CS 25-328 - Tenant Self-Service Portal

Project Proposal

Prepared for

Mahesh Nair

Capital One

By

Alan Dorn, Austin Glass, Prakhar Mathur, Krish Patel

Under the supervision of

Preetam Ghosh

10/11/24

Executive Summary

The Tenant Self-Service Portal project aims to streamline and automate the onboarding and dataset management processes for platform tenants at Capital One. Currently, these processes require manual intervention from platform engineers, leading to inefficiencies, delays, and repeated back-and-forth communications. Our goal is to simplify this workflow by developing a self-service web portal that allows clients to manage their datasets and use cases independently, reducing dependency on engineers and improving overall operational efficiency.

The proposed solution involves building a multi-tenant platform where users (Clients) can perform tasks such as dataset registration, data mapping creation, batch process tracking, and document lifecycle management. The portal will also include an approval system where Admins can review and provide feedback on submitted datasets and use cases. This portal will enhance user experience by allowing users to save progress during dataset creation, view approval status, and make necessary updates based on feedback.

Key project objectives include:

- Developing a self-service portal for clients to manage datasets and use cases.
- Enabling an approval workflow for dataset submissions, with Admin feedback loops.
- Building a user-friendly interface that supports navigation, progress-saving, and dataset updates.
- Ensuring the solution is scalable and secure, adhering to multi-tenant platform management principles.

The project will be implemented using open-source UI stacks, with room to explore AWS tools, and databases such as MongoDB. The focus will first be on building the Dataset Registry, followed by the Use Case Registry. This project presents a significant opportunity for students to gain hands-on experience in full-stack development, platform management, and enterprise-level solutions. It will also benefit Capital One by reducing manual workload, improving data management efficiency, and providing a scalable solution to support multiple business functions.

The expected outcome is a robust, reliable, and user-centric portal that will reduce time loss, minimize redundancy, and improve the overall onboarding experience for platform tenants at Capital One.

Table of Contents

Section A. Problem Statement

The Tenant Self-Service Portal project aims to address a significant inefficiency in the current onboarding process for platform tenants at Capital One. Presently, tenants must rely on a manually intensive workflow that involves frequent back-and-forth communication with platform engineers for tasks such as dataset registration, data mapping creation, and configuration updates. This manual process creates operational delays, increases the likelihood of human error, and burdens engineers with repetitive tasks that could be automated. By streamlining and automating these workflows, the project seeks to reduce operational redundancies, improve efficiency, and minimize the time required for data submissions and approvals.

The problem affects two key groups: platform tenants (clients) who need to register datasets and use cases, and platform engineers who are responsible for assisting with these tasks. For clients, the lack of a self-service option means they must wait for engineers to complete routine tasks, delaying their access to services. Engineers, in turn, are bogged down by repetitive manual work, which could otherwise be automated, reducing their productivity and limiting their focus on higher-priority tasks. The inefficiencies introduced by this manual process occur frequently across the platform, impacting many tenants and internal teams. As a result, the problem is widespread, and the delays incurred by this outdated process can lead to significant time losses, increased operational costs, and potential missed business opportunities.

Capital One is the primary client for this project, with stakeholders including multiple teams whose workflows depend on efficient and timely onboarding of datasets. These stakeholders require a solution that is both technically robust and scalable, ensuring the platform can accommodate future growth. The goal is to create an automated self-service portal that enables tenants to handle key tasks like dataset registration and updates without relying on engineers. By automating the approval process and providing clients with tools to manage their submissions, the solution will address a critical need in Capital One's platform operations.

This project falls under the financial technology (fintech) sector, with a specific focus on enterprise platform management and multi-tenant systems. Within the fintech industry, automation has become a key trend, as companies strive to improve customer experience by reducing manual intervention and streamlining processes. The self-service portal aligns with this trend, providing an automated, user-friendly interface that allows tenants to manage datasets and use cases more efficiently. The project will also leverage modern technologies, including open-source UI stacks, with room to explore AWS tools, and MongoDB, allowing for experimentation with different architectures and tools to deliver a scalable solution.

Historically, Capital One and other financial institutions have used manual processes for onboarding datasets and services, leading to inefficiencies and delays. Although some internal tools and scripts have been developed to partially automate these processes, none have fully addressed the need for a comprehensive self-service solution. The proposed portal improves on prior attempts by empowering clients to manage their submissions independently while maintaining an approval process for quality control. Admins will be able to review submissions, provide feedback, and ensure that datasets meet the required standards, all while significantly reducing the time and resources previously required for manual intervention.

In summary, the Tenant Self-Service Portal will automate critical tasks in Capital One's platform, reducing operational costs, improving client experience, and enabling platform engineers to focus on more strategic initiatives. The solution will advance automation efforts within the fintech industry, providing a scalable and efficient platform for managing datasets and use cases in a multi-tenant environment.

Section B. Engineering Design Requirements

The Tenant Self-Service Portal project aims to automate the onboarding process for platform tenants at Capital One, providing a streamlined experience for dataset registration, updates, and approvals. To ensure the design effectively meets stakeholder expectations and addresses the project's objectives, clear goals, design objectives, and specifications are outlined below.

B.1 Project Goals (i.e. Client Needs)

The primary goals of the project are derived from the needs of Capital One and its stakeholders, who require an automated solution to reduce manual intervention and improve the efficiency of the tenant onboarding process. The project will deliver a platform that automates key tasks for tenants while maintaining a feedback and approval system for admins.

The goals of the project include:

- **To create a self-service portal** that allows tenants to independently register, update, and manage datasets without engineering involvement.
- **To reduce manual back-and-forth** between tenants and engineers, minimizing time spent on repetitive tasks and improving overall platform efficiency.
- **To provide transparency** in the onboarding process, ensuring users can track the approval status of submissions, receive feedback, and make necessary adjustments.
- **To streamline the approval workflow** by enabling admins to review and provide feedback in a structured and automated manner.

These goals are designed to improve operational efficiency, reduce time delays, and empower platform users with a user-friendly interface.

B.2 Design Objectives

The key objectives of the design are focused on creating a robust and scalable self-service platform that meets the operational needs of Capital One's tenants and engineers. These objectives adhere to the SMART framework, ensuring that they are Specific, Measurable, Achievable, Realistic, and Time-bound.

The design will:

- **Provide a step-by-step process** for tenants to submit datasets in a minimum of 9 screens, allowing users to save progress and return later.
- **Automate the approval process**, allowing admin users to review, comment, and provide feedback on tenant submissions within a maximum of 2 business days.
- **Enable dataset updates** where users can view, update, and manage approved datasets, ensuring that only valid, accurate data is processed.
- **Offer a status dashboard** for clients to track the approval status and feedback for all submitted datasets.

These objectives ensure that the portal provides a streamlined, intuitive experience while adhering to performance and usability standards.

B.3 Design Specifications and Constraints

The design of the Tenant Self-Service Portal will be governed by a set of technical and operational constraints, ensuring that the system integrates seamlessly with existing processes while maintaining reliability and scalability. These constraints are measurable and realistic to meet the project's objectives.

Functional Constraints:

- The platform must be similar with Capital One's existing tech stack, using Node.js for the backend, Vue.js for the frontend, and a MongoDB database (tentatively chosen).
- The platform must allow tenants to submit datasets within a maximum of 15 minutes per complete submission.
- Admins should provide feedback within two business days to ensure a smooth approval process.

Data Constraints:

- Dataset submissions must adhere to data validation protocols, including format checks, required fields, and compatibility with existing data standards.

Interoperability Constraints:

- The system must support RESTful API integration with Capital One's internal systems to allow data processing, approval, and feedback sharing.
- It must integrate with Capital One's authentication protocols to ensure secure user access and data protection.

Cost Constraints:

- The project is developed using open-source tools (Vue.js, Node.js, MongoDB), resulting in minimal upfront costs during development.
- Future costs for full deployment within the client's proprietary environment are not included in the current phase and will depend on integration with internal systems and possible infrastructure upgrades.

Maintainability Constraints:

- The platform is built with open-source tools that have large developer communities, ensuring continued updates and support.
- Maintenance beyond the development phase will depend on integration with the client's internal systems and the handover of long-term upkeep protocols to Capital One's engineering team.

Usability and Security Constraints:

- The system must support role-based access control (RBAC), distinguishing between tenant users and admin users.
- The platform must comply with Capital One's data security standards, including encryption of sensitive information and secure data transmission.

By adhering to these specifications and constraints, the Tenant Self-Service Portal will meet the operational needs of Capital One while delivering a scalable, efficient, and secure solution.

Section C. Scope of Work

The project scope defines the boundaries of the Tenant Self-Service Portal project, outlining the key objectives, timeline, milestones, and deliverables. It clearly delineates the responsibilities of the team, including the process by which the proposed work will be verified and approved. Establishing a clear scope facilitates understanding, reduces ambiguities and risk, and manages expectations. Along with stating the responsibilities of the team, this section explicitly defines the tasks outside the team's responsibilities. The project scope also sets boundaries on timeline, available resources, and promised deliverables, preventing scope creep and ensuring control over changes. The section further specifies the development approach to be used, which will follow an agile methodology, allowing for flexibility and regular adjustments as the project progresses. Effective communication with the project sponsor and faculty advisor is essential for staying on track with scope, timeline, and budget.

C.1 Deliverables

The project deliverables are the tangible outcomes the project team is responsible for providing to the project sponsor, produced through the engineering design and development process. These include technical components like code, user interfaces, functional diagrams, and documentation. Academic deliverables, required for course completion, are also listed below.

Project Deliverables:

- **Functional Dataset Registry**: A portal feature that allows clients to submit datasets, and admins to review and approve them.
- **API Simulation**: RESTful APIs to simulate integration with external systems, demonstrating data processing and approval workflows.
- **User Interfaces**: Vue.js-based front-end for both client and admin users, providing essential interaction points for dataset submission, feedback, and approvals.
- **Documentation**: Full technical and user documentation outlining the developed features, APIs, and system architecture.
- **Testing Suite**: A basic testing framework to validate the core functionalities and ensure the platform operates as intended.

Academic Deliverables:

- Team Contract
- Project Proposal
- Preliminary Design Report
- Fall Poster and Presentation
- Final Design Report
- Capstone EXPO Poster and Presentation

C.2 Milestones

The milestones are major tasks or phases that need to be completed to ensure that the project deliverables are achieved. The following milestones break down the development timeline into manageable tasks and provide a structured approach to completing the project.

Fall Semester Milestones (Weeks 1-16):

- **Context and Requirements Gathering (Weeks 1-4)**
  Objective: Understand the project scope, gather system requirements, and research relevant technologies.
- **Initial Development of Dataset Registry (Weeks 5-8)**
  Objective: Begin coding the front-end and back-end components of the Dataset Registry.
- **API Simulation and Backend Development (Weeks 9-12)**
  Objective: Implement basic RESTful APIs and finalize core Dataset Registry functionality.
- **Testing and Validation (Weeks 13-16)**
  Objective: Conduct testing, ensure functional accuracy, and refine UI/UX.

Spring Semester Milestones (Weeks 17-32):

5. **Use Case Registry Development (Weeks 17-20)**
   Objective: Begin developing the Use Case Registry, if time permits.
6. **Refinement and Enhancement (Weeks 21-24)**
   Objective: Refine existing features, based on feedback and testing results.
7. **Final Testing and Documentation (Weeks 25-28)**
   Objective: Finalize testing, prepare the final project for presentation, and complete all documentation.
8. **Final Presentation and Submission (Weeks 29-32)**
   Objective: Present the project and submit all required academic deliverables.

C.3 Resources

To complete the project, the following resources will be required. These resources will be provided by the project sponsor or sourced from open-source platforms.

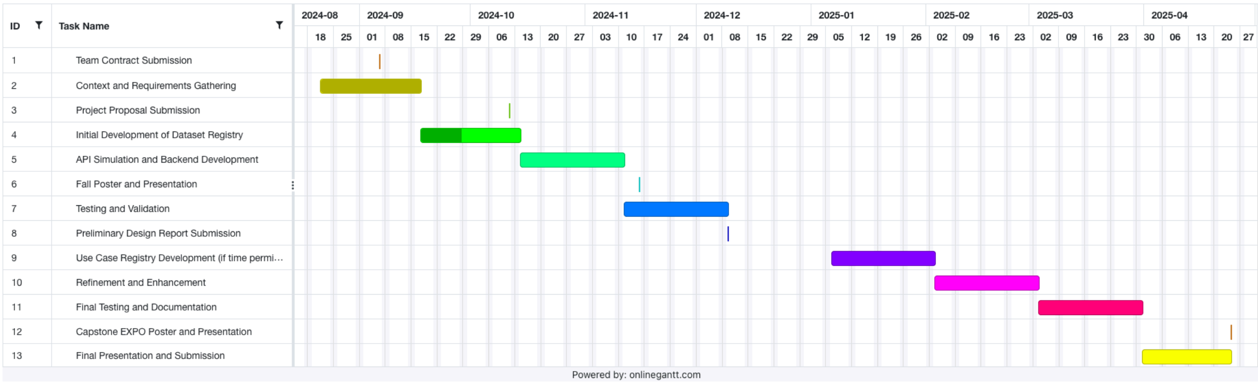
Hardware & Software Resources:

- **Development Environment**: Access to cloud-based IDEs and version control systems (GitHub).
- **Data**: Mock data to simulate tasks and processes
- **APIs**: Access to relevant APIs for simulating system integration (mocked RESTful services).
- **Front-end & Back-end Technologies**: Vue.js for front-end development and Node.js with MongoDB for back-end.
- **Testing Tools**: Basic testing frameworks for validating functionality.

External Resources:

- **Libraries and APIs**: Access to libraries for building RESTful APIs and data validation processes.
- **Reference Materials**: Documentation and online resources for implementing authentication, UI components, and secure data handling.

Appendix 1: Project Timeline



Appendix 2: Team Contract (i.e. Team Organization)

Step 1: Get to Know One Another. Gather Basic Information.

Team Member Name	*Strengths each member bring to the group*	*Other Info*	*Contact Info*
Prakhar Mathur	*Organization, communication, Professional experience, data science + ML, information systems, software development, IT infrastructure + architecture*	*Python, R, SQL (Data Science, ML)* *Java, C* *Statistics, Math, Visualization* *Azure DevOps, Project Management*	*mathurp6@vcu.edu* *804-591-7609*
Alan Dorn	*Effective communication, Organization, software development*	*Java, Python, C, Databases, SQL, HTML, CSS, JS*	*dornn@vcu.edu* *703-713-5629*
Krish Patel	*Workforce experience, organization, software development*	*Microsoft Power Platform, Java, Python, Databases, HTML, CSS, JS, SQL*	*patelkp7@vcu.edu* *757-386-8642*
Austin Glass	*Effective communication, backend development,*	*Java, C, C++, Python, Linux*	*glassal@vcu.edu* *804-714-7969*

Other Stakeholders	***Notes***	***Contact Info***
Faculty Advisor	*Dr. Ghosh*	pghosh@vcu.edu
Sponsor	*Mahesh Nair*	mahesh.bahulleyannair@capitalone.com
Mentor	*Balakrishnan Muthusamy* *Ashish Kulkarni* *Shannon Hsu*	balakrishnan.muthusamy@capitalone.com ashish.kulkarni2@capitalone.com shannon.hsu@capitalone.com

Step 2: Team Culture. Clarify the Group's Purpose and Culture Goals.

Culture Goals	***Actions***	***Warning Signs***
Being on time to every meeting	- *Set up meetings in shared calendar* - *Send reminder email in day before meeting*	- *Student misses first meeting, warning is granted* - *Student misses meetings afterwards – issue is brought up with faculty advisor*
Informing the group of any delays in completing assignments	- *Stay up to date with each other's project responsibilities* - *Set reasonable deadlines and note when an extension is needed*	- *Student shows up for weekly meeting with no considerable work done*
Communicating effectively about delays	- *Let other members know if you are running behind or are unable to make it*	- *Student does not communicate whereabouts*

Step 3: Time Commitments, Meeting Structure, and Communication

Meeting Participants	***Frequency*** ***Dates and Times / Locations***	***Meeting Goals*** ***Responsible Party***
Students Only	*As Needed, On Discord Voice Channel*	*Update group on day-to-day challenges and accomplishments*
Students + Faculty advisor	*Meet as needed (more of sponsor guidance + involvement)* *Weekly/Biweekly*	*Check-in with Dr. Ghosh on any guidance or expertise needed.*
Project Sponsor	*Weekday evening (waiting on sponsor communication)* *Weekly/Biweekly*	*Check-in with sponsor for questions (technical/nontechnical), issues, etc.*

Step 4: Determine Individual Roles and Responsibilities

Team Member	***Role(s)***	***Responsibilities***
Prakhar Mathur	*Project Manager/SDE*	✓ *Define project scope and tasks* ✓ *Facilitate team communication* ✓ *Mitigate risks and allocate resources* ✓ *Monitor progress and quality*
Krish Patel	*Full-stack developer/SDE*	✓ *Designing system architecture and UI/UX design* ✓ *Developing backend requirements for software application* ✓ *Debugging and testing software* ✓ *Improving and optimizing accessibility and responsiveness*
Alan Dorn	*Devops / Architect personnel*	✓ *Selecting tools* ✓ *Defining infrastructure as code* ✓ *Setting the stage for an automated, efficient pipeline to accelerate innovation*
Austin Glass	*Co-Project Manager/SDE*	✓ *Assist in defining project scope and technical tasks* ✓ *Bridge communication between technical and non-technical team members* ✓ *Contribute to coding while managing risks and resources* ✓ *Ensure progress tracking and code quality*

Step 5: Agree to the above team contract

Team Member: *Krish Patel* ***Signature:***

Team Member: *Alan Dorn* ***Signature:*** Alan Dorn

Team Member: *Prakhar Mathur* ***Signature:*** Prakhar Mathur

Team Member: Austin Glass ***Signature:*** Austin Glass