technologies, and materials.
- **Risk Assessment:** Identify potential risks and create mitigation strategies.

3. Execution Phase

- **Development Sprints:** Iterative development cycles that include designing, coding, and testing the UHMS features. Each sprint ends with a review and retrospective to guide the next one.
- **Ongoing Research:** Conduct continuous research on technology trends, user feedback, and healthcare regulations that may impact the project.
- **Documentation:** Create and update documentation, such as the architecture documents and user manuals, throughout the development process.

4. Quality Assurance Phase

- **Testing:** Rigorous testing of the system, including unit tests, integration tests, and user acceptance testing to ensure reliability and performance.
- **Refinement:** Based on testing feedback, refine, and adjust features to ensure they meet user needs and quality standards.

5. Deployment Phase

- **User Training:** Conduct training sessions with end-users to ensure they are comfortable with the system.
- **System Deployment:** Deploy the web and desktop applications in the client's environment, ensuring proper configuration and operation.

6. Handover Phase

- **Handover Documentation:** Deliver all project documentation to the client, ensuring they have all the information required to use and maintain the system.
- **Final Review:** Conduct a final project review with stakeholders to ensure all objectives have been met.

7. Evaluation and Reflection Phase

- **Project Evaluation:** Assess the project's success against the initial goals and objectives.

8. Wrap-up Phase

- **Portfolio/Thesis Development:** For internship projects, dedicate time to compiling a portfolio or writing a thesis that summarizes the project's approach, findings, and outcomes.
- **Project Closure:** Formal closure of the project, releasing project resources and providing any final deliverables to the client.

3.2 Time plan and milestones

Phasing	Description	Start date	Finish date
1 Project Kick-off & Problem Analysis	Understanding project scope, stakeholder interviews, and initial requirement gathering.	19/02/2024	04/03/2024
2 Initial Development Cycle	Development of basic user interfaces and database schema setup.	05/03/2024	18/03/2024
3 Feature Development Cycle	Implementation of core functionalities for patient and doctor modules.	19/03/2024	01/04/2024
4 Feature Development & Testing	Further development of advanced features and initial integration testing.	02/04/2024	15/04/2024
5 System Integration & Testing	Focus on system integration, security features, and comprehensive testing.	16/04/2024	29/04/2024
6 Finalization & Bug Fixes	Addressing feedback, bug fixing, and system optimization.	30/04/2024	13/05/2024
7 User Acceptance Testing	Conducting user acceptance testing and final adjustments based on feedback.	14/05/2024	27/05/2024
8 Handover & Documentation	Preparing user manuals, maintenance guides, and handover documentation.	28/05/2024	10/06/2024
9 Evaluation & Reflection	Project evaluation, retrospective meetings, and learning documentation.	11/06/2024	17/06/2024
10 Wrap Up & Portfolio/Thesis Development	Finalizing all project documentation and preparing the portfolio/thesis.	18/06/2024	28/06/2024

4. Testing strategy and configuration management

4.1 Testing strategy

Unit Testing:

- **Strategy:** Each piece of code will be tested individually to ensure it performs as expected.
- **Goal:** Aim for at least 80% code coverage with unit tests.
- **Automation:** All unit tests will be automated and run with each build to catch issues early.

Component Testing:

- **Strategy:** Test individual components of the application, such as user interface layers and database access layers, to verify they function correctly both independently and when integrated with other parts of the system.

Integration Testing:

- **Strategy:** Focus on the interactions between integrated components and systems, ensuring that they work together seamlessly.
- **Automation:** Integration tests will be automated where possible, particularly for backend services and data integration points.

System Testing:

- **Strategy:** Test the entire system's functionality, security, and performance to validate the software against the requirements.
- **Automation:** Key system tests will be automated, while others may be manual, especially for performance and security.

Acceptance Testing:

- **Strategy:** Conducted with stakeholders to ensure the system meets business needs and is ready for deployment.

Justification and Quality Measures:

- The layered approach to testing allows for the early detection of defects and ensures that by the time the product reaches acceptance testing, most issues have been identified and resolved.
- Additionally, code reviews and pair programming practices will be implemented to further ensure quality and knowledge sharing within the team.

4.2 Test environment and required resources

Development environment:

- Local development & testing
- Development database in fhict.local
- Git

Production environment:

- .NET luna server
- Production data in fhict.local

4.3 Configuration management

The project will employ a robust configuration management strategy, focusing on maintaining the integrity and traceability of the software versions throughout its lifecycle.

Version Management:

Tooling: Git will be the version control system of choice due to its wide adoption and support for distributed development. It provides robust mechanisms for tracking changes, branching, and merging.

Branching Strategy: The project will use the 'Git Flow' branching strategy.

Promotion Strategy: Code changes will be promoted through branches and from feature branches into develop for all development work, from develop to release branches for release preparation, and from release branches to master for production deployment.

Release Strategy: Releases will be scheduled at the end of each sprint cycle. Release branches will be created from develop, and once the release is finalized it will be merged into master.

Baseline Strategy: The master branch at the point of each release becomes a baseline. Past releases can be traced through tags, and hotfixes can be applied if necessary.

5. Risk

5.1 Risk and mitigation

Risk	Prevention activities	Mitigation activities
1 Technology Integration Issues	Conduct feasibility studies before finalizing technology choices. Create a proof of concept for critical integrations.	Have a backup plan with alternative solutions. Allocate time for additional development and testing if needed.
2 Loss of Key Personnel	Ensure cross-training within the team and maintain detailed project documentation.	Have a succession plan and identify potential replacements or redistribute workload among current team members.
3 Scope Creep	Clearly define project scope and requirements upfront and enforce change control processes.	Review and approve changes strictly. Adjust project plans and timelines if necessary.