Software Engineering

Name: Minh Nga Nguyen

Introduction

Subsystem breakdown

An information system for internal use in prisons will consist of the following subsystems:
1. Prisoner Management System: handles all information related to prisoners, including personal details, conviction data, behaviour records, work and study programs, visitor logs, and health records. It would also manage temporary leaves (holidays), court appearances, and release dates.

2. Staff Management System: focuses on prison staff, maintaining records of their qualifications, length of service, work schedules, access rights, and any restrictions regarding their interaction with prisoners. It could also include performance evaluation and training records.

3. Security and Access Control System: manages access to various parts of the prison and information within the system. It includes security protocols for physical access to areas within the prison and digital access to sensitive data, ensuring that users can only view or edit information relevant to their role.

4. Health Services Management: manages prisoners' health records, including general health checks, dental treatments, illnesses, and treatments. It facilitates the exchange of health-related information with external health services.

5. External Interfaces and Integration System: ensures seamless data exchange between the prison's internal systems and external information systems (e.g., VRK, Risen, OKM), including the police and health services. It manages data import/export and ensures data integrity and security during these processes.

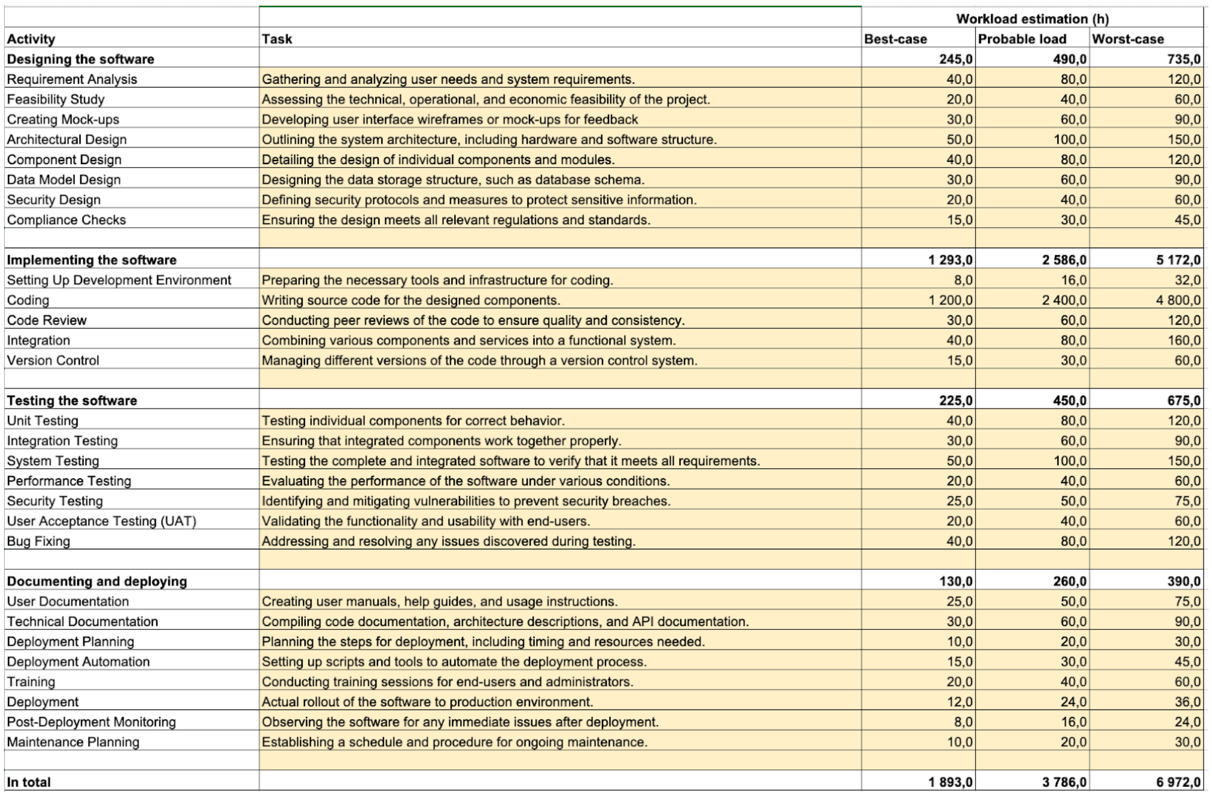
6. Facility and Logistics Management: manages the physical aspects of the prison, including maintenance schedules, cleaning rosters, and inventory management for supplies needed by both inmates and staff.

7. Reporting and Analytics: A system-wide support subsystem that provides tools for generating reports and analytics across all other subsystems, aiding in decision-making and operational optimization.

Target for Design

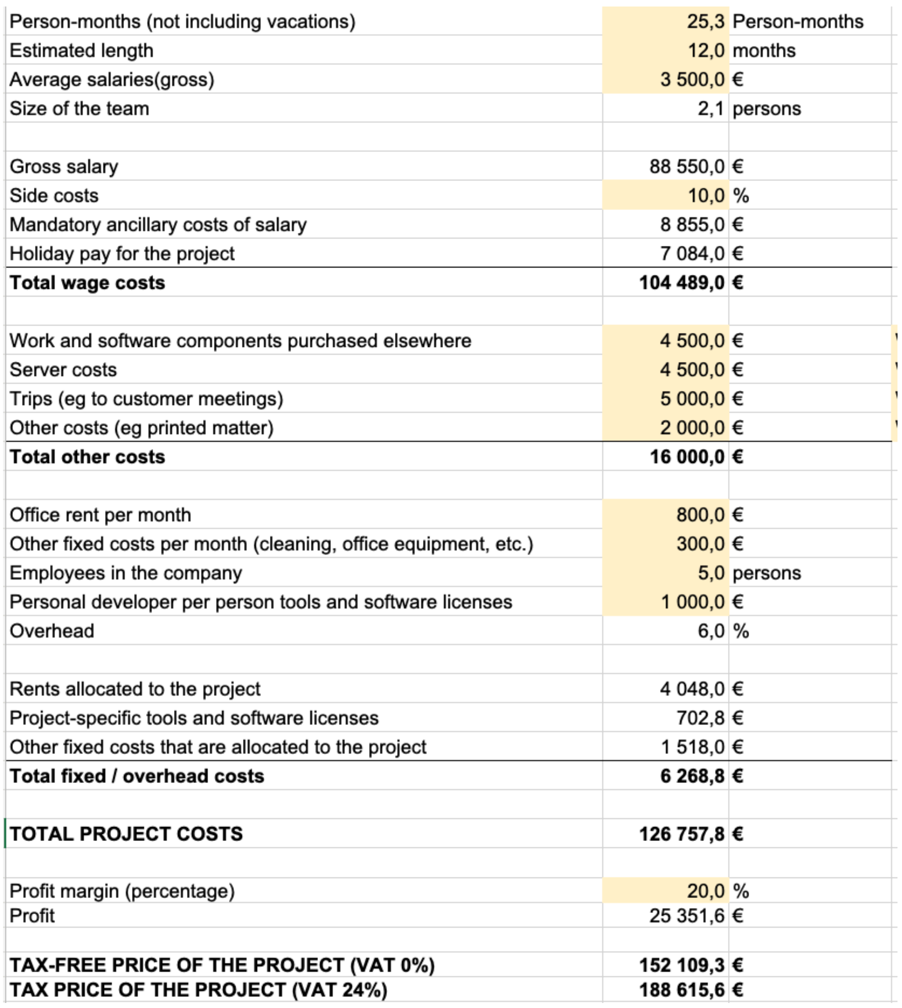
I choose Health Services Management as the target for this plan because in Finland, where happiness, well-being, and equality are highly valued. Prisoners deserve good treatment and access to healthcare just like anyone else. It's part of making sure that everyone, no matter their situation, is treated fairly and has a chance to get better, both mentally and physically.

Budget estimate



		Workload estimation (h)		
Activity	**Task**	**Best-case**	**Probable load**	**Worst-case**
Designing the software		**245,0**	**490,0**	**735,0**
Requirement Analysis	Gathering and analyzing user needs and system requirements.	40,0	80,0	120,0
Feasibility Study	Assessing the technical, operational, and economic feasibility of the project.	20,0	40,0	60,0
Creating Mock-ups	Developing user interface wireframes or mock-ups for feedback	30,0	60,0	90,0
Architectural Design	Outlining the system architecture, including hardware and software structure.	50,0	100,0	150,0
Component Design	Detailing the design of individual components and modules.	40,0	80,0	120,0
Data Model Design	Designing the data storage structure, such as database schema.	30,0	60,0	90,0
Security Design	Defining security protocols and measures to protect sensitive information.	20,0	40,0	60,0
Compliance Checks	Ensuring the design meets all relevant regulations and standards.	15,0	30,0	45,0
Implementing the software		**1 293,0**	**2 586,0**	**5 172,0**
Setting Up Development Environment	Preparing the necessary tools and infrastructure for coding.	8,0	16,0	32,0
Coding	Writing source code for the designed components.	1 200,0	2 400,0	4 800,0
Code Review	Conducting peer reviews of the code to ensure quality and consistency.	30,0	60,0	120,0
Integration	Combining various components and services into a functional system.	40,0	80,0	160,0
Version Control	Managing different versions of the code through a version control system.	15,0	30,0	60,0
Testing the software		**225,0**	**450,0**	**675,0**
Unit Testing	Testing individual components for correct behavior.	40,0	80,0	120,0
Integration Testing	Ensuring that integrated components work together properly.	30,0	60,0	90,0
System Testing	Testing the complete and integrated software to verify that it meets all requirements.	50,0	100,0	150,0
Performance Testing	Evaluating the performance of the software under various conditions.	20,0	40,0	60,0
Security Testing	Identifying and mitigating vulnerabilities to prevent security breaches.	25,0	50,0	75,0
User Acceptance Testing (UAT)	Validating the functionality and usability with end-users.	20,0	40,0	60,0
Bug Fixing	Addressing and resolving any issues discovered during testing.	40,0	80,0	120,0
Documenting and deploying		**130,0**	**260,0**	**390,0**
User Documentation	Creating user manuals, help guides, and usage instructions.	25,0	50,0	75,0
Technical Documentation	Compiling code documentation, architecture descriptions, and API documentation.	30,0	60,0	90,0
Deployment Planning	Planning the steps for deployment, including timing and resources needed.	10,0	20,0	30,0
Deployment Automation	Setting up scripts and tools to automate the deployment process.	15,0	30,0	45,0
Training	Conducting training sessions for end-users and administrators.	20,0	40,0	60,0
Deployment	Actual rollout of the software to production environment.	12,0	24,0	36,0
Post-Deployment Monitoring	Observing the software for any immediate issues after deployment.	8,0	16,0	24,0
Maintenance Planning	Establishing a schedule and procedure for ongoing maintenance.	10,0	20,0	30,0
In total		**1 893,0**	**3 786,0**	**6 972,0**

For estimating the budget, I started by breaking down the activities into smaller tasks and then figuring out how many hours each one might take, leaning on the Pareto principle. This principle suggests that a big chunk of work, like 70% to 80%, usually comes from just a few key tasks – in this case, it's writing the source code and actually getting the software to work. I thought about the best case first, like if everything went super smooth and without any problems. Then, to get a more realistic estimate, I doubled the best case hours. For the worst case, where everything that can go wrong does, I tripled the best case number, just to be on the safe side. This way, I've got a range that should cover the best scenario and the worst scenario.

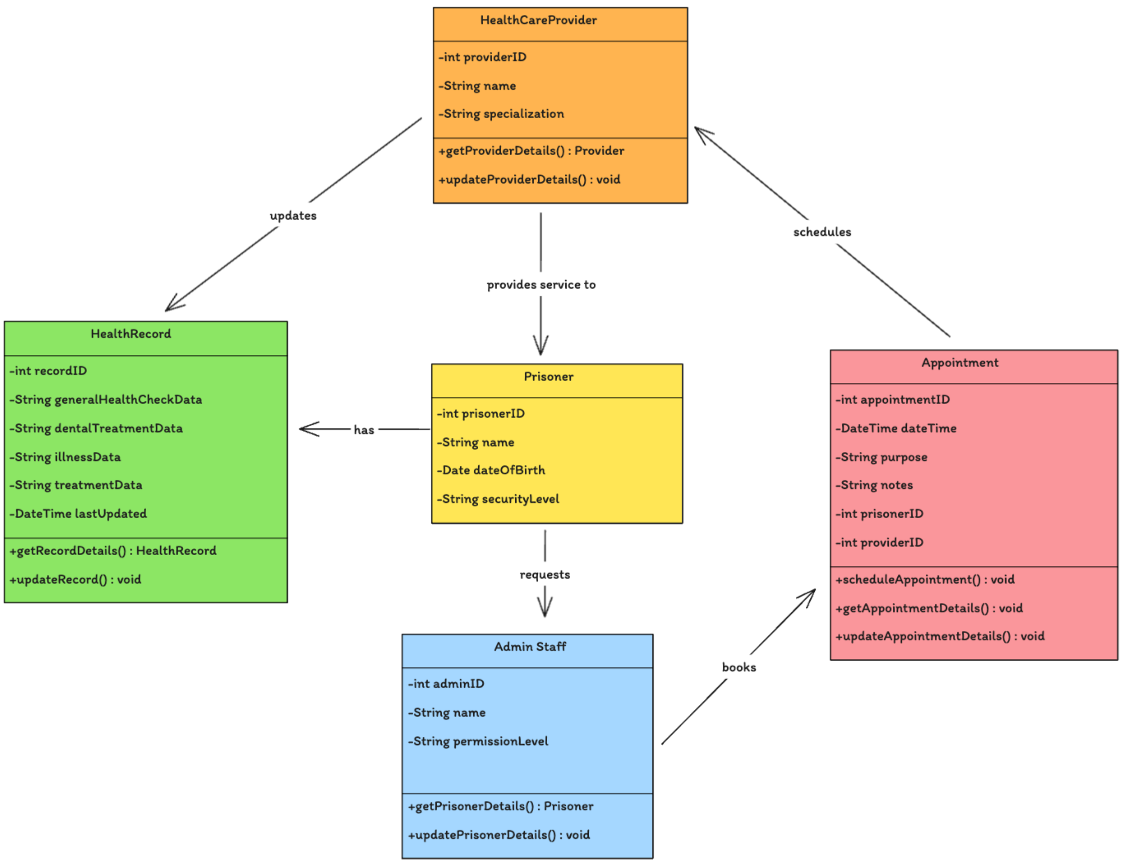


Person-months (not including vacations)	25,3	Person-months
Estimated length	12,0	months
Average salaries(gross)	3 500,0	€
Size of the team	2,1	persons
Gross salary	88 550,0	€
Side costs	10,0	%
Mandatory ancillary costs of salary	8 855,0	€
Holiday pay for the project	7 084,0	€
Total wage costs	**104 489,0**	**€**
Work and software components purchased elsewhere	4 500,0	€
Server costs	4 500,0	€
Trips (eg to customer meetings)	5 000,0	€
Other costs (eg printed matter)	2 000,0	€
Total other costs	**16 000,0**	**€**
Office rent per month	800,0	€
Other fixed costs per month (cleaning, office equipment, etc.)	300,0	€
Employees in the company	5,0	persons
Personal developer per person tools and software licenses	1 000,0	€
Overhead	6,0	%
Rents allocated to the project	4 048,0	€
Project-specific tools and software licenses	702,8	€
Other fixed costs that are allocated to the project	1 518,0	€
Total fixed / overhead costs	**6 268,8**	**€**
TOTAL PROJECT COSTS	**126 757,8**	**€**
Profit margin (percentage)	20,0	%
Profit	25 351,6	€
TAX-FREE PRICE OF THE PROJECT (VAT 0%)	**152 109,3**	**€**
TAX PRICE OF THE PROJECT (VAT 24%)	**188 615,6**	**€**

After having the person months, I estimated the cost for wage, software, servers, and other costs. I based this logic on my knowledge on software development cost.

High level UML plan

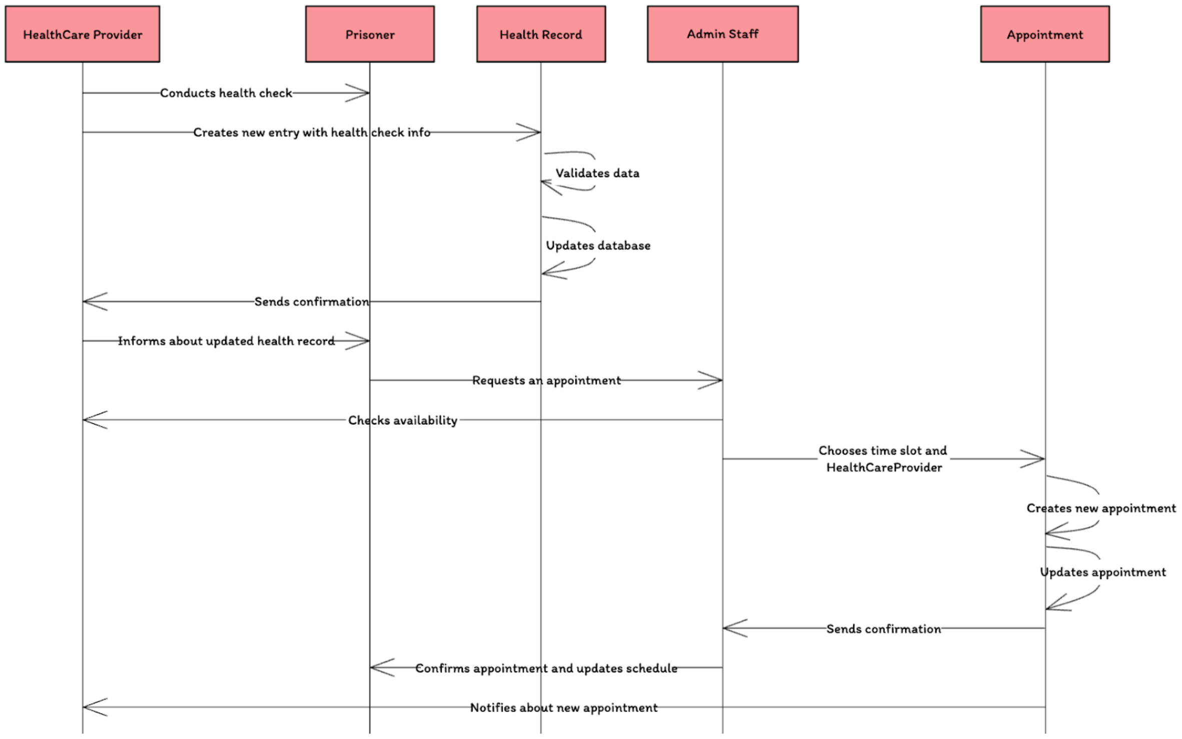
Class diagram



In this class diagram,

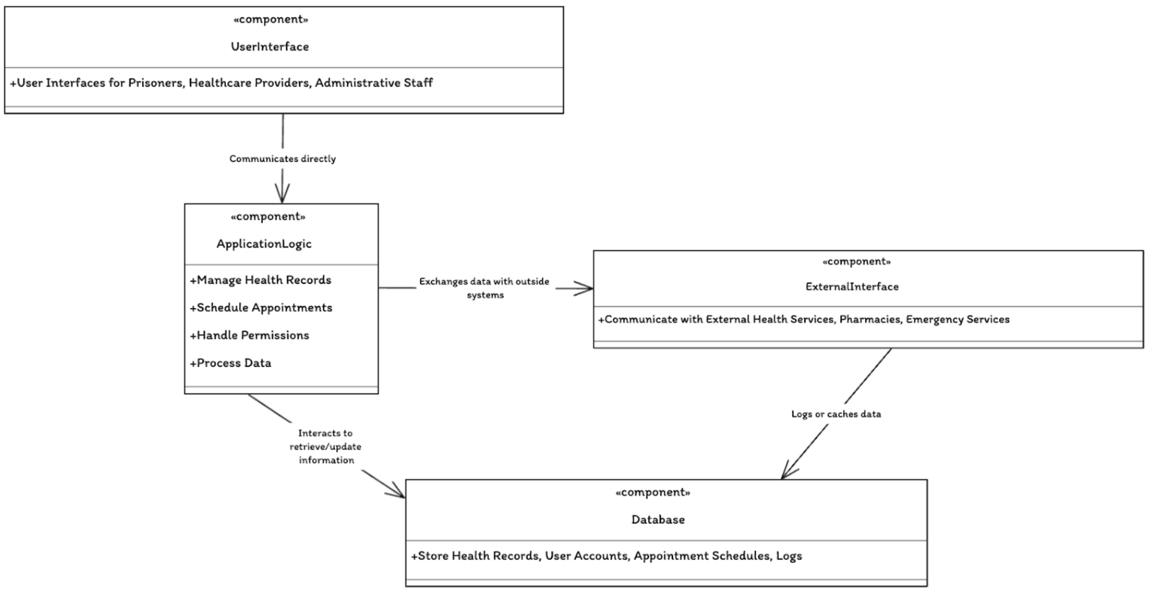
  - The Prisoner class represents a prisoner and would include personal details and security level
- The HealthRecord class is linked to the Prisoner class and contains detailed health records for each prisoner, including various types of health data and the date it was last updated.
- The HealthCareProvider class includes details about medical staff or external health care providers.
- The Appointment class would manage the scheduling and details of health-related appointments for prisoners.
- The Admin Staff class is the prison staff who manages and takes care of the prisoner. When the prisoner needs a health care check up as requested by health care provider or by the prisoner, the admin staff will book an appointment with health care service providers.

Sequence diagram of key activities



1. Adding a New Health Record
   - The HealthCareProvider initiates the process by conducting a health check on the Prisoner.
   - The HealthCareProvider collects health check information.
   - The HealthCareProvider interacts with the HealthRecord system, creating a new entry.
   - The HealthRecord system validates the provided data.
   - The HealthRecord system updates the database with the new health record.
   - The HealthRecord system sends a confirmation back to the HealthCareProvider.
   - The HealthCareProvider informs the Prisoner that the health record has been updated.
2. Scheduling an Appointment
   - The Prisoner requests an appointment.
   - The AdministrativeStaff checks the availability of HealthCareProviders.
   - The AdministrativeStaff chooses an available time slot and HealthCareProvider.
   - The Appointment system creates a new appointment entry.
   - The Appointment system updates the database with the appointment details.
   - The Appointment system sends a confirmation to the AdministrativeStaff.
   - The AdministrativeStaff confirms the appointment with the Prisoner and updates their schedule.
   - A notification is sent to the relevant HealthCareProvider about the new appointment.

Component Diagram


«component»
UserInterface
+User Interfaces for Prisoners, Healthcare Providers, Administrative Staff
Communicates directly
«component»
ApplicationLogic
+Manage Health Records
+Schedule Appointments
+Handle Permissions
+Process Data
Exchanges data with outside systems
«component»
ExternalInterface
+Communicate with External Health Services, Pharmacies, Emergency Services
Logs or caches data
Interacts to retrieve/update information
«component»
Database
+Store Health Records, User Accounts, Appointment Schedules, Logs

Components:

- User Interface (UI): This is the front-end component through which users interact with the system. It would include separate interfaces for different types of users, such as prisoners, healthcare providers, and administrative staff.
- Application Logic: This is the backend component where the main processing happens. It includes modules for managing health records, scheduling appointments, handling permissions, and processing data.
- Database: This stores all persistent data, including health records, user accounts, appointment schedules, and logs.
- External Interface: Components that manage communication with external systems like other health services, pharmacies, or emergency services.

Relationships:

- The User Interface communicates directly with the Application Logic, sending and receiving data based on user actions.
- The Application Logic interacts with the Database to retrieve or update information as needed.
- The Application Logic also communicates with External Interfaces to exchange data with outside systems.
- External Interfaces may interact with the Database for logging purposes or to cache data received from external sources.

User stories

Key user stories

1. As a healthcare provider, I want to access a prisoner's health record, so I can review their medical history before a consultation.
2. As an administrative staff member, I want to schedule health check appointments for prisoners, so they receive timely medical evaluations.
3. As a healthcare provider, I want to update a prisoner's health record after a consultation, so the record reflects the most current information.
4. As a prisoner, I want to request a healthcare appointment, so I can receive necessary medical treatment.
5. As an administrative staff member, I want to view and manage the schedule of healthcare providers, so I can efficiently allocate their time.

Sprints structure

Each sprint is 2 weeks

Sprint 1 - User Story 1
- Design the UI for healthcare providers to view health records.
- Implement backend logic for retrieving health records.
- Test the health record view feature.

Sprint 2 - User Story 2
- Design the appointment scheduling interface for administrative staff.
- Develop the appointment scheduling logic in the backend.
- Test the scheduling feature.

Sprint 3 - User Story 3
- Create UI for updating health records.
- Implement update functionality in the backend.
- Test the record update process.

Sprint 4 - User Story 4
- Design the UI for prisoners to request healthcare appointments.
- Develop the logic for handling and tracking appointment requests.
- Test the appointment request feature.

Sprint 5 - User Story 5
- Create UI for managing appointments.
- Implement update functionality in the backend.
- Test the appointment update process.

Sprint plan

Key Development Principles

- Agile Methodology: Embracing agile principles, the team will work in short sprints, focusing on delivering working software quickly and iterating based on feedback.
- User-Centric Design: Prioritizing user experience by involving healthcare providers and administrative staff in the design process to ensure the system meets their needs.
- Security and Privacy by Design: Given the sensitive nature of health records, incorporating strong security and privacy measures from the outset.
- Modularity: Building the system with modular components to facilitate easier updates and maintenance.

Justification

The focus of the first two sprints is to establish core functionalities of the Health Services Management system, which are accessing prisoner health records and scheduling health evaluations. The decision to prioritize these features and adhere to the outlined principles stems from the need to quickly establish a foundation for the system that addresses critical user needs while ensuring the security and privacy of sensitive health information. Engaging users early in the development process and adopting an agile methodology allows for iterative improvements, ensuring the system evolves in alignment with user expectations and project objectives.

Sprint 1: Accessing Prisoner Health Records

Sprint Goal

Enable healthcare providers to securely access prisoners' health records for consultation preparation.

Sprint Backlog

User Interface Design for Health Records
Task: Design a user-friendly interface for healthcare providers to view health records.
Duration: 3 days
Actor: UI/UX Designer

Backend Development for Health Record Retrieval
Task: Develop backend services to retrieve health records from the database.
Duration: 4 days

Actor: Backend Developer

Integration of UI with Backend Services
Task: Integrate the health record viewing interface with backend services.
Duration: 2 days
Actor: Frontend Developer

Security Measures Implementation
Task: Implement security measures for accessing health records, including authentication and authorization.
Duration: 3 days
Actor: Security Specialist

Testing and Feedback Collection
Task: Conduct testing with healthcare providers and collect feedback.
Duration: 2 days
Actor: QA Engineer, Healthcare Providers

Sprint 2: Scheduling Health Check Appointments

Sprint Goal

Equip administrative staff with tools to efficiently schedule health check appointments for prisoners, ensuring timely medical evaluations.

Sprint Backlog

Appointment Scheduling Interface Design
Task: Design the appointment scheduling interface for administrative staff.
Duration: 3 days
Actor: UI/UX Designer

Appointment Management Backend Development
Task: Develop backend logic for creating, updating, and cancelling appointments.
Duration: 4 days
Actor: Backend Developer

Integration of Scheduling Interface with Backend
Task: Ensure the scheduling interface communicates effectively with the backend.
Duration: 2 days
Actor: Frontend Developer

Implement Notification System
Task: Develop a system to notify healthcare providers and prisoners of scheduled appointments.
Duration: 3 days

Actor: Backend Developer

Testing and Adjustments

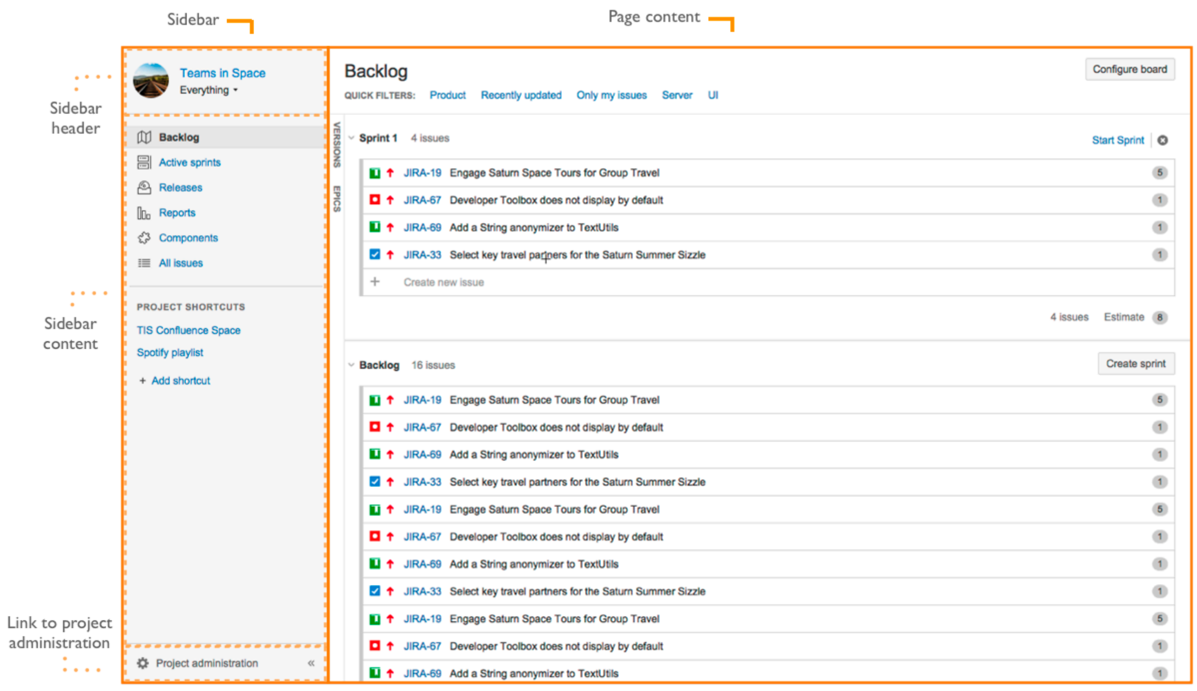
Task: Test the scheduling system and make necessary adjustments based on feedback.
Duration: 3 days
Actor: QA Engineer, Administrative Staff

Working environment

The working environment for developing a Health Services Management system is centered around collaboration, efficiency, and security. The tools chosen for task tracking, code versioning, project publishing, and communication are important in creating a cohesive and productive development process.

Tools

Task Tracking: Jira
Used by: Project Managers, Developers, QA Engineers, Designers
Purpose: To manage tasks, sprints, backlogs, and workflows in an agile development process.
Justification: Jira is widely recognized for its agile project management capabilities, including customizable boards for Scrum and Kanban. It supports detailed reporting, which is crucial for tracking progress and productivity.



Backlog

Sprint 1 4 issues

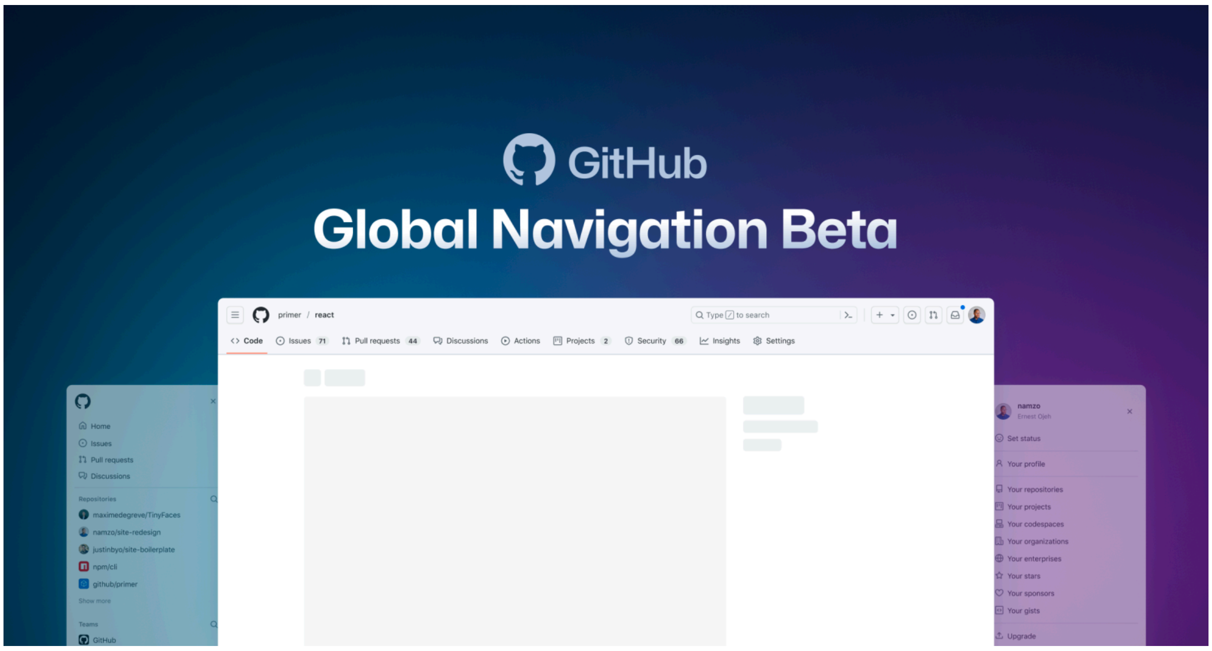
JIRA-19	Engage Saturn Space Tours for Group Travel	5
JIRA-67	Developer Toolbox does not display by default	1
JIRA-69	Add a String anonymizer to TextUtils	1
JIRA-33	Select key travel partners for the Saturn Summer Sizzle	1

4 issues Estimate 8

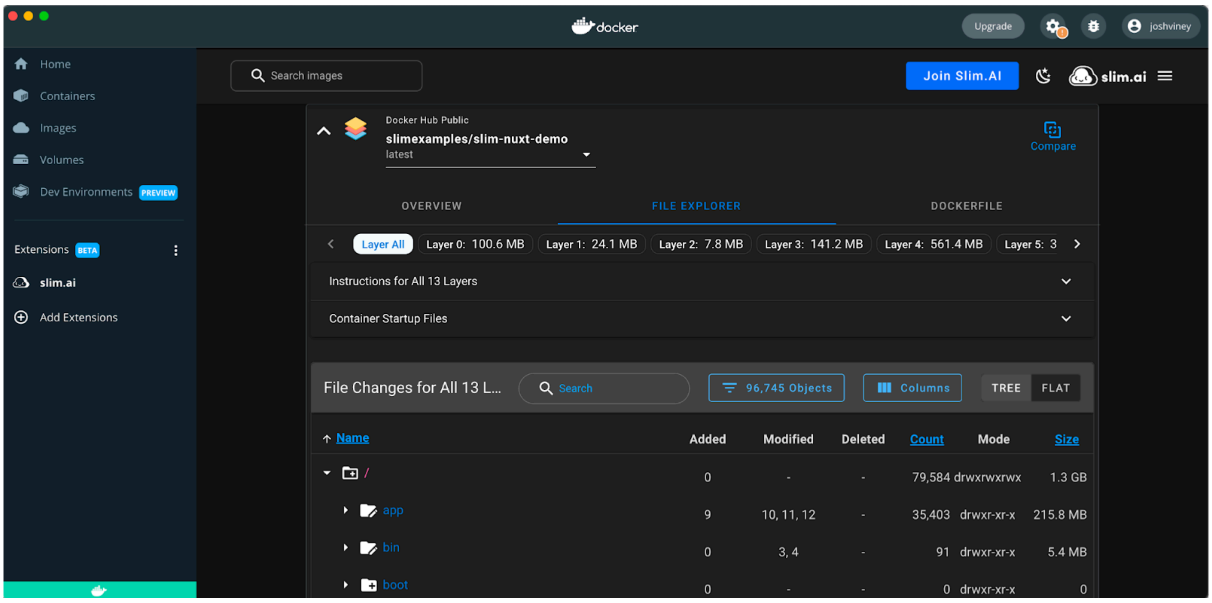
Backlog 16 issues

JIRA-19	Engage Saturn Space Tours for Group Travel	5
JIRA-67	Developer Toolbox does not display by default	1
JIRA-69	Add a String anonymizer to TextUtils	1
JIRA-33	Select key travel partners for the Saturn Summer Sizzle	1
JIRA-19	Engage Saturn Space Tours for Group Travel	5
JIRA-67	Developer Toolbox does not display by default	1
JIRA-69	Add a String anonymizer to TextUtils	1
JIRA-33	Select key travel partners for the Saturn Summer Sizzle	1
JIRA-19	Engage Saturn Space Tours for Group Travel	5
JIRA-67	Developer Toolbox does not display by default	1
JIRA-69	Add a String anonymizer to TextUtils	1

Code Versioning: Git with GitHub
Used by: Developers, QA Engineers
Purpose: For version control of codebase, collaboration among developers, and code review processes.
Justification: Git is the standard for version control, offering robust branching and merging capabilities. GitHub provides a collaborative platform with features like pull requests and issue tracking, fostering open communication and peer review.



Project Publishing: Docker and Kubernetes
Used by: Developers, DevOps
Purpose: To containerize the application for consistent deployment across environments and manage container orchestration.
Justification: Docker simplifies configuration and deployment, ensuring the application runs the same in every environment. Kubernetes allows for scalable and automated deployment, scaling, and management of containerized applications, essential for maintaining high availability.



Docker Hub Public
slimexamples/slim-nuxt-demo
latest

OVERVIEW | FILE EXPLORER | DOCKERFILE

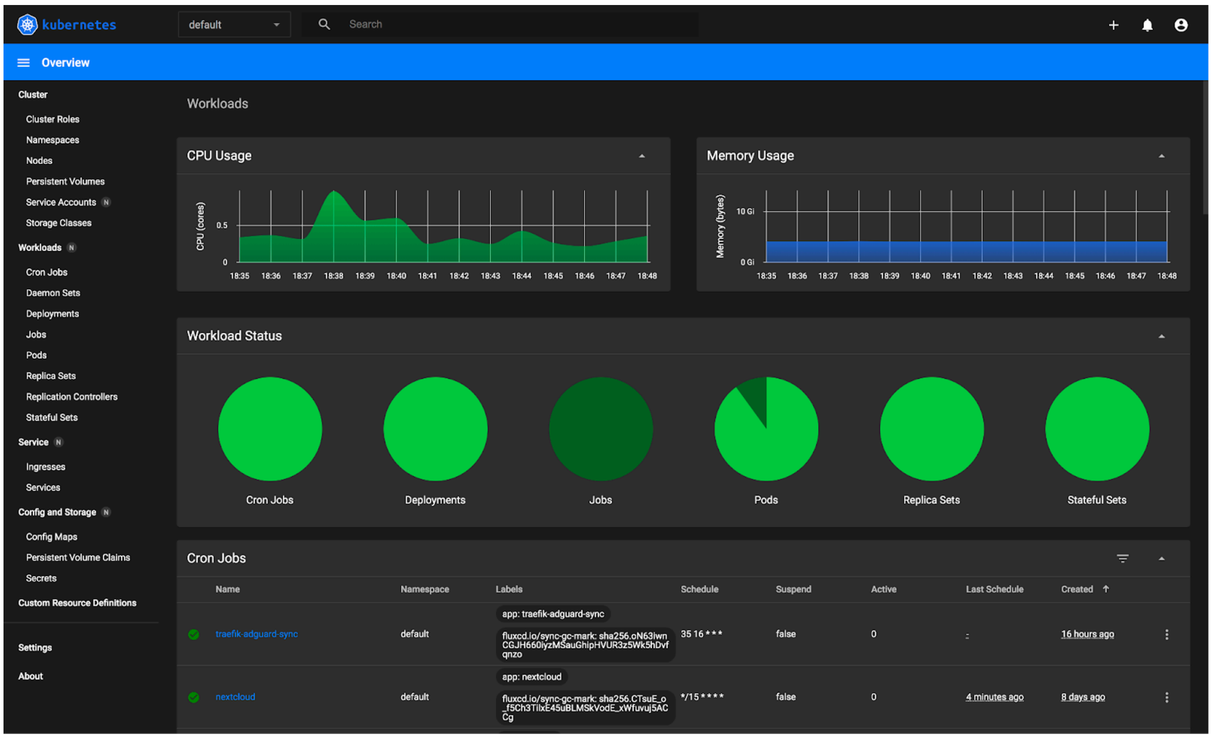
Layer All | Layer 0: 100.6 MB | Layer 1: 24.1 MB | Layer 2: 7.8 MB | Layer 3: 141.2 MB | Layer 4: 561.4 MB | Layer 5: 3

Instructions for All 13 Layers

Container Startup Files

File Changes for All 13 L... | 96,745 Objects

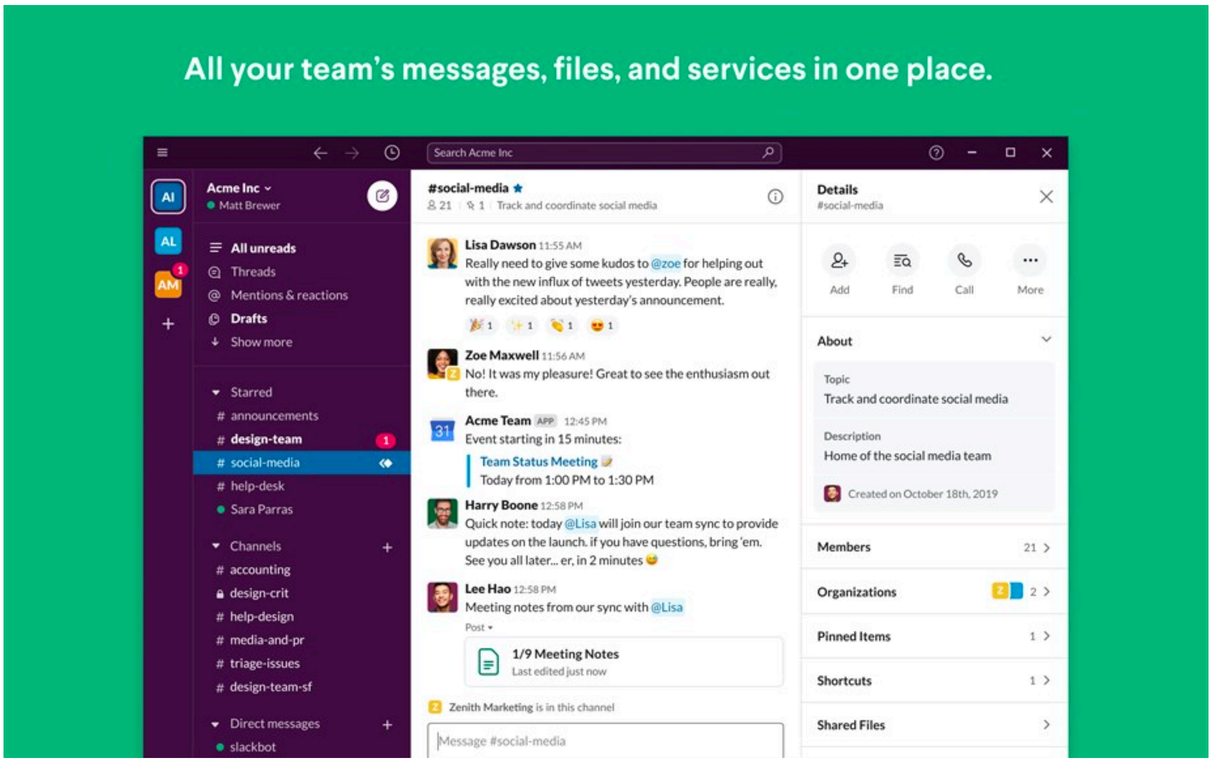
Name	Added	Modified	Deleted	Count	Mode	Size
/	0	-	-	79,584	drwxrwxrwx	1.3 GB
app	9	10, 11, 12	-	35,403	drwxr-xr-x	215.8 MB
bin	0	3, 4	-	91	drwxr-xr-x	5.4 MB
boot	0	-	-	0	drwxr-xr-x	0



Cron Jobs

Name	Namespace	Labels	Schedule	Suspend	Active	Last Schedule	Created
traefik-adguard-sync	default	app: traefik-adguard-sync; fluxcd.io/sync-gc-mark: sha256.oN63iwnCGJH660iyzMSauGhipHVUR3z5Wk5hDvfqnzo	35 16 * * *	false	0	-	16 hours ago
nextcloud	default	app: nextcloud; fluxcd.io/sync-gc-mark: sha256.CTsuE_o_f5Ch3TilxE45uBLMSkVodE_xWfuvuj5ACCg	*/15 * * * *	false	0	4 minutes ago	8 days ago

Communication: Slack
Used by: Everyone on the Project Team
Purpose: For day-to-day communication, updates, and team collaboration.
Justification: Slack is a powerful tool for team communication offering direct messaging, channels (for topics, teams, or projects), integration with other tools (e.g., Jira, GitHub), and file sharing, making it an all-in-one communication platform.


All your team's messages, files, and services in one place.

Lisa Dawson 11:55 AM
Really need to give some kudos to @zoe for helping out with the new influx of tweets yesterday. People are really, really excited about yesterday's announcement.

Zoe Maxwell 11:56 AM
No! It was my pleasure! Great to see the enthusiasm out there.

Acme Team APP 12:45 PM
Event starting in 15 minutes:
Team Status Meeting
Today from 1:00 PM to 1:30 PM

Harry Boone 12:58 PM
Quick note: today @Lisa will join our team sync to provide updates on the launch. if you have questions, bring 'em. See you all later... er, in 2 minutes

Lee Hao 12:58 PM
Meeting notes from our sync with @Lisa
Post
1/9 Meeting Notes
Last edited just now

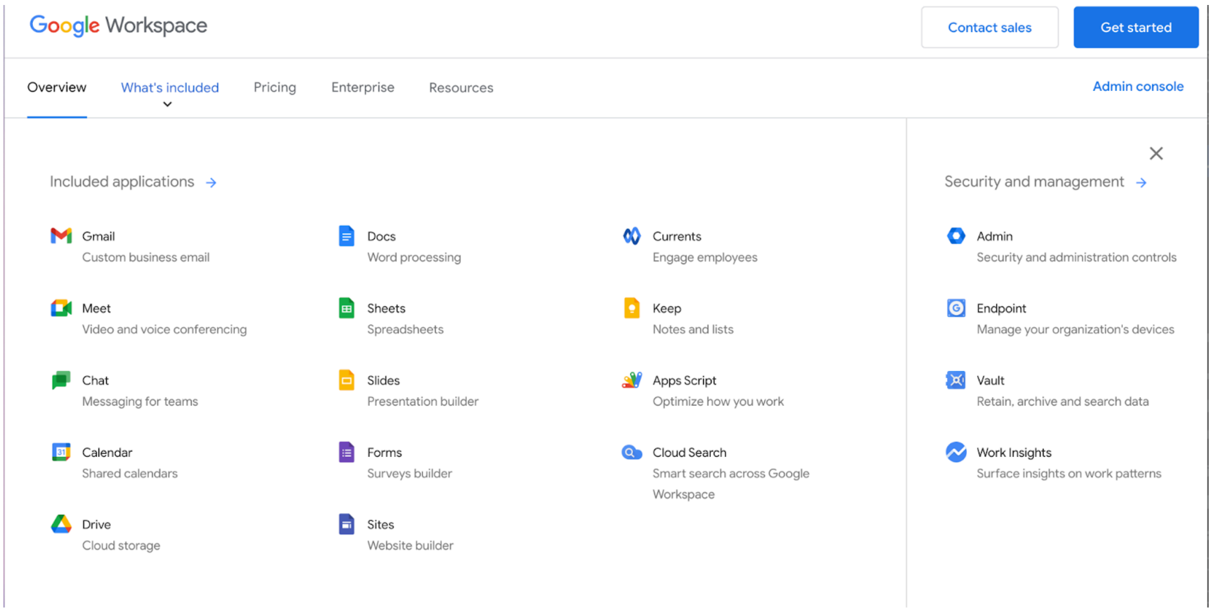
Zenith Marketing is in this channel

Topic
Track and coordinate social media

Description
Home of the social media team

Created on October 18th, 2019

Document Sharing and Collaboration: Google Workspace
Used by: Everyone on the Project Team
Purpose: To share documents, spreadsheets, and presentations; collaborate in real-time; and schedule meetings.
Justification: Google Workspace facilitates easy collaboration with its suite of applications like Docs, Sheets, and Slides. It's cloud-based, allowing team members to work from anywhere and share files securely.



Included applications

Gmail	Custom business email
Meet	Video and voice conferencing
Chat	Messaging for teams
Calendar	Shared calendars
Drive	Cloud storage
Docs	Word processing
Sheets	Spreadsheets
Slides	Presentation builder
Forms	Surveys builder
Sites	Website builder
Currents	Engage employees
Keep	Notes and lists
Apps Script	Optimize how you work
Cloud Search	Smart search across Google Workspace

Security and management

Admin	Security and administration controls
Endpoint	Manage your organization's devices
Vault	Retain, archive and search data
Work Insights	Surface insights on work patterns

Environment Setup and Tool Integration

The development environment will be cloud-based to ensure accessibility and flexibility for the team, who may be distributed geographically. Virtual machines (VMs) or cloud workstations can be used for coding and testing, equipped with development software and tools like IDEs (Integrated Development Environments), Postman (for API testing), and database management tools.

Integration between these tools is key. For instance, GitHub can be integrated with Jira to link commits and pull requests to tasks, Slack can be integrated with Jira and GitHub to provide real-time updates on issues and code changes, and Kubernetes deployments can be triggered from successful GitHub actions.

Time and Skills Needed

Setup Time: Setting up this environment, including account creation, configuration, and integrations, could take approximately 1-2 weeks, depending on the complexity of the setup and integrations.
Skills Needed: Knowledge of agile project management for Jira, familiarity with Git and GitHub for version control, experience with Docker and Kubernetes for deployment, and proficiency with Slack and Google Workspace for communication and collaboration. Training may be required for team members unfamiliar with these tools.
Choosing these tools and setting up an integrated development environment is justified by the need for efficient collaboration, secure and scalable project deployment, and agile project management to adapt to changing requirements and feedback during the project lifecycle.