Software Requirement Specification Document for BREACTOR

Nader El-Sheikh, Doaa Hamed, Abdelrahman Shahata,
Ahmed Hany
Supervised by: Dr. Sarah Nabil
Teaching Assistant : Menna Gamil

November 15, 2017

1 Introduction

1.1 Purpose of this document

The purpose of Software Requirement Specification is explaining functional and non-functional requirements as we can find that driver can use system to create new account as new driver with new data set , also we can find that driver can check the system and screen as they're working or not . The system should work on reading EEG signals from the biosignals and filter signals from noise and detect the channels to classify specific signals, then BREACTOR can compare between new signals and data set to find if any abnormal behavior detected system should send alert and system should do reports across every trip and at the end of trip system shall output summary report for driver and save all summaries reports to be taken by industry automotive .

1.2 Scope of this document

Our document Scope starts from reading signals from drivers brain and most of these signals need to be filtered from noise to detect main signal we need, The second Step is the pre-processing as we begin to filter the signal from any type of noise that make interruption to extract the signal. The Third Step is to make the signal readable by Computer which leads us to the Forth step that classify all types of signals by specific algorithm then, Fifth step that we start to compare between the extracted signal and the data-set of abnormal behaviors, The Sixth Step is interacting with driver as sending warning messages and alerts at same time of detecting the abnormal behavior of driver. The last Step is if the type of behavior not exist in data-set and Signal was detected as new behavior so it have to be updated in the data-set as a new Behavior.

1.3 Overview

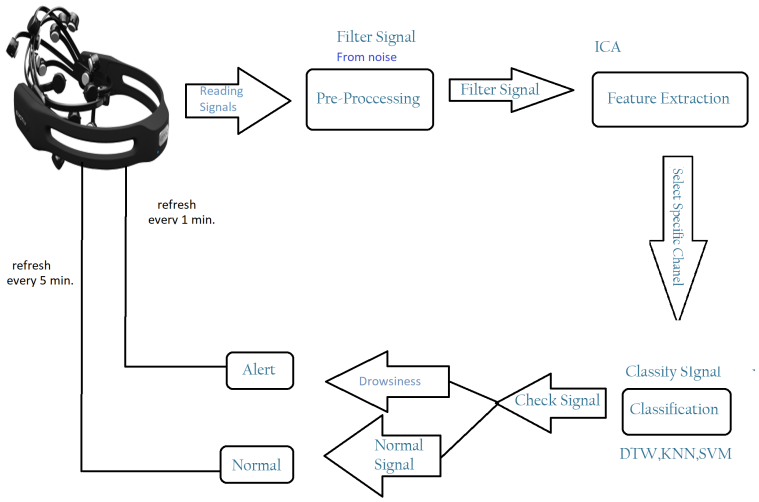

Figure 1: BREACTOR's Overview Diagram

First step as in the figure 1 the emotive must be ready to work and start reading signals after that the signals will be filtered from noise, to extract the specific signals then can classify the signals where we will use some algorithm to classify the signals . Finally the signal will be saved and checked, if signal result is drowsiness the system will alert the driver and repeat the whole process every one minute on the other hand if the signal is normal the process will be repeated every 5 minutes.

1.4 Business Context

The main reason for car accidents are human as 94% of car accidents are caused by human error, it means that drivers can significantly reduce their own risk of becoming involved in an accident, so from here we came up with idea to help in reducing car accidents; which is to detect the diver when driver starts to loss his concentration on driving by reading the biological signal specially the brain signals and the muscle signals this we can know if driver is able to drive or no moreover by creating an account on this program the user can choose what type of alert would he prefer to be alert with. In case of finding the driver not concentrating on driving the program will alert him and send message to a member will be chosen while creating an account. using this program driver will feel more comfortable and safe while driving and this will affect in reducing the road traffic accidents. [5]

2 General Description

2.1 Product Functions

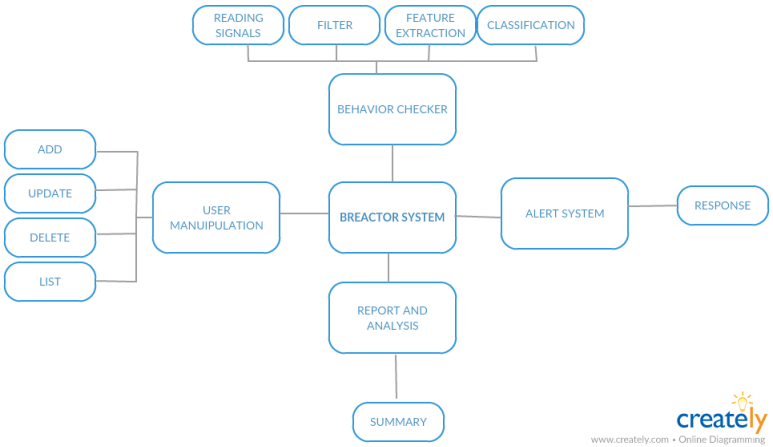

Figure 2: BREACTOR's Block Diagram

2.1.1 Module 1: User Manipulation

ADD User can active a new account on his car as new driver.
DELETE User can delete or update his profile.
UPDATE User can add some option about how to react when system detect abnormal behavior.
LIST User can show all user using same car

2.1.2 Module 2: Checking Behavior

System will be checking behavior to detect any abnormal behavior by READING,FILTER,FEATURE EXTRACTION and CLASSIFICATION the signal

2.1.3 Module 3: Real-time Interaction

Checking the response of the driver with the alert system to test accuracy of system

2.1.4 Module 4: Reporting and Analysis

User can get feedback about his driving attitude

2.2 Similar System Information

The motivation of "EEG-based motor imagery classification in BCI system by using unscented Kalman filter" work is to improve the feature data extraction from the system. It briefly detect the driver behavior using UKF classification with CSSP filter applied. There was a various results because the system is tested with CSSP compiling with other algorithms in order to get the best results. The first algorithms for BCI competition data set is CSSP + LDA + UKF and its best accuracy was 93.3 % with 30.1s training time and 82.8-100 % confident interval. The upcoming Results are Accuracy results of BCI competition III data set IIIa when using seven channels, with and without applying UKF. The first algorithm in it was CSSP + LDA with best accuracy of 86.1 % and training time 0.7s. The second algorithm was CSSP + LDA + UKF which gave us the accuracy of 99.7 % in 5.6 training time. Actually this paper is one of the important papers Ive read because it includes many resources I needed as the Algorithm used and filters. [4]

2.3 User Characteristics

The system contains many user as Drivers where The driver that is using this system not necessary to be fluent in English or have an experience with dealing with computers because all the system needs from him as an input is his signals as he didn't need to make any effort except to wear the device that reads his signals, and even the output will be very simple and visual report to suits with all the levels of education. Also there is the industry of automotive here is all they need is a report from the driver about their percentage of concentration while driving also they will could provide a well alert condition to wake them up as a vibrated chair.in addition of the owner of the car, he have the privilege to manage the user data as creating a new user for the car and also he can remove users from the car data moreover he can update and reed all the users and their information in the car. Moreover the alarm system it is responsible for the alert which is will be the output for the driver, also there is many types of alert as sound and messages.

2.4 User Problem Statement

The main problem in most of car accidents are caused by abnormal behavior from the driver , according to statistics it have been proved that 90 percent of car accidents caused by human error which leads nearly to 1.3 million people die in road crashes each year. So we have detected the problem and we have found the solution to avoid most of car accidents.[6]

2.5 User Objectives

The drivers are aiming for more security with less effort, so they want an easy way to be safe, The cars with more luxuries options in safety are more attractive

so that these are the most important aims from the driver toward the vehicle: -flexible, gradual and smooth distribution of tasks between driver and automation to better handle critical driving situations, to monitor, understand, assess and anticipate the driver, also reducing processing time and try to fit the memory. These solutions will serve two main goals, test the safety, efficiency and effectiveness and consider security, legal and societal issues.

2.6 General Constraints

-Emotive device
-high processing
-big memory to store data set
-Both the web portal and the mobile application will be constrained by the capacity of the database. Since the database is shared between both application it may be forced to queue incoming requests and therefor increase the time it takes to fetch data.
-alarm to wake him up
-report about the driver status
-We must use many laptops as each car need a laptop. we used laptop instead of mobile because mobile can't handle the heavy process

3 Functional Requirements

BREACTOR / SRS / Functional Requirements	
Code	Owner01
Name	Add new driver
Type	Functional Requirement
Critically	High
Input	Driver Name ,user name , password, confirm password , Date of birth
Output	Confirmation message and new record is added to data
Description	user data will be saved in the memory card
Priority	10/10
Expected Risks	Adding too many drivers
Pre-conditions	Must turn on car and launch the System
Post-conditions	New driver added to system
Dependencies	None

BREACTOR / SRS / Functional Requirements	
Code	Owner02
Name	Update Driver
Type	Functional Requirement
Critically	High
Input	Drivers name ,Date of birth, password, user name
Output	Successful message
Description	This function allow user to update his account
Priority	9/10
Expected Risks	
Pre-conditions	Owner want to update account
Post-conditions	Updated record
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	Driver01
Name	Edit Driver
Type	Functional Requirement
Critically	High
Input	driver's password
Output	Successful message
Description	This function allow user to update his password
Priority	9/10
Expected Risks	
Pre-conditions	Owner want to update his password
Post-conditions	Updated record
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	Owner03
Name	Delete Driver
Type	Functional Requirement
Critically	High
Input	Drivers user name
Output	Successful message for the deleted driver
Description	This function allow user to delete driver account from this system
Priority	10/10
Expected Risks	None
Pre-conditions	Driver will not drive the car again
Post-conditions	Record Deleted , No data for this driver
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	Owner04
Name	List All Drivers
Type	Functional Requirement
Critically	High
Input	None
Output	List of drivers names
Description	This function allow user to list all the drivers on the system
Priority	9/10
Expected Risks	None
Pre-conditions	Car owner want to see who is the drivers on this car
Post-conditions	List of drivers shown
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	ALERT01
Name	Send Alert
Type	Functional Requirement
Critically	High
Input	None
Output	Alert message with some options from the preferable
Description	This functions generated to alert the driver to concentrate on the road
Priority	10/10
Expected Risks	Alerting while the driver's signal is normal
Pre-conditions	System is detecting abnormal behavior
Post-conditions	Alerts for user to concentrate
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	BREACTOR01
Name	Check Connection Emotiv
Type	Functional Requirement
Critically	High
Input	None
Output	confirmation that Emotiv is ready to use.
Description	the Emotiv will power on when user launch the Car
Priority	
Expected Risks	
Pre-conditions	User launch his car .
Post-conditions	check if the device is unable to launch, give the user a warning, else itll return true.
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	BREACTOR02
Name	Detect Drowsiness
Type	Functional Requirement
Critically	High
Input	data set result, classification result
Output	0 = drowsing , 1 = awake
Description	the system should output the condition according to the detected signals.
Priority	10/10
Expected Risks	system get bad accuracy
Pre-conditions	there must be some data set.
Post-conditions	the screen will announce the user condition
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	Driver02
Name	show summary
Type	Functional Requirement
Critically	Medium
Input	
Output	summary analysis percentage to each condition since the launch of the application till the end of the trip
Description	the system will output visualize summary to the user
Priority	7/10
Expected Risks	
Pre-conditions	there must be some dataset.
Post-conditions	the screen will announce the user condition
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	BREACTOR03
Name	Save trip summary
Type	Functional Requirement
Critically	Medium
Input	Conditions summary
Output	
Description	user data will be saved in database once he finish the trip.
Priority	
Expected Risks	
Pre-conditions	the user switched off the car
Post-conditions	it will add the user data summery to the database
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	DRIVER03
Name	Load Trip Summary
Type	Functional Requirement
Critically	Medium
Input	user name
Output	Summary
Description	user data will be loaded from database once he finished the trip.
Priority	
Expected Risks	
Pre-conditions	the user switched off the car
Post-conditions	it will add the user data summery to the database
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	BREACTOR04
Name	Save Driver Signal
Type	Functional Requirement
Critically	High
Input	signal, date time, finger ID
Output	
Description	user data will be saved in the memory card
Priority	10/10
Expected Risks	
Pre-conditions	the system have read a new signal.
Post-conditions	data will be inserted in the user database
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	BREACTOR05
Name	filter signals
Type	Functional Requirement
Critically	High
Input	signals
Output	filtered Signals.
Description	the used signals will be filtered according to a certain algorithm to remove noise from the signals.
Priority	10/10
Expected Risks	
Pre-conditions	system start receiving signals
Post-conditions	go to function BREACTOR04 .
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	DRIVER04
Name	Login
Type	Functional Requirement
Critically	High
Input	User name , password
Output	Logged into system
Description	The user create account to the system
Priority	10/10
Expected Risks	
Pre-conditions	The user cant use the system
Post-conditions	The user can use the system
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	DRIVER05
Name	Password encryption
Type	Functional Requirement
Critically	medium
Input	Password the user entered
Output	Encrypted password
Description	the password will encrypt to increase the security
Priority	10/10
Expected Risks	
Pre-conditions	the password is not encrypted in the database
Post-conditions	Password encrypted in the database
Dependencies	Security of the user information

BREACTOR / SRS / Functional Requirements	
Code	BREACTOR06
Name	Reading the signals
Type	Functional Requirement
Critically	High
Input	EEG signals
Output	Data of signals
Description	This function is responsible for reading the signals from Dataset
Priority	10/10
Expected Risks	
Pre-conditions	check Connection , No data is read to detect drivers state Detecting the signals
Post-conditions	To extract and classifies the data
Dependencies	Check Connection

BREACTOR / SRS / Functional Requirements	
Code	BREACTOR07
Name	Extracting Features
Type	Functional Requirement
Critically	High
Input	Data set contains many signals
Output	The exact data needed
Description	Takes from the signals what I will use to detect drivers state
Priority	10/10
Expected Risks	
Pre-conditions	The system contains all the signals from the driver
Post-conditions	Only signals that the system will use
Dependencies	Classification of data

BREACTOR / SRS / Functional Requirements	
Code	BREACTOR08
Name	Classifying Testing signals
Type	Function Requirement
Critically	High
Input	Signals from driver
Output	The state of signal
Description	Classifing of the signal
Priority	10/10
Expected Risks	
Pre-conditions	Having signals without knowing what the meaning of each
Post-conditions	Knowing what the signals contains
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	BREACTOR09
Name	Classifying Training signals
Type	Function Requirement
Critically	High
Input	Signals from driver
Output	The state of signal
Description	Classifying of the signal
Priority	10/10
Expected Risks	
Pre-conditions	Having signals without knowing what the meaning of each
Post-conditions	Knowing what the signals contains
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	DRIVER06
Name	Choose Alert
Type	Function Requirement
Critically	High
Input	Type of alerts
Output	Confirmation message
Description	Driver can choose type of alerts
Priority	7/10
Expected Risks	The chosen type of alert is not the best choice to make driver concentrate
Pre-conditions	Adding New Driver
Post-conditions	New Alert is inserted into database
Dependencies	OWNER01

BREACTOR / SRS / Functional Requirements	
Code	DRIVER07
Name	Switch User
Type	Function Requirement
Critically	High
Input	
Output	All user names
Description	The Driver wants to switch to his account
Priority	9/10
Expected Risks	The Driver didn't find his account
Pre-conditions	Another Driver want to log in
Post-conditions	Driver switched Successfully
Dependencies	OWNER04

BREACTOR / SRS / Functional Requirements	
Code	OWNER05
Name	Get Reports
Type	Function Requirement
Critically	High
Input	the owner request to view the drivers report
Output	reports for each driver using the owners car
Description	the owner wants data about the drivers so he requests their reports
Priority	8/10
Expected Risks	
Pre-conditions	the owner logged in the system and want to retrieve data about the drivers
Post-conditions	the drivers data will be displayed for the owner
Dependencies	owner can find statistics for the drivers

BREACTOR / SRS / Functional Requirements	
Code	OWNER06
Name	Control Report
Type	Function Requirement
Critically	High
Input	Object from report and reportUI
Output	display the report
Description	the function use an object from the report to control the data to display it to the user
Priority	8/10
Expected Risks	
Pre-conditions	Report Exists
Post-conditions	Report displayed
Dependencies	OWNER05

BREACTOR / SRS / Functional Requirements	
Code	OWNER07
Name	Control User
Type	Function Requirement
Critically	High
Input	Object from user and user UI
Output	Display Controlled data from user
Description	This function control the user interface and the model of user
Priority	9/10
Expected Risks	
Pre-conditions	Owner want to manage drivers accounts
Post-conditions	Owner manage a specific driver account
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	DRIVER08
Name	Control password
Type	Function Requirement
Critically	High
Input	Object from Driver and Driver UI
Output	Can update driver's password
Description	This function can update driver's password
Priority	9/10
Expected Risks	Forget the old password
Pre-conditions	Driver want to update his password
Post-conditions	Updated Password Successfully
Dependencies	Login

BREACTOR / SRS / Functional Requirements	
Code	DRIVER08
Name	Control Alert
Type	Function Requirement
Critically	High
Input	object from alert and alert UI
Output	display the alert
Description	This function can control alert types
Priority	9/10
Expected Risks	
Pre-conditions	Driver want to choose alert
Post-conditions	Driver selected alert type
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	DRIVER09
Name	Control Switched User
Type	Function Requirement
Critically	High
Input	object from driver and driverUI
Output	display the lists of usernames
Description	the driver can switch to his account easily
Priority	10/10
Expected Risks	the driver won't find his name
Pre-conditions	another driver account will be logged in
Post-conditions	the current driver account will be logged in
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	DRIVER10
Name	Control Summary
Type	Function Requirement
Critically	High
Input	Object from summary and from summary UI
Output	display drivers summary
Description	it controls the saved summary to be displayed for the driver
Priority	9/10
Expected Risks	
Pre-conditions	Trip ended
Post-conditions	Trip Summary displayed
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	CARENGINEER01
Name	Export Data
Type	Function Requirement
Critically	High
Input	
Output	data
Description	This function works on exporting the report of driver's summaries
Priority	10/10
Expected Risks	
Pre-conditions	Reports saved into database
Post-conditions	Car Engineer exported reports successfully
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	DRIVER11
Name	Give Feedback
Type	Function Requirement
Critically	High
Input	drivers input his feedback
Output	percentage of drowsiness
Description	the drives gives the feedback of his trip to be compared
Priority	9/10
Expected Risks	driver input wrong data
Pre-conditions	the system saved the driver predicted signals
Post-conditions	compare the predicted and the driver feedback signals
Dependencies	BREACTOR02

BREACTOR / SRS / Functional Requirements	
Code	BREACTOR10
Name	Add Channel
Type	Function Requirement
Critically	High
Input	channel location
Output	saved new channel location
Description	adding new channels to the database
Priority	10/10
Expected Risks	
Pre-conditions	the user want to add new channel
Post-conditions	a new channel is added
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	BREACTOR11
Name	Remove Channel
Type	Function Requirement
Critically	High
Input	channel name
Output	successfully removed
Description	removing channel
Priority	9/10
Expected Risks	
Pre-conditions	Channel is exist
Post-conditions	Channel is deleted
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	BREACTOR12
Name	Add Report
Type	Function Requirement
Critically	High
Input	report specifications
Output	new report generated
Description	the function generates report for drivers
Priority	7/10
Expected Risks	
Pre-conditions	Many summaries saved in database
Post-conditions	new report will be added
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	BREACTOR13
Name	Control Report
Type	Function Requirement
Critically	High
Input	object from report and report UI
Output	display report
Description	controls how to display the reports
Priority	9/10
Expected Risks	
Pre-conditions	Reports are saved to database
Post-conditions	reports are successfully displayed
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	BREACTOR14
Name	Control Feedback
Type	Function Requirement
Critically	medium
Input	object from the feedback and from feedback UI
Output	user interface to take the driver feedback
Description	the control feedback will take object from feedback class to take the drivers feedback and display its user interface
Priority	7/10
Expected Risks	the driver enter any feedback
Pre-conditions	the system wants to know the driver feedback
Post-conditions	the driver entered his feedback
Dependencies	increase the accuracy for classifying data

BREACTOR / SRS / Functional Requirements	
Code	BREACTOR15
Name	Get Instance
Type	Function Requirement
Critically	High
Input	
Output	instance of connection object
Description	This function return object of connection if not exist it will generate new object of connection
Priority	9/10
Expected Risks	
Pre-conditions	Usage of database functions
Post-conditions	Check the connection
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	BREACTOR16
Name	Insert into Data base
Type	Function Requirement
Critically	High
Input	Query
Output	successfully the data saved
Description	this function is inserting data in the database
Priority	10/10
Expected Risks	
Pre-conditions	data want to be saved
Post-conditions	successfully data saved
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	BREACTOR17
Name	Retrieve From database
Type	Function Requirement
Critically	High
Input	database Statement
Output	data from database
Description	retrieve the data will be used from database
Priority	10/10
Expected Risks	
Pre-conditions	The System want to make a process on saved data in database
Post-conditions	data retrieved from database
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	BREACTOR18
Name	Fit
Type	Function Requirement
Critically	High
Input	Features and Labels
Output	
Description	This function works on training on data set with labels
Priority	10/10
Expected Risks	
Pre-conditions	
Post-conditions	System is trained with entered feature with labels
Dependencies	

BREACTOR / SRS / Functional Requirements	
Code	BREACTOR19
Name	Predict
Type	Function Requirement
Critically	High
Input	Sample of features
Output	Label
Description	This function test the sample features and the output the predicted label
Priority	10/10
Expected Risks	
Pre-conditions	
Post-conditions	
Dependencies	BREACTOR18

BREACTOR / SRS / Functional Requirements	
Code	DRIVER12
Name	List Users
Type	Function Requirement
Critically	High
Input	the driver request to list all the car drivers username
Output	List User Objects
Description	The Driver want to list all users
Priority	8/10
Expected Risks	the driver didnt find his username
Pre-conditions	Driver want to switch to his account
Post-conditions	System list all users
Dependencies	more than one driver use the same car

BREACTOR / SRS / Functional Requirements	
Code	BREACTOR20
Name	Calculate Distance
Type	Function Requirement
Critically	High
Input	Features , initial points
Output	Distance data sorted
Description	This function sort data
Priority	10/10
Expected Risks	
Pre-conditions	
Post-conditions	
Dependencies	

4 Interface Requirements

4.1 User Interfaces

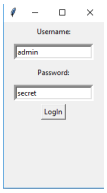

Figure 3: BREACTOR's LogIn UI

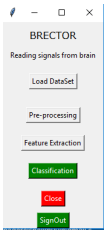

Figure 4: BREACTOR's Functionality UI

4.1.1 CLI

-git (Bitbucket)
This command line tool is a utility for Bitbucket Server (where you down-

load and host bitbucket yourself). If you are a user of Bitbucket Cloud (bitbucket.org) then this tool is not for you. [1]

4.1.2 API

-Emotiv SDK and API
Cortex is our new API powerhouse for creating BCI applications and integrating data streams from our headsets with third party software. The API is built on JSON and WebSockets, making it easy to access from a variety of programming languages and platforms. You can use Cortex to create games and apps, or record data for experiments. Cortex is a wrapper around our Software Development Kit (SDK) and houses all the tools required to develop with EMOTIV. It provides API access to different EMOTIV data streams, tiered out across three license levels. Join our worldwide community of developers and start creating your own application now.[2]
-tensorflow API :
using NN in tensorflow for neural network. using tflearn as high-level API in deep learning for tensorflow .

4.1.3 Diagnostics or ROM

Emotiv Epoc device is a Neuro headset that offer high and full spatial resolution. The device is a 14 channels that read wireless EEG. It is easy to use and flexible as its a wireless device that work up to 12 hours and rechargeable. Emotiv is compatible with Windows, OSX, Linux, Android, and iOS. It has a reference In the CMS/DRL noise cancellation configuration P3/P4 locations.

For device specifics, we dont need all of the 14 channels (AF3, F7, F3, FC5, T7, P7, O1, O2, P8, T8, FC6, F4, F8, AF4). As for drowsiness, only channels as (Fz, Cz, C3, C4, Pz).in signal resolutions, the sample method is Sequential sampling. Sample rates are 128 SPS or 256 SPS (2048 HZ). Resolution is 14 bits 1 LSB = 0.51V (16 bit ADC, 2 bits instrumental noise floor discarded), or 16 bits. Bandwidth are 0.2 43Hz, digital notch filters at 50Hz and 60Hz.Filtering are built in digital 5 th order Since filter. It has Dynamic range (input referred) of 8400V. The device work 12 hours on wireless connection or 6 hours in Bluetooth

4.2 Hardware Interfaces

The application require Emotiv device to read signals and a display screen to interact with user. Also it must have an internal memory ship to store each drivers behavior of a certain car,it needs an alert device to wake up the driver if he felt drowsiness. All that hardware interface shall be connected together. The main component of BREACTOR is the software application inside that system. The Emotiv must be as its battery life is 12 hours using wireless and 6 hours using Bluetooth.[3]

4.3 Communications Interfaces

4.4 Software Interfaces

5 Performance Requirements

System shall read and check signals every 10 minutes and if accuracy compared to it found to be near to be abnormal so it will refresh with smaller interval of time

6 Design Constraints

6.1 Standards Compliance

This section includes the design constraints on the software caused by the hardware. Other constraints such as limited memory and processing power are also worth considering. BREACTOR is meant to be quick and responsive, even when dealing with large data and classifications, so each feature must be designed and implemented with efficiency in mind.

6.2 Hardware Limitations

Internal memory size:
TAG: Memory ship space
GIST: the memory card space.
SCALE: GB.
METER: Observations done from the performance log during testing
MUST: No more than 2 GB.
PLAN: No more than 1.5 GB
WISH: No more than 1 GB
Operate System: DEFINED: The Operate System which the application is stored on.
DESC: As every single car will have the data of the users and their signals with the learned dataset so we choosed the limitation wisely. In addition on , the data will not exceed 2GB because the data that's older than 1 year will be erased.

Application memory usage:
TAG: Application Memory Usage
GIST: The amount of Operate System memory occupied by the application.
SCALE: GB.
METER: Observations done from the performance log during testing on deep learning
MUST: No more than 8 GB.
PLAN: No more than 6 GB
WISH: No more than 4 GB

Operate System: DEFINED: The Operate System which the application is running on.
GB DEFINED: Gega byte

7 Other non-functional attributes

BREACTOR / SRS / Non-Functional Requirements	
Code	BREACTOR21
Type of Requirement	Security
TAG	Authentication Security
GIST	The security of the system.
SCALE	The messages should be encrypted for log-in communication, so others cannot get user-name and password from those messages.
METER	attempts to get user-name and password through obtained messages on 1000 log-in session during testing.
MUST	100 percent of the message is encrypted
DESC	The user information should be Encrypted to avoid public access by any other user

BREACTOR / SRS /Non-Functional Requirements	
Code	BREACTOR22
Type of Requirement	Reliability
TAG	System Reliability
GIST	The reliability of the system.
SCALE	The reliability that the system get the most accurate result to the user.The system should have an average performance rate and at any failure ,system should get a feedback
METER	Measurements obtained from 36+ trained data set files
MUST	More than 96 % of the prediction.
PLAN	More than 98 % of the prediction
WISH	99.9 % of the prediction

BREACTOR / SRS / Non-Functional Requirements	
Code	BREACTOR23
Type of Requirement	Maintainability
TITLE	Application extendable
DESC	The application should be easy to extend thought the implementation of MVC, Observer, Template, Strategy, Decorative and Singleton design pattern.
RAT	In order for future functions to be implemented easily to the application.

BREACTOR / SRS / Non-Functional Requirements	
Code	BREACTOR24
Type of Requirement	Maintainability
TITLE	Application test-ability
DESC	Test environments should be built for the application to allow testing of the applications different functions.
RAT	In order to test the application

BREACTOR / SRS / Non-Functional Requirements	
Code	BREACTOR25
Type of Requirement	Re-usability
DESC	The system can fit on any vehicle. it doesn't require a complex installations.
RAT	In order to test the application

8 Preliminary Object-Oriented Domain Analysis

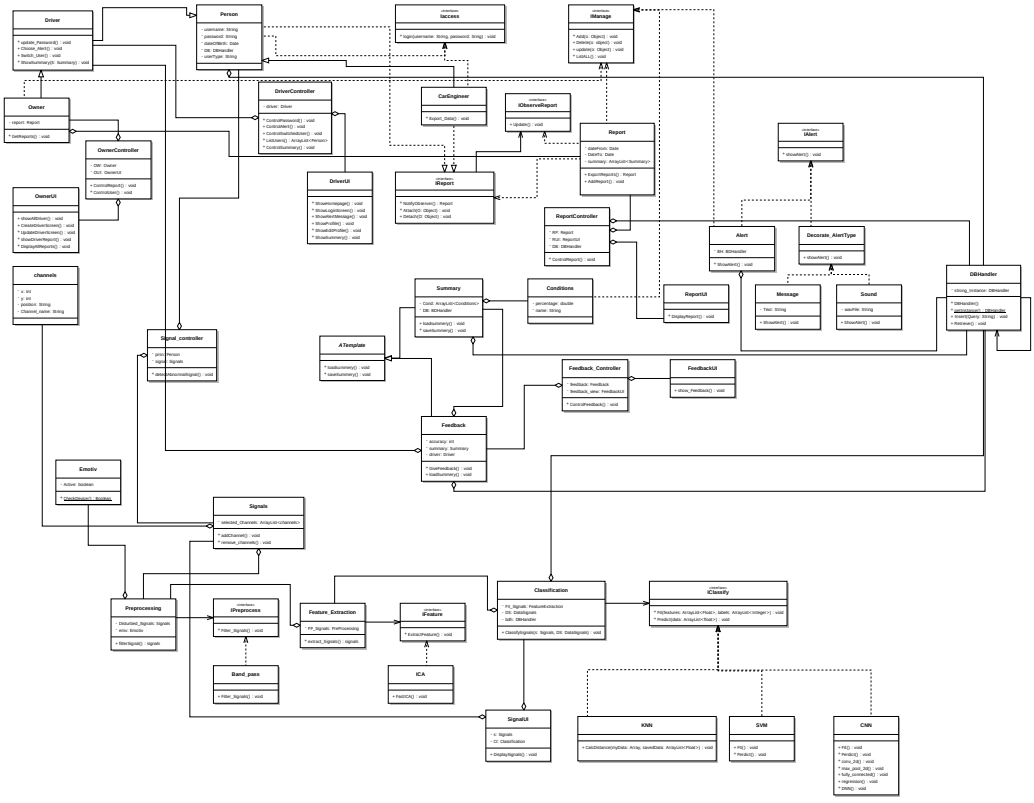

Figure 5: BREACTOR's Class Diagram

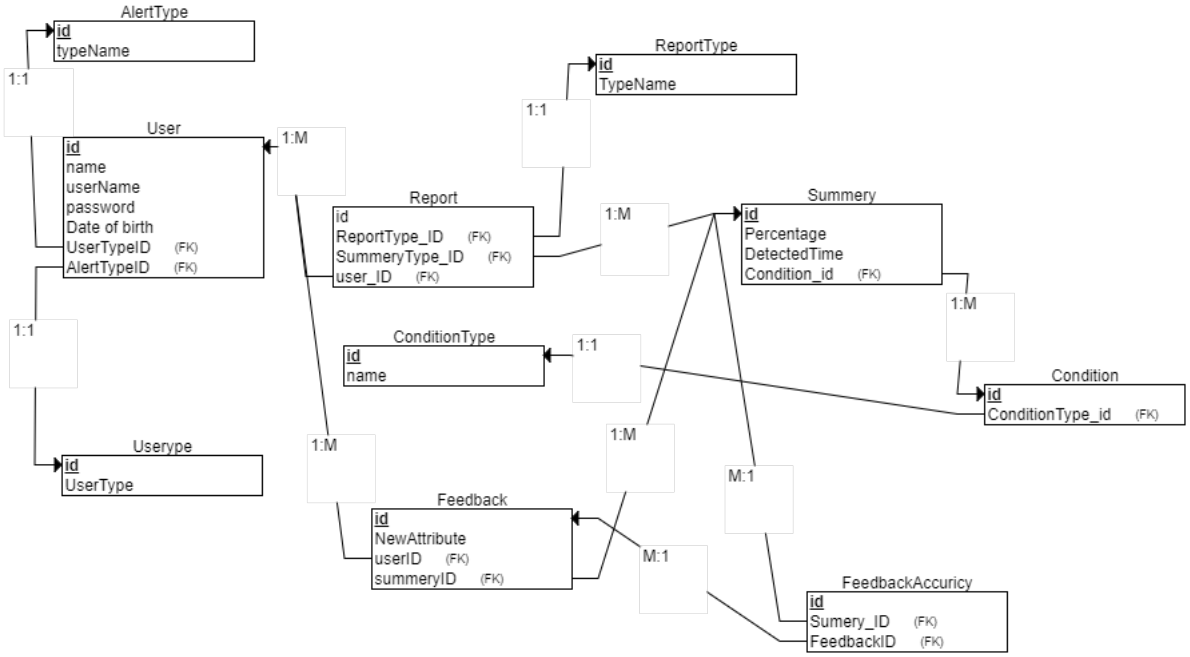

Figure 6: BREACTOR's Database Diagram

8.1 Person

8.1.1 Class Name :

Person

8.1.2 Concrete class :

8.1.3 List of Super classes:

None.

8.1.4 List of Subclasses:

Driver

8.1.5 Purpose :

Main class has users information.

8.1.6 Collaborations :

None.

8.1.7 Attributes:

-username: String
-password: String
-DateOfBirth: Date
-userType: String

8.1.8 Operations :

None

8.2 CarEngineer:

8.2.1 Concrete class :

8.2.2 List of Super classes:

Person.

8.2.3 List of Subclasses:

None

8.2.4 Purpose :

Exporting the reports for each car

8.2.5 Collaborations :

None.

8.2.6 Attributes :

None

8.2.7 Operations :

ExportData(): void

8.3 Driver :

8.3.1 Concrete class :

8.3.2 List of Super classes :

Person.

8.3.3 List of Subclasses :

Owner

8.3.4 Purpose :

Driver can set his own settings

8.3.5 Collaborations :

None.

8.3.6 Attributes :

none

8.3.7 Operations :

Updatepassword(): void ChooseAlert(): void SwitchUser(): void ShowSummary(S: Summary): void

8.4 DriverController :

8.4.1 Concrete class :

8.4.2 List of Super classes :

None.

8.4.3 List of Subclasses :

none

8.4.4 Purpose :

Controls the data from the driver.

8.4.5 Collaborations :

None.

8.4.6 Attributes :

driver: Driver

8.4.7 Operations :

ControlPassword(): void ControlAlert(): void ControlSwitchedUser(): void ControlSummary(): void

8.5 DriverUI :

8.5.1 Concrete class :

8.5.2 List of Super classes:

None.

8.5.3 List of Subclasses :

none

8.5.4 Purpose :

The screens that will appear to the Driver

8.5.5 Collaborations :

None.

8.5.6 Attributes :

none

8.5.7 Operations :

ShowHomePage(): void
ShowLoginScreen(): void
ShowAlertMessage(): void
ShowProfile(): void
ShowEditProfile(): void
ShowSummary(): void

8.6 Owner :

8.6.1 Concrete class :

8.6.2 List of Super classes :

Driver.

8.6.3 List of Subclasses :

none

8.6.4 Purpose :

Know the summaries for all drivers of the same car.

8.6.5 Collaborations :

None.

8.6.6 Attributes :

none

8.6.7 Operations :

GetReports(): void

8.7 OwnerController :

8.7.1 Concrete class :

8.7.2 List of Super classes :

None.

8.7.3 List of Subclasses :

none

8.7.4 Purpose :

Controls the data from the Owner to output the data in the view OwnerUI.

8.7.5 Collaborations :

None.

8.7.6 Attributes :

OW: Owner
OUI: OwnerUI

8.7.7 Operations :

ControlReport(): void
ControlUser(): void

8.8 OwnerUI :

8.8.1 Concrete class :

8.8.2 List of Super classes :

None.

8.8.3 List of Subclasses :

none

8.8.4 Purpose :

The screens that will appear to the Owner of the car.

8.8.5 Collaborations :

None.

8.8.6 Attributes :

none

8.8.7 Operations :

ShowAllDriver(): void
CreateDriverScreen(): void
UpdateDriverScreen(): void
ShowDriverReport(): void
DisplatAllReports(): void

8.9 Channels :

8.9.1 Concrete class :

8.9.2 List of Super classes :

None.

8.9.3 List of Subclasses :

none

8.9.4 Purpose :

To know the channel position.

8.9.5 Collaborations :

None.

8.9.6 Attributes :

x: int
y: int
position: String
Channel$_name$: *String*

8.9.7 Operations :

None

8.10 Iaccess :

8.10.1 Concrete class :

8.10.2 List of Super classes :

None.

8.10.3 List of Subclasses :

None

8.10.4 Purpose :

The person and the car engineer log into the system with difference interface

8.10.5 Collaborations :

None.

8.10.6 Attributes :

None

8.10.7 Operations :

Lpgin(username: String, Password: String): void

8.11 Imanage :

8.11.1 interface class

8.11.2 List of Super classes:

None.

8.11.3 List of Subclasses :

none

8.11.4 Purpose :

The owner, the conditions and the alert have the privilege to manage the data.

8.11.5 Collaborations:

None.

8.11.6 Attributes:

None

8.11.7 Operations

Add(o: object): void Delete(o: object): void Update(o: object):void ListALL(): void

8.12 Report:

8.12.1 Concrete class

8.12.2 List of Super classes:

None.

8.12.3 List of Subclasses:

None

8.12.4 Purpose:

Reporting the summary of the driver and add new types of reports

8.12.5 Collaborations:

None.

8.12.6 Attributes:

-dataFrom: Data DateTo: Date Summary: ArrayList¡Suummary¿

8.12.7 Operations

ExportReports(): Report
AddReport(): void

8.12.8 ReportController:

8.12.9 Concrete class

8.12.10 List of Super classes:

None.

8.12.11 List of Subclasses:

None

8.12.12 Purpose:

Manage the data from the summary to export the report

8.12.13 Collaborations:

None.

8.12.14 Attributes:

RP: Report
RUI: ReportUI

8.12.15 Operations

ControlReport(): void

8.13 ReportUI:

8.13.1 concrete class

8.13.2 List of Super classes:

None

8.13.3 List of Subclasses:

None

8.13.4 Purpose:

The view of the report

8.13.5 Collaborations:

None

8.13.6 Attributes:

None

8.13.7 Operations

DisplayReport(): void

8.14 Alert:

8.14.1 Concrete class

8.14.2 List of Super classes:

None.

8.14.3 List of Subclasses:

None

8.14.4 Purpose:

8.14.5 Collaborations:

None.

8.14.6 Attributes:

DB: DBHandler

8.14.7 Operations

showAlert().

8.15 Signals:

8.15.1 Concrete class

8.15.2 List of Super classes:

None.

8.15.3 List of Subclasses:

None

8.15.4 Purpose:

This class have signal information

8.15.5 Collaborations:

None.

8.15.6 Attributes:

Selecte$_C hannels : ArrayList < Channels >$

8.15.7 Operations

addChannel(): void
Remove$_C hannels() : void$

8.16 channels:

8.16.1 Concrete class

8.16.2 List of Super classes:

None.

8.16.3 List of Subclasses:

None

8.16.4 Purpose:

This class have signal information

8.16.5 Collaborations:

This class is aggregated with Signals.

8.16.6 Attributes:

X: int
Y: int
Position: String
Channel$_name$: $String$

8.16.7 Operations

None.

8.17 Preprocessing:

8.17.1 Concrete class

8.17.2 List of Super classes:

None.

8.17.3 List of Subclasses:

None

8.17.4 Purpose:

This class is for filtering the signals from any noises

8.17.5 Collaborations:

This class is aggregated with Signals.

8.17.6 Attributes:

Disturbed$_Signals$: $Signals$

8.17.7 Operations

filterSignal() : signals

8.18 IPreprocess:

8.18.1 Interface class

8.18.2 List of Super classes:

None.

8.18.3 List of Subclasses:

None

8.18.4 Purpose:

Its interface class and satisfying design pattern

8.18.5 Collaborations:

none

8.18.6 Attributes:

8.18.7 Operations

filterSignal() : signals

8.19 Feature$_{E}xtraction$:

8.19.1 Concrete class

8.19.2 List of Super classes:

None.

8.19.3 List of Subclasses:

None

8.19.4 Purpose:

Extract the specific signals

8.19.5 Collaborations:

none

8.19.6 Attributes:

FilteredSignals: Signals

8.19.7 Operations

Extract$_S$$ignals() : Signal$

8.20 IFeature:

8.20.1 Interface class

8.20.2 List of Super classes:

None.

8.20.3 List of Subclasses:

None

8.20.4 Purpose:

Its interface class.

8.20.5 Collaborations:

None

8.20.6 Attributes:

None

8.20.7 Operations

ExtractFeature() : void

8.21 Classification:

8.21.1 concrete class

8.21.2 List of Super classes:

None.

8.21.3 List of Subclasses:

None

8.21.4 Purpose:

Classifies the signals extracted from feature extraction step

8.21.5 Collaborations:

None

8.21.6 Attributes:

S: Signal
DS: DataSignals
Bdh: DBHandler

8.21.7 Operations

classifySignals(s: Signal, DS:DataSignals) : void

8.22 IClassify:

8.22.1 Interface class

8.22.2 List of Super classes:

None.

8.22.3 List of Subclasses:

None

8.22.4 Purpose:

Its interface class.

8.22.5 Collaborations:

None

8.22.6 Attributes:

None

8.22.7 Operations

Fit(features: ArrayList¡float¿, labels:
ArrayList¡Integer¿): void
Predict(data: ArrayList¡float¿): void

8.23 Signal$_{c}ontroller$:

8.23.1 concrete class

8.23.2 List of Super classes:

None.

8.23.3 List of Subclasses:

None

8.23.4 Purpose:

Manage the data from the signals

8.23.5 Collaborations:

None

8.23.6 Attributes:

Prsn: Person
Signal: Signals

8.23.7 Operations

detectAbnormalSignals(): void

8.24 Emotiv:

8.24.1 concrete class

8.24.2 List of Super classes:

None.

8.24.3 List of Subclasses:

None

8.24.4 Purpose:

To know if the device is working or not

8.24.5 Collaborations:

None

8.24.6 Attributes:

Active: boolean

8.24.7 Operations

CheckDevice(): Boolean

8.25 DBHandler:

8.25.1 concrete class

8.25.2 List of Super classes:

None.

8.25.3 List of Subclasses:

None

8.25.4 Purpose:

Handle the database

8.25.5 Collaborations:

None

8.25.6 Attributes:

None

8.25.7 Operations

DBHandler()
Insert(): void
Retrieve(): void

8.26 Decorate$_{A}lertType$:

8.26.1 abstract class

8.26.2 List of Super classes:

iAlert.

8.26.3 List of Subclasses:

Sound, message

8.26.4 Purpose:

Its abstract class.

8.26.5 Collaborations:

8.26.6 Attributes:

None

8.26.7 Operations

ShowAlert(): void

8.27 Message:

8.27.1 concrete class

8.27.2 List of Super classes:

Decorate$_A lertType$.

8.27.3 List of Subclasses:

None

8.27.4 Purpose:

None

8.27.5 Collaborations:

None

8.27.6 Attributes:

Wavfile: String

8.27.7 Operations

ShowAlert() : void

8.28 message:

8.28.1 concrete class

8.28.2 List of Super classes:

Decorate$_A lertType$.

8.28.3 List of Subclasses:

None

8.28.4 Purpose:

None

8.28.5 Collaborations:

None

8.28.6 Attributes:

Text: String

8.28.7 Operations

ShowAlert() : void

8.29 Condition:

8.29.1 concrete class

8.29.2 List of Super classes:

None

8.29.3 List of Subclasses:

None

8.29.4 Purpose:

Know each signal where it is belong in the classifiers

8.29.5 Collaborations:

None

8.29.6 Attributes:

Percentage: double
Name: String

8.29.7 Operations

None

8.30 summary:

8.30.1 concrete class

8.30.2 List of Super classes:

None

8.30.3 List of Subclasses:

None

8.30.4 Purpose:

To know the condition of the driver after each trip

8.30.5 Collaborations:

None

8.30.6 Attributes:

Cond: ArrayList¡Conditions¿

8.30.7 Operations

None

8.31 Ireport:

8.31.1 interface class

8.31.2 List of Super classes:

None

8.31.3 List of Subclasses:

Iaccess, Car$_e ngineer$

8.31.4 Purpose:

Interface class

8.31.5 Collaborations:

None

8.31.6 Attributes:

None

8.31.7 Operations

NotifyObserver(): Report
Attach(O: Object)
Detach(O: Object): void

9 Operational Scenarios

9.1 Use Case Diagram

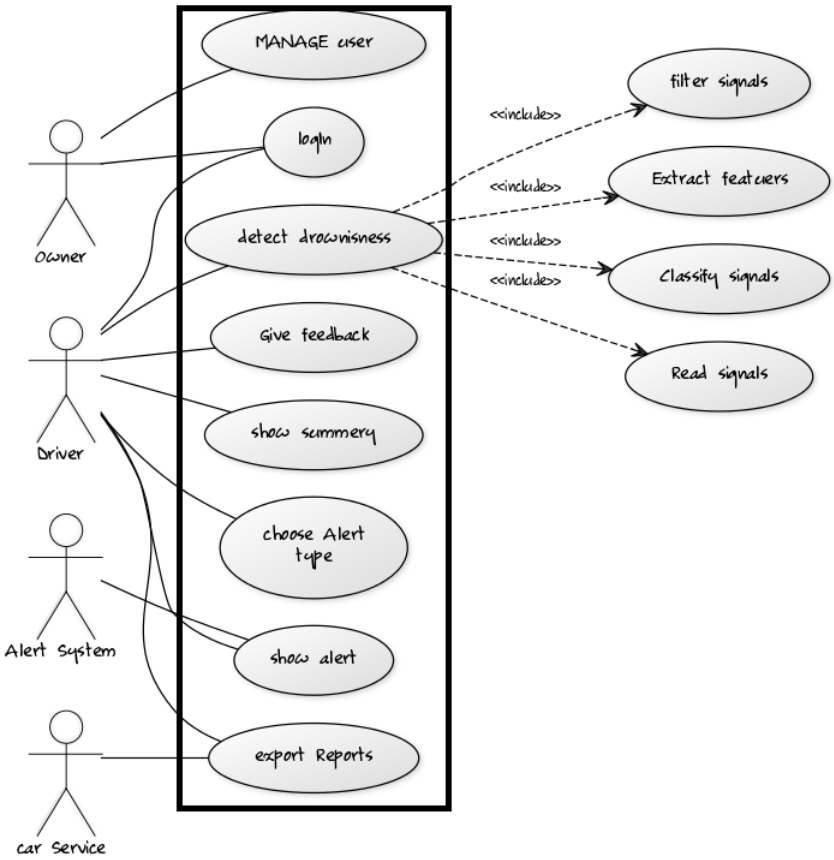

Figure 7: BREACTOR's Use Case Diagram

Mange user: owner have more privileges and can control the drivers. Choose alert: give the driver options of the alert to choose if there is many options Ex: Sound, light and vibration.
Detect drowsiness: the main function of the system that go throw 3 steps to get if driver is consented or not (Filter, Extract, classify, read signal).
Login: give users access throw system using user name and password.
Show Alert: The output of the alert that done if there is any detection of drowsiness.
Show Summary: This function is to show report after every trip there is summary will be created about the status of the driver.
Export Report: its a quality assurance for the car industry to know that our system is running well so they compare between reports and the driver survey like rating.
Give feedback: after the trip finish the system ask for a feedback to compare it with the report its as a survey from 0 to 10 (where 0 is very sleepy and 10 is so concentrated).

10 Preliminary Schedule Adjusted

Tasks will be handled according to the waterfall model since we have abundant time and this is a graduation project. We will go through its 6 phases

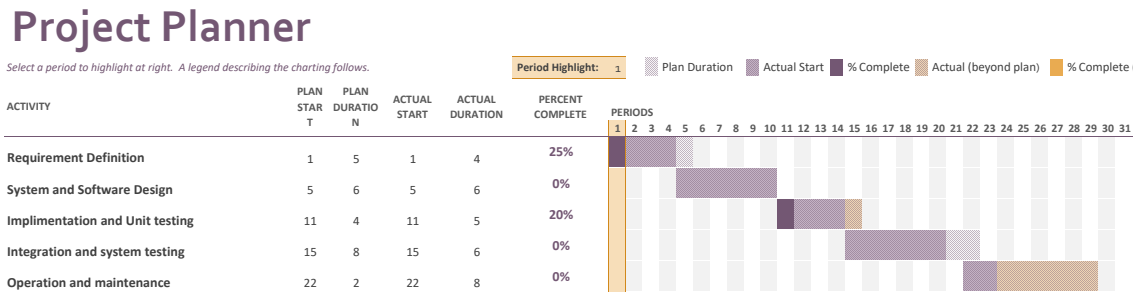

Figure 8: BREACTOR's Project Time Planner

11 Preliminary Budget Adjusted

Currently Our Budget includes the following :
Emotiv epoc+ : 299 $
MYO Device : 250 $
Output Device : 2000 LE

12 Appendices

12.1 Definitions, Acronyms, Abbreviations

BREACTOR / SRS / Definitions, Acronyms, Abbreviations	
User	Someone who interacts with the system application. Also the owners of projects who interact with the system
Stakeholder	Any person who has interaction with the system who is not a developer.
System	The application which present special facilities for the user behaviours.
DESC	Description
RAT	Rational
DEP	Dependency
TAG	A unique, persistent identifer contained in a PLanguage statement .
GIST	A short, simple description of the concept contained in a PLanguage statement.
SCALE	The scale of measure used by the requirement contained in a Planguage statement .
METER	The process or device used to establish location on a SCALE contained in a PLanguage statement .
MUST	The minimum level required to avoid failure contained in a PLanguage statement.
PLAN	The level at which good success can be claimed contained in a PLanguage statement.
WISH	A desirable level of achievement that may not be attainable through available means contained in a PLanguage statement
Biosignal	Biological Signals
UKF	Unscented kalman Filter
CSSP	Common Spatial Spectral Pattern
BCI	Brain Computer Interaction
LDA	Latent Dirichlet allocation
SVM	Support Vector Machine
ICA	Independent Component Analysis
KNN	K-Nearest Neighbour
CNN	Conventional Neural Network
Hz	Hertz
ADC	Analog Digital Converter
SDK	Software Development Kit
API	Application Programming Interface
JSON	Java Script Object Notation
EEG	Electroencephalography

13 References

References

[1] Bitbucket. *Bitbucket Command Line*. URL: https://bitbucket.org/atlassian/bitbucket-server-cli.

[2] *Emotiv SDK API (Cortex)*. URL: https://www.emotiv.com/developer/.

[3] Donald Knuth. *Knuth: Computers and Typesetting*. URL: https://www.emotiv.com/epoc/.

[4] Nik Khadijah Nik Aznan Kyung-Moo Huh Yeon-Mo Yang. "EEG-based motor imagery classification in BCI system by using unscented Kalman filter". In: (2016). DOI: http://dl.acm.org/citation.cfm?id=3013771.

[5] NHTSAs National Center for Statistics and Analysis. "Critical Reasons for Crashes Investigated in the National Motor Vehicle Crash Causation Survey". In: (2015). DOI: https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812115.

[6] Association For Safe International Road Travel. "Road Crash Statistics". In: (2015). DOI: http://asirt.org/initiatives/informing-road-users/road-safety-facts/road-crash-statistics.