Guide 1. Definition of APT Project Capstone Course

1. **PART I**



| Student Name | **Erick Brandon San Martín Zuñiga** |
| --- | --- |
| ID | **21.449.917-7** |
| Major | **Computer Engineering** |
| Location | **Maipú** |

| Student name | **John Andres Herrera Montes** |
| --- | --- |
| ID | **21.367.897-3** |
| Major | **Computer Engineering** |
| Location | **Maipú** |

| Student name | **Alexander Javier Pulgar Allende** |
| --- | --- |
| ID | **21.091.734-9** |
| Major | **Computer Engineering** |
| Location | **Maipú** |



| Project name | OfiSync: Comprehensive Building Management System |
| --- | --- |
| Performance area(s) | Software Development Systems Analysis  IT Service Management  Technology Project Management |
| Competencies | Manage the configuration of environments, application services, and databases   * Configure VPS, centralized database, and development/testing environments.   Offer IT solution proposals by comprehensively analyzing processes   * Define modules according to profiles (Administrator, Concierge, Cleaning, Customers).   Develop a software solution using systematic development and maintenance techniques   * Iterative development in sprints, version control, testing, and documentation.   Build scalable data models   * Design and standardization of the centralized database.   Program queries or routines to manipulate information in the database   * CRUD for common expenses, reservations, inventories, logs, and payments.   Build programs and routines of varying complexity with good coding practices   * Developing the web portal and mobile application with database integration.   Performing quality testing on both products and processes.   * Unit, integration, functional, and load testing. |

|  | Build the architectural model of the systemic solution.   * Definition of multi-platform architecture (web + app + centralized database).   Implement comprehensive system solutions to automate business processes.   * Automation of common expenses, reservations, payment validation, inventory control, and notifications.   Resolving systemic vulnerabilities and complying with security standards   * User validations, credential encryption, security testing in the database and application. |
| --- | --- |



| Relevance of the APT project | This project seeks to address a real problem in building management in Chile: the lack of digitization and automation in the daily management of common expenses, service reservations, cleaning logs, and entry and exit records. Currently, many communities still use manual processes, Excel spreadsheets, or non-specialized software that is not adapted to the specific needs of each role (administrator, tenant, cleaning staff, and concierges).  The project is mainly located in the Metropolitan Region, especially in districts such as Las Condes, Providencia, Huechuraba, and Vitacura, where there is a high concentration of office buildings, and also in districts such as Santiago Centro, where there is a high concentration of residential buildings. It would directly impact building administrators, cleaning workers, janitors, and tenants, improving their experience, efficiency, and traceability in processes.  The choice of this topic responds both to its real impact on the work environment and its alignment with current trends in digital transformation, automation, and smart services. |
| --- | --- |
| APT Project Description | The project consists of developing a multi-platform system comprising: A web portal developed with custom views for:   * Administrator: calculation and management of common expenses, receipt and validation of payment receipts, management of service reservations, control of inventory supplies. * Concierge: recording of reception logs, access control. |

|  | * Cleaning staff: management of supplies inventory and recording of activities in the cleaning log.   A mobile application aimed at:   * Customers (Tenants): checking common expense amounts, uploading payment receipts, and reserving common building services.   A centralized database, hosted on a VPS, that synchronizes information between the web portal and the mobile application in real time, ensuring data consistency and availability for all user profiles.  The objective of the system is to modernize and automate office building management processes, optimizing operational efficiency and improving the experience of the various users involved. |
| --- | --- |
| Relevance of the project to the graduate profile | The project is directly related to the graduate profile because:   * It requires the design and development of comprehensive technological solutions, taking into account different types of users and devices. * It involves technology project management, sprint planning, deliverable control, and progress monitoring. * It applies principles of interoperability, data synchronization, and IT security, which are part of the graduate profile. * It is based on agile methodologies, such as Scrum, which allow value to be delivered iteratively and continuously. * The selected competencies are essential to the success of the project, as they allow both the technical and management aspects to be addressed. |
| Relationship to professional interests | This project aligns perfectly with our professional interests in software development.  It allows us to work in multiple areas of development such as frontend, backend, databases, and systems architecture, as well as strengthening our experience with agile methodologies and multi- platform development.  Its implementation will give us the opportunity to improve our technical skills and expand our portfolio with a solid and functional project that is useful for the job market. |
| Feasibility of APT Project Development | Strengths:   * The project is well defined in its main functionalities and has a backlog organized by epics and user stories. * The estimated time (10 weeks) is sufficient, divided into 3 sprints, allowing for progressive delivery of functionalities. * The technologies used (Ionic, JavaScript, Django) are   manageable and realistic for this academic context. |

|  | * The required materials are accessible (free software or with academic licenses).   Weaknesses/Potential difficulties:   * Synchronization between platforms may create technical complexity. * Possible difficulties with VPS connection and configuration. * Careful planning is required to avoid overloading the sprints.   Solution:   * Mitigate these weaknesses by applying agile methodologies and conducting frequent testing. * Prioritize critical functionalities in the first sprints. * Consult teachers/technical tutors in case of technical difficulties. |
| --- | --- |

1. **PART II**



| General objective | Develop a multi-platform system consisting of a web portal and a mobile application that modernizes and automates the administration of office buildings by digitizing processes, optimizing resources, and improving the experience of all users involved. |
| --- | --- |
| Specific objectives | Strategic objectives:   * Reduce administrative errors by digitizing and automating key processes such as common expenses, reservations, and logs. * Improve transparency and tenant confidence in payment and reservation management through an accessible and secure system. * Save time and resources by optimizing the workflows of administrators, concierges, and cleaning staff. * Increase user satisfaction by offering a modern, intuitive, and efficient platform. * Develop a scalable and flexible solution capable of adapting   to future improvements or functional enhancements. |

|  | * Implement agile methodologies that ensure efficient development management and timely delivery. * Foster effective collaboration between development team members and stakeholders.   Project Objectives:   * Develop a web portal with different views for:   + Administrators (management of common expenses, reservations, receipts, inventories, agenda).   + Concierges (reception logs, access control, assigned tasks).   + Cleaning staff (activity log and supply control). * Develop a mobile application aimed at customers (tenants) with features such as:   + Checking the amount of common expenses.   + Upload payment receipts.   + Reservation of common services. * Ensure real-time synchronization between the mobile application and the web portal through a centralized database in the cloud. * Design intuitive and accessible interfaces for all types of users, ensuring ease of use. * Implement a secure data storage and query system, backed up in the cloud. * Structure a robust database that allows for efficient management of common expenses, reservations, records, and logs. * Enable the uploading and validation of payment receipts to facilitate financial control by administrators. * Implement a digital log for concierges and cleaning staff, with the possibility of attaching photos and comments. * Develop an efficient reservation system, avoiding scheduling conflicts and duplications. * Create a cleaning supplies inventory module, with low stock alerts to facilitate replenishment. * Meet the established deadlines for development without compromising the technical or functional quality of the product. |
| --- | --- |







| **Type of evidence**  **(progress or final)** | **Name of**  **evidence** | **Description** | **Justification** |
| --- | --- | --- | --- |
| Progress | Scrum Charter | Initial document establishing the project objectives, the Scrum team, roles, and initial scope. | It allows the team and stakeholders to align around a formal starting framework. |
| Progress | Product Backlog | Prioritized list of user stories and features to be developed. | Ensures that the team has clarity on scope and priorities. |
| Progress | Complete common expenses module (web) | Functional deliverable that allows common expenses to be calculated, viewed, and managed on the web platform. | Represents real progress on the project and validates frontend- backend integration. |
| Progress | Complete reservations module (mobile) | Functional deliverable that allows users to make reservations from the mobile application. | Ensures early availability of a key feature for users. |
| Progress | Sprint backlog for the first sprint. | Selection of user stories and technical tasks defined for Sprint 1. | Shows detailed work planning for the first cycle. |
| Progress | Retrospective minutes from the first sprint. | Document summarizing lessons learned, problems, and improvements identified at the end of Sprint 1. | Enables the team to reflect and continuously improve. |
| Progress | First sprint testing report. | Results of unit, integration, and user tests from Sprint 1. | Ensures the quality of the progress delivered and provides objective evidence of validation. |
| Final | Fully functional system. | Final delivery of the product with all web and mobile functionalities integrated. | Demonstrates that the project meets the agreed scope and is ready for production. |
| Final | Sprint backlog for remaining sprints. | Planning of tasks and functionalities worked on in subsequent sprints. | Evidence of control and monitoring throughout development. |
| Final | Retrospective minutes for the rest of the sprints. | Documents summarizing the lessons learned from sprints 2 and 3. | Reinforces continuous improvement and adaptation of the Scrum process. |

| Final | Complete testing report | Consolidated report of all tests performed on the entire system. | Provides final quality assurance and reduces the risk of production failures. |
| --- | --- | --- | --- |



| **APT Project Work Plan** | | | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| Competence or units of competence | Name of Activities/Task s | Description of Activities/Task s | Resources | Duration of activity | Responsible[1](#_heading=h.ya06hdejfgjo) | Comments |
| Team communicatio n and coordination. | Daily meetings | 15-minute meetings every day to synchronize progress, obstacles, and next steps. | Video calls, chat, Scrum board. | 15 to 20 minutes a day for the entire project | Full team | Key to maintaining alignment and communication. |
| Manage environment configuration, application services, and databases. | VPS and environment configuration | AWS server configuration, installation of dependencies, and setup of development/ testing environments. | AWS, Docker, Git. | 2 days | Full team | Mandatory initial activity for all development. |
| Build scalable data models / Program queries or routines to manipulate information in the database. | Database configuration | Design and creation of the centralized database in PostgreSQL. | PostgreSQL, pgAdmin. | 3 days | Complete team | Requires defining initial schema and relationships. |
| Develop a software solution using systematic | Common expenses module | Screens for viewing and editing common | React, Vite | 1 week | John Herrera | Connects to common expense API. |



1 If the APT Project is a group project, the names of those responsible for each task or activity should be indicated in this column. This will subsequently allow for individual assessment of each member.

| development and maintenance techniques / Implement comprehensiv e systemic solutions. | interface (web) | expenses in Django frontend. |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Develop a software solution using systematic development and maintenance techniques / Build programs and routines of varying complexity with good coding practices. | Backend calculation of common expenses | Development of logic and endpoints for the calculation of common expenses. | Node, JavaScript Express, PostgreSQL. | 2 weeks | Erick San Martín and John Herrera | Direct integration with frontend and app. |
| Develop a software solution using systematic development and maintenance techniques / Implement comprehensiv e systemic solutions. | Reservation module (mobile app) | Development of screens and booking logic in Ionic. | React Native, Expo, Node, JavaScript Express, PostgreSQL. | 2 weeks | Alexander Pulgar | Requires backend booking endpoints. |
| Perform quality testing on both products and processes. | Initial testing | Testing of common expense and reservation modules. | Jest, manual testing. | Half a week | Full team | Validation of the first product increment. |
| Perform quality tests on both products and processes. | Supply inventory interface (web) | Development of web form for CRUD of supplies. | React, Vite | 1 week | John Herrera | Stock validations. |

| Build scalable data models / Program queries or routines to manipulate information in the database. | Backend inventory of supplies | CRUD API for supplies with persistence in PostgreSQL. | Node, JavaScript Express, PostgreSQL. | 2 weeks | Erick San Martín and John Herrera | Connects to web interface. |
| --- | --- | --- | --- | --- | --- | --- |
| Develop a software solution using systematic development and maintenance techniques / Implement comprehensiv e systemic solutions. | Receipt upload module (mobile app) | Function in the app for uploading receipts (PDF/image). | React Native, Expo, Node, JavaScript Express, PostgreSQL. | 1 week | Alexander Pulgar | Validate file format before uploading |
| Develop a software solution using systematic development and maintenance techniques / Implement comprehensiv e systemic solutions. | Common expenses query module (mobile app) | App screen to view common expenses to be paid. | React Native, Expo, Node, JavaScript Express, PostgreSQL. | 1 week | Alexander Pulgar | Requires stable common expense API. |
| Perform quality testing on both products and processes. | Intermediate testing | Testing inventory, receipts, and common expenses on mobile devices. | Jest, pruebas manuales. | Half a week | Complete team | Cross- validations of modules. |
| Develop a software solution using systematic development and maintenance techniques / | Reservation query interface (web) | Screens for concierge/ad ministrator to view reservations. | React, Vite | Half a week | John Herrera | Connects to the reservation backend. |

| Implement comprehensiv e systemic solutions. |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Build programs and routines of varying complexity / Implement comprehensiv e systemic solutions. | Reservation query backend | Endpoints to return reservations to the web frontend. | Node, JavaScript Express, PostgreSQL. | 1 week | Erick San Martín and John Herrera | Connection with mobile reservation module. |
| Develop a software solution using systematic development and maintenance techniques / Implement comprehensiv e systemic solutions. | Logbook interface (web) | Screen for notes, access control, logbook registration. | React, Vite | Half a week | John Herrera | No comments. |
| Build programs and routines of varying complexity / Program queries or routines to manipulate information in the database. | Backend log | CRUD API for storing log records. | Node, JavaScript Express, PostgreSQL. | 1 week | Erick San Martín and John Herrera | Must guarantee persistence and fast querying. |
| Develop a software solution using systematic development and maintenance techniques / Implement comprehensiv | Login and registration view (web) | Development of user login and registration screens. | React, Vite | Half a week | John Herrera | Requires integration with backend. |

| e systemic solutions. |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Build programs and routines of varying complexity / Resolve systemic vulnerabilities and comply with security standards. | Backend login and registration | Authentication API and user roles. | Node, JavaScript Express, PostgreSQL. | Half a week | Erick San Martín | Includes credential validation and encryption. |
| Implement comprehensiv e systemic solutions / Resolve systemic vulnerabilities and comply with security standards. | Mobile app login | Implementatio n of login in the mobile app (Ionic). | React Native, Expo, Node, JavaScript Express, PostgreSQL. | Half a week | Alexander Pulgar | No comments |
| Build the architectural model for the systemic solution / Implement comprehensiv e systemic solutions. | Module integration | Connection of all web, backend, and mobile app modules. | Git, APIs REST, Postman | 2 weeks | Full team | May require endpoint and UI  adjustments. |
| Perform quality testing of both products and processes / Implement comprehensiv e systemic solutions. | Final testing and closure | Regression, functional, and usability testing for the entire system. | Jest, manual testing. | 1 week | Full team | Final validation prior to launch. |



| Find a Gantt chart format that suits you and organize the activities planned in the previous point, taking |
| --- |
| into account the period assigned for the development of your APT Project. You must maintain the |
| academic period schedule in the development of the three phases covered by the Degree Portfolio Course. |

