**Guide 2: APT Project Development**

**Capstone Course**

|  |
| --- |
| **1. Summary of Progress - APT Project** |
| A continuación, encontrarás distintos campos que deberás completar con la información solicitada. |

|  |  |
| --- | --- |
| Summary of APT Project Progress | The TechApps - SCAV (Vehicle Access Control System) project has progressed significantly, reaching Sprint 10. The main objective is to optimize vehicle access management at the Vista Parque condominium, improving traffic flow and security.  **Achievements so far:**   * **Implementation of the license plate reader using Python**, which has been presented as a functional prototype. * **Development of the REST API in Spring Boot**, enabling database integration and vehicle validation through an external API. * **Creation of the Entity-Relationship Model (ERM)** for the database using SQL Server. * **Partial development of the mobile application in Flutter**, focusing on the administrative module. |
| Objectives | **Objectives:**  **General Objective:** Develop an integrated vehicle access control system to improve security and expedite vehicle entry and exit at the Vista Parque condominium, ensuring exclusive control for residents and authorized personnel.  Specific Objectives:   1. Implement an automatic vehicle license plate reader using Python. 2. Develop a mobile application in Flutter to allow residents to register visits in advance. 3. Create a REST API in Spring Boot to manage vehicle access permissions and integrate with external APIs. 4. Design a database in SQL Server to manage residents, visits, and commercial vehicles. 5. Generate vehicle access reports and enable real-time monitoring. 6. Ensure the system meets information security standards. |
| Methodology | The methodology followed is Scrum, with 9 sprints of two weeks each. The activities include requirements planning, modular development (license plate reader, API, mobile application, database), integration testing, and the future deployment of the system. |
| Evidence of Progress | **License Plate Reader Prototype:** A functional prototype implemented in Python was presented, demonstrating the system's ability to detect license plates and associate them with registered vehicles.  **Database Design:** An ERD was designed in SQL Server.  **REST API Development:** The API is partially implemented and connected to the database, enabling vehicle validation and access to information on residents, visitors, and logs. |

|  |
| --- |
| **2. Monitoring of the Work Plan** |
| Examina cuidadosamente tu plan de trabajo, enfocándote especialmente en la columna de estado de avance y ajustes. |

**Competencies or Units of Competence**

* **Administering the configuration of environments, application services, and databases (Competence 1):**  
  The development environment has been configured for the SQL Server database and the REST API in Spring Boot. The database is connected to the different system modules to ensure operational continuity.  
  This work was carried out from Sprint 5 to Sprint 8, during which the database design and API integration were completed.
* **Providing IT solution proposals by analyzing processes (Competence 2):**  
  From the early stages, requirements gathering was conducted to identify the issues at the Vista Parque condominium and propose a system to efficiently manage vehicle access.  
  This was primarily implemented during the initial planning phase of the project and throughout Sprint 1 and Sprint 3 with the definition of the system architecture.
* **Developing a software solution using systematic techniques (Competence 3):**  
  Up to week 10, multiple system modules have been developed, including the license plate reader in Python, the REST API in Spring Boot, and the initiation of the mobile application in Flutter. All of this follows an agile methodology (Scrum) that systematizes the development process.  
  This competence has been active throughout Sprints 5-10, with clear progress in the technical implementation of the software.
* **Building data models (Competence 4):**  
  The ERD (Entity-Relationship Diagram) for the SQL Server database was developed, which manages records of residents, visitors, and vehicles, ensuring scalability for future expansions.  
  This task was completed during Sprint 7.
* **Programming queries or routines to manipulate information (Competence 5):**  
  The REST API has been implemented, allowing real-time queries to the database on residents and visitors.  
  This has been worked on during Sprints 7-10, when part of the API development and its connection to the database were completed.
* **Implementing comprehensive systemic solutions (Competence 6):**  
  The license plate reader, database, and REST API are integrated to automate the vehicle access process, optimizing condominium operations.  
  This has been part of the development since Sprint 5, with substantial progress made during Sprints 8-10.

**Work Plan and Monitoring**

**Competencies or Units:**

* **Project Management:** Sprint planning, requirements gathering, and backlog creation.
* **Software Development:** Implementation of the license plate reader, development of the REST API, database, and mobile application.
* **Software Quality:** Security, functionality, and integration testing.

**Activities and Progress Status:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Activity** | **Status** | **Difficulties** | **Materials/Physical Resources** | **Languages/Environment** |
| License Plate Reader | Completed | Challenges testing live on the street due to weather and traffic conditions | Video camera, PC | Python, OpenCV |
| Database Design | Completed | N/A | PC, Google Cloud SQL | Kubernetes |
| REST API | In Progress | N/A | PC, GCP (Google Cloud Platform) API server | Java, Spring Boot |
| Mobile Application in Flutter | In Progress | N/A | PC, smartphone for testing | Flutter, Dart |
| Integration Testing | Not Started | Coordination for real-time testing with a camera in a controlled environment | Video camera, PC, local network for integration | Python, Java, SQL, Flutter |

**Future Activities:**

* Perform integration tests between modules.
* Prepare the system for launch and provide training to end users.

**Adjustment:**

1. **Methodology: Correction to Scrum**

From the start of the TechApps - SCAV project, the Scrum methodology was proposed and established as the guiding framework to manage development. However, during the early phases of implementation, the team mistakenly began managing activities with a more traditional approach. This deviation occurred due to the organization of tasks as independent blocks, with fixed delivery dates and limited interaction between the modules under development (license plate reader, REST API, database, and mobile application).

Upon identifying this issue, an immediate adjustment was made to return to the agile Scrum approach, which enabled the following:

1. **Recovery of Iterative and Incremental Planning:**  
   Tasks were reorganized into sprints, ensuring greater flexibility to incorporate changes as needed.
2. **Improved Coordination and Collaboration:**  
   Daily meetings were reintroduced to align the team and quickly address any blockers.
3. **Frequent Partial Deliveries:**  
   Incremental deliveries resumed for each sprint, allowing the presentation of functional progress, such as the license plate reader and the integration of the API with the database.
4. **Continuous Feedback:**  
   Constant interaction with stakeholders was encouraged to ensure that each module was developed according to the expectations and requirements of the project.

This adjustment has been essential to regaining the agile dynamic that had been lost. The team realigned with the Scrum methodology, resulting in better synchronization between modules, optimized development timelines, and ensuring the project progresses as planned.

1. **Project Timeline: Sprint Adjustment**

Initially, the TechApps - SCAV project was organized into a different number of sprints, which did not allow for proper planning to implement all the modules. After reviewing the schedule, a correction was made to the number and duration of the sprints to optimize development and meet the objectives.

**New Schedule:**

* **Total Sprints:** 9 Sprints
* **Duration of Each Sprint:** 2 weeks each

**Module Distribution:**

1. **Project Management and Backlog:** Sprint 1 (S1) and (S2), Sprint 2 (S3) and (S4).
2. **Development of the License Plate Recognition System:** Sprint 3 (S5) and (S6).
3. **Database Model Development:** Sprint 4 (S7) and (S7).
4. **REST API Implementation:** Sprint 5 (S9) and (S10).
5. **Mobile Application in Flutter:** Adjusted to two full sprints: Sprint 6 (S11) and (S11), Sprint 7 (S13) and (S14).
6. **Quality and Integration Testing:** Sprint 8 (S15) and (S16).
7. **Soft Launch and Training:** Sprint 9 (S17) and (S18).

**Impact of the Adjustment:**

* **Time Redistribution:** The dates were adjusted to allocate more time to critical modules, such as the development of the mobile application in Flutter, which will be completed over two full sprints.
* **Workflow Optimization:** The remaining modules will be developed in single sprints to avoid work fragmentation.

1. **Application Development**

**Unification into a Single App with Administrative and Resident Roles**

During the initial planning phase of the **TechApps - SCAV** project, we initially proposed developing two separate applications: one for administrators and another for residents. However, as we progressed with the development and after reviewing the flow and usability, we identified that this division created redundancies and increased the system's maintenance burden.

**Decision to Change: Unification into a Single Application with Dynamic Roles** Instead of managing two separate applications, we decided to integrate all functionalities into a single mobile application. This new strategy simplifies development, facilitates maintenance, and enhances the user experience by unifying operations on one platform.

**How the New Structure Works:**

* Upon login, the application detects the **account type** (Administrator or Resident) based on the user's credentials.
* Depending on the user's role, they are redirected to different screens and functionalities:
  + **Administrators**:
    - Management of residents, visits, and commercial vehicles.
    - Access to reports and real-time monitoring of vehicle access.
    - Administration of entry and exit permissions.
    - Automation and management of fines.
  + **Residents**:
    - Pre-registration of visits.
    - Real-time notifications about the arrival of visitors and deliveries.
    - Viewing personal access history.
    - Requesting and authorizing access for commercial services.

**Benefits of the Unified Approach**

1. **Reduced Complexity**: By unifying both applications into one, redundancies are eliminated, simplifying system management.
2. **Flexible Role Usage**: The app dynamically adapts to the user's role, providing an intuitive and centralized flow.
3. **Simplified Maintenance**: Facilitates update management since any improvements are reflected in a single app.
4. **Enhanced User Experience**: Both administrators and residents can access all functionalities from a single platform without needing to switch applications.
5. **Cloud Services Implementation and Database Migration**

**Context of the Change**: In the initial phases of the **TechApps - SCAV** project, the system was configured to operate in a **local environment**, which introduced certain limitations in terms of access and scalability. To improve performance and ensure more robust and secure access, we decided to migrate the application and database to a cloud environment using **Google Cloud Platform (GCP)**.

**Details of the Implementation**:

1. **Database Migration**:
   1. The database, originally developed in **SQL Server**, was migrated to a cloud service using **Google Cloud SQL**.
   2. This migration enables faster and more reliable access to information on residents, visits, and commercial vehicles from anywhere.
   3. **Benefits**:
      1. Eliminates the dependency on a localhost environment, improving availability and accessibility.
      2. Enhances security by leveraging GCP's security policies and encryption.
      3. Facilitates database scaling according to system demand.
2. **Deployment of the Authentication Service (Microservice msautenticar)**:
   1. The first microservice (msautenticar) was deployed in the cloud using **Google Kubernetes Engine (GKE)**.
   2. This allows for an **automated and scalable deployment** of the authentication system, which is crucial for managing access for both administrators and residents.
   3. **Benefits**:
      1. Improves scalability and availability of the authentication service.
      2. Facilitates load management based on the number of concurrent users.
      3. Reduces downtime by allowing hot updates without affecting users.

**Impact of These Changes on the Project**

1. **Elimination of Local Dependencies**:
   1. By migrating to the cloud, the need to run services on localhost is eliminated, simplifying the integration and deployment of the application in production environments.
2. **Enhanced Scalability and Reliability**:
   1. Cloud services, combined with Kubernetes, allow the system to scale automatically based on usage, ensuring optimal performance.
3. **Improved Security and Accessibility**:
   1. Leveraging GCP's robust security measures ensures that the system remains secure while being accessible from anywhere.

These changes significantly improve the overall efficiency, scalability, and security of the **TechApps - SCAV** project, aligning it with the needs of the **Vista Parque** residential complex.

1. **Use of Kubernetes for Container Management**

**Context of the Change**: To optimize the **orchestration** and **scalability** of the system, the decision was made to implement Kubernetes using **Google Kubernetes Engine (GKE)**. This has enabled a more modern and efficient approach to deploying the services for the **TechApps - SCAV** project.

**Implementation Details**:

1. **Container Orchestration with Kubernetes**:

* All microservices in the system, such as the license plate reader, the **API REST**, and the notification system, are now **containerized using Docker** and managed by Kubernetes.
* Kubernetes automates the **deployment, scaling, and management** of these containers, ensuring that the system operates smoothly under varying levels of load.

1. **Benefits of Using Kubernetes**:

* **Automatic Scalability**: Kubernetes automatically adjusts the number of replicas for each microservice based on demand.
* **Continuous Deployment (CI/CD)**: Facilitates continuous integration and deployment, allowing updates to be made without interruptions.
* **High Availability**: Due to its ability to relocate failed services and manage resource availability, the system remains operational even in the event of node failures.

**Impact of These Changes on the Project**

1. **Elimination of Local Dependencies**:

* With the migration to the cloud and the use of Kubernetes, the need to run services on localhost is eliminated, simplifying the **integration and deployment** of the application in production environments.

1. **Improved Scalability and Availability**:

* The system can now **dynamically scale** based on demand, which is essential for a project aimed at optimizing vehicle access in a large condominium like **Vista Parque**.

1. **Facilitates Collaboration and Development**:

* By centralizing services in the cloud, all team members can access the **development and production environments** remotely, speeding up testing and collaborative development.

|  |
| --- |
| **3. Adjustments Based on Monitoring** |
| Profundiza en las observaciones de tu plan de trabajo. Analiza las actividades planificadas y señala qué aspectos facilitaron u obstaculizaron la ejecución del plan. Plantea cómo abordaste y/o abordarás los obstáculos. Por último, señala los ajustes que realizaste al plan de trabajo a partir de este análisis. |

|  |
| --- |
| **Factors That Facilitated the Development of the Work Plan:**   * **Team Coordination:** Implementing daily meetings through the Scrum methodology enabled better communication and quick problem-solving. * **Appropriate Technological Tools:** A well-configured development environment, including Python, SQL Server, and Spring Boot, facilitated the developers' work. * **Early Functional Prototype:** The license plate reader was completed in the initial phases of the project, allowing rapid progress in the integration with other modules.   **Factors That Hindered the Development of the Work Plan:**   1. **Change in Application Structure:**    * **Difficulty:** Initially, a single application was developed for both residents and administration, complicating management and increasing the workload.    * **Action Taken:** The development was divided into two applications: one for administration and another for residents, with the administrative app being prioritized for now. 2. **Error in the Application of the Scrum Methodology:**    * **Difficulty:** Although the plan was to follow Scrum, a traditional approach was mistakenly adopted during the first weeks, affecting planning and deliverables.    * **Action Taken:** An immediate correction was made to return to the Scrum methodology, establishing 2-week sprints with daily meetings and incremental deliveries. 3. **Adjustment in Sprint Schedule:**    * **Difficulty:** The initial sprint plan did not allocate enough time for some key modules, such as the Flutter application.    * **Action Taken:** Tasks were redistributed across 9 sprints, with 2 full sprints dedicated to Flutter development, and the timelines for other modules were adjusted accordingly. |

|  |
| --- |
| **Adjusted or Eliminated Activities**  **Adjustments Made:** **1. Unification of Applications for Administration and Residents**  * **Reason**: Initially, the plan was to develop two separate applications (one for Administration and another for Residents). However, as development progressed, we realized that maintaining two separate applications created redundancy and increased management complexity. * **Adjustment**: All functionalities were unified into a **single mobile application**. Depending on the user role (Administrator or Resident), upon logging in, the app redirects to specific screens and functionalities according to the profile. * **Impact**: The unification simplifies **maintenance and updates**, enhances the user experience, and reduces the development workload. Additionally, the system can now **scale more efficiently** by centralizing all functionalities on a single platform.   2. Migration of Backend to the Cloud with Kubernetes   * Reason: Initially, the backend (API REST in Spring Boot) was configured to run on localhost, which limited system access and scalability. * Adjustment: The API REST developed in Spring Boot was migrated to Google Cloud Platform (GCP), using Google Kubernetes Engine (GKE) for orchestration. * Impact:   + Elimination of localhost: The system now connects to the backend through a cloud service, improving availability and reliability.   + Scalability and Availability: Kubernetes enables automatic scaling of the API based on user load, ensuring optimal performance at all times.   + Simplified Maintenance: The use of containers and Kubernetes allows for continuous updates without service interruptions, ensuring the system is always available to users.   3. Adjustment in Sprint Schedule   * Reason: The initial planning assigned one sprint per module, which did not provide sufficient time for developing more complex components, such as the mobile application in Flutter. * Adjustment: The schedule was revised, allocating two sprints for the development of the Flutter mobile application and maintaining one sprint for simpler modules. This new structure ensures better planning and helps avoid delays.   **Eliminated Activities:**   * **At this time, no activities have been eliminated,** as all the initially defined phases remain relevant to achieving the project's objectives.   **Justification:**  These adjustments were made to ensure development efficiency and improve system quality, enabling us to meet deadlines without compromising key functionalities. The separation of applications and the reorganization of the schedule have been essential in maintaining the project's flow according to expectations. |

|  |
| --- |
| **Activities Not Started or Delayed**  If there are activities that you have not started or are delayed according to your planning, indicate the reasons for not meeting the deadlines and what strategies you would use to advance these activities without affecting your APT project.  **Integration Testing Between Modules:**   * **Current Status:** All the main modules have been developed: the license plate reader, the REST API, and the SQL Server database. However, integration testing to validate that these components interact correctly with each other has not yet been conducted. * **Strategy:** Integration tests are scheduled for the upcoming sprints. They will be carried out in a controlled environment that simulates real conditions, ensuring the system operates smoothly before moving to production.  ****Development of the Mobile Application in Flutter:****  * **Current Status:** The development of the mobile application has not yet started because it is planned for later sprints according to the established schedule. This application will focus on functionalities for residents, such as visitor registration and real-time notifications. * **Strategy:** Development will begin in the corresponding sprints (6 and 7). In the meantime, the team is focusing on completing the priority modules, such as the administrative application and backend component integration.  ****System Testing (Marcha Blanca) and Staff Training:****  * **Current Status:** Although significant progress has been made with the main modules, the system testing phase ("marcha blanca") and staff training have not yet started, as they depend on the completion of integration testing. * **Strategy:** Staff training is planned to be conducted in parallel with the final system tests. This will ensure that personnel are prepared when the system becomes operational. The system testing phase will begin once it has been validated that all modules interact correctly. |