SOEN 6841 - Software Project Management

Winter 2024

Phase II

IntelliHealth: AI-Enhanced Health Monitoring Platform

Submitted by:-

Group 24

Rishi Ravikumar (40269971)
Md Abdul Hai (40270829)
Tania Sanjid (40255010)
Vedant Gadhavi (40269585)

IntelliHealth: AI-Enhanced Health Monitoring Platform

Feasibility Study

Scope and Functionalities

IntelliHealth aims to revolutionize health management using intelligence. The goal is to provide tailored health insights, support monitoring of health metrics and offer interaction through a virtual assistant. This platform caters to a range of users including those focused on being, individuals managing chronic conditions and healthcare professionals seeking tools for improved patient care with actionable information.

Technical Feasibilities Assessment

- **Incorporating Advanced Technology, Tools and Infrastructure**

The IntelliHealth platform is being developed with cutting edge AI and machine learning technologies. By leveraging cloud computing services like AWS the solution benefits from computing power and extensive storage capacity required to process volumes of health data in real time. Employing development tools and frameworks allows for the creation of an user friendly platform accessible, via both mobile devices and web browsers.

- **Overcoming Technical Challenges**

Recognizing the obstacles related to data privacy and security, integration of data sources and management of amounts of real time data. The project incorporates encryption methods, robust data management practices and scalable cloud solutions to safeguard user data integrity while ensuring operation of the platform.

Operational Feasibility Examination

- **Alignment with the organization's vision**

IntelliHealth exemplifies the thinking objective of utilizing technology to improve healthcare outcomes. It equips individuals with insights into their health promoting an approach to wellness management that aligns with the changing landscape of healthcare.

➢ **Incorporation into organizational workflows and procedures**

The system is crafted to blend with existing healthcare workflows offering healthcare professionals tools for monitoring patients remotely. Its user-friendly design ensures integration reducing disruptions to established healthcare practices.

Economic Feasibility Evaluation

➢ **Thorough examination of costs and benefits**

A detailed analysis highlights the economic value proposition of IntelliHealth emphasizing its ability to notably cut down healthcare expenses through care and early intervention measures. This assessment compares savings from decreased hospital admissions and lower medication costs against ongoing outlays forecasting the platform's financial sustainability in the long run.

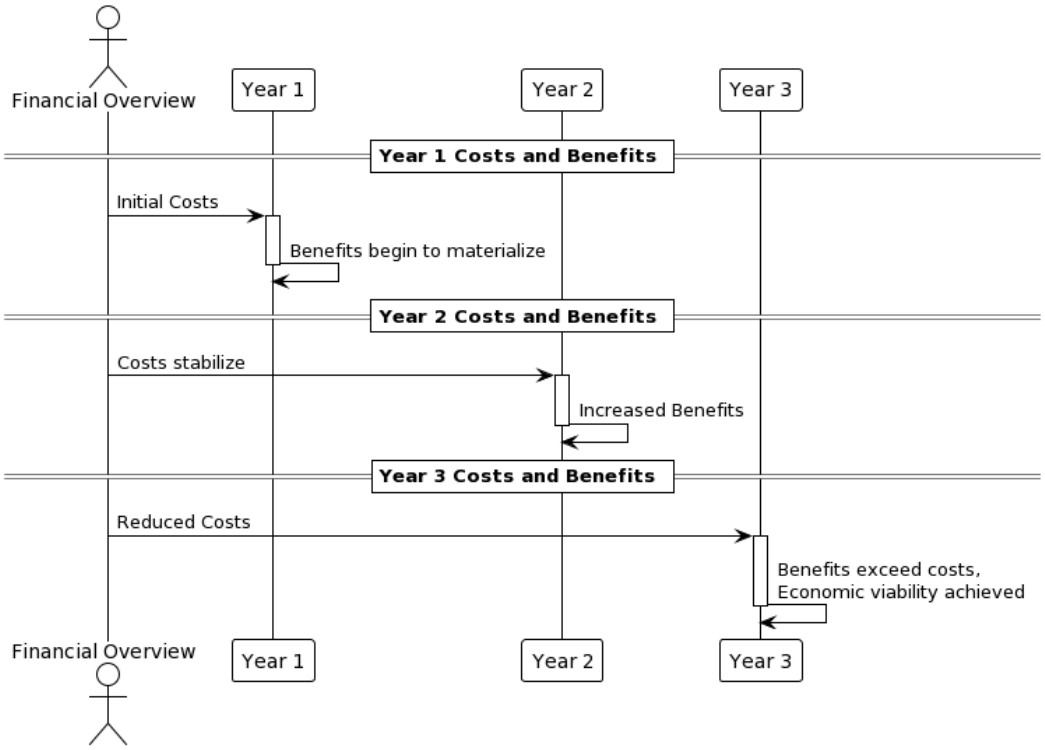


Fig: Cost Benefit Analysis

➢ **Comprehensive estimation of project costs**

The financial plan, for the IntelliHealth project covers personnel, infrastructure, services, and overhead expenses. Managing budgets is crucial for balancing responsibilities with the expected returns ensuring the project's financial well being and long term viability.

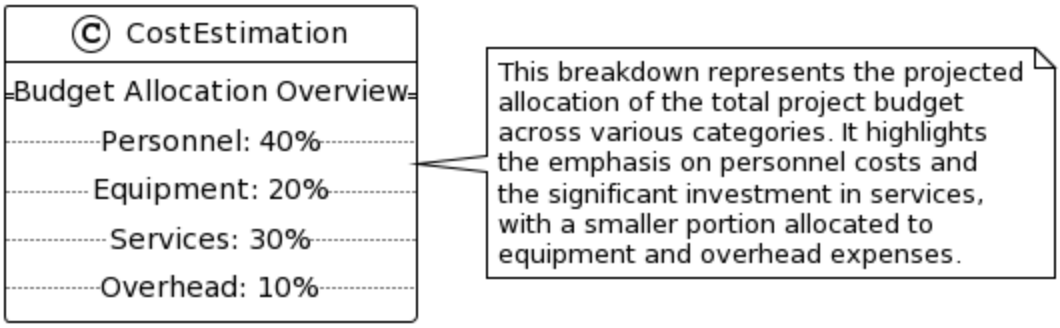


Fig: Project Cost Estimation

Scheduling Feasibility Outlook

➢ **Strategic Planning for Agile Project Timelines**

By adopting a development method IntelliHealth meticulously plans its project schedule to allow for adaptability and gradual advancements. This plan covers the phases of prototype creation, beta testing and full scale deployment integrating user feedback to improve the platform's features.

➢ **Recognizing Key Project Dependencies and Milestones**

The project's success hinges on handling dependencies like forming tech partnerships meeting standards and securing funding. Key checkpoints are established to monitor advancement toward reaching the objectives of the platform such as introducing a product iteration enhancing user interaction and demonstrating effectiveness.

Solution Proposal

Solution Approach

In order to provide a seamless experience for the users, ensuring that all the core features are developed with the flexibility of new feature requirements, the software solution would consist of 4 subsystems. The rationale behind splitting the system into the subsystems is to ensure a clear separation of concerns between subsystem development teams, while ensuring parallel progress.

➢ **Cross Platform Mobile Application**

This subsystem provides the interface where users and health professionals can interact with the platform. The application would facilitate health data inputs, help visualize smart and personalized health insights, and allow users to communicate with the virtual AI assistant. The application would also help users connect with health professionals to then share health data securely and procure consultations within the app.

➢ **Web Application**

The online web application offers users and medical professionals detailed dashboards with real time statistics. It provides insights from predictive analytics from individual user-generated health data. It also includes safe payment methods for premium services and subscription management.

➢ **Data Analytics Engine**

This subsystem encompasses the data analytics components responsible for analyzing large volumes of user health data. It includes data pipelines for collecting raw data from various sources, to then derive personalized health recommendations using continuously improving machine learning models. The engine would be developed to ensure that the data used for model training is anonymous and protected.

➢ **Backend Engine**

The backend engine is the subsystem that handles the core backend infrastructure for the user facing APIs for the mobile and web applications. The primary functionality of the subsystem is to manage user authentication, session management and to orchestrate communication between other subsystems.

Key Features

The software solution aims to deliver the following functional and non-functional requirements.

Functional Requirements

- **Real-time data ingestion**: The system must be able to ingest real time data from the installed user applications
- **Data processing**: The system must process the real time incoming data efficiently. The information should be kept in a real-time database that the data analytics team may access to facilitate quick retrieval and historical data tracking
- **Data analytics**: The system must be able to perform complex event processing to detect errors, as well as generate meaningful insights
- **Monitoring**: The system must have support to visualize data and system load metrics of data generated by the various data sources
- **Alerting and Notifications**: The system must be capable of generating smart insights as push alerts/notifications
- **Configuration**: The system must be mutable to change software configuration and features

Non Functional Requirements

- **Reliability**: The system shall be able to handle fluctuations in data volume and new data sources
- **Scalability**: The system shall be equipped to handle data from increasing number of new users
- **Performance**: The system shall adhere to stated quality of service (QoS) specification for latency, throughput, resource utilization, and service level metrics
- **Traceability**: The system shall have system traces and metric collection backed in to enable troubleshooting
- **Security**: The system shall be built to ensure secure communication between subsystems
- **Service Availability**: The system shall have 99.999 (three nines) uptime

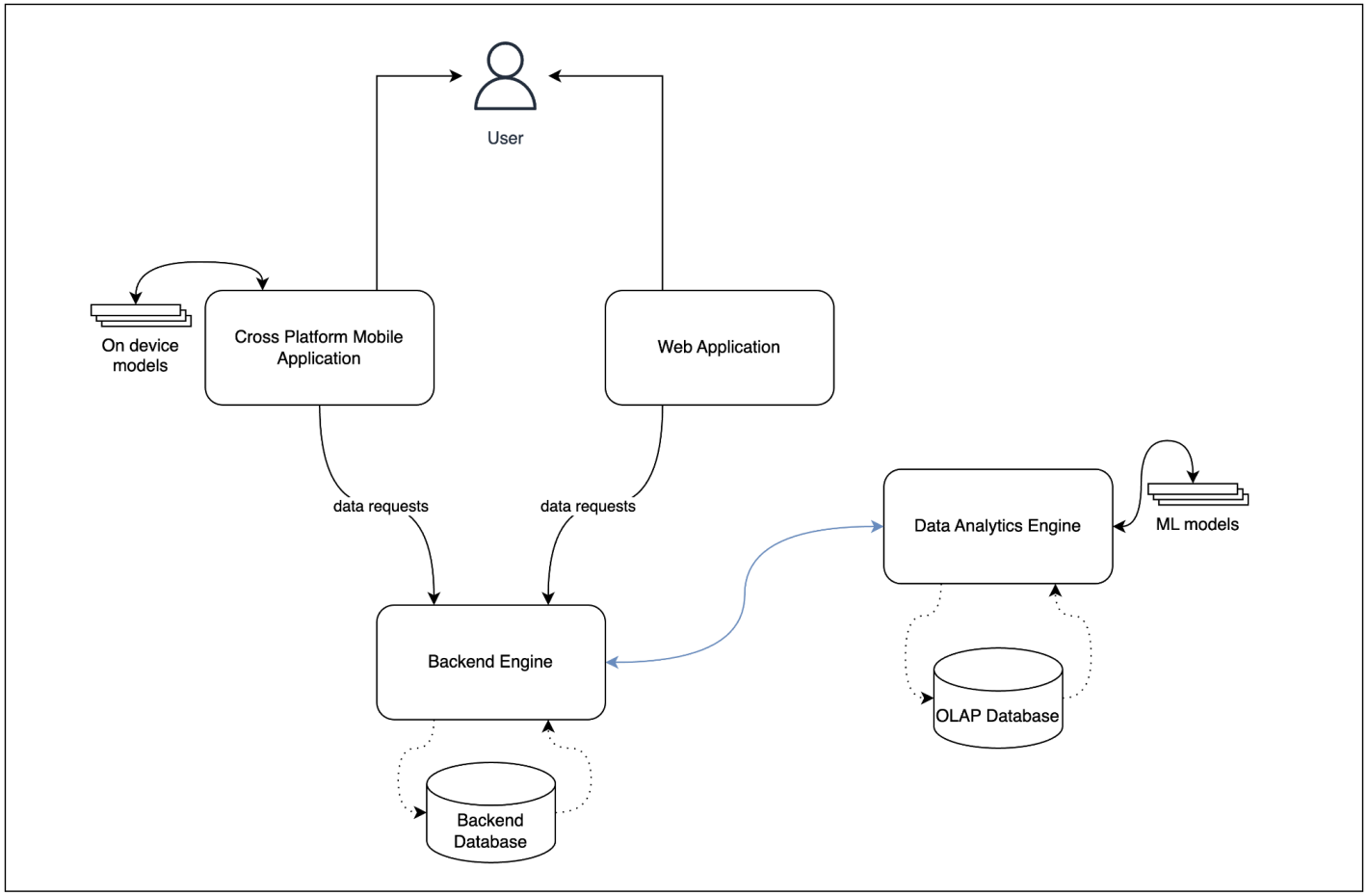
Additionally, the software system will make sure that it is built to be malleable, with new features being added regularly, without disrupting the existing functions. Additionally, support teams would work in shifts to ensure that any outlier software faults are caught in real time, and fixed promptly.

System Design

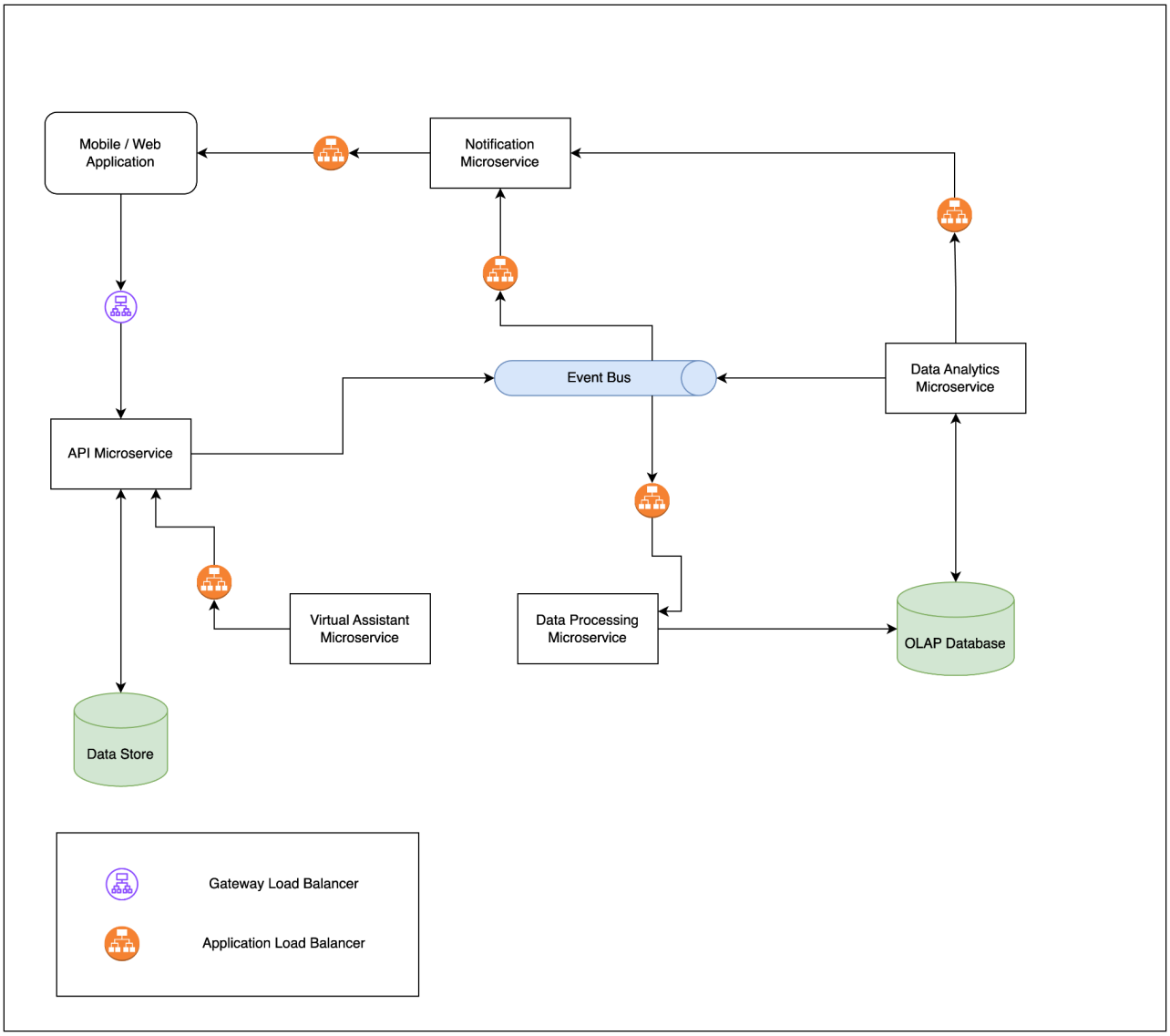
To build this system, the following architectural design patterns would be used

- Event-driven design pattern using message queues
- Microservice design pattern
- Replicated Load-Balanced Services (RLBS)

High Level System Diagram



Data Flow Diagram



System Components

Event Bus

The system uses a Messaging bus (such as Apache Kafka[1]) to handle real time events from the user applications. This ensures real time data processing, while being durable by distributing the storage of messages across replicated fault tolerant servers.

The messages are produced by the API Microservice, and by the Data Analytics Mocroservice. These messages are consumed by the Data Processing Microservice, and the Notification Microservice.

API Microservice

The API microservice receives requests over public endpoints through a Gateway Load Balancer (GWLB), and forwards them to the message queue. The communication protocol used would be HTTPS with the REST protocol, with secured authorization.

The use of a GWLB ensures that the non-functional requirements of security and scalability are met. It acts as a managed entry point that routes traffic to the correct destination. It also ensures intrusions from non-trusted sources.

The API Microservice provides the applications with consistent data updates and quick and efficient information retrieval from the data store

Data Processing Microservice

This microservice acts as a consumer of the event broker, to consume the messages, process, clean and store the processed in the real time database.

The microservice would perform basic data sanity and validation checks. The data output is stored into the OLAP database in a format that is quickly accessible by the data analytics engine. The microservice also ensures that the data is secure, and personal data is encrypted.

Data Analytics Microservice

The Data Analytics Microservice reads the pre-processed data from the OLAP database for trend analysis and stores processed multidimensional models back into the OLAP database(such as Amazon Redshift[2]).

The microservice would use descriptive analytics to summarize and visualize data to understand patterns and trends. Additionally, the microservice would use predictive analytics using historical data to make predictions about future outcomes, through techniques such as regression analysis, decision trees, and random forests can be used to predict health outcomes such as disease risk, medication adherence.

Virtual Assistant Microservice

This microservice powers the mobile and web applications with a powerful virtual assistant, with dynamic replies. The microservices uses the API microservice to provide the front-facing interfaces with endpoints to communicate with users applications.

Data Store

The data store is a NoSQL database (Couchbase[3]) to store health data provides a greater advantage as they are designed to be highly available, support a high write throughput and are optimized for low latency data accesses. Additionally, they are ideal for unstructured data, hence is the most rational choice for the application.

OLAP Database

The system integrates an Online Analytical Processing(OLAP) database (such as Amazon Redshift[2]) to store, manage and retrieve OLAP workloads. Additionally, the database can be seamlessly integrated with BI tools for advanced analytics and reporting.

Notification Microservice

The notification microservices consume messages from the event bus to provide alerts to the user. Additionally, it listens to requests from the data analytics microservice to provide personalized user health insight notifications.

Project Plan

Development Plan

The **Agile Development Plan** would be best for developing the AI-Enhanced Health Monitoring Platform. The plan would include the following characteristics.

- **Sprint Planning:** At the start of each sprint, the team picks tasks from the product backlog to work on. This covers creating new features, fixing bugs, and using feedback.
- **Daily Stand-ups:** Quick daily meetings help the team update each other on their progress, talk about any problems, and plan for the day ahead. This keeps everyone informed and helps solve issues quickly.
- **Sprint Reviews:** When a sprint ends, the team shows what they've done to stakeholders, including feedback from tests. This is important to make sure the product meets user needs and business objectives.
- **Sprint Retrospectives:** After showing the completed work, the team looks back at how the sprint went. They figure out what can be better and plan to make those improvements in the next sprint.
- **Feedback Loop**: An essential part of the Agile methodology is the continuous feedback loop from users, stakeholders, and team members. This feedback will be incorporated into the product backlog for future sprints, ensuring that the platform evolves in response to real-world use and feedback.

The team would use the following tools to ensure effective collaboration:

- **Project Management Tools**: Tools like Jira[4] or Trello[5] will be used for tracking tasks, sprints, and progress.
- **Communication Tools**: Slack[6] for daily communication, Zoom[7] for meetings, and Confluence[8] for documentation.
- **Version Control**: Git with GitHub or Bitbucket to manage code versions and collaborate on code development.

Milestones and Deliverables Detailed Breakdown

Requirement Gathering and Analysis (Weeks 1-2)
Goals: Understand user needs, technical requirements, and regulatory constraints. Finalize feature list and prioritize for development.
Deliverables: Requirements document, feature prioritization list.

System Architecture Design (Weeks 3-4)
Goals: Define the high-level architecture of the system, including how subsystems will interact, data flow, and technology stack.
Deliverables: System architecture diagrams, technology stack decisions.

Development Sprints (Weeks 5-20)
Goals: Develop, test, and refine features in alignment with sprint themes.
Deliverables: Working software increment at the end of each sprint, sprint review reports.

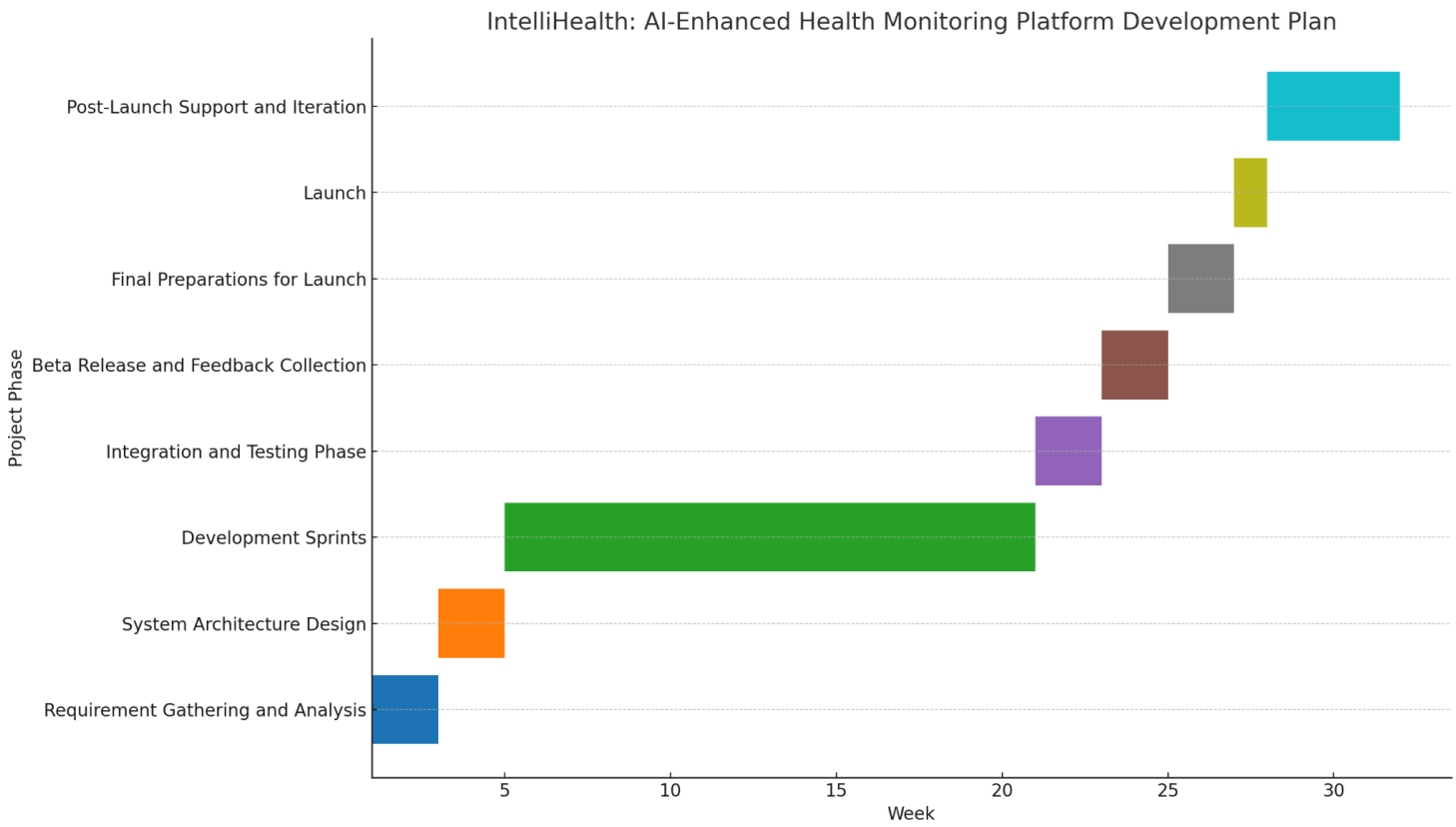
Integration and Testing Phase (Weeks 21-22)
Goals: Integrate subsystems, conduct comprehensive testing (including load testing and security testing), and prepare for beta release.
Deliverables: Integrated system, testing reports, beta release candidate.

Beta Release and Feedback Collection (Weeks 23-24)
Goals: Deploy beta release to a limited user group, collect feedback, and identify any issues or areas for improvement.
Deliverables: Beta release feedback report, list of issues and improvements.

Final Preparations for Launch (Weeks 25-26)
Goals: Implement feedback, finalize documentation, and prepare marketing materials. Conduct final security and performance checks.
Deliverables: Final product, user documentation, marketing materials.

Launch (Week 27)
Goals: Make the platform available to the general public, monitor system performance, and address any immediate issues.
Deliverables: Launch announcement, operational platform.

Post-Launch Support and Iteration (Weeks 28+)
Goals: Provide ongoing support, collect user feedback, and continue iterating on the product based on usage data and feedback.
Deliverables: Regular product updates, support documentation updates, continuous improvement



Resource Requirements

The project would require the following core development team.

- **Data Privacy and Security Experts**: Specialists who ensure the platform adheres to global data protection regulations (e.g., GDPR, HIPAA) and implement state-of-the-art security measures
- **Cloud Infrastructure Architects**: Experts in cloud computing architectures to design a scalable, resilient, and cost-efficient infrastructure on platforms on AWS[9]
- **AI and Machine Learning Engineers**: To refine and implement predictive models, and ensure continuous improvement of the health insights generated by the platform
- **User Experience (UX) Researchers**: To conduct ongoing research on user interaction, making the platform intuitive and accessible to all user demographics
- **Community Managers**: To build and maintain engagement with the platform's user community, facilitating feedback loops and support channels

To enable team training and development, the following additional resources would be utilized.

- **Professional Development**: Implement ongoing training programs for staff to keep up with the latest technologies, methodologies, and best practices in their respective fields

- **Cross-functional Workshops**: Foster collaboration and knowledge sharing across teams through regular workshops, ensuring a unified approach to the project's challenges

Additionally, there are technical and infrastructure resources that would help with the project development, as mentioned below

Technical Resources

- **Security Software**: Advanced tools for encryption, secure data transmission, and intrusion detection to safeguard user data

- **Analytics and BI Tools**: Software for data visualization and business intelligence to extract actionable insights from user data

- **Development and Productivity Tools**: Licenses for integrated development environments (IDEs), code repositories, and project management tools that facilitate Agile development practices

Infrastructure Requirements

- **High-Performance Computing Resources**: For data processing and analytics, including GPU-accelerated servers for machine learning model training

- **Robust Storage Solutions**: To securely store and manage the vast amounts of health data collected, including both structured and unstructured data formats

- **Networking Equipment**: High-speed networking gear to ensure seamless data transfer between the platform's components and the end-users with minimal latency

Risk Assessment

In order to ensure the successful development and implementation of the AI-enhanced health monitoring platform, the main goal of this risk assessment and mitigation plan is to identify potential obstacles and uncertainties related to the IntelliHealth project and to develop practical strategies to mitigate or minimize their impact.

Risk Identification

Technical Risks

- **Integration challenges with wearable devices:** Because of possible compatibility problems and technical complexity, integrating different wearables can be risky.
- **Complexity in developing AI algorithms:** There are technical complexities and difficulties involved in creating AI algorithms for individualized health advice.
- **Data accuracy concerns:** The platform's reputation and efficacy depend on the quality and dependability of the health insights it generates.

Operational Risks

- **Inadequate user adoption:** User adoption rates may be hampered by subpar user experience design.
- **Technical glitches:** System failures or platform crashes may leave users unhappy and erode their confidence in the system.
- **Insufficient training data for AI models:** The precision of health advice and forecasts may be jeopardized by insufficient training data for AI models.

Economic Risks

- **Budget constraints:** A platform's competitiveness and value proposition may be impacted by its inability to grow and expand due to a lack of funding.
- **Market fluctuations:** The demand for AI-powered health monitoring solutions may be impacted by shifts in market trends, which could jeopardize the project's sustainability.
- **Increased competition:** Increasing competition could put pressure on prices and hurt IntelliHealth's profitability in the healthcare technology sector.

Risk Impact Analysis

1. **Integration challenges with wearable devices**

➢ **Impact:** User frustration and decreased engagement.
Likelihood: Moderate
➢ **Severity:** High

2. **Complexity in developing AI algorithms**

➢ **Impact:** Developer's frustration and time consuming.
➢ **Likelihood:** Moderate
➢ **Severity:** Moderate

3. **Data accuracy concerns**

➢ **Impact:** Loss of credibility and improper health insights
➢ **Likelihood:** Low
➢ **Severity:** High

4. **Inadequate user adoption due to poor user experience design**

➢ **Impact:** Low user retention rates and negative reviews.
➢ **Likelihood:** High
➢ **Severity:** Medium

5. **Technical glitches**

➢ **Impact:** Undermined user satisfaction and experience
➢ **Likelihood:** Moderate
➢ **Severity:** Moderate

6. **Insufficient training data for AI models**

➢ **Impact**: Reduced accuracy of predictions
➢ **Likelihood**: Moderate
➢ **Severity**: High

7. **Budget constraints affecting scalability**

➢ **Impact:** Limitation of growth potential and feature enhancements.
➢ **Likelihood:** Low
➢ **Severity:** High

8. **Market fluctuations**

➢ **Impact**: Instability of Revenue and demand
➢ **Likelihood**: Moderate
➢ **Severity**: Moderate

9. **Increased competition**

➢ **Impact**: Reduced profitability and pricing
➢ **Likelihood**: High
➢ **Severity**: Low

Risk Mitigation Strategies

1. **Integration challenges with wearable devices**

To address integration issues promptly, conduct thorough compatibility testing and establish clear communication channels with manufacturers.

2. **Complexity in developing AI algorithms**

Implement Agile methodologies to streamline problem-solving within the development team.

3. **Data accuracy concerns**

To ensure integrity, implement rigorous data validation procedures and invest in data quality assurance tools.

4. **Inadequate user adoption due to poor user experience design**

Conduct extensive user testing to identify issues early by placing a high priority on user-centric design principles.

5. **Technical glitches**

Establish proactive maintenance protocols and robust monitoring systems to ensure stable performance.

6. **Insufficient training data for AI models**

Collaboration with healthcare institutions will help access larger datasets by leveraging data augmentation techniques.

7. **Budget constraints affecting scalability**

Seek strategic partnerships for resource expansion and prioritize investments in scalable infrastructure.

8. **Market fluctuations**

Adapt pricing strategies to changing market conditions by diversifying revenue streams.

9. **Increased competition**

Develop innovative and value-added services and invest in brand awareness campaigns to differentiate yourself from the competition.

Budgeting Plan

Budget Breakdown

The project consists of phases and milestones as described in the project plan. The risks have also been identified and an appropriate mitigation plan is established, and the project would be actively monitored to track and prevent the identified risks. Considering the solution, project plan as well as the potential risks associated with this project, the costs mainly fall into the following categories:

People Costs

These costs include the salaries for the following employees.

- ➔ Upper management team and department heads
- ➔ Core developer team
- ➔ QA and production support team
- ➔ Data scientists in the AI team,
- ➔ UX designers of the mobile and web applications
- ➔ Project and program managers
- ➔ HR Team

Equipment Costs

These costs are associated with the operating environment of the systems being developed. These are the components such as:

- ➔ Renting the cloud servers and compute engines from AWS[],
- ➔ The purchase of persistent cloud storage
- ➔ Networking equipment and Virtual Private Cloud resources
- ➔ Developer workstations, Mobile devices and other utilities

Service Costs

These costs include the following services used throughout the development process

- ➔ Cloud computing services from AWS
- ➔ Third-party APIs (such as the Google Maps API[])
- ➔ Software Licenses (Github Organisation for version control, Atlassian Jira & Confluence for product management and documentation)
- ➔ Consulting services

Overhead Costs

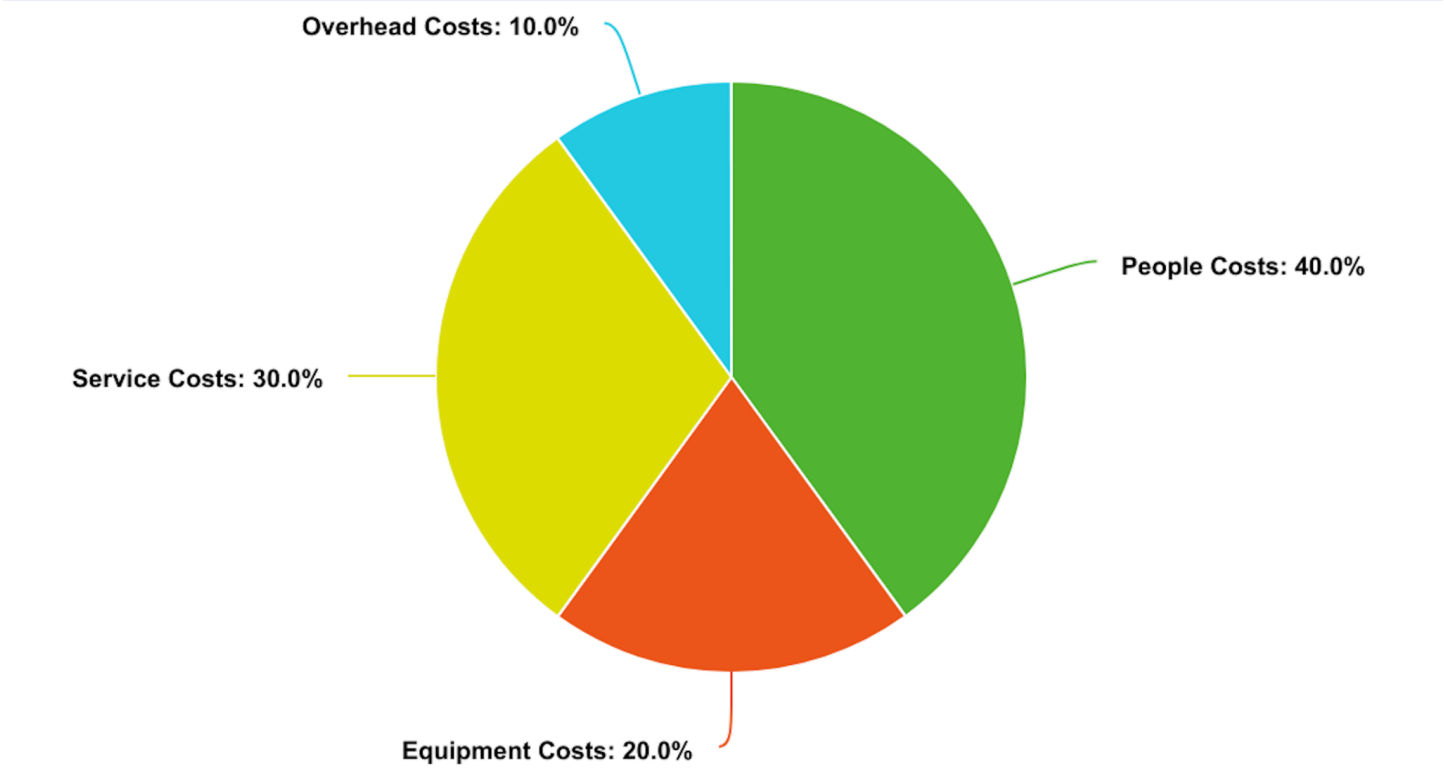
These are the miscellaneous costs of the software product development process. This is inclusive of unforeseen costs due to unexpected expenses, delays, or scope changes. Some of the costs are listed as follows:

- ➔ Office space, and furnishing
- ➔ Office utilities
- ➔ Insurance

- ➔ Administrative Expenses
- ➔ Contingency funds

Budget Allocation

Inorder to keep a fair split up across the different costs associated with the project, the total budget is allocated to each of the major sub costs as shown in the following pie chart:



People Costs: 40% of the total budget

Equipment Costs: 20% of the total budget

Service Costs: 30% of the total budget

Overhead Costs: 10% of the total budget

Using the above breakdown of costs, the finances of the project are as follows:

Category	**Budget Allocated**
People Costs	CA$400,000
Equipment Costs	CA$200,000
Services	CA$300,000
Contingency Reserve	CA$100,000
Overhead Costs	CA$100,000
	Total Budget: $1,100,000

Budget Monitoring and Control

It's important to establish a process to keep track and monitor project expenses. In order to ensure an effective budget management for the project success, the following plan would be adopted.

- ➢ **Regular Expense Reviews**

The upper management would schedule regular reviews on a monthly basis to ensure that the categories and their respective budget allocation is within the targets. Inorder to facilitate this meeting, expense tracking tools such as QuickBooks would be used.

- ➢ **Expense Approval Process**

In order to maintain a strict process, all expenses over a predetermined threshold would have to go through a robust approval process, with request tracking for all stakeholders.

- ➢ **Forecast Future Expenses**

The stakeholders would use historical data, and revised project plans to forecast and plan for future expenses. This would ensure that expenses are continuously evaluates and the budget targets are met.

References

[1] https://kafka.apache.org/

[2] https://aws.amazon.com/redshift/

[3] https://www.couchbase.com/

[4] https://www.atlassian.com/software/jira

[5] https://trello.com/

[6] https://slack.com/

[7] https://zoom.us/

[8] https://www.atlassian.com/software/confluence

[9] https://aws.amazon.com/