| **Guide 2. APT Project Development Capstone Course**    **1. APT Project Progress Summary** |
| --- |
| Below, you will find various fields that you must complete with the requested information. |

| APT Project Progress Summary | During Sprint 1 (Weeks 1-3), the team successfully completed the foundational activities of the OfiSync project. The development and testing environments were set up, and the centralized database was created.  At the same time, the first key functional modules were developed: the "Common Expenses Module" for the web portal (including interface and backend) and the "Reservations Module" for the mobile application. Finally, the initial testing cycle for these modules was executed.  These advances directly meet the following specific objectives:   * "Develop a web portal with differentiated views..." (Started with the common expenses module). * "Develop a customer-oriented mobile application..." (Started with the service reservation functionality). * "Ensure real-time synchronization... through a centralized database..." (Achieved through the configuration of the database). * "Structure a robust database..." * "Develop an efficient reservation system..."   Despite the difficulties encountered, such as compatibility issues that forced a change in programming language and challenges in testing implementation, the team managed to adapt by actively researching and applying their knowledge (React, Jest) to complete the defined activities within the established timeframes. |
| --- | --- |
| Objectives | No adjustments have been made to the project objectives defined in Phase 1. |

| Methodology | No adjustments have been made to the Scrum methodology. This has been maintained and has been fundamental in managing unforeseen technical  difficulties (such as the change of programming language) efficiently, allowing the team to adapt and meet the sprint deadlines. |
| --- | --- |
| Evidence of progress | The evidence presented for this progress corresponds to the key deliverables of Sprint 1, as planned:   1. Product Backlog 2. First Sprint Backlog: Document detailing the user stories and technical tasks selected from the Product Backlog to be developed in this first cycle. 3. Functional Deliverables (Software):    * Complete common expenses module (web).    * Complete reservations module (mobile). 4. First Sprint Testing Report: Results of unit and manual tests performed on the developed modules. 5. First Sprint Retrospective Minutes: Document summarizing the lessons learned, problems, and improvements identified at the end of Sprint 1.   Justification and Quality Assurance: This evidence demonstrates the actual progress of the project by showing a complete Scrum cycle. The Sprint Backlog provides evidence of detailed planning. The functional modules are tangible proof that functional software has been built that meets specific objectives and demonstrates the development of the competencies "Develop a software solution" and "Build programs and routines."  To ensure quality, the Scrum methodology was applied and testing activities were carried out (Competency: "Perform quality testing"). Although there were difficulties in implementing tests (Jest), the team ensured quality by actively researching documentation and applying identified best practices to validate components. The Retrospective Minutes demonstrate a commitment to continuous process improvement. |

| **2. Monitoring the Work Plan** |
| --- |
| Carefully examine your work plan, focusing especially on the progress status and adjustments column. |

| **APT Project Work Plan** | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Competency or competency units | Name of Activities  /Tasks | Descriptio n of Activities/ Tasks | Resourc es | Duratio n of activity | Responsibl e1 | Comments | Progress | Adjustments |
| Team communica tion and coordinatio n. | Daily meeting s | 15-minute meetings every day to synchroniz e progress, obstacles, and next steps. | Video calls, chat, Scrum board. | 15 to  20  minutes per day/Ent ire project | Entire team | Key to maintaining alignment and communicatio n. | Ongoing | None |
| Manage environme nt configurati on, application services, and databases. | VPS and environ ment configur ation | AWS  server configurat ion, installatio n of dependen cies, and setup of developm ent/testin g environm ents. | AWS,  Docker, Git. | 2 days | Full team | Mandatory initial activity for all development. | Completed | None |
| Build scalable data models / | Databas e configur ation | Design and creation of the | Postgre SQL,  pgAdmi n. | 3 days | Complete team | Requires defining initial schema and relationships. | Completed | None |



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.

| Program queries or routines to manipulate informatio n in the database. |  | centralize d database in PostgreSQ L. |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Develop a software solution using systematic developme nt and maintenan ce techniques  / Implement comprehen sive systemic solutions. | Commo n expense s module interfac e (web) | Screens for viewing and editing common expenses in Django frontend. | React, Vite | 1 week | John Herrera | Connects to common expenses API. | Completed | The technology was changed from "Django frontend" to "React, Vite" due to compatibility issues. |
| Develop a software solution using systematic developme nt and maintenan ce techniques  / Build programs and routines of varying complexity with good coding practices. | Backend calculati on of commo n expense s | Developm ent of logic and endpoints for the calculatio n of common expenses. | Node, JavaScri pt Express  ,  Postgre SQL. | 2 weeks | Erick San Martín and John Herrera | Direct integration with frontend and app. | Completed | Technology changed to Node. |
| Develop a software solution using systematic developme | Reservat ion module (mobile app) | Developm ent of screens and booking | React Native, Expo, Node, JavaScri pt | 2 weeks | Alexander Pulgar | Requires backend booking endpoints. | Completed | The technology was changed from "Ionic" to "React Native, Expo" |

| nt and maintenan ce techniques  / Implement comprehen sive systemic solutions. |  | logic in Ionic. | Express  ,  Postgre SQL. |  |  |  |  | due to compatibility issues. |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Perform quality testing on both products and processes. | Initial testing | Testing of common expense and reservatio n modules. | Jest, manual testing. | Midwee k | Full team | Validation of the first product increment. | Completed | Took longer than estimated due to difficulties in implementin g Jest, requiring additional research. |
| Perform quality testing on both products and processes. | Supplies inventor y interfac e (web) | Developm ent of web form for CRUD of supplies. | React, Vite | 1 week | John Herrera | Stock validations. | Not started | None |
| Build scalable data models / Program queries or routines to manipulate informatio n in the database. | Backend inventor y of supplies | CRUD API  for supplies with persistenc e in PostgreSQ L. | Node, JavaScri pt Express  ,  Postgre SQL. | 2 weeks | Erick San Martín and John Herrera | Connects to web interface. | Not started | None |
| Develop a software solution using systematic developme nt and maintenan ce | Receipt upload module (mobile app) | Function in the app for uploading receipts (PDF/imag e). | React Native, Expo, Node, JavaScri pt Express  , | 1 week | Alexander Pulgar | Validate file format before uploading | Not started | None |

| techniques  / Implement comprehen sive systemic solutions. |  |  | Postgre SQL. |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Develop a software solution using systematic developme nt and maintenan ce techniques  / Implement comprehen sive systemic solutions. | Commo n expense s query module (mobile app) | App screen to view common expenses to be paid. | React Native, Expo, Node, JavaScri pt Express  ,  Postgre SQL. | 1 week | Alexander Pulgar | Requires stable common expense API. | Not started | None |
| Perform quality testing on both products and processes. | Interme diate testing | Inventory, receipts, and common expense testing on mobile devices. | Jest, manual testing. | Midwee k | Full team | Cross- validations of modules. | Not started | None |
| Develop a software solution using systematic developme nt and maintenan ce techniques  / Implement comprehen sive systemic solutions. | Reservat ion query interfac e (web) | Screens for concierge/ administra tor to view reservatio ns. | React, Vite | Midwee k | John Herrera | Connects to the reservation backend. | Not started | None |

| Build programs and routines of varying complexity  / Implement comprehen sive systemic solutions. | Reservat ion query backend | Endpoints to return reservatio ns to the web frontend. | Node, JavaScri pt Express  ,  Postgre SQL. | 1 week | Erick San Martín and John Herrera | Connection with mobile reservation module. | Not started | None |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Develop a software solution using systematic developme nt and maintenan ce techniques  / Implement comprehen sive systemic solutions. | Logbook interfac e (web) | Screen for notes, access control, logbook. | React, Vite | Midwee k | John Herrera | No comments. | Not started | None |
| Build programs and routines of varying complexity  / Program queries or routines to manipulate informatio n in the database. | Backend log | CRUD API  for storing log records. | Node, JavaScri pt Express  ,  Postgre SQL. | 1 week | Erick San Martín and John Herrera | Must guarantee persistence and fast querying. | Not started | None |
| Develop a software solution using systematic developme nt and | Login and registrat ion view (web) | Developm ent of user login and registratio n screens. | React, Vite | Midwee k | John Herrera | Requires integration with backend. | Not started | None |

| maintenan ce techniques  / Implement comprehen sive systemic solutions. |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Build programs and routines of varying complexity  / Resolve systemic vulnerabilit ies and comply with security standards. | Backend login and registrat ion | Authentic ation API and user roles. | Node, JavaScri pt Express  ,  Postgre SQL. | Midwee k | Erick San Martín | Includes credential validation and encryption. | Not started | None |
| Implement comprehen sive systemic solutions / Resolve systemic vulnerabilit ies and comply with security standards. | Mobile app login | Implemen tation of login in the mobile app (Ionic). | React Native, Expo, Node, JavaScri pt Express  ,  Postgre SQL. | Midwee k | Alexander Pulgar | No comments | Not started | None |
| Build the architectur al model of the systemic solution / Implement comprehen sive systemic solutions. | Module integrati on | Connectio n of all web, backend, and mobile app modules. | Git, REST APIs,  Postma n | 2 weeks | Full team | May require endpoint and UI  adjustments. | Not started | None |

| Perform quality testing on both products and processes / Implement comprehen sive systemic solutions. | Final testing and closure | Regressio n, functional, and usability testing for the entire system. | Jest, manual testing. | 1 week | Full team | Final validation prior to launch. | Not started | None |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |

| **3. Adjustments based on monitoring** |
| --- |
| Delve deeper into the observations in your work plan. Analyze the planned activities and point out which aspects facilitated or hindered the execution of the plan. Discuss how you addressed and/or will address the obstacles. Finally, point out the adjustments you made to the work plan based on this analysis. |





