D3.6 - First dialogue system with integrated physiological monitoring

Project acronym:	ALIAS
Project name:	Adaptable Ambient Living Assistant
Strategic Objective:	ICT based solutions for Advancement of Social Interaction of Elderly People
Project number:	AAL-2009-2-049
Project Duration:	July, 1st 2010 – June, 30th 2013 (36months)
Co-ordinator:	Prof. Dr. Frank Wallhoff
Partners:	Technische Universität München Technische Universität Ilmenau Metralabs GmbH Cognesys GmbH EURECOM Guger Technologies Fraunhofer Gesellschaft pme Familenservice YOUSE GbR

D3.6

Version:	1.0
Date:	2011-06-30

Author:
Tobias Rehrl
Jürgen Geiger
Maja Golcar
Stefan Gentsch
Jan Knobloch

Dissemination status: PU

This project is co-funded by the Ambient Assisted Living (AAL) Joint programme, by the German BMBF, the French ANR, the Austrian BMVIT.

Once completed please e-mail to WP leader with a copy to

eric.bourguignon@tum.de and frank@wallhoff.de.

Del 3.6	Executive Summary
This deliverable describes the physiological monitoring system within the dialog system. The physiological monitoring system is implemented as a web-based solution and in this way integrated into the ALIAS system on the robotic platform. The storing of the physiological data is achieved via a SQL database, where multiple users can be handled. The connection towards the SQL database and the system is established via Hibernate, which provides a mapping between Java objects and SQL databases. The physiological monitoring system is a web-based service, thus the data entry as well as the data visualization are managed via the web browser. This approach facilitates remote access onto the data from authorized persons (e.g. doctors, relatives, etc.) as well as an easy interface for the primary user.	

Dissemination Level of this deliverable	
PU	Public
Nature of this deliverable	
R	Report

Due date of deliverable	2011-06-30
Actual submission date	2011-06-30
Evidence of delivery	2011-06-30

Authorisation			
No.	**Action**	**Company/Name**	**Date**
1	Prepared	TUM-MMK, Tobias Rehrl	2011-06-30
2	Approved	Guger Technologies, Christoph Hintermüller	2011-07-19
3	Released	Cognesys, Stephanie Lamp-Emden	2011-08-16

Table of Contents

List of Abbreviations

ALIAS	Adaptable Ambient Living ASsistant
ECG	Electrocardiograph
GSR	Galvanic Skin Response
HR	Heart Rate
HRV	Heart Rate Variability
HTML	Hypertext Markup Language
IDP	Interdisciplinary Project
JDBC	Java Database Connectivity
JDK	Java Development Kit
JPA	Java Persistance API
MVC	Model View Controller
PIN	Personal Identification Number
RMSSD	Root Mean Square Of Duration Variations Of Adjacent RR Intervals
RR	R Wave To R Wave Interval
WHO	World Health Organization
XML	Extensible Markup Language

1 Introduction

One aspect of the ALIAS project is helping elderly people suffering from diseases. Thus, a mobile robot system is to be created that interacts with elderly users, monitors, and provides cognitive assistance in daily life and promotes social inclusion by creating connections to people and events in the wider world.

By providing a communication platform with a wide spectrum of functionalities, the ALIAS project can assist elderly people whose quality of life is restricted by diseases. The ALIAS platform has no physical manipulators. Its main purpose is to help elderly people to sustain their social contacts and everyday communications. Diseases like musculoskeletal diseases or hip fractures are not of relevance for this project. However, by providing a multimodal dialog system, the ALIAS platform can help people suffering for example from ear and eye diseases. People suffering from diabetes can be assisted by providing reminder functions and cognitive assistance.

Within this deliverable, physiological monitoring is being addressed. Together with additional systems for recording physiological data, the ALIAS platform can impersonate a helpful assistant to record, monitor and store physiological data.

Cardiovascular diseases happen to be the prime cause of mortality all over the world. The World Health Organization (WHO) is calling it a global epidemic. Information- and communication systems like ALIAS can provide various tools for fighting against it and improving the health and well-being of the population.

Main goal of such tools should be the assistance to enable people suffering from cardiovascular diseases, like hypertension, or diabetes to go on living a normal life (Rocha et al. 2010). Therefore, important physiological data include vital signs like body temperature, pulse rate (or heart rate), blood pressure as well as respiration. The alias robot will provide a system for monitoring these physiological parameters.

The recorded vital signals can be inspected remotely by the users, their relatives or their personal doctor through the ALIAS platform. There are two ways of providing the physiological data to ALIAS. The first possibility is to directly transmit the data by a recording system. Alternatively, the user can record the data on his own and submit it to ALIAS via the multimodal ALIAS user interface (i.e. per speech or touch screen).

The platform should store the physiological in the database, visualize it and trigger appropriate alarm messages when a parameter exceeds the predefined thresholds. Additionally, patient information like age, sex and recorded personal data can be stored by the ALIAS system, which is of great importance when the robot is used by more than one person (e.g. nursing homes).

The ALIAS platform contains a database which records the various physiological signals and features captured by sensors or added manually through a web interface. The recorded data has to be processed such that it can be visualized and evaluated in order to inspect critical changes of the vital signs. To ensure easy handling of the module, it should be equipped with an easy interface for the user via direct speech input and the built-in touch screen.

2 Related Work

Previous work in the Ambient Assistant Living sector has reflected upon the usability requirements of Information Technology based systems for elderly people: These systems have to deal with the special conditions of their users, like reduced cognitive capacities, sight loss, hearing loss and minor user experience with interactive systems. This can be achieved by offering flexible font sizes, naming buttons with significant/clear words, offering one-level navigation or using headlines as major information. These aspects are also known as the ease of use-approach (Lorenz et al. 2007).

Beer and Takayama (2011) performed some more in-depth research in terms of robot acceptance. Due to the fact that elderly people usually do not have much experience in using interactive systems, they are only of interest for them, when they offer more comfort, assistance, independence and an easier management of the social life. Rocha et al. (2010) deal with the monitoring process of vital signals in the Cardiovascular field as the most relevant and advanced technology developed in the field of Assisted Ambient Living (AAL) where wearable computing plays the key role of constant vital monitoring.

Mufti et al. (2009) consider ubiquitous wireless computing, where different sensors measure various values of all kind of sorts. These sensors are combined to an “Infrastructure For Elderly Assistance” as the task for the future where the sensors serve as health indicators or full-time attendants. This approach is supported by Brell et al. (2010) with the “Mobile Robot Peekee II” representing such a mobile infrastructure. Within their presented concept, the robot and sensors are completely independent from the infrastructure.

MobiSense is another mobile health monitoring system for ambulatory patients introduced by Waluyo et al. (2010). This system is able to detect postures of people like lying, sitting and standing while it is also able to manage its’ own resources like re-configuration of parts. Dinh et al. (2009) show a similar approach with their physical activity monitoring system with build-in vital-sign and fall detection. Again, this system includes a wearable device collecting physical and activity data with various sensors.

A similar approach was introduced by Bai et al. in 2000 with presenting their Home Telemonitoring Framework, which they divided in two categories: the daily activity monitoring category for the elderly and the vital sign monitoring category for patients recovering at home. Thus, the home care needs were separated into different levels. The system itself was divided in the home monitoring unit mainly responsible for the data storage, a hospital monitoring center and a communication network interconnecting these components.

This deliverable addresses the following questions from the perspective of older adults: How can ALIAS assist elderly people regarding sensor-independent and sensor-dependent physiological monitoring while maintaining their independence and how can a remote access to stored data e.g. by relatives or a doctor be realized? To assess these questions, use cases were created. These use cases, the corresponding activity and state diagrams and some security aspects are presented in Chapter 3. The developed database concept is introduced in Chapter 4. Finally, the developed web application is presented in Chapter 5.

3 System Overview

The ALIAS physiological monitoring module is provided as a web application, which means that it runs in a web browser. Integration into the ALIAS dialogue system is thus possible through the web browser which is built into the graphical user interface (GUI) of ALIAS. The module can be started and controlled for example through the direct speech input interface of ALIAS, which relates the recognized control commands to the web browser. Data can be entered via the ALIAS symbolic keyboard or speech.

The capabilities of the system are defined by various use cases, which are described in the next section. Along these use case descriptions, the involved processes are described using activity diagrams and details about data security are discussed.

3.1 Use Cases

In order to properly realize the database concept, it is necessary to define the use cases the system shall handle. The main purpose of use cases is to identify and model the functionality of the system the user expects.

In the following sections, we will introduce and explain the use cases realized in the system and explain the tasks the system has to perform in response to a specific request from the user.

3.1.1 Registration and Login

The database is a web application, therefore, the dialogue manager has to open the corresponding ALIAS physiological homepage, when the users wants to interact with the database. In general, three different steps can be determined, when a user wants to interact with the system. The first step is to register the user with the system. For the interaction with the system, the user has to login, and afterwards to logoff again.

- System Registration: Concerning system registration and login, the user has three options to choose. First, if the user has no account yet, he needs to register himself to the system. This will be done exactly once.
- Login: If the user is already registered on the system, he can use his/her login and password in order to log onto the system. After login he can perform all the tasks and exercises explained in the following sections.
- Logout: At the end of the session, when the user has finished his tasks and exercises he may or shall log out, to prevent other users from accessing his/her data.

3.1.2 Sensors

After successful login the user can use one or both of the two supported sensors if connected:

g.tec Physio Module: This system records the following biosignals and physiological features thereof.

- ECG (Electrocardiograph): This task inherits the measurement of:
  - HR (heart rate): the mean heart rate in beats per minute computed in the past N seconds.
  - HRV (heart rate variability): the heart rate variability over the past N seconds.
  - Ratio of normalized HR and HRV: both values will be normalized before computing the ratio.

    - RMSSD: Root Mean Square of duration variations of adjacent RR (R wave to R wave interval) intervals for the past N seconds.
    - HR/RMSSD: Ratio between normalized HR and normalized RMSSD.
    - pNN50: Percentage of adjacent RR intervals differing by more than 50 ms within the past N seconds.
    - HR/pNN50: Ratio between normalized HR and normalized pNN50.
- GSR (Galvanic Skin Response):
    - ERD: Normalized relative means (in percent) for the past N seconds.
- Respiration:
    - Respiration Rate.
    - Duration of inspiration and expiration interval.
    - Depth of inspiration and expiration
    - Intermittend breaks within the inspiration and expiration cycle
- Stress: Initial HR/RMSSD will be used to indicate the stress-level.

Source and further information: (Zeitlinger et al. 2010, p.6).

Zephyr:

The Zephyr HxM Sensor enables a Real-time feedback which includes:

- Heart rate
- Speed
- Distance
- Cadence
- RR-telemetry

(Source: http://www.zephyr-technology.com/store/hxm-development-kit.html)

3.1.3 Manual Data Entry

The user is able to enter the following values manually:

- Blood pressure
- Blood sugar
- Heart rate
- Pulse
- Respiration
- Temperature and
- Weight

For all those values, the user can also choose to display the data he already entered. Here, he has two options: Either he chooses a timeframe he can select by a selection box – or if there is a specific data, from when on entered values are to be displayed, he is free to insert this date as well.

3.1.4 Administration

Another important task to be realized is the system administration. The administrator (a person of trust e.g. a relative, geriatric nurse) can manipulate the system in four different ways:

- Set Language: The language can be set by the administrator.

- Set Min/Max Values (alarm limits): The minimum- and maximum-values for the tasks can be set manually by the administrator if necessary (cf. Figure 6).
- Reset Password: Dealing with elderly people, it is necessary to offer the option to change their password in case it is forgotten. It is assumed, that the administrator is a person of trust and can thus comfortably change the password for the user.
- Delete User: In case a user has to be removed from the system, the administrator is able to perform this deletion.

3.2 Activity Diagrams

After the system requirements from the users' point of view are identified, it is important to recognize the corresponding processes the system needs to perform in order to realize the identified use cases. This is usually done by Activity Diagrams. In the following, the diagrams will be displayed and briefly explained.

3.2.1 Registration and Login

3.2.1.1 Registration

When the user has no account yet, he has to register to the system by providing the required user data such as first name, last name, birthday, height, gender, language and most comfortable font size. Furthermore, the user has to choose a login name which uniquely identifies him on the system and a corresponding password. When the data is entered, the system checks whether the login name chosen is already in use or not. If the login is already assigned, the user has to choose another one. The account data is stored within the underlying database if the registration was successful. The next time the user can log onto the system using his/her login and password. The registration activity diagram is shown in Figure 1.

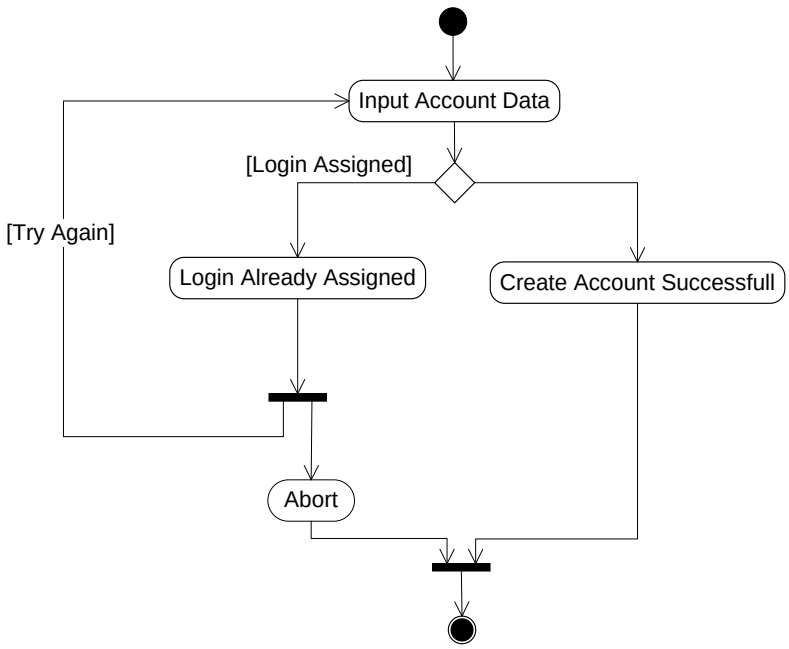

Figure 1: Registration Activity Diagram.

3.2.1.2 Login

The system takes the login and password entries of the user and checks, if the entered values are correct. If they do not match, the login is declined. When the provided username and password are valid, the user is successfully logged in and the corresponding user data is transferred from the database to the system. The login activity diagram is shown in Figure 2.

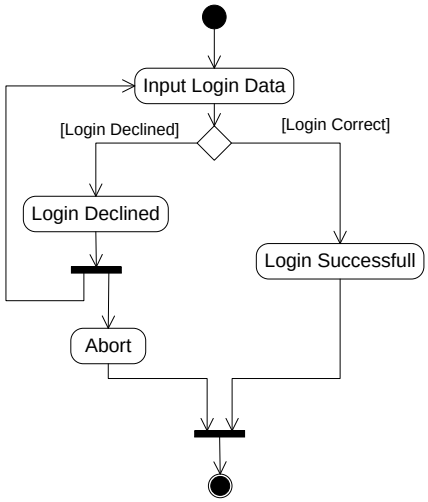

Figure 2: Login Activity Diagram.

3.2.2 Sensors

After the starts recording live data, the system awaits the user either to select the g.tec Monitoring Service or the Zephyr Monitoring Service respectively. When it receives the user input, data logging is started. This data is being stored into a queue, until enough data has been collected to start the monitoring. After a dataset value has been displayed, the system stores it into the underlying database. When the user chooses to stop logging and forwards his/her choice to the system, logging, monitoring and storing data is stopped immediately and thus the live data performance is finished. The activity diagram for the live data is shown in Figure 3.

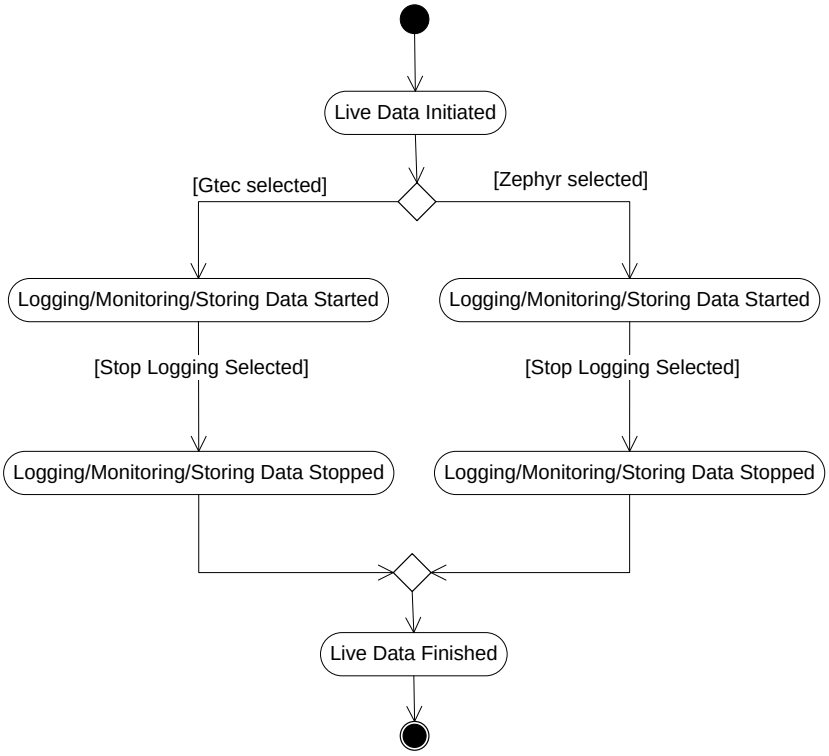

Figure 3: Live Data Activity Diagram.

3.2.3 Manual Data Entry

It is possible to enter physiological data manually into the database, therefore the user selects the kind of data, which should be stored into the database and selects the corresponding button in the web application. For instance, when the blood pressure shall be recorded, the system waits for the systolic and diastolic values as well as the pulse to be entered. After the user entered these values and pushes the confirm button, the system displays the entered values again to the user and offers him/her two options:

The first option is to edit the data to correct mistyped values. In this case, the values are deleted and the system waits for the new ones to be entered. . The second option the user has is to confirm the values he entered. The underlying data base is only updated and thereby modified if the entered vales values have been confirmed by the user. This ensures that the database does never contain invalid values. Figure 4 shows the activity diagram for entering a blood pressure value into the database.

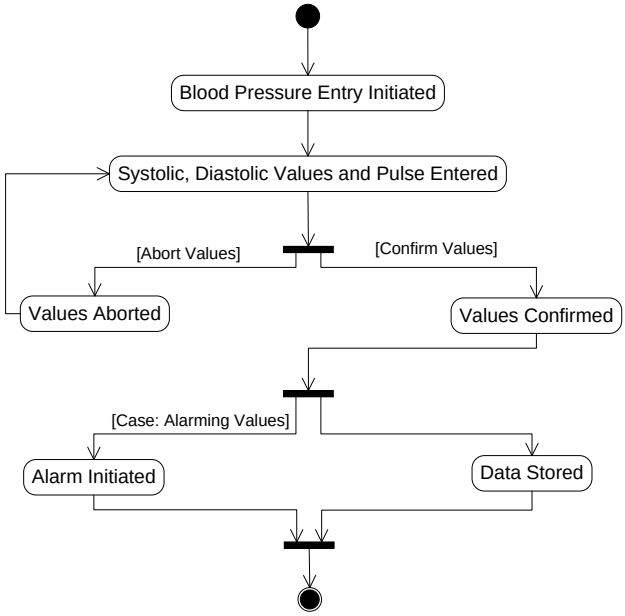

Figure 4: Blood Pressure Entry Activity Diagram.

It would be redundant to show the Activity Diagrams for all different value types (blood sugar, heart rate, respiration, temperature, weight). Therefore the single-valued Value types are combined in the Single-Valued Diagram (cf. Figure 5). The process is analogous to the blood pressure process. The only difference is, that the input of blood pressure values requires to enter more than one value (systolic, diastolic and pulse) while all other value types only require the input of one single value.

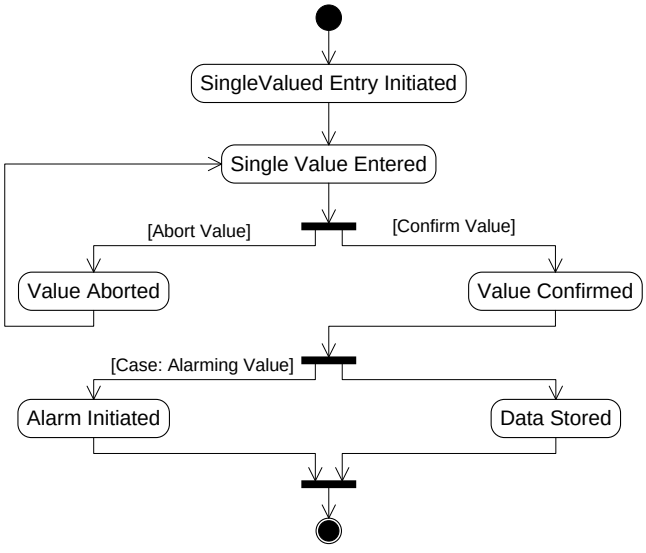

Figure 5: Single Valued Entry Activity Diagram.

The system checks if the entered values exceed any of the predefined lower and upper alarm limits defined in the database (see Figure 6).

Value Type	Maximum Value	Minimum Value

Systolic	140	105
Diastolic	90	60
Pulse	100	60
Blood Sugar	110	70
Heartrate	100	60
Respiration	20	12
Temperature	37.5	35.8
Weight (BMI)	24.9	18.5

Figure 6: Minimum/ Maximum alarm limits for the different types of values.

3.2.4 Administration

3.2.4.1 User Deletion

When a user has to be removed from the system, the system administrator initiates the deletion of the account. Thereby all user data is removed from the database. At the same time, all data records associated with this user will be deleted from the system, too. After a successful deletion, a confirmation is displayed to the system administrator. Figure 7 shows the activity diagram for deleting a user from the system.

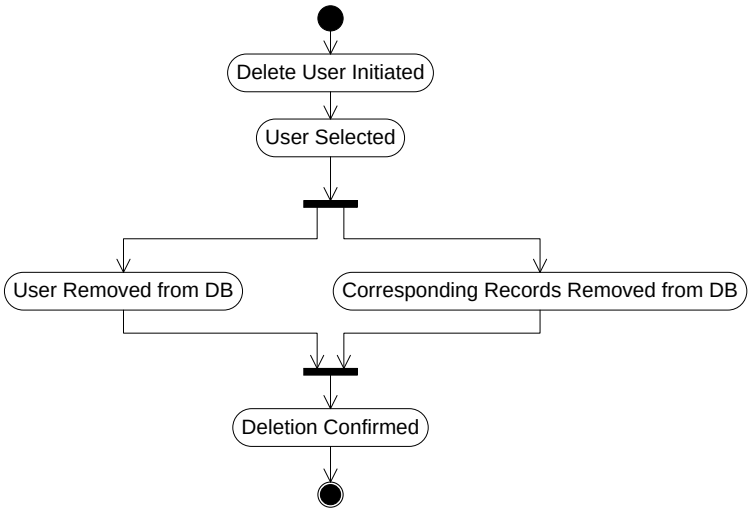

Figure 7: User Deletion Activity Diagram.

3.2.4.2 Password Reset

Offering the possibility to reset a password is necessary in every system. Regarding the fact, that elderly people play the role as users to the system, it appears likely, that they are unable to cope with this task. Thus, we decided that – assuming the system administrator to be a person of trust – he takes this task. If the system administrator initiates a password reset and selects the user to reset the password, he can put the new password into the system. The system displays a confirmation message to the system administrator after the reset was successful. Figure 8 shows the activity diagram for a password reset.

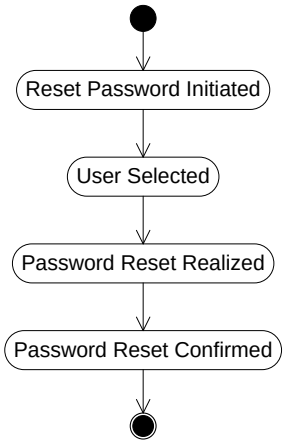

Figure 8: Reset Password Activity Diagram.

3.3 Security

3.3.1 Authentication

The Web offers three forms of authentication, namely authentication through knowledge, possession and property or any combination thereof (Sabzevar, Stavrou, 2008).

Authentication through knowledge includes for example entering a PIN code when turning on your cell phone or entering a password to check one's email. This authentication mechanism is cheap to implement, which explains its widespread use today. Authentication here depends solely on secret knowledge, which is pre shared between the end user and the service provider. Proof of knowledge of the secret can subsequently be used by the end user to prove his/her identity to the service provider. The downside of this approach is that users often for-get their password, which leads to expensive recovery mechanism in companies (for further reading see http://unisec.blogspot.com/2007/11/three-types-of-authentication.html).

Secondly, authentication can be done through possession such as hardware tokens. Authentication through possession relies on the ownership of a physical object. For example, if a bank provides a customer with an RSA token when opening a bank account, then at a later point in time, the user can prove his/her identity, by proving to be in possession of the token. The bank knows that they gave the RSA token out to a certain customer, and then can be sure they are dealing with that same customer when being provided with the token.

A major disadvantage of authentication by possession is the relatively high cost. Authentication by knowledge requires no infrastructure, and is cheap to implement and maintain. Authentication by possession, on the other hand, requires an infrastructure (such as RSA tokens, or RFID chips and RFID chip readers), as well as processes to ensure the safe operation and maintenance of the system. Just like in authentication by knowledge users may forget or lose their passwords, in authentication by possession, users may also lose their tokens, or even have them stolen. The cost of deactivating the lost token, and cost of replacing it is significantly higher than that for a lost password.

Finally, authentication can be performed by characteristics such as biometrics. Biometrics is applied increasingly, such as fingerprint readers on laptop computers. The biggest advantage of biometric authentication is that we always carry the used characteristic (such as fingerprint, voice, iris, etc.) with us. Our fingerprints cannot be lost or stolen. Just like in authentication by possession, biometrics

also requires investments in the infrastructure, i.e. the biometirc reading devices, plus capacities to store the data and evaluate requests. Furthermore, there is a tradeoff between usability and security in biometrics. No two biometric values will ever be the same. In other words, if one person repeatedly scans his/her fingerprints, he will never get the exact same value twice. This means that the biometric system needs to accept a range of values, rather than an exact value. For example apassword is not accepted unless it matches its stored version exactly. When the range of acceptable values is too small, users will be annoyed by having to repeatedly scan their finger, until the system accepts it. When the acceptable range is too high, there is an increased risk of the system matching two unrelated datasets.

As both biometrics and authentication through possession require investments in hardware and a technological infrastructure, we chose to use authentication by knowledge in the physiological module of ALIAS.

As previously discussed, authentication by knowledge (i.e. password) has some disadvantages, such as cost of password recovery, users selecting simple and therefore guessable passwords or passwords witten on post its placed close to the computer. ALIAS is not a Web application, which is designed to be used by millions of users on the Internet. It will run locally on a machine, which significantly reduces the risk of a security attack.

3.3.2 Password Storage

The most straightforward method of storing passwords is storing them directly in the database. While this approach is easiest to implement, it comes with significant security concerns. Users tend to use the same user name/password combination for their different ac-counts.

In addition, users with direct access to the database (such as administrators) can see all passwords, and try them out on accounts the user may have at other providers. Furthermore, vulnerabilities such as injection attacks could potentially give out plaintext passwords of all users.

In order to obviate from these risks, we decided to store encrypted passwords rather than plaintext passwords. The encryption is done using hash functions which convert an input of varying length into an output string of fixed-length. Specialized cryptographic hash functions furthermore ensure that it is not possible to deduce the original plaintext from its hash.

A difficulty in the use of cryptographic hash functions is that hash functions that are safe today may not always stay safe in the future. With ever-increasing computing power, some hash functions that were too complex to break even ten years ago, can now easily be broken with modern hardware. For this reason we decided to make use of two independent cryptographic hash functions, SHA-256 and MD5. The use of two cryptographic hash function ensures that if either one of the two were broken, the security of our password-storage mechanism won't be compromised.

SHA-256 belongs to the family of SHA-2 cryptographic hash functions. SHA-2 replaced SHA-1, in which security flaws were identified in 2005. SHA-256 has an output size of 256 bits. MD5 has an output length of 128 bits.

In order to generate the value stored in the database, the entered password is first hashed with the SHA-256 hash function. The output produced then serves as input to the second function, MD5, whose output subsequently gets stored in the database.

4 Database

4.1 Database Concepts

There are different types of databases available and they provide different kind of features, a more extensive information about databases can be found in (Oppel (2009), Silberschatz/ Korth/ Sudarshan(2006)).

In general a database is a collection of data, which has a certain kind of interrelation and can be accessed via a specific set of functions or programs. For databases a specific architecture was defined comprising three different layers (see Figure 9): physical layer (containing the data file on physical disk drives), logical layer (first abstraction level: data is represented in a common abstract structure form) and the external layer (second abstraction level: the form of the data accessed by the users). Different database models (flat-files, Hierarchical Model, Network Model, Relational Model, Object-Relational Model, etc.) have evolved over time. A short overview is given in deliverable D3.5. For the reason mentioned in D3.5, we decided for an object-relational approach for the physiological monitoring database.

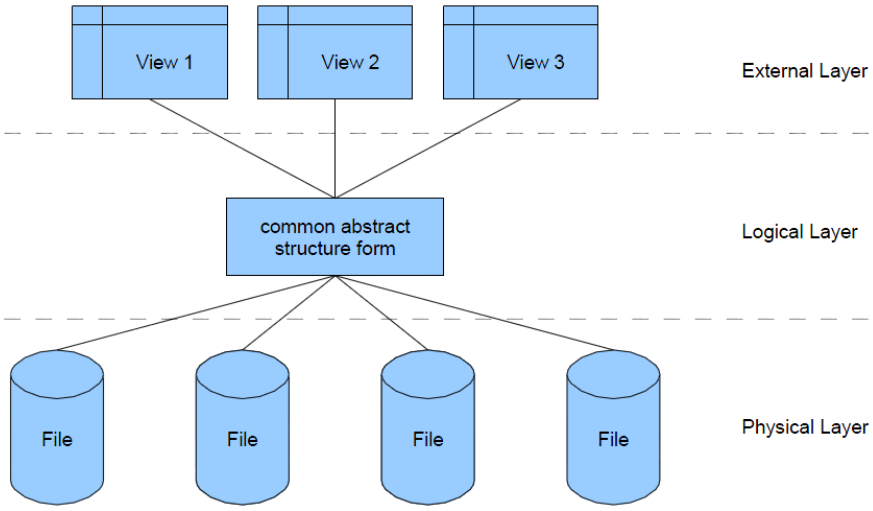

Figure 9: Database Architecture.

4.2 Database Concept for the Physiological Monitoring

The database concept for the physiological monitoring system envisioned for the ALIAS project has to fulfill several properties, which are relevant for handling physiological data of elderly people. First, the data should be stored over longer time periods enabling to perceive changes in the course of the data (e.g. blood pressure). Second, the database should be able to handle different users and their related physiological data (this is relevant for the usage of the ALIAS system in care facilities). Third, the physiological data should be accessed either directly on the ALIAS system or via remote access, thus enabling doctors and the authorized health care provider to access the physiological data.

In order to achieve a generic, resource-efficient database, an elaborate database concept has to be developed. Figure 10 depicts the generated database concept consisting of the following entities: user, dataset, dataset value, dataset value type. These four entities will be described in the following.

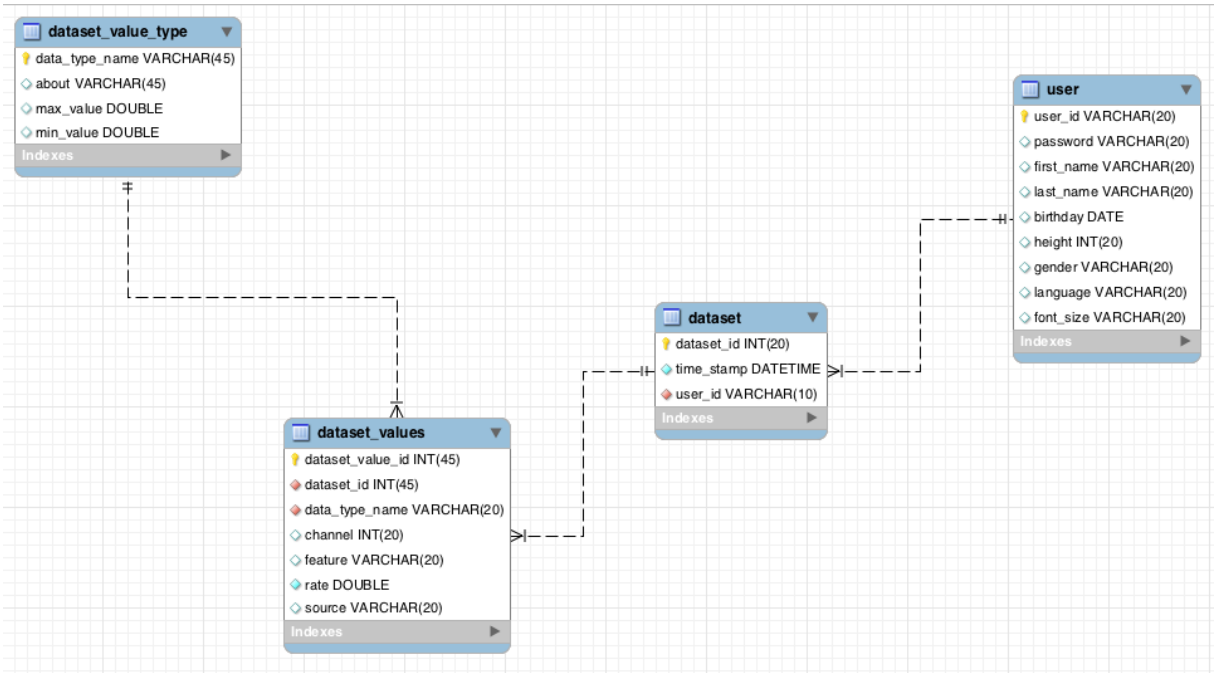

Figure 10: Database Concept.

4.2.1 User

In order to enable various users to share the same ALIAS-platform, we created the table “user”. Here the user id, corresponding to the login name, is used as the primary key of the user. This ensures that each user can be uniquely identified by his/her chosen user id, at the cost that no two users can choose the same id.

Furthermore, the user has to choose a password. The user id and the corresponding password are used to grant registered users access to the data base and into the system. In addition the following information about the user is provided by the user-entity:

- **First Name**
- **Last Name**
- **Birthday**
- **Height**
- **Gender**
- **Language**
- **Font size**
- **Login**
- **Password**

Finally, a user can be associated with various datasets. This is realized by a one-to-many relationship between a user and datasets.

4.2.2 Dataset

In order to uniquely identify a dataset, it contains an automatically generated “dataset id”. The timestamp enables to reconstruct, at which date and time a dataset was recorded. Furthermore, a dataset has a foreign key which assigns it to a specific user. A single dataset can contain multiple dataset values, which is realized by a one-to-many relationship between dataset and dataset values.

The dataset relation is necessary, because a sensor may record multiple biosignals and extract many physiological features thereof at one time. If for example pulse and blood sugar are measured at the same time, this would lead to a dataset containing two dataset values, one for the pulserate and one for the blood sugar level respectively. Having a dataset relation allows us to logically group data that was recorded simultaneously, thereby separating if from data collected at different points in time.

4.2.3 Dataset Value

The automatically generated field “dataset value id” allows to reference each dataset value in a unique way. In order to differentiate between multiple connected sensors, from which dataset values are transferred to the system, the field “channel” is invented. The field “feature” allows for capturing further information of the sensor, if necessary. The dataset value itself is stored by the field “rate” and the sensor from which the dataset value is originated is captured by the field “source”.

The dataset values table contains two foreign keys. The first assigns the current dataset-value to its corresponding data set. The second assigns a specific dataset-value–type to it.

4.2.4 Dataset Value Type

The “dataset value type”-table enables the resource-effective and generic framework to be realized: in this table, the various dataset value types that can be recorded, for example blood pressure, respiration or heart rate. These different types are recorded only once and are then assigned to the dataset values.

It contains the field “data type name”, which also acts as unique identifier for the dataset value type. Further information about the dataset value type can be stored in the field “about”, if necessary. The fields “minValue” and “maxValue” define the thresholds of the lower and upper alarm limits if any is set.

One dataset value type can be assigned to any number of dataset values

4.3 Hibernate

For the reasons described above, Hibernate was chosen as a persistent data storage frame-work. Hibernate is a framework that maps objects form an object-oriented Java environment to a relational database.

Thus, it allows making Java objects persistent by storing them in a relational database. The programmer does not have to worry about the underlying relations.

4.3.1 Hibernate Mode

Hibernate provides two ways of mapping java classes to relational database tables: through Java annotations and through hibernate-mapping XML files.

In order to persist Java objects with hibernate, it is necessary to provide some information to the hibernate framework. In the past, this was done with XML-mapping files that described how to save a given Java object to the database. For example, if you have a class named user with three properties id, login and password, you have to add following mapping file to the Java classpath:

```
<hibernate-mapping>
    <class name="user.User" table="USER">
        <id name="id" column="id">
```

```
            <generator class="native"/>
        </id>
        <property name="login"/>
        <property name="title"/>
    </class>
</hibernate mapping>
```

Using hibernate-mapping.xml files certainly is not wrong, but there are some disadvantages:

- Information about Java classes has to be maintained in external files.
- XML can become difficult to write for large classes.
- For a large number of classes, the XML files become unreadable by a human.
- Errors in XML-files can propagate all over the program.

Fortunately, with Java 5 a new Java based artifact was introduced – the Java annotations. Annotations empower developers to add detail information about java classes without changing any code inside a Java class or method. Thus, Java annotations can be used instead of XML-mapping files. An example of Java annotations can be seen in Figure 11.

```
@Entity
public class User {
	private Long id;
	private String login;
	private String password;
	public Long getId() {
		return id;
	}
	@Id
	@GeneratedValue
	public void setId(Long id) {
		this.id = id;
	}
	public String getLogin() {
		return login;
	}
	public void setLogin(String login) {
		this.login = login;
	}
	public String getPassword() {
		return password;
	}
	public void setPassword(String password) {
		this.password = password;
	}

}
```

Figure 11: Hibernate Annotations Example.

The `@Entity`, `@Id` and `@GeneratedValue` annotations replace the econtent of the corresponding XML mapping files. The Java class maintains its own mapping information, which is easier to handle a class by class basis. In addition to this, introducing new classes or removing persistent classes form the domain model becomes much easier (McKenzie 2008, p. 19-21).

For the reasons described above, Java annotations were chosen for the mapping.

4.3.2 Technical Requirements

In order to work with Hibernate and JPA (Java Persistence API) annotations, you have to fulfill following technical requirements:

- A JDBC compliant database including apropriate JDBC drivers; without itno Java persistence is possible. We chose to use the widespread MySQL Database.
- The JDK 1.5 or newer which offers Java annotations
- Various files and libraries associated with the Hibernate Core and Hibernate Annotations.
- The hibernate.cfg.xml file must be accessible through the runtime classpath. This configuration file defines the name of the database, the database driver to be used for connecting toit, and the address and prot to connect to when accessing it. Further it contains the username and password for establishing a valid connection to the database.(see Figure 12). (McKenzie (2008), p. 33, p.52)

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
"-//Hibernate/Hibernate Configuration DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
      <session-factory>

            <property name="connection.url">jdbc:mysql://localhost/hibernate</property>
            <property name="connection.username">username</property>
            <property name="connection.password">password</property>

            <property name="dialect">org.hibernate.dialect.MySQLDialect</property>
            <property name="connection.driver_class">com.mysql.jdbc.Driver</property>

            <property name="transaction.factory_class">org.hibernate.transaction.JDBCTransactionFactory</property>
            <property name="current_session_context_class">thread</property>
            <property name="hibernate.show_sql">true</property>

            <mapping class="de.tum.connectionprovider.hibernate.auto_create.User"/>
            <mapping class="de.tum.connectionprovider.hibernate.auto_create.Dataset"/>
            <mapping class="de.tum.connectionprovider.hibernate.auto_create.DatasetValues"/>
            <mapping class="de.tum.connectionprovider.hibernate.auto_create.DatasetValueType"/>

      </session-factory>
</hibernate-configuration>
```

Figure 12: "hibernate.cfg.xml" File

5 Web Application

The web-application allows to manually record physiological data and to remotely display them. The advantage of the web-application is that it displays all contents within the web-browser which is embedded within the graphical user interface of ALIAS. Figure 13 illustrates the login page of the web application.

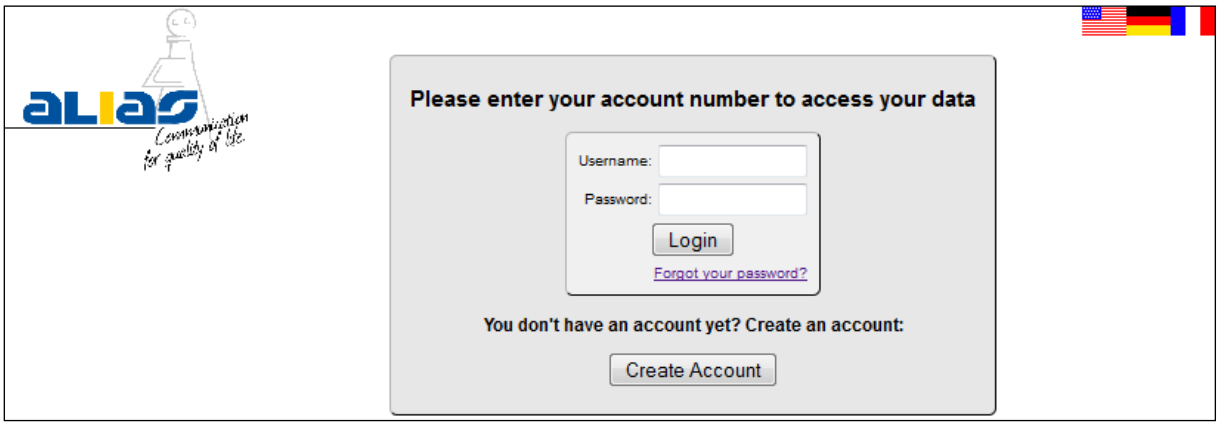

Figure 13: ALIAS Login Page.

5.1 Model View Controller Architecture

A web application consists of three major components: Data Beans, in which the data objects can be manipulated and persisted into the database, JSP's (Java Server Pages) and a controller. These components are also known as model, view and controller (MVC).

5.1.1 Model

In a web application, it is defined within a model which data to use in the application and which operations to perform on these data. This is realized with the Data Beans. The model is responsible for the data processing. Its main role is to encapsulate the data and all of the methods that work on it.

5.1.2 View

The main purpose of the view is to display data to the user, i.e. to represent it. At the same time, the view is not allowed to perform any processing of the data. Usually, a web application contains multiple views: each JSP acts as a separate view.

5.1.3 Controller

The program that links the views and the models is called the controller. This is realized by controller servlets. The controller plays an important role for it handles the requests from the browser and sends responses back to the server. (Downey 2007, p.91)

5.1.4 Servlet Engine

The installation of a servlet engine is necessary in order to run servlets and JSPs. The web server calls the servlet engine which is an application that handles JSPs and servlets. We chose to use Tomcat for the ALIAS-project for the Apache project Tomcat is a very popular and stable servlet engine to use. Tomcat is an open source project and can thus be downloaded for free from http://tomcat.apache.org/. (Downey 2007, p 27).

5.1.5 Session Handling

HTTP is in its nature a stateless protocol. This means that every request sent to the server is treated as an independent request, and does not depend on previous communication between the client, and the server. Session management is especially useful in Web applications, where many different users may be logged in, and using the system simultaneously. Without session management, there would be no way to separate the requests from one another. Each request would have to include information about which user is making the request, password information, etc. This would inevitably add great redundancy, and unnecessary overhead to the application.

In order to ease the programming of Web applications, the J2EE framework provides an interface to its session management module. Sessions allow the Web application to create a relationship with the Client that extends over the lifespan of multiple request/response pairs. The session management module provides an easy way for Web applications to create something like a session over the HTTP protocol.

When the User initially logs in through the login page, a session is created on Server-side. The Server now creates a unique identifier, which it associates with the session. In the response to the initial request from the Client, the Server includes a reference to the unique identifier (also known as Cookie).

Upon receiving this response, the Browser now securely stores the Cookie, and associates it with the URL of the Web site which sent the Cookie. Every subsequent request sent by the Browser to the corresponding Web site will contain the content of the Cookie. When receiving a request, the Server tries to match the session identifier stored in the Cookie with the corresponding session. If a match is found, the Server automatically knows which user has made the request.

Therefore, as soon as the user has authenticated with the application, every request sent to the Server includes the Cookie, which enables the Server to identity the user behind the request.

5.2 Creating a New Account and Login

5.2.1 Creating a New Account

Before a user can login onto the system, he has to create an account. The user data first name, last name, birthday, height, gender, language and font size are stored. Additionally, the user has to choose a login name, which serves as unique identifier and a password. When the user pushes the “Create”-Button the system checks, if the login name is already used by another user. In this case the user has to choose another one. Otherwise, the account is created and the user data is stored into the underlying database. The web page for creating an account can be seen in Figure 14.

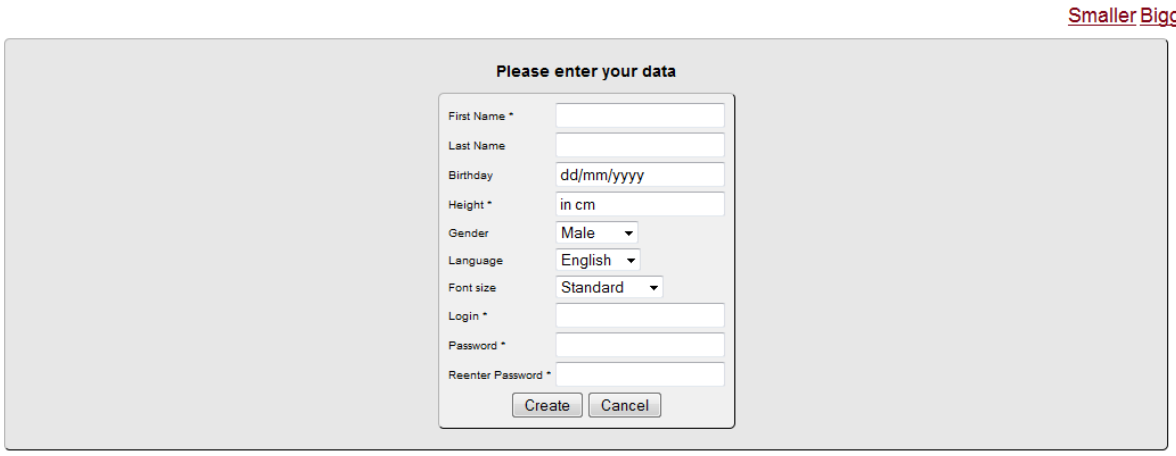

Figure 14: ALIAS Create Account Page.

5.2.2 Login

Before utilizing the system the user has to logon to it using his/her login name and password chosen when the account was created. When the user pushes the “Login”-Button after he inserted his/her values, the system checks, if the values are correct. If they are not, an error message is displayed to the user (see Figure 15).

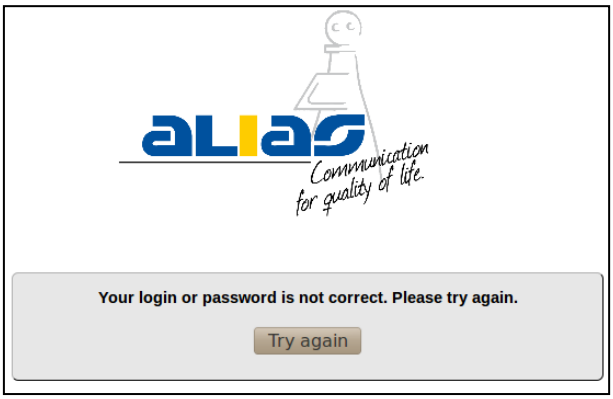

Figure 15: ALIAS Login Error Message.

For the case that the user forgets his/her password a “forgot password”-link was added to the login-page. Through this link a user can ask for help. When clicking the link, the forgot-password pop-up opens, displaying information on how to reset the password or obtain a new one (see Figure 16).

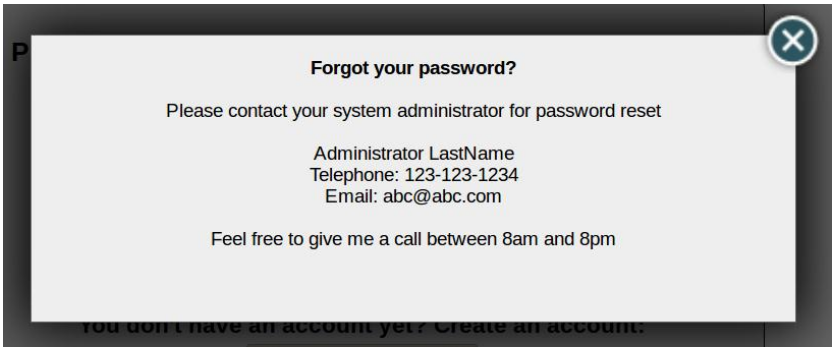

Figure 16: ALIAS Forgot Password Pop-Up.

5.3 Recording and Live Displaying of Sensor-Generated Data

5.3.1 Data Visualization

For data visualization, the charting library Highcharts was used. It allows to integrate interactive charts within a web application. It is compatible with all modern browsers like Firefox, Opera etc. and also supports mobile platforms like the iPhone/iPad.

Highcharts offers numerous chart types like line, spline, area, areaspline, etc. and allows multiple modifications. The programmer can also comfortably configure options in a JavaScript object notation structure. Therefore, in this interdisciplinary project we chose to use Highcharts for data visualization.

For Non-Commercial use, Highcharts comes for free, while the pricing for commercial websites can be found at http://www.highcharts.com/license. An example graph generated with Highcharts is shown in Figure 17.

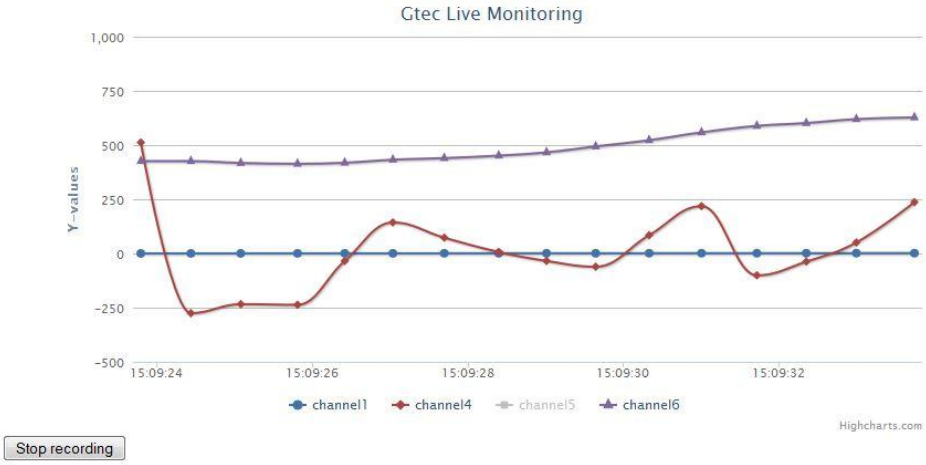

Figure 17: Example Graph Generated With Highcharts.

5.3.2 ALIAS Home Page for Physiological Monitoring

When the user has successfully logged in, the alias home page is displayed. Now, the user can choose between several options. The most important ones are manually adding data and life recording through an attached recording systems (Figure 18).

Figure 18: Bar of the ALIAS Physiological Monitoring Home Page.

5.3.3 Selecting Sensors

When the user selects “LIVE DATA”, he can choose between the recording systems from g.tec- and the Zephyr Monitoring system. In case the physiological data shall be recorded using the g.tec device, the recording devices is started and a new window opens. Within this window, these recorded data are being live displayed. At the same time, the visualized data is persisted to the database. When the

user wants to stop the monitoring, he has to push the button “Stop recording”. Analogous, the Zephyr monitoring is to handle for the user.

5.4 Adding Data Manually

5.4.1 ALIAS Home Page

On the ALIAS Home Page, the User has the possibility to manually add values indicating blood pressure, blood sugar levels, heart and pulse, respiration rate, body temperature and weight.

5.4.2 Blood Pressure: Inserting New Data

In case that the user decides to enter new blood pressure values from a blood pressure reading, the blood pressure page opens. Here, the user has three options (cf. Figure 19):

- He can record new results from the blood pressure reading,
- select a timeframe for blood pressure data to be displayed or
- choose a date for blood pressure data to be displayed.

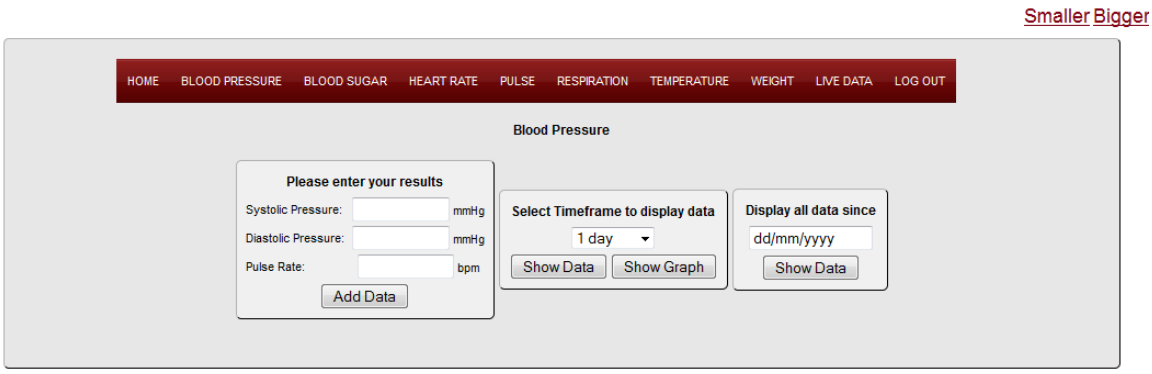

Figure 19: Blood Pressure Edit Page.

If the user decides to record new blood pressure values from a reading, he has to confirm it by pressing the Add Data button.. The data he inserted is then displayed again, where he can either edit the data in case he entered a false value by pushing the “Edit”-Button and returning to the edit page or – if the entered values are correct – store the values to the database by pushing the “Process”-Button (cf. Figure 20).

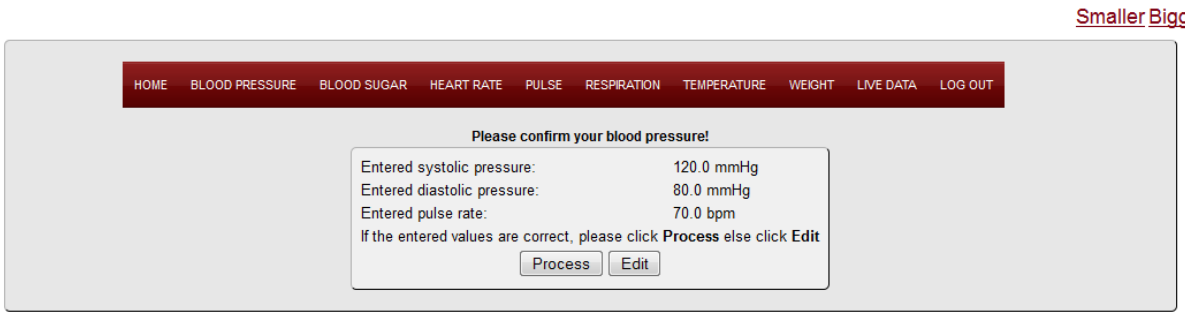

Figure 20: Blood PressureConfirm Page.

When the Process button is pushed, the user gets a message that the data has been recorded successfully (cf. Figure 21). When the data has been recorded, the user can choose between

returning to the ALIAS home page, selecting any other data to manipulate, to choose vital signals to be stored by a sensor while being at the same time live displayed or to log out.

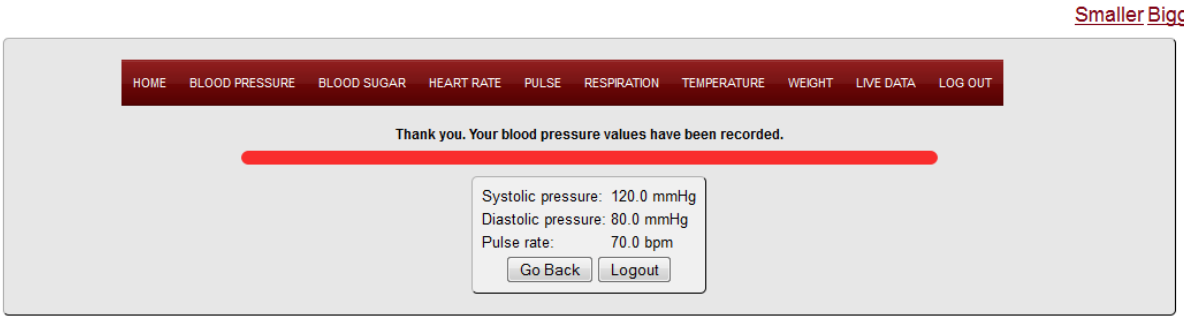

Figure 21: Blood Pressure Process Page.

5.4.3 Blood Pressure: Displaying Data

If the user chooses to display data from preceding blood pressure readings, he can choose to display data by selecting a timeframe or by inserting a date. By pushing either one of the “Select” Buttons, the data will be displayed (cf. Figure 22). Optionally, the user can display the data entered in a Highcharts graph by pushing the “Show Graph”-Button.

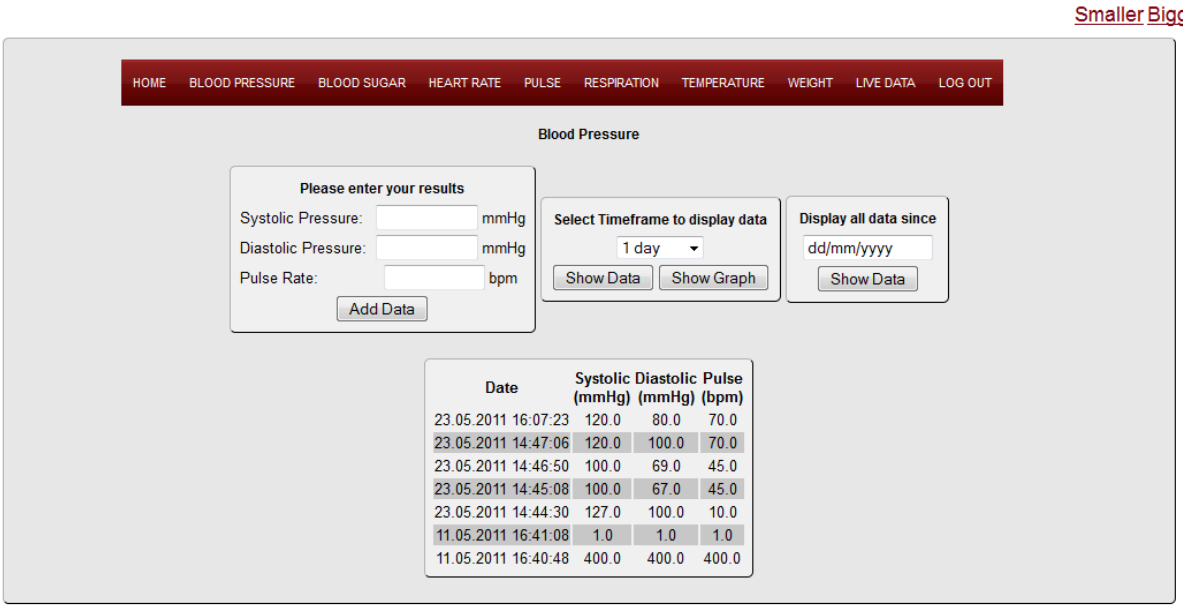

Figure 22: Blood Pressure Displaying Page.

5.5 System Administration

5.5.1 Admin Interface

The admin interface (cf.Figure 23) allows the system administrator to perform four general tasks: setting the language, defining the alarm limits for each stored data type, resetting a user’s password and deleting a user from the system. Within the language settings, the system administrator can manipulate the contact information introduced in Figure 16 within the necessary language. In the following, the three latter tasks will be explained in detail.

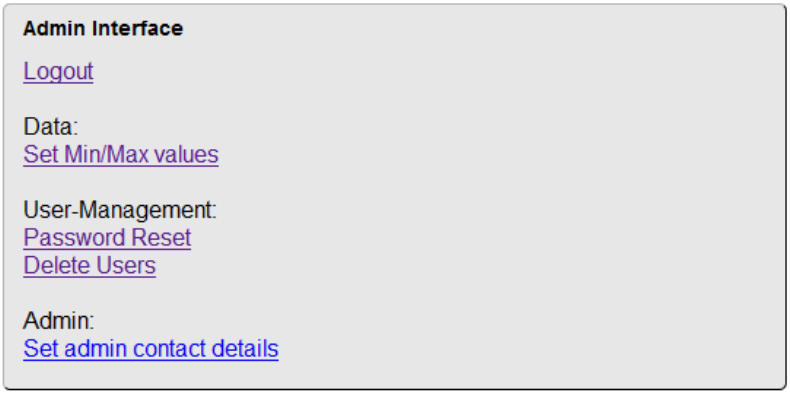

Figure 23: Admin Interface.

5.5.2 Setting Min/Max Alarm limits

We decided it is an important task for the system administrator to be able to manipulate the lower and upper alarm limits for the different data values. Therefore, if the administrator – again assumed, that he is a person of trust – can perform this task comfortably by using the interface presented in Figure 24.

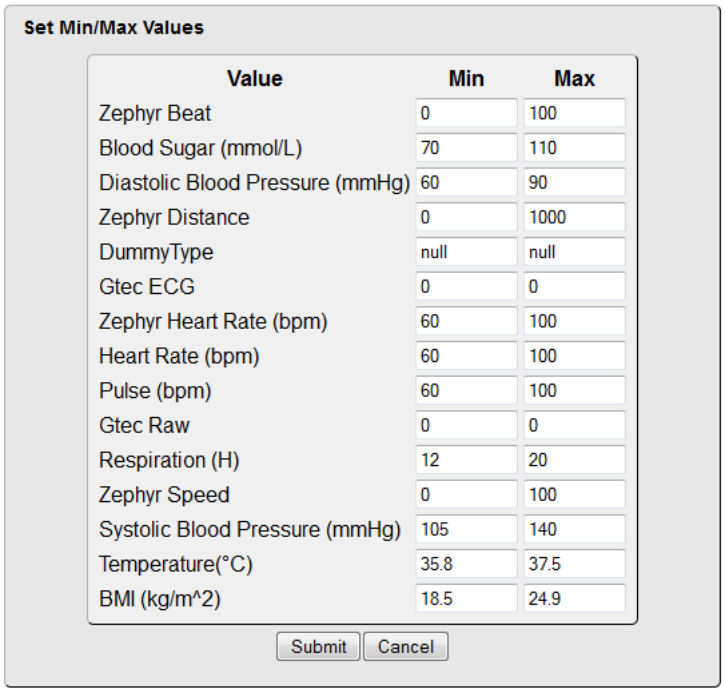

Figure 24: Setting Min/Max Values.

5.5.3 Deleting a User

Out of various reasons it can become necessary to delete a user from the system. Due to security reasons, only the administrator should be able to perform this task.

When the admin intends to delete a user, he can comfortably select a user by a drop-down box (cf. Figure 25) which displays all users stored in the database. When he selects the proper user and pushes the “Delete”-Button, the selected user as well as all corresponding values stored in the database are removed from the database and thus deleted from the system. When the user deletion is performed successfully, a confirmation message is displayed to the administrator (cf. Figure 26).

Figure 25: Delete a User.

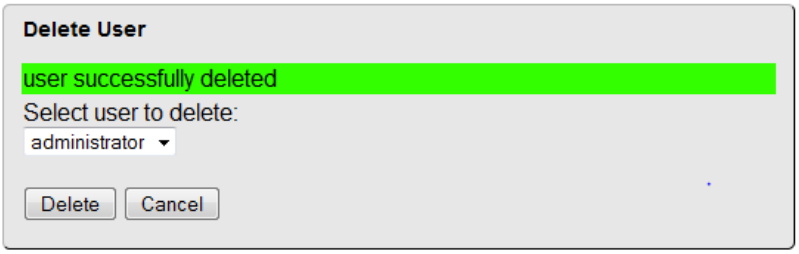

Figure 26: Confirm Deletion.

Certainly, the administrator should not be allowed to delete its' own account. In order to prevent him doing this by mistake, the system prohibits the deletion of the admin-account.

5.5.4 Resetting a Password

As explained above, the administrator is a person of trust and thus has the ability to change the password on behalf of the elderly users. In this case, the new password can be stored in the respective field, where the corresponding user can be selected by a drop-down-box (cf. Figure 27).

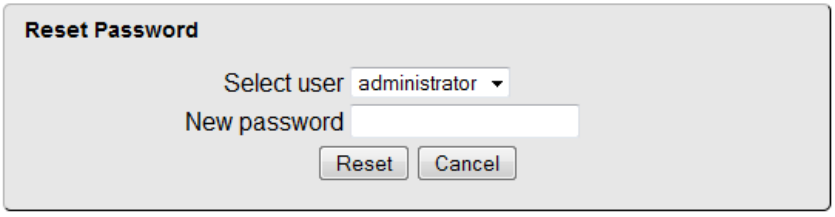

Figure 27: Reset a Password.

By pushing the "Reset"-Button, the system immediately stores the new password within the corresponding user data. When the password reset is successfully performed, the system displays a confirmation message to the administrator (cf. Figure 28).

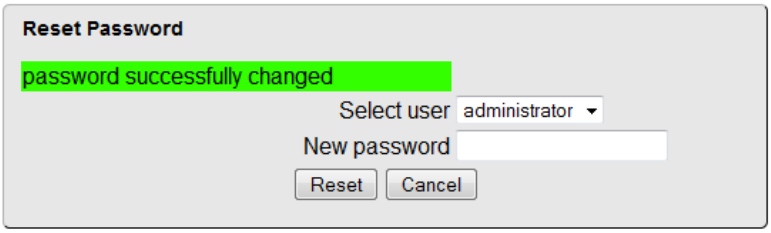

Figure 28: Cofirm Reset Password.

6 Conclusion

In this deliverable the realized version of the physiological monitoring was presented. As a web-based service, the physiological monitoring is integrated within the web browser of the ALIAS robot. The main tasks for the physiological monitoring can be summarized as follows:

- Storage of corresponding physiological data in the database.
- Visualization of data.
- Individual definition of alarm limits for each single physiological parameter.
- Storage of patient information to enable the usage of one single ALIAS-Robot by multiple users.
- Remote Access.

Therefore, a database concept was generated capable of fulfilling these requirements; however, there is still space for improvement left. Some further tasks imaginable are:

- Applying Machine Learning on the data stored: Salvador, Nogueira and Valadas (2009) for example developed two Markovian Models – one of them is the Markov Modulated Poisson Process Model known from traffic analysis in communication networks for monitoring medical signals. The constructed models are based on empirical data from public domain medical signal databases. This approach could be interesting for further work, however, the interpretation of physiological data has to be handled very carefully. Thus this task is not feasible within the ALIAS project.
- Integrating further sensors, for example SPO2 (Oxygen Saturation) and fuse the data from all integrated sensors in the database in order to determine a complete overview of the patients' fitness level in real time at any given time.
- Implementing a method to establish a personal health profile based on the data analysis performed on the data stored.
- Improving data visualization by utilizing visualization tools on the enhancements.

7 References and Web Sources

In the following, the used references as well as the used internet sources are presented.

7.1 References

Bai, J.; Zhang, Y.; Cui, Z.; Zhang, J. (2000): Home Telemonitoring Framework Based on Integrated Functional Modules. In: Proceedings of the 22nd Annual EMBS International Con-ference, pp. 778-781.

Beer, J.M.; Takayama, L. (2011): Mobile Remote Presence Systems for Older Adults: Ac-cptance, Benefits and Concerns. In: ACM Proceedings of Human Robot Interaction, pp. 19-26.

Brell, M.; Meyer, J.; Frenken, T.; Hein, A. (2010): A mobile robot for self-selected gait velocity assessments in assistive environments: a robotic driven approach to bring assistive technologies into established homes. In: Proceedings of the 3rd International Conference on Pervasive Technologies Related to Assistive Environments (PETRA '10), pp. 15:1 – 15:8.

Date, C.J.; Darwen, H. (1987): A Guide to the SQL Standard. Addison-Wesley, New York.

Dinh, A.; Teng, D.; Chen, L; Shi, Y.; McCrosky, C.; Basran, J.; Del Bello-Hass, V. (2009): Implementation of a Physical Activity Monitoring System for The Elderly People With Built-in Vital Sign and Fall Detection. In: Sixth International Conference on Infor-mation Technology: New Generations, pp. 1226-1231.

Downey, T. (2007): Web Development with Java. Springer Science + Business Media.

Lektorat Pflege (2007): Pflege Heute. 4. Auflage. Elsevier GmbH, Urban & Fischer Verlag.

Lorenz, A.; Mielke, D.; Oppermann, R.; Zahl, L. (2007): Personalized Mobile Health Monitoring for Elderly. In: Proceedings of the 9th international conference on Human com-puter interaction with mobile devices and services, pp. 297-304.

McKenzie, C. (2008): Hibernate made easy. Simplified Data Persistence with Hibernate and JPA Annotations. www.hiberbooks.com.

Mufti, M.; Agouridis, D.; ud Din, S.; Mukhtar, A. (2009): Ubiquitous wireless infrastructure for elderly care. In: Proceedings of the 2nd International Conference on Pervasive Tech-nologies Related to Assistive Environments, pp 22:1 – 22:5.

Oppel, A. (2009): Databases A Beginner's Guide. McGraw-Hill, Inc., Edition 1.

Rocha, V.; Borza, P.; Correia, J.; Goncalves, G.; Puscas, A.; Seromenho, R.; Mascioletti, A.; Picano, A.; Cocorada, S.; Carp, M. (2010): Wearable computing for patients wth coro-nary diseases: Gathering efforts by comparing methods. In: International Conference on Au-tomation Quality and Testing Robotics, pp. 1-9.

Sabzevar, A. P.; Stavrou, A., (2008): Universal Multi-Factor Authentication Using Graphical Passwords, IEEE International Conference on Signal Image Technology and Internet Based Systems (SITIS).

Salvador, P.; Nogueira, A.; Valadas, R. (2009): Markovian Models for Medical Signals on Wireless Sensor Networks. In: IEEE International Conference on Communications Work-shops, pp. 1-5.

Silberschatz, A.; Korth, H.; Sudarshan, S. (2006): Database System Concepts, McGraw-Hill, inc. Edition 5.

Waluyo, A.B.; Yeoh, W.; Pek, I.; Yong, Y.; Chen, X. (2010): MobiSense: Mobile Body Sensor Network for Ambulatory Monitoring. In: ACM Transactions on Embedded Compu-ting Systems, pp. 13:01-13:30.

Zeitlinger, M.; Bruckner, M.; Guger, C.; Hintermüller, C. (2010): Specification Physio Module, v 1.10.4.

7.2 Web Sources

http://www.zephyr-technology.com/store/hxm-development-kit.html

http://unisec.blogspot.com/2007/11/three-types-of-authentication.html

http://tomcat.apache.org/

http://www.highcharts.com/license

http://www.springsource.org