Indiana University Southeast

RF-3

Jonathan Roberts
Sebastian Burman
Xiaokun Li
Capstone No Kill Louisville Check In
Professor Finkbine
10/13/2023

SAS Report: No Kill Louisville Animal Shelter Check-In System

Introduction:

The No Kill Louisville Animal Shelter aims to streamline its check-in process through the implementation of a check-in system. This system will be integrated with the Acuity scheduling system using an API and will leverage Twilio's text message service to facilitate seamless user check-ins. Additionally, the system will be developed using the .NET Core Web API for efficient management and communication between the systems.

Use Cases:

The check-in system's primary use cases include enabling users to conveniently check-in for their appointments, facilitating efficient data transmission between the Acuity scheduling system and the shelter's database, and sending timely notifications to users via Twilio's text message service.

Functions:

Functions within the check-in system will include the transformation of data between the Acuity scheduling system and the shelter's internal database, as well as the real-time determination of appointment statuses and availability.

Triggers:

Triggers for the check-in system will be primarily transaction-based, activated when users check in for their appointments using the integrated interface. Additionally, time triggers will be utilized for sending timely notifications and reminders.

Data Stores and Data Flows:

The system will incorporate a central data store for storing user check-in information and appointment data. Data flows will facilitate the movement of information between the Acuity scheduling system and the shelter's database, ensuring the seamless integration of check-in data.

Data Elements:

Data elements within the check-in system will encompass essential information such as user details, appointment time slots, check-in timestamps, and any additional relevant user preferences or notes.

Processes:

The check-in system will rely on automated processes for handling data transmission between the Acuity scheduling system and the shelter's database. Additionally, human processors will oversee the system's operation and address any potential issues or discrepancies.

Data Storage and Data Connections:

Data storage will be managed through a secure repository utilizing advanced cloud-based solutions, ensuring the safety and accessibility of stored data. Data connections will be established through robust communication pipelines to facilitate the seamless flow of information between the integrated systems.

Actors/External Entities:

External entities interacting with the check-in system include the shelter's clients utilizing the Acuity scheduling system and the Twilio text message service for check-in procedures and updates.

Component Properties:

- API: Acuity scheduling system API for seamless integration.
- Twilio: Text message service for timely notifications and communication.
- .NET Core Web API: Framework for efficient system development and management.

Flexibility in Gathering Component Properties:

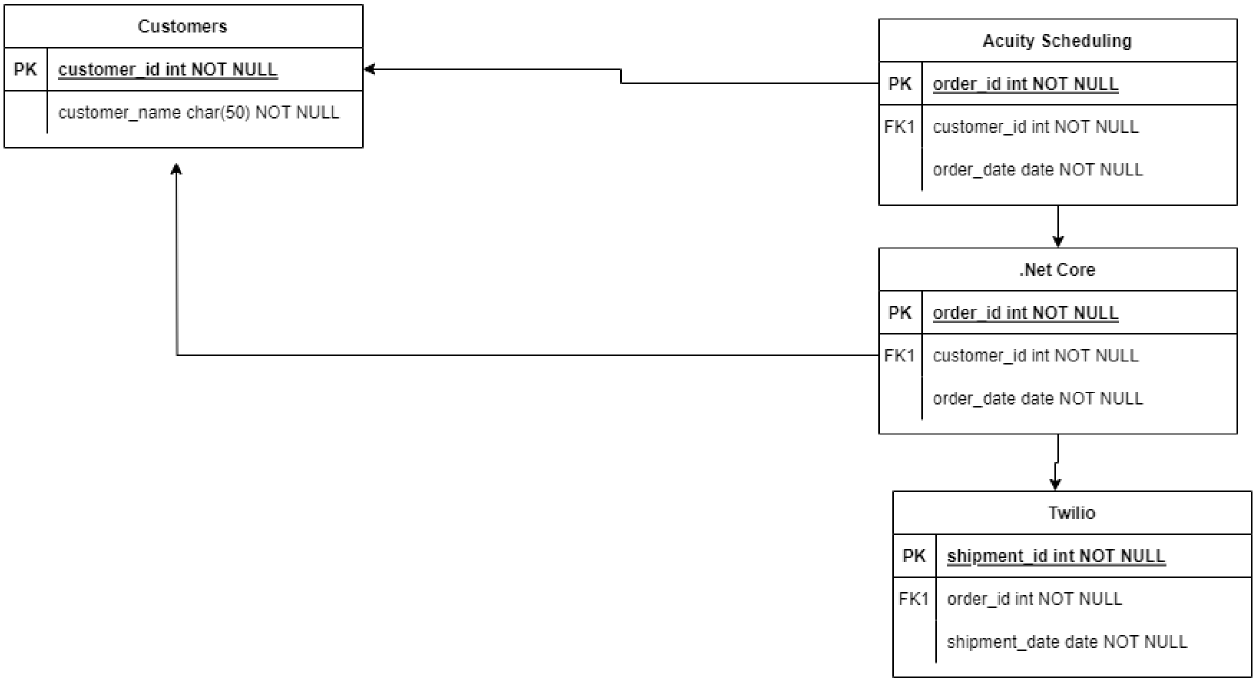
The project's flexible approach allows for the dynamic gathering and definition of component properties based on the specific requirements of the No Kill Louisville Animal Shelter's check-in system, ensuring a customized solution tailored to the organization's needs.

Conclusion:

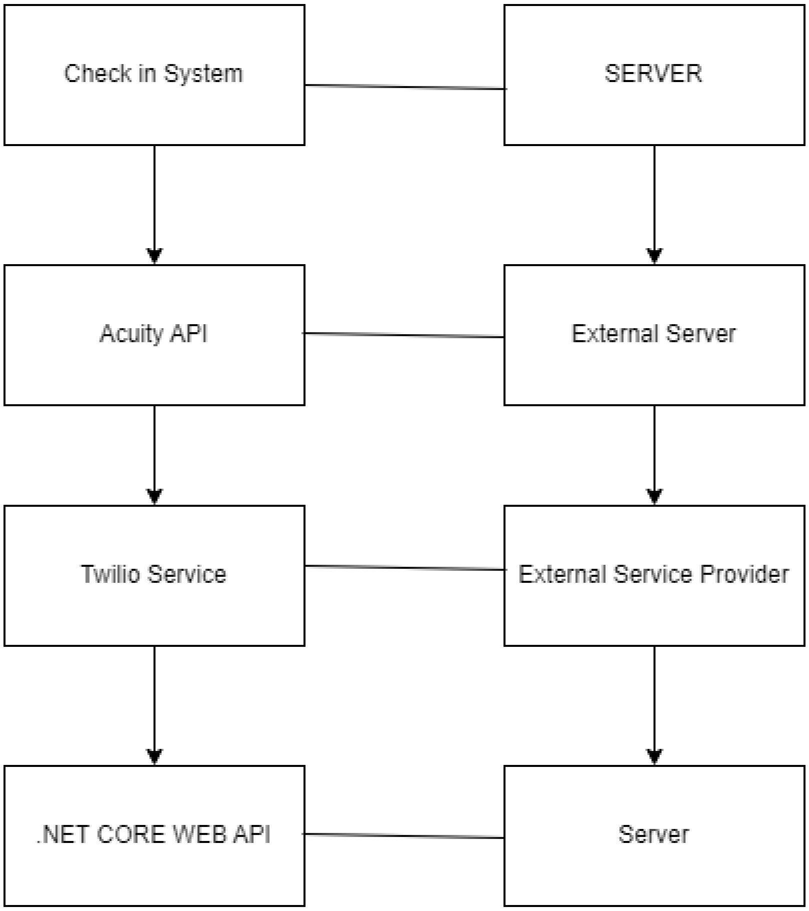
The proposed check-in system presents a comprehensive solution for the No Kill Louisville Animal Shelter, effectively streamlining their check-in process and enhancing user experience. By leveraging advanced technologies such as the Acuity API, Twilio's text messaging service, and .NET Core Web API, the shelter will significantly improve operational efficiency and communication with its clients.

Diagrams:

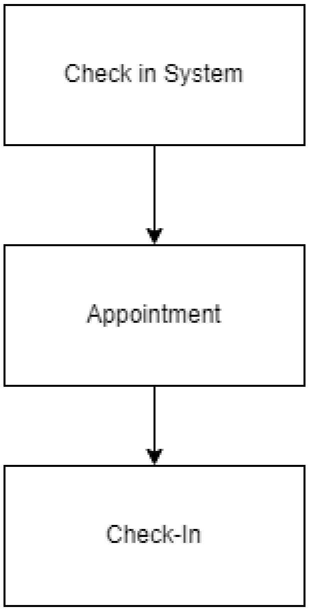
Subsystem Decomposition:



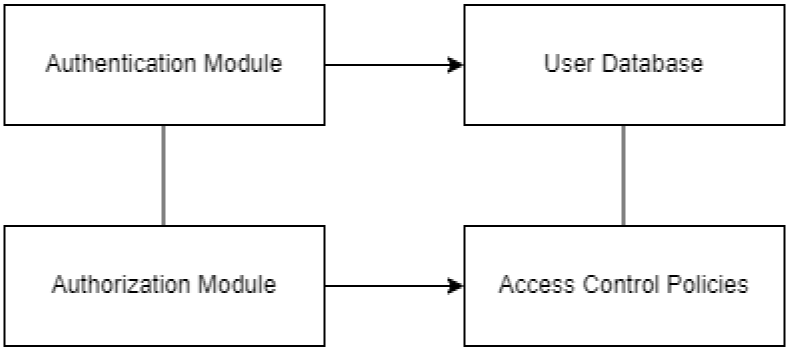
Hardware/Software Mapping:



Persistent Data Management:



Access Control and Security:



Global Software Control:

- Activity Diagram:
- Procedural Control Flow for Check-In Process
- Sequence Diagram:
- Event-Driven Flow for Notification Process
- State Diagram:
- State Transition for Appointment Status

Boundary Conditions:

- Activity Diagrams:
- System Startup and Shutdown Processes
- Error Handling and Exception Responses

- Deployment Diagram:
- Migration to New Server
- Activity Diagrams:
- Bulk Data Dump and Load Processes

Individual Contributions

Li:

- **System Integration:** Li will primarily focus on integrating the check-in system with the Acuity scheduling system using the API. This will involve thorough understanding of API functionalities, data mapping, and ensuring seamless data transfer between the systems.

- **Data Management:** Li will be responsible for overseeing the data storage and data flow aspects, ensuring that the data from the check-in system is accurately stored, updated, and synchronized with the shelter's database.

- **Quality Assurance:** Li will play a critical role in testing the integrated system, identifying potential issues, and ensuring that the check-in system meets the highest quality standards before its deployment.

Jonathan:

- **Twilio Integration**: Jonathan will focus on integrating the Twilio text message service into the check-in system, enabling the system to send real-time notifications and updates to users. This will involve setting up the necessary communication protocols and ensuring the smooth functioning of the messaging service.

- **User Experience:** Jonathan will be responsible for designing an intuitive and user-friendly interface for the check-in system, ensuring that users can easily navigate through the check-in process and receive clear and concise notifications via the Twilio service.

- **Documentation:** Jonathan will oversee the documentation process, ensuring that all system functionalities, user guides, and technical documentation are well-documented and readily available for future reference and troubleshooting.

Sebastian:

- **.NET Core Developer:** Sebastian will focus on developing the core functionalities of the check-in system using .NET Core Web API. This will involve designing and implementing the necessary backend processes, data handling functionalities, and communication protocols for the system.

- **Technical Lead:** Sebastian will be the main point of contact for resolving any technical issues or concerns related to the check-in system. This will involve providing timely support to team members, addressing any technical challenges, and ensuring the smooth operation of the system throughout its lifecycle.

- **Project Coordinator:** Sebastian will take on the responsibility of coordinating tasks, managing timelines, and ensuring effective communication between team members. Sebastian will also be responsible for tracking project milestones and ensuring that the project progresses according to the established timeline.