Project Blueprint

IT in a real organization

André Martins, Manon, Elvin, Leonardo Coimbra and Tomás Coimbra

Lamtco Solutions

0230991223@uni.lu

November 10, 2025

Contents

1 Introduction

In this document you will find the main definition of this project. You will find the Specification definition as well as the main system design. There will also be a section dedicated to the current system's risks.

2 Specification

The Specification has been defined in three different levels: **Epics, User stories and Use cases**. All the different definitions can be found in the following pages.

2.1 Epics

2.1.1 Epic: User system

Epic ID: EPIC-1 **Status:** `Proposed` **Priority:** `High`
Theme: Connecting all the users (employees) to the app infrastructure
Created: 15/10/2025 **Last Updated:** 15/10/2025
Target Release: 08/12/2025

Epic Overview

Business Problem:

Currently managing working hours always requires each employee to have additional hardware (badge system, need of a computer, ...) or they still write all their working hours in a paper format.

In order to solve this issue, he have decided to implement a mobile application software that would allow us all employees to use their personal phones in order to book all their working hours.

Business Value:

The software eliminates the need for additional hardware and digitalizes the entry of working hours, improving efficiency and avoiding trickery.

Success Metrics: - Metric 1: Being able to add hours into the system by no more that 5 button actions - Metric 2: Support all platforms - Metric 3: [Still to continue]

Scope

In Scope

- Build an intuitive, easy-to-use mobile application

- Send localization and timings of the user to the data server
- Make attractive design for employees

Out of Scope

- Application integrated into company car software
- Fully automated service (by entering a certain location with your device)

User Personas

- **Employee User:** Enters the start/end of their shift.

User Stories

Story ID	Story Title	Status	Priority
[US-1]	Employer Shift Monitoring	`Backlog`	`Low`
[US-2]	Employee Shift Punch-in (Start Shift)	`Backlog`	`High`
[US-3]	Employee Shift Punch-out (End Shift)	`Backlog`	`High`
[US-4]	Employee Reminders	`Backlog`	`Low`

2.1.2 Epic: Timelink Admin Platform

Epic ID: EPIC-002 **Status:** `Proposed` **Priority:** `High` **Theme:** Helping managers and CEO's manage the working hours of their employees

Created: 15/10/2025 **Last Updated:** 16/10/2025 **Target Release:** 08/12/2025

Epic Overview

Business Problem: Currently all managers do quite struggle to manage and monitor the working hours of their workers and working places respectively. It is really easy to make lose the amount of labor that has been relaised in a working place or individual in live and in an easy way.

Business Value: The Admin Platform helps companies save time and reduce errors by showing all information in one clear system. It makes it easier to see all the data represented in several graphical representations.

Success Metrics: - Reduce manual work for administrators by at least 60% - 100% synchronization between Admin Platform and Data Server - Reach 95% user satisfaction in the first release - System uptime above 99.9% during testing

Scope

In Scope

- Build a web-based Admin Platform using React Native
- Connect the Admin Platform to the main Data Server (Spring Boot / SQL)
- Create login and role-based access (admin users only)
- Add dashboards with tables and graphs for all the employee data
- Allow basic CRUD functions (create, read, update, delete) for all employee data

Out of Scope

- Full integration with external systems (like GPS tracking)
- Data export in XML or other formats
- Mobile version for admin users (web only for now)

User Personas

- **Admin User:** Works in HR or management. Needs to see and manage all employee data, hours, and working places quickly.
- **Company Manager:** Supervises multiple workplaces and wants to be able to easily review statistics and reports.

User Stories

Story ID	Story Title	Status	Priority
US-5	As an admin, I want to log in securely to the Admin Platform	`Backlog`	`High`
US-6	As an admin, I want to view employee working hours and working places	`Backlog`	`High`
US-7	As an admin, I want to see graphs of total hours worked per working place	`Backlog`	`Medium`
US-8	As an admin, I want to add and remove users from the system	`Backlog`	`Medium`

Story ID	Story Title	Status	Priority
US-9	As an admin, I want to see connection status with the Data Server	`Backlog`	`Low`
US-10	As an admin, I want to filter employee data by department or project, so that I can analyze work distribution more efficiently	`Backlog`	`Medium`
US-11	As an admin, I want to edit working places and assign employees to them, so that I can maintain an up-to-date organizational structure	`Backlog`	`High`
US-12	As an admin, I want to view an activity log of recent changes in the system, so that I can track who modified what and when	`Backlog`	`Medium`

Technical Considerations

Architecture Impact: - The Admin Platform will communicate directly with the Data Server using a REST Controller. - Requires a new web frontend built with React Native for Web. - Backend data comes from the existing Spring Boot server with SQL database.

Dependencies: - Data Server must be ready before Admin Platform API integration.

Risks & Mitigations: - **Risk 1:** Admin Platform may load slowly with big data sets – *Mitigation:* Optimize queries. - **Risk 2:** Inconsistent data between server and UI – *Mitigation:* Add sync checks and server-side validation.

Non-Functional Requirements:

- **Performance:** Pages should load in under 2 seconds. - **Security:** Only authorized admin users can log in. - **Scalability:** Should support up to 100 admin users per company. - **Reliability:** 99.9% uptime expected in testing phase.

2.2 User Stories

2.2.1 User Story 01

Epic: EPIC-01

01: Employer Shift Monitoring **Status:** `Backlog`

As a(n) Employee,
I want to see a list of all my logged hours,
So that I can monitor attendance and verify shift compliance.

Acceptance Criteria: - [ ] Show weekly, monthly, or yearly totals per employee. - [ ] Show hours in graphical diagram (example: histogram). - [ ] View detailed employee information (site, total hours, employee since, etc.).

Technical Notes: - Graphical Diagrams could be generated at the end of each week.

Estimate: 2-3 Story Points **Priority:** `Low`

2.2.2 User Story 02

Epic: EPIC-01

02: Employee Shift Punch-in (Start Shift) **Status:** `Backlog`

As a(n) Employee,
I want to start my shift by pressing a "Start Shift" button in the application,
So that the system may begin logging my working hours accurately.

Acceptance Criteria: - [ ] The application prompts a confirmation before starting a shift. - [ ] Access the "Start Shift" button within 3 taps or fewer from opening application. - [ ] The application indicates clearly that a shift has started.

Technical Notes: - Button sizes should be considerably large to take gloves and thicker fingers into consideration.

Estimate: 2 Story Points **Priority:** `High`

2.2.3 User Story 03

Epic: EPIC-01

03: Employee Shift Punch-out (End Shift) **Status:** `Backlog`

As a(n) Employee,
I want to end my shift by pressing a "End Shift" button in the application,
So that the system may log the end of my shift accurately.

Acceptance Criteria: - [ ] The app prompts a confirmation before ending a shift. - [ ] Getting to "End Shift" button should take no more than 3 clicks. - [ ] The application

indicates clearly that the shift has ended. - [ ] Upon ending a shift, the total hours worked are shown.

Technical Notes: - Button sizes should be considerably large to take gloves and thicker fingers into consideration.

Estimate: 2 Story Points
Priority: `High`

2.2.4 User Story 04

Epic: EPIC-01

04: Employee Reminders **Status:** `Backlog`

As a(n) Employee,
I want to receive a notification if I forget to start or end my shift,
So that I don't miss logging my hours.

Acceptance Criteria: - [ ] The app checks for missing start/end times daily. - [ ] Notifications appear on the phone at customizable times.

Estimate: 1 Story Points **Priority:** `Low`

2.2.5 US-5

Epic: EPIC-002

US-101: As an admin, I want to log in securely to the Admin Platform **Status:** `Backlog`

As a admin **I want to** log in securely to the Admin Platform **So that** only authorized users can access administrative functions

Acceptance Criteria: - [ ] The login form requires a valid username and password - [ ] Passwords are stored and transmitted securely (e.g. hashed and encrypted) - [ ] Invalid login attempts are logged - [ ] Successful login redirects to the Admin Dashboard

Technical Notes: - Use secure authentication (e.g. JWT) - Implement HTTPS for all connections - Include session timeout after inactivity

Estimate: 3 Story Points **Priority:** `High`

2.2.6 US-6

Epic: EPIC-002

US-102: As an admin, I want to view employee working hours and working places **Status:** `Backlog`

As a admin **I want to** view employee working hours and their working locations **So that** I can monitor attendance and manage work distribution

Acceptance Criteria: - [ ] Admin can view a list of all employees - [ ] Each employee entry displays total working hours and current/last working place - [ ] Data can be filtered by date or employee - [ ] Information updates dynamically from the Data Server

Technical Notes: - Fetch data via API from Data Server - Ensure data consistency for large datasets

Estimate: 3-4 Story Points **Priority:** `High`

2.2.7 US-7

Epic: EPIC-002

US-103: As an admin, I want to see graphs of total hours worked per working place Status: `Backlog`

As a admin **I want to** see visual representations of total working hours per location **So that** I can easily analyze workload distribution

Acceptance Criteria: - [ ] Admin can view bar or pie charts of total hours per working place - [ ] Charts update dynamically when filters change - [ ] Data source matches what is displayed in the working hours overview

Technical Notes: - Use a chart library (e.g. d3.js) - Ensure responsiveness for desktop and tablet views

Estimate: 4-5 Story Points **Priority:** `Medium`

2.2.8 US-8

Epic: EPIC-002

US-104: As an admin, I want to add and remove users from the system Status: `Backlog`

As a admin **I want to** add and remove users from the system **So that** I can manage access and maintain an up-to-date user list

Acceptance Criteria: - [ ] Admin can create a new user with role and credentials - [ ] Admin can deactivate or delete existing users - [ ] Changes are reflected immediately in the system - [ ] Confirmation dialog appears before user removal

Technical Notes: - Implement CRUD endpoints for user management - Validate inputs and handle duplicate user entries - Ensure role-based access control

Estimate: 1 Story Points **Priority:** `Medium`

2.2.9 US-9

Epic: EPIC-002

US-105: As an admin, I want to see connection status with the Data Server **Status:** `Backlog`

As a admin **I want to** see the connection status with the Data Server **So that** I know if data is up-to-date and if there are connectivity issues

Acceptance Criteria: - [ ] Connection status is displayed clearly (e.g. green = connected, red = disconnected) - [ ] Status updates automatically in real time - [ ] Admin receives a warning message if the server is unreachable

Technical Notes: - Implement a ping mechanism to check connection status

Estimate: 3 Story Points **Priority:** `Low`

2.2.10 US-10

Epic: EPIC-002

US-106: As an admin, I want to filter employee data by department or project, so that I can analyze work distribution more efficiently. **Status:** `Backlog`

As a admin **I want to** filter employee data by department or project **So that** I can analyze work distribution more efficiently

Acceptance Criteria: - [ ] The dashboard updates in real-time according to filters. - [ ] Filter persists across sessions until reset.

Estimate: 3 Story Points **Priority:** `Medium`

2.2.11 US-11

Epic: EPIC-002

US-107: As an admin, I want to edit working places and assign employees to them, so that I can maintain an up-to-date organizational structure. **Status:** `Backlog`

As a admin **I want to** edit working place and assign employees to them **So that** I can maitain an up-to-date organizational structure

Acceptance Criteria: - [ ] Admin can add, rename, or remove working places. - [ ] Admin can assign or reassign employees to a working place. - [ ] Changes reflect instantly in the dashboard and reports.

Technical Notes: - Data integrity enforced with foreign keys (SQL).

Estimate: 2 Story Points **Priority:** `High`

2.2.12 US-12

Epic: EPIC-002

US-108: As an admin, I want to view an activity log of recent changes in the system, so that I can track who modified what and when. Status: `Backlog`

As a admin **I want to** view an activity log of recent changes in the system **So that** I can track who modified what and when

Acceptance Criteria: - [ ] All admin actions (add, edit, delete) are logged. - [ ] The log shows username, action type and timestamp. - [ ] Logs are viewable from a dedicated "Activity Log" tab.

Technical Notes: - Implement audit logging.

Estimate: 1 Story Points **Priority:** `Medium`

2.3 Use Cases

2.3.1 Use Case: Monitor Employee Shifts

Use Case ID: UC-1
Version: 1.0
Created: 18/10/2025
Last Updated: 25/10/2025
Priority: `Low`
Status: `Draft`
Related US: US-1

Primary Actor: Employee **Secondary Actors:** Employee
Stakeholders: Company

Brief Description: Monitoring user shifts in an easy and comprehensive way.

Trigger: User (employee) trigger the review your work button.

Preconditions: - The application is turned on and connected to the respective user

Postconditions: - Employee's working hours are displayed - Employee's information is displayed (name, id, employee_since, etc.)

Main Success Scenario

Step	Actor Action	System Response
1	Employee clicks on the "Review work button"	Trigger the system to the respective page

Step	Actor Action	System Response
2	Employee selects time that the time period	Trigger the system to generate all the visualizations during the selected time

Extensions (Alternative Flows)

2a. User does not select a time period

- **Condition:** User choses to not select any prefered time period
- **Action:** Triggers the system to automatically review the last month
- **Result:** Reviews his last month's work

3a. User does not have any hours of work executed in the respective time

- **Condition:** User choses a time period where he hasn't executed any hours of work
- **Action:** The system prompts a message saying that there are no hours executed in the selected time period
- **Result:** No visualizations will be shown

2.3.2 Use Case: Employee Shift Punch-in (Start Shift)

Use Case ID: UC-2
Version: 1.0
Created: 19/10/2025
Last Updated: 19/10/2025
Priority: `High`
Status: `Draft`
Related US: US-2

Primary Actor: - Employee (Application User)

Secondary Actors: - Employer - Team Leader - Management

Stakeholders: - Company

Brief Description: This use case describes how an employee starts logging their working hours by using a "Start Shift" function in the mobile application.
The process records the current timestamp and the GPS location (Car GPS in the future) of the user. This data is sent to the central server after the start of a shift, as soon as there is a stable internet connection (in case of connectivity issue), or after marking the end of the shift.

This feature ensures accurate time and location tracking and minimizes manual input or reporting errors.

Trigger: The user opens the mobile application and in no more than 3 taps, presses "Start Shift" button to begin recording their working hours.

Preconditions: - The user must be already logged into their account in the application. - The application must have the appropriate permissions to access time and location data of the device and time servers (critical to prevent time manipulation). - The server or local storage must be available to record the shift start data.

Postconditions: - The system successfully records the shift start timestamp and GPS coordinates. - The data is stored on the central (or queued locally for later synchronization). - The applications displays a confirmation that the shift has started.

Main Success Scenario

Step	Actor Action	System Response
1	User opens the mobile application.	The application loads the home/dashboard screen.
2 (Optional)	User navigates to shifts of a specific site/location.	The application loads a screen with a "Start Shift" button for the specific site/location.
3	User taps the "Start Shift" button.	The system validates that no active shift exists for the user.
4		The system records the current timestamp and GPS coordinates.
5		The system stores the data in the local database and/or sends it to the central server.
6		The application displays a confirmation message (example: "Shift started at 08:32").
7	User taps "Ok" on the confirmation message.	The application updates the UI to show the ongoing shift status (example: "Shift started at 08:32 (Time worked: 02:45)").

Extensions (Alternative Flows)

2a. No Internet Connection

- **Condition:** The application cannot reach the central server.

- **Action:** The system stores the shift start data locally.
- **Result:** Data will be automatically synced once the device reconnects to the internet.

3a. GPS unavailable

- **Condition:** The phone's GPS is disabled or unavailable.
- **Action:** The system records the timestamp only and logs a "Location Missing" flag or return location as "null".
- **Result:** The shift still starts successfully, butwith a warning message.

4a. Duplicate Shift Attempt

NOTE: This should probably never occur but safety fallbacks should still be implemented!

- **Condition:** User tries to start another shift while one is already active.
- **Action:** The system prevents the action and shows an error message (example: "Shift already active").
- **Result:** The application handles the issue in an automated manner without creating duplicate entries.

Special Requirements

Performance:

- The system should record the shift start within **about 2 seconds** of user action. - The "Start Shift" button should be accessible within **3 taps or fewer** from the launch of the application.

Security:

- User authentication via secure credentials or company SSO. - Shift data must be encrypted during transmission.

Reliability:

- System should handle temporary network outages by caching data locally. - The application should automatically retry synchronization when connection is restored.

Technical Constraints:

- Mobile app must be compatible with Android at launch (iOS can be released at a later stage). - Integration of GPS should keep vehicle tracking system in mind.

Open Issues

- Should location tracking be mandatory or optional due to privacy concerns?
- How will unsuccessful shift start be handled when the user performs the actions correctly but there is a failure in the system? (Avoid creating issues to employees)

2.3.3 Use Case: Employee Shift Punch-in (Start Shift)

Use Case ID: UC-3
Version: 1.0
Created: 19/10/2025
Last Updated: 19/10/2025
Priority: `High`
Status: `Draft`
Related US: US-3

Primary Actor: - Employee (Application User)

Secondary Actors: - Employer - Team Leader - Management

Stakeholders: - Company

Brief Description: This use case describes how an employee stops logging their working hours by using a "End Shift" function in the mobile application.
The process records the current timestamp and the GPS location (Car GPS in the future) of the user. This data is sent to the central server after the end of a shift or as soon as there is a stable internet connection (in case of connectivity issue).
This feature ensures accurate time and location tracking and minimizes manual input or reporting errors.

Trigger: The user opens the mobile application and in no more than 3 taps, presses "End Shift" button to stop recording their working hours.

Preconditions: - The user must be already logged into their account in the application. - The application must have the appropriate permissions to access time and location data of the device and time servers (critical to prevent time manipulation). - The server or local storage must be available to record the shift end data.

Postconditions: - The system successfully records the shift end timestamp and GPS coordinates. - The data is stored on the central (or queued locally for later synchronization). - The applications displays a confirmation that the shift has ended. - The total hours worked are calculated (slightly rounded up) and displayed to the user. - The shift entry is marked as completed in the database.

Main Success Scenario

Step	Actor Action	System Response
1	User opens the mobile application.	The application loads the home/dashboard screen.
2 (Optional)	User navigates to shifts of a specific site/location.	The application loads a screen with a "End Shift" button for the specific site/location.

Step	Actor Action	System Response
3	User taps the “End Shift” button.	The system validates that an active shift exists for the user.
4		The system records the current timestamp and GPS coordinates.
5		The system stores the data in the local database and/or sends it to the central server.
6		The application displays a confirmation message (example: “Shift ended at 15:28. Total hours worked: 7 h 56 min”).
7	User taps “Ok” on the confirmation message.	The application updates the UI to show “Start Shift” once again.

Extensions (Alternative Flows)

2a. No Internet Connection
- **Condition:** The application cannot reach the central server.
- **Action:** The system stores the shift end data locally.
- **Result:** Data will be automatically synced once the device reconnects to the internet.

3a. User Cancels Confirmation
- **Condition:** User presses “Cancel” in the confirmation dialog.
- **Action:** The system aborts the operation.
- **Result:** The shift remains active and the recorded shift end data is wiped.

4a. GPS unavailable
- **Condition:** The phone’s GPS is disabled or unavailable.
- **Action:** The system records the timestamp only and logs a “Location Missing” flag or return location as “null”.
- **Result:** The shift still ends successfully, but with a warning message.

5a. No Active Shift Found
NOTE: This should probably never occur but safety fallbacks should still be implemented!
- **Condition:** User attempts to end a shift when none is active.
- **Action:** The application displays an error (example: “No active shift to end”).
- **Result:** Action terminates without recording any data.

Special Requirements

Performance:
- End-of-shift confirmation and processing should complete **within 3 seconds**.
- The "End Shift" button should be accessible within **3 taps or fewer** from the launch of the application.

Security:
- User authentication via secure credentials or company SSO. - Shift data must be encrypted during transmission.

Reliability:
- System should handle temporary network outages by caching data locally.
- The application should automatically retry synchronization when connection is restored.

Technical Constraints:
- Mobile app must be compatible with Android at launch (iOS can be released at a later stage).
- Integration of GPS should keep vehicle tracking system in mind.

Open Issues

- Should location tracking be mandatory or optional due to privacy concerns?
- How will unsuccessful shift end be handled when the user performs the actions correctly but there is a failure in the system? (Suggestion: Have a fixed amount of hours for the shift of each employee (usually 8 hours). Once 8 hours have passed, end shift automatically.)
- Should the confirmation dialog include additional notes or comments (e.g., "Reason for early departure")?

2.3.4 Use Case: Employee Reminder

Use Case ID: UC-4
Version: 1.0
Created: 19/10/2025
Last Updated: 19/10/2025
Priority: `Low`
Status: `Draft`
Related US: US-4

Primary Actor: - Employee (Application User)

Secondary Actors: - Employer - Team Leader - Management

Stakeholders: - Company

Brief Description: This use case describes how the mobile app automatically reminds an employee to start or end their shift if they have forgotten to do so.
At scheduled times before or after the expected shift start and end times, the system checks for missing punch-in our punch-out entries and triggers a notification to prompt the user to take action.
This helps maintain accurate work-hour tracking and reduces administrative corrections.

Trigger: User manually configures one or more reminder times in the application settings.

Preconditions: - The user must be already logged into their account in the application. - The application must have the appropriate permissions to access system time. - Push notifications must be enabled for the application. - There must be at least one shift available to associate with the reminder.

Postconditions: - User receives a notification at the configured times before or after the expected shift start or end times. - User can open the application directly from the notification to start or end the shift (Suggestion: If possible implement a "Start Shift" and "End Shift" option directly to the notification. This avoids user even opening the application.).

Main Success Scenario

Step	Actor Action	System Response
1	User configures a reminder for his shift (e.g., 5 min before shift starts).	The application saves this reminder into its configuration files.
2		The application sends a push notification at defined time (e.g., "Your shift starts in 5 minutes").
3	User opens the notification on their device.	The notification provides direct action buttons (e.g., "Start Shift" or "Open App").
4a	User taps the notification.	The application opens directly to "Start Shift" or "End Shift" button screen.
or		
4b	User uses direct notification buttons to start or end shift.	System updates shift status (records data) and notification disappears

Extensions (Alternative Flows)

2a. Notifications Disabled by User
- **Condition:** The user has disabled notifications on their device (or specifically for this application).
- **Action:** The application logs that reminders cannot be sent.
- **Result:** No notification is displayed.

3a. Shift Already Started or Ended
NOTE: In case reminders are set for after shift actions.
- **Condition:** The user already took the required action before the reminder time.
- **Action:** The system cancels or suppresses the reminder.
- **Result:** No notification sent.

4a. Device Off or Do Not Disturb Mode On
- **Condition:** The user's device is off or in "Do Not Disturb" mode.
- **Action:** The notification is queued and sent once conditions allow.
- **Result:** Employee receives the notification later, with correct timestamp.

Special Requirements

Performance:
- Notifications must be delivered on time or at the very least **within 30 seconds of expected time**.

Security:
- Notification must not display sensitive data (only generic text such as "Shift Start Reminder").

Reliability:
- Reminder service should retry failed deliveries up to 3 times.
- Missed notifications (notification not sent to user) must be logged for later review.

Technical Constraints:
- Requires knowledge handling background scheduling API (Android WorkManager or iOS BackgroundTasks)

Open Issues

- Should employer be able to create a reminder which cannot be turned off?
- Should the application include recurring reminders (alert user that shift should have already started or ended)?

2.3.5 Use Case: Secure Admin Login

Use Case ID: UC-5
Version: 1.0
Created: 16/10/2025
Last Updated: 17/10/2025
Priority: `High`
Status: `Draft`
Related US: US-5
Primary Actor: Admin
Secondary Actors: Authentication Service
Stakeholders:

- **Admin:** needs secure access to the platform
- **IT/Security Team:** requires proper authentication and data protection
- **Company Management:** wants controlled access to administrative functions

Brief Description:
This use case describes how an admin securely logs in to the Admin Platform to access administrative functions, ensuring that only authorized users can enter the system.
Trigger:
The admin attempts to access the Admin Platform.
Preconditions:
- Admin has valid credentials - The Admin Platform is online and accessible
Postconditions: - Admin is authenticated and redirected to the Admin Dashboard

Main Success Scenario

Step	Actor Action	System Response
1	Admin opens the Admin Platform	System displays the login form
2	Admin enters username and password	System securely transmits and validates credentials
3	Admin is authenticated	System redirects to the Admin Dashboard

Extensions (Alternative Flows) 2a. Invalid Login
- **Condition:** Credentials are invalid
- **Action:** System displays an error and logs the attempt - **Result:** Admin may retry login

Special Requirements Performance:
- Login validation within 2 seconds

Security:

- HTTPS for all connections
- Passwords hashed and encrypted
- JWT-based authentication
- Session timeout after inactivity

Reliability:
- Authentication service uptime >= 99.5%

Technical Constraints:
- Integration with Authentication Service via REST API

Open Issues

- Define session timeout duration

2.3.6 Use Case: View Employee Working Hours and Locations

Use Case ID: UC-6
Version: 1.0
Created: 16/10/2025
Last Updated: 17/10/2025
Priority: `High`
Status: `Draft`
Related US: US-6

Primary Actor: Admin
Secondary Actors: Data Server
Stakeholders:

- **Admin:** monitors attendance and work distribution

- **Company Management:** requires up-to-date employee data
- **IT Team:** ensures data integrity and system performance

Brief Description:
This use case describes how an admin views employee working hours and locations to monitor attendance and manage workload distribution.

Trigger:
Admin selects the "Employee Overview" section.

Preconditions:

- Admin is authenticated
- Data Server connection is active

Postconditions:

- Employee working hours and locations are displayed accurately
- Filters adjust the displayed data dynamically

Main Success Scenario

Step	Actor Action	System Response
1	Admin selects “Employee Overview”	System displays employee list with hours and locations
2	Admin applies filters (by date or employee)	System updates the data dynamically
3	Admin reviews updated results	Displayed data matches selected filters

Extensions (Alternative Flows) 2a. No Data for Filter

- **Condition:** No results for selected filters
- **Action:** System shows “No data available” message
- **Result:** Admin can reset filters

Special Requirements Performance:
- Data retrieval within 2 seconds
- Filter update within 1 second

Security:
- Data transmitted securely over HTTPS

Reliability:
- System availability 99.5% - Automatic error handling for missing data

Technical Constraints:
- Integration with Data Server API
- Support for large datasets

Open Issues

- Define page size for large employee lists

2.3.7 Use Case: View Workload Charts by Location

Use Case ID: UC-7
Version: 1.0
Created: 16/10/2025
Last Updated: 17/10/2025
Priority: `Medium`
Status: `Draft`
Related US: US-7
Primary Actor: Admin
Secondary Actors: Data Server
Stakeholders:

- **Admin:** analyzes workload distribution visually
- **Company Management:** requires visual insights for decision-making
- **IT Team:** maintains data consistency and responsiveness

Brief Description:
This use case describes how an admin views visual charts (bar/pie) of total hours worked per location for better workload analysis.

Trigger:
Admin selects the "View Charts" section.

Preconditions:

- Admin is logged in
- Employee work data is available

Postconditions:

- Charts display total working hours per location
- Charts update dynamically when filters change

Main Success Scenario

Step	Actor Action	System Response
1	Admin navigates to "View Charts"	System generates visual charts for total hours per location

Step	Actor Action	System Response
2	Admin applies filters	Charts update dynamically

Extensions (Alternative Flows) 2a. No Data for Chart

- **Condition:** No matching data for filter
- **Action:** System shows "No data available" message
- **Result:** Admin may adjust filters

Special Requirements Performance:
- Chart rendering $<$ 2 seconds
Security:
- Data retrieved securely via API
Reliability:
- Charts reflect real-time data updates
Technical Constraints:
- Charts implemented with d3.js or equivalent
- Responsive design for desktop and tablet

Open Issues

- Determine chart types (bar, pie, line)

2.3.8 Use Case: Manage Users (Add/Remove)

Use Case ID: UC-8
Version: 1.0
Created: 16/10/2025
Last Updated: 17/10/2025
Priority: `Medium`
Status: `Draft`
Related US: US-8
Primary Actor: Admin
Secondary Actors: Authentication Service
Stakeholders:
- **Admin:** manages access and user roles
- **IT/Security Team:** ensures controlled access
- **Company Management:** maintains updated user records

Brief Description:
This use case describes how an admin adds or removes users to manage access and keep the system user list current.

Trigger:
Admin navigates to the “User Management” section.

Preconditions:

- Admin is logged in
- Admin has permission to manage users

Postconditions: - User list is updated and changes are reflected immediately

Main Success Scenario

Step	Actor Action	System Response
1	Admin selects “Add User” or “Remove User”	System displays user form
2	Admin enters or selects user details	System validates inputs
3	Admin confirms action	System applies change and displays confirmation

Extensions (Alternative Flows) 3a. Invalid or Duplicate Entry

- **Condition:** User data invalid or duplicate detected
- **Action:** System shows error message
- **Result:** Admin corrects input and retries

Special Requirements Performance:
- CRUD operations complete within 2 seconds

Security:
- Role-based access control
- Input validation for user data

Reliability:
- Changes instantly applied in user list

Technical Constraints:
- CRUD endpoints integrated with Authentication Service

Open Issues

- Define detailed role and permission levels

2.3.9 Use Case: Monitor Data Server Connection

Use Case ID: UC-9
Version: 1.0
Created: 16/10/2025
Last Updated: 17/10/2025
Priority: `Low`
Status: `Draft`
Related US: US-9
Primary Actor: Admin
Secondary Actors: Data Server
Stakeholders:

- **Admin:** needs visibility on data availability
- **IT Team:** monitors server uptime and connectivity
- **Company Management:** depends on data accuracy and availability

Brief Description:
This use case describes how an admin monitors the real-time connection status to the Data Server to ensure the system is up-to-date.

Trigger:
Admin checks the connection status indicator or system shows a warning.

Preconditions:

- Admin is logged in
- Connection check mechanism is active

Postconditions: - Connection status (green/red) displayed and updated in real time

Main Success Scenario

Step	Actor Action	System Response
1	Admin opens dashboard	System shows Data Server connection status

Step	Actor Action	System Response
2	Connection changes	System automatically updates the indicator

Extensions (Alternative Flows) **2a. Data Server Unreachable**

- **Condition:** Ping to server fails

- **Action:** System shows warning message and marks status red
- **Result:** Admin is informed and may contact IT support

Special Requirements **Performance:**
- Status check performed every 10 seconds
Security:
- All communication over HTTPS
Reliability:
- System availability >= 99.5%
- Good handling of temporary disconnections
Technical Constraints:
- Ping mechanism for connectivity - Real-time status updates

2.3.10 Use Case: Filter employee data by department or project

Use Case ID: UC-10
Version: 1.0
Created: 18/10/2025
Last Updated: 18/10/2025
Priority: `Medium`
Status: `Draft` **Related US:** US-10

Primary Actor: Admin
Secondary Actors: Data Server
Stakeholders:

- **Admin:** needs to quickly analyze employee distribution by department or project.

- **HR Department:** uses filtered data for performance tracking and reporting.
- **Company Managers:** need targeted insights for project management and ressources allocation.

Brief Description:

This use case describes how an admin filters employee data by department or project in order to analyze work distribution. The system dynamically updates the dashboard according to selected filtersand maitain these filters across sessions until they are reset.

Trigger:

The admin selects one or more filters (department, project) on the admin dashboard.

Preconditions:

- Admin is authenticated and logged into the admin platform.
- Employee and project data are synchronized with the data server.
- The dashboard is loaded and operational.

Postconditions:

- The system displays only employee data that match the selected filters.
- The applied filters persist across sessions until manually reset.

Main Success Scenario

Step	Actor Action	System Response
1	Admin navigates to the "employee dashboard"	System displays all employee data by default
2	Admin opens the filter panel	System displays the filter options(department, project)
3	Admin selects one or more filters	System queries filtered data from the data server
4	System retrieves the matching results	Dashboard updates dynamically to display filtered employee data
5	Admin closes the filter panel	System stores the selected filters

Extensions (Alternative Flows) 2a.No matching data

- **Condition:** there is no matching data for the given filters
- **Action:** System displays a message ("There is no matching data for the filters: ...")
- **Result:** Admin may choose other filters

Special Requirements Performance:
- Filtered results must appear in under ? seconds.

Usability:
- Filters should be triggered and combinable intuitively.

Persistence:
- Filters must persist locally until reset.

Scalability:
- Filtering must handle up to ? departments and ? projects without performance degradation.

2.3.11 Use Case: Edit working places and assign employee to them

Use Case ID: UC-11
Version: 1.0
Created: 18/10/2025
Last Updated: 18/10/2025
Priority: `High`
Status: `Draft`
Related US: US-11

Primary Actor: Admin
Secondary Actors: Data Server
Stakeholders:

- **Admin:** needs to keep employee assignments and working places up to date
- **HR Department:** relies on accurate working place data for reports
- **Company Management:** need updated employee locations for supervision and resource management

Brief Description:
This use case describes how an admin creates, edits, or deletes working places and assigns or reassigns employees to those places. The goal is to maintain an accurate and current organizational structure visible on the dashboard and in reports.

Trigger:
The admin selects the "Manage Working Places" option from the navigation menu.

Preconditions:

- Admin is authenticated and logged into the admin platform.

- Data server is online and synchronized.

- Admin has sufficient permissions to modify employee and working place data.

Postconditions: - Working place and employee assignment data are updated successfully in the system. - Updated information is reflected in dashboards and reports immediately.

Main Success Scenario

Step	Actor Action	System Response
1	Admin navigates to the "Working Places" management page	System displays the list of all current working places with assigned employees
2	Admin clicks "Add New" or selects a working place to edit	System opens a form for adding, renaming, or deleting a working place
3	Admin enters or modifies the working place details	System validates the input
4	Admin confirms changes	System saves the working place data and updates the list
5	Admin assigns or reassigns employees to a working place	System updates employee records
6	Admin saves assignments	Dashboard and reports reflect updated structure in real-time

Extensions (Alternative Flows) 2a. Invalid or duplicate working place name

- **Condition:** Admin enters invalid or duplicate working place name

- **Action:** System displays an error message and prevents saving

- **Result:** Admin should enter another name

Special Requirements Performance:
- Updates should be reflected on the dashboard within ? seconds.

Security:
- Only admin users with edit permissions can modify working place data.

Technical Constraints:
- Real-time update

2.3.12 Use Case: View system activity log

Use Case ID: UC-108
Version: 1.0
Created: 18/10/2025
Last Updated: 18/10/2025
Priority: `Medium`
Status: `Draft`
Related US: US-12

Primary Actor: Admin
Secondary Actors: Data Server
Stakeholders:

- **Admin:** needs visibility on recent actions to track modifications in the system
- **IT/Security Team:** requires a full record of admin activities for compliance
- **Company Management:** wants accountability and traceability for data integrity

Brief Description:
This use case describes how an admin accesses and reviews a detailed activity log of all recent administrative actions (add, edit, delete). The log provides transparency and accountability by displaying who performed each action, what was changed, and when it occurred.

Trigger:
The admin selects the "Activity Log" tab from the Admin Platform menu.

Preconditions:

- Admin is authenticated and logged into the Admin Platform.
- Activity data exists in the database

Postconditions:

- Admin can view the list of recent changes made in the system.
- The activity log is accurate and complete.

Main Success Scenario

Step	Actor Action	System Response
1	Admin navigates to the "Activity Log" section	System retrieves and displays recent admin actions
2	Admin clicks on a specific log entry for more details	System displays detailed information

Extensions (Alternative Flows) 2a. No activity logs available

- **Condition:** there are no activity logs
- **Action:** System displays message ("No recent activity recorded.")

- **Result:** Admin may retry later

2.4 Technologies used

Under these rather small paragraphs you shall find all the current technologies that have been defined under the context of this project. The chosen technologies for this project are the following:

- Data Server
  - `Base:` SpringBoot application
  - `Programming language:` Java
  - `User interaction:` PHPMyAdmin framework
- Timelink mobile/admin
  - `Base:` React native framework + necessary libraries for UI/design
  - `Programming language:` React
  - `User interaction:` implemented front-end

3 System Design

We have decided to define the project architecture and design in four different levels, also called **C1, C2, C3, C4**.

Currently we have defined the first 2 levels of architecture. In order to define them we have used the PlantUML tool to create several types of diagram in a code based environment.

Below, you will be able to take a look at both levels of architecture.

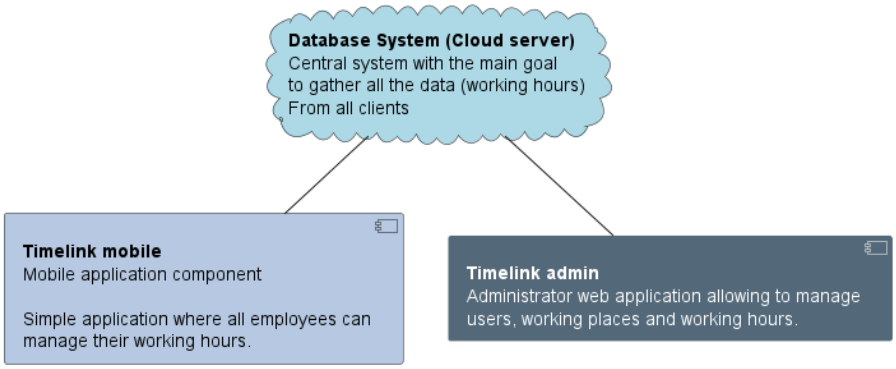


Figure 3.1: C1 architecture level

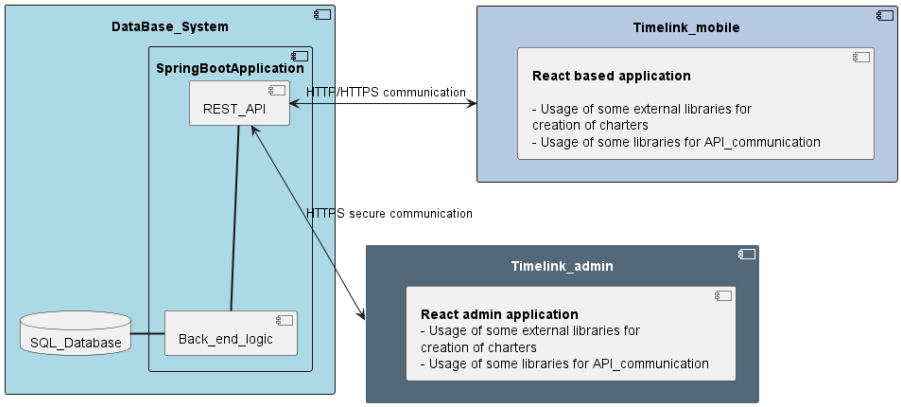


Figure 3.2: C2 architecture level

3.1 Data/Network flows

Additionally, we have also started the main communication flows for both systems, you can find down below two data flows, one for each application.

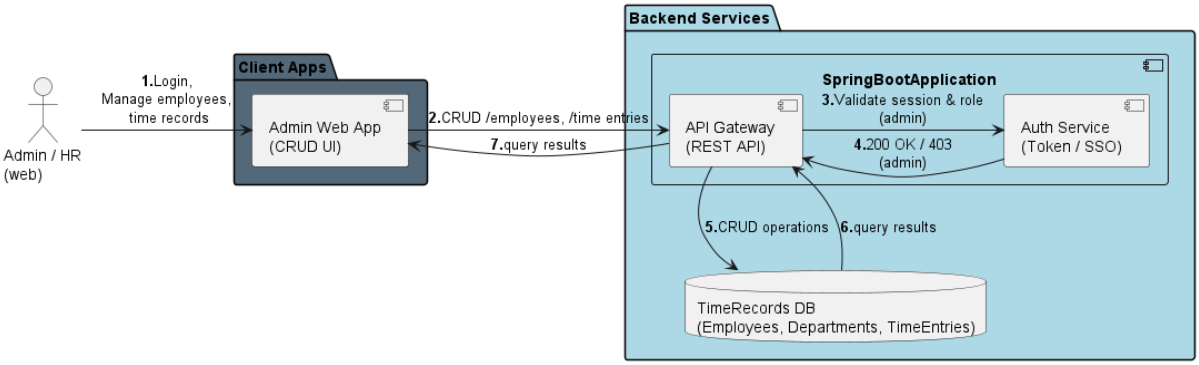


Figure 3.3: Admin app main data flow

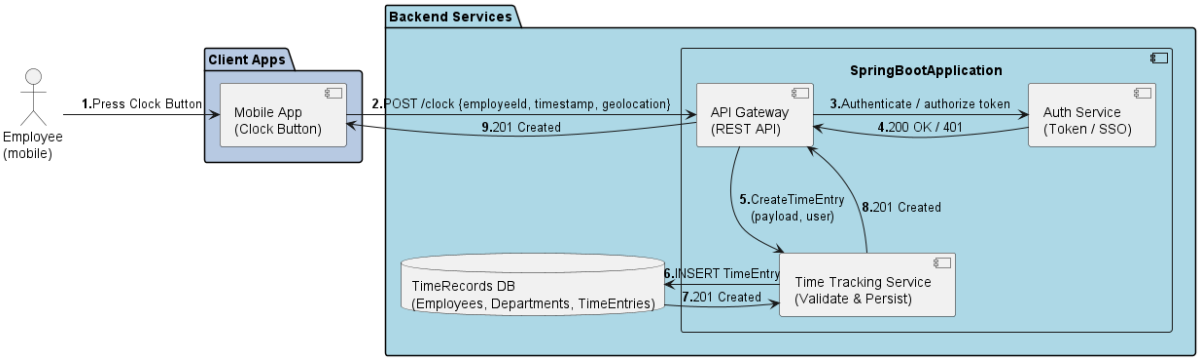


Figure 3.4: Mobile app main data flow

3.2 UI designs

Our team has also created some UI designs that will be used as a basis when implementing the UI of our apps.

3.2.1 Timelink mobile

3.2.2 Timelink admin

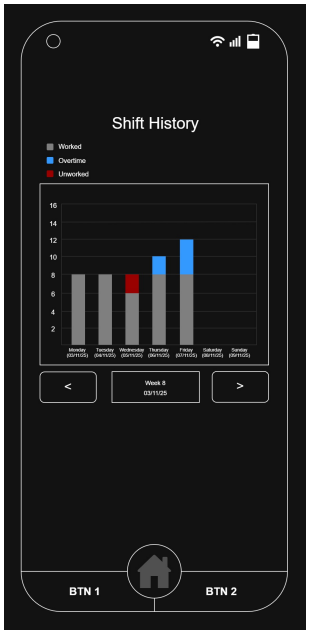


Figure 3.5: UI related to UC-01

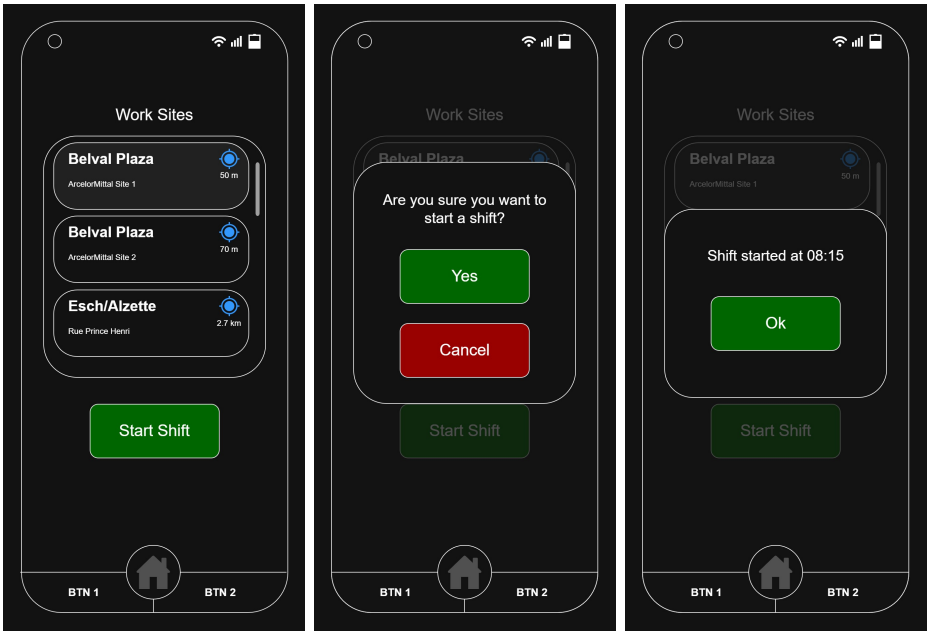


Figure 3.6: UI's related to UC-02

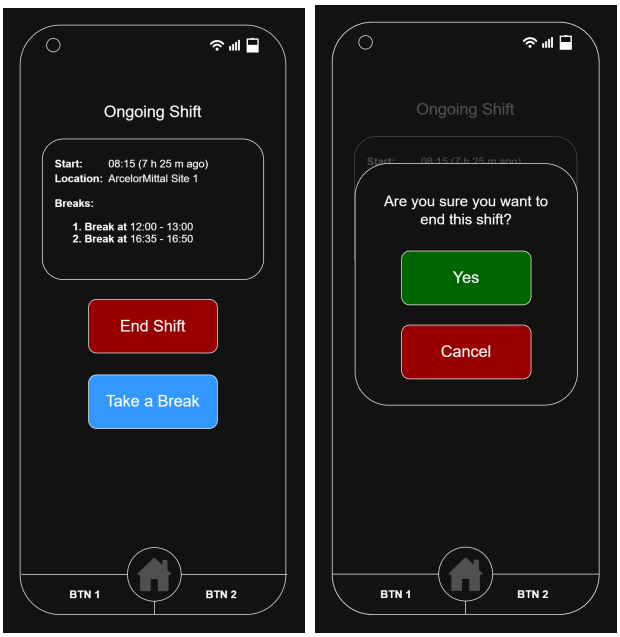


Figure 3.7: UI's related to UC-03

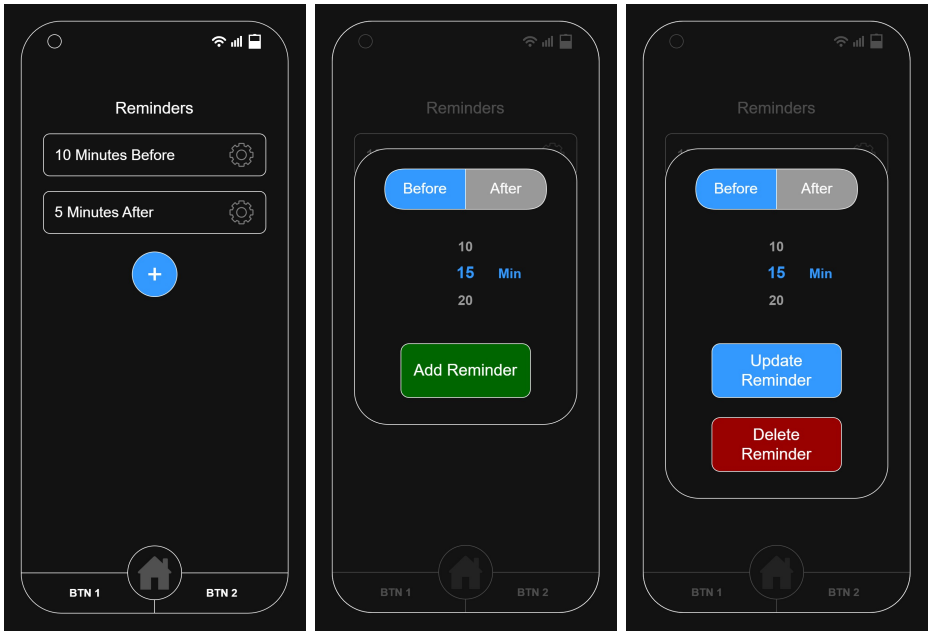


Figure 3.8: UI's related to UC-04

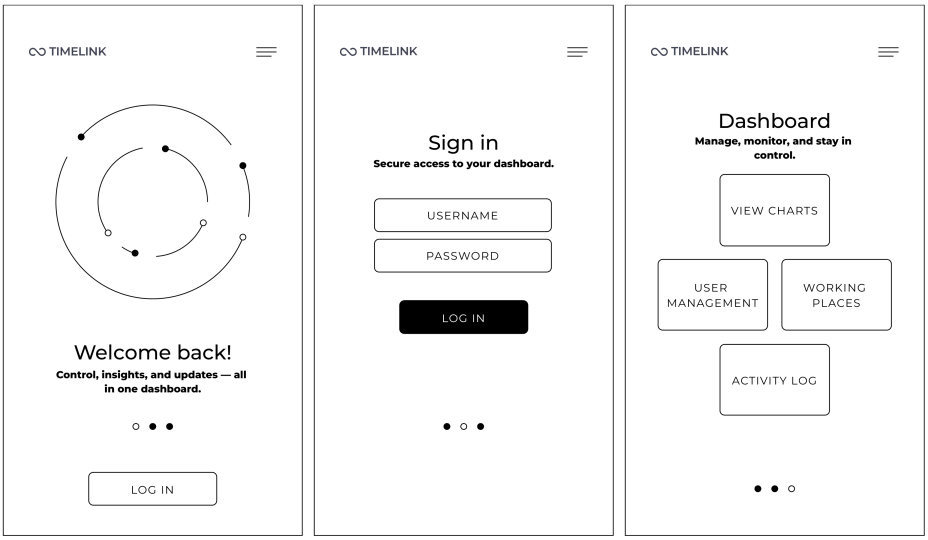


Figure 3.9: UI's related to UC-05 and UC-09

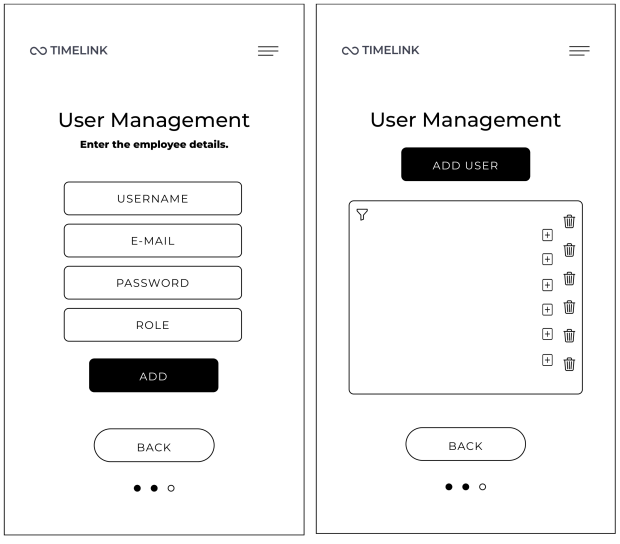


Figure 3.10: UI's related to UC-06 and UC-08

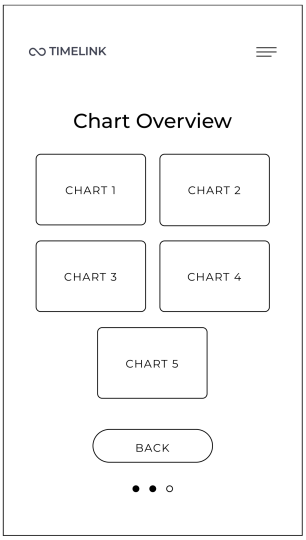


Figure 3.11: UI related to UC-07

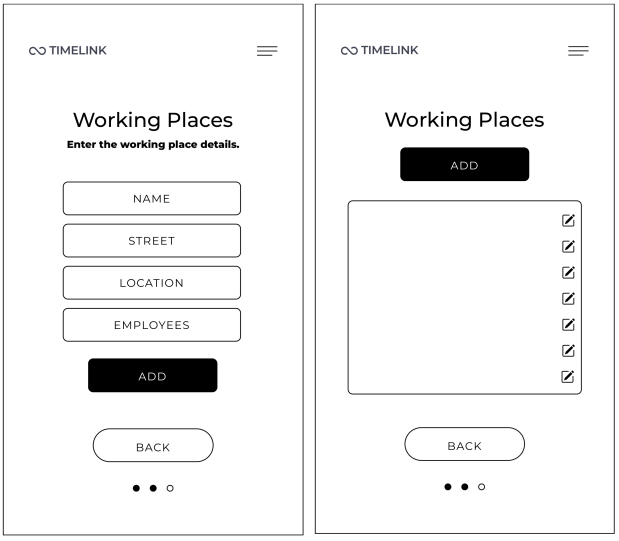


Figure 3.12: UI's related to UC-11

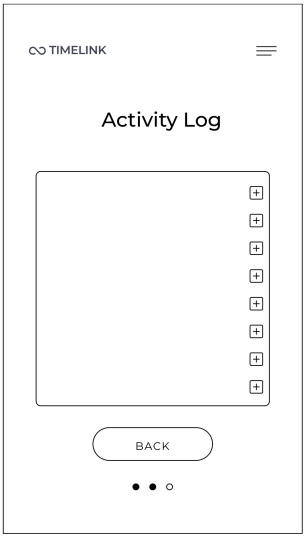
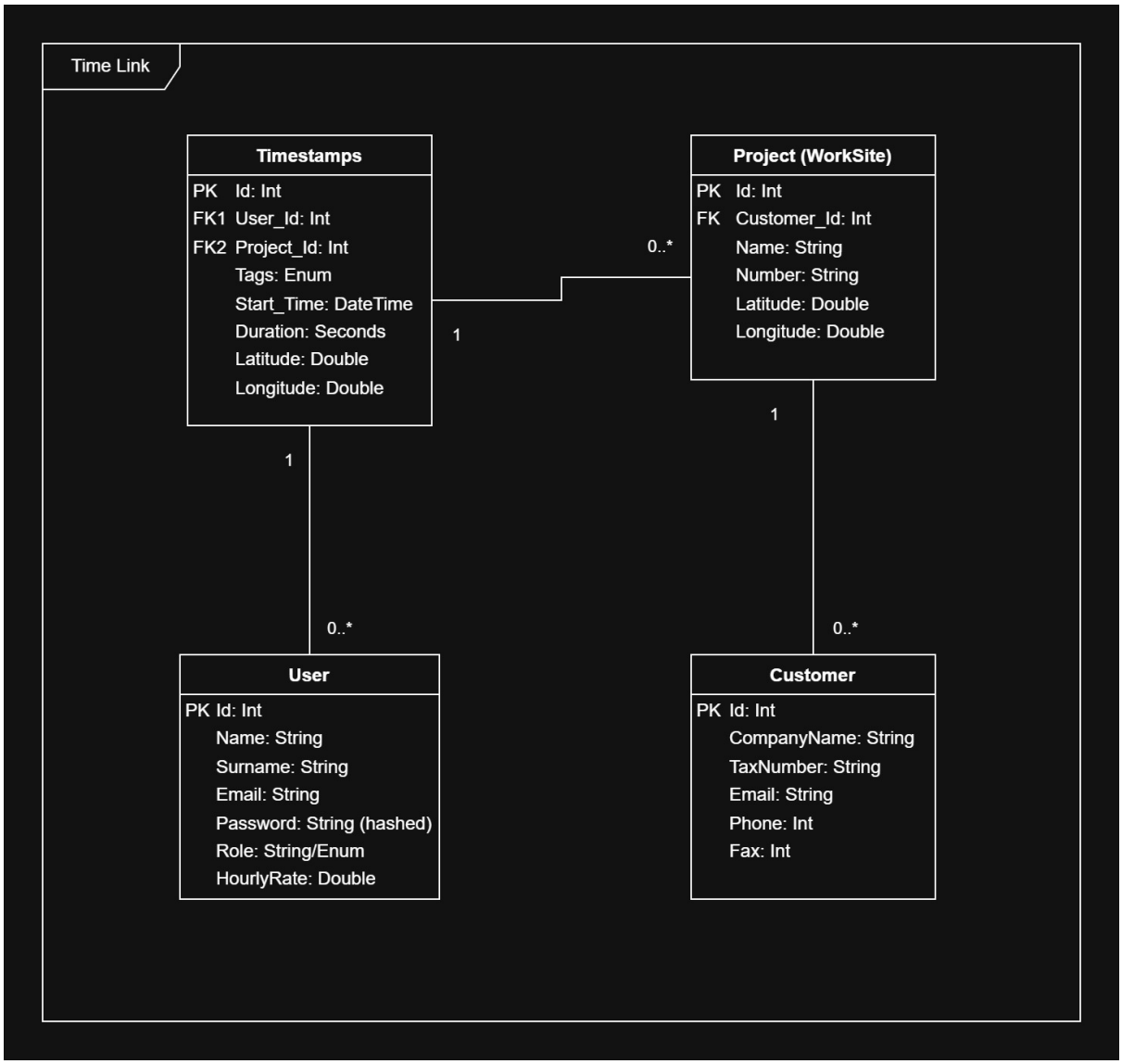


Figure 3.13: UI related to UC-12

4 Data Models

Currently we have define main data model that will be used as a base for our data server, here is the diagram that was designed as a data definition:



5 User roles

Currently, since our main focus for developing during the university's timeline will be to develop the mobile app (Timelink mobile), the main permissions that this user shall have are the following:

- General Employee user
- System owner/Admin (Someone who has access to the Data server)

5.1 General Employee user

The context of this user is someone who only will be interacting with the app via the Mobile app, here are the following permissions the user shall have:

- Creating new working hours under his name
- Change existing working hours under his name
- Delete existing working hours under his name
- Have access to his working hours in a visualization format
- Have access to his profile and change his personal preferences

One more detailed version will be created once the development allows us to identify several important points that will need to be resolved.

5.2 System owner

This user will have access to the data-server. The main interaction will be under a `PHPMyAdmin` platform (current work-around until Timelink admin will be deployed). The user shall have all the same permissions that a General employee has but shall be able to do it for all users. Additionally, the user shall also be able to do the following:

- Create new users
- Delete/modify existing users

6 Status report

In this section of the document you shall find all the current information related to the the current status of the system.

6.1 Current Achievements

Currently, we have completely achieved both `Milestone 1` and `Milestone 2`. More specifically, both specifications as well as the main design of system have been defined

6.2 Current project plan

After creating all the project specifications and starting to adding all the tasks in the project Backlog, I've realized that the current Scope will be to much for the current timeline that we have left.

To compensate this issue I've taken the decision to take the application `Timelink admin` out of development since this would possibly remove all the scope creep that we are currently suffering.

Taken this information into account, the current tasks are the following:

- Development of the Data Server
- Development of the Mobile app
- Planning and execution tests that will be executed

These current tasks will start as soon as possible and will be divided under the respective people. Since the Business analyst and the PM will not have a lot to do during the time left, they will try to help the others as much as possible.

6.3 Financial's

The current financial status of the project is the following:

- Time spent
  - 24 days spend
  - 36 days still left to deliver the final product

6.4 Project forecast

After having removed one of the deliverables of the scope (Timelink admin), I do think that this project can still be achieved within the defined timeline.

7 Testing

In this section of this document you may find all the current deliverables that are related to testing for our project.

7.1 Current identified risks

7.2 Risks

Risk name	Risk Description	Risk Impact	Solution
Regulatory non-compliance (GDPR)	The system can expose sensible data (name, ID, hire date) without respecting GDPR.	High, legal risk and reputation risk	We can either accept this risk because our project will not be public, or we can try to add compliant measures (hahsing)
Data Privacy	A user see sensible data, but he doesn't have the rights to see these.	High, legal risk and reputation risk	We need to decide which type of user can see what type of information and then implement these. Or accept this risk.

Risk name	Risk Description	Risk Impact	Solution
Incorrect data, not-updated, missing	The represented data (shifts, working hours, employees assigned to a working place, working place details, etc. . .) are not synchronized correctly because of a bug, a failure,	High	We accept it for the moment
Insufficient log retention (for activity log UC-12)	Logs expire too soon or are not archived	Low	Apply the right parameters (choose the right time to keep activity logs)
Modification conflict	Two administrators modify the same data at the same time.Overwriting of modifications, inconsistency between assignments.	High	When development, think of concurrent modification to prevent this risk
No validation prior to major modification (add, remove, edit a user or a working place)	No confirmation before modification or deletion.Risk of irreversible human error.	High	Implement a validation system, a message (“Are you sure you want to. . . ”)
Poor management of duplicates	Users, project, working places, etc.. are created multiple times	Medium	When development, think of managing duplicates to prevent this risk
Authentication service unavailable	Admin can no longer access their account, critical operations blocked.	High, service interruption	Accept this risk.

Risk name	Risk Description	Risk Impact	Solution
Unauthorized access, security failure	A compromised admin account, or a password leak that allow access to the system.	High, GDPR sanctions, data breaches, manipulation of sensitive information	Use security measures (multi-factor authentication, strong password, etc. . .)