User Web Interactions Predictor

Software Requirements Specification

Version 1.1

March 13th, 2021

Gabriel Falcones Paredes

Prepared for
ICSI 418Y Software Engineering Course
Instructor: Dr. Mei Hwa Chen
Spring 2021

Revision History

Date	Description	Author	Comments
13/03/21	Version 1.1	Gabriel Falcones	First Revision

Document Approval

The following Software Requirements Specification has been accepted and approved by the following:

Signature	Printed Name	Role	Date
Gabriel Falcones	Gabriel Falcones	Software Developer	13/03/21

Table of Contents

1 Introduction

This Software Requirements Specification (SRS) aims to describe the Purpose, General description, Specific requirement, Functionality, Design model of the software system: User Web Interactions.

1.1 Purpose

The purpose of this SRS is to describe and define the functions, requirements and limitations of the software: User Web Interactions Predictor.

1.2 Scope

Description of the confirmation of the results that you will be produced with the relevant benefits, objectives and goals. Relate the software goals and business strategies.

1.3 Definitions, Acronyms and Abbreviations

API	application programming interface
EC2	Amazon Elastic Compute Cloud
GPU	Graphics Processing Unit
CUDA	Compute Unified Device Architecture. Interface that allows developers to use a GPU for general purpose processing

1.4 References

- *AWS: https://aws.amazon.com/*
- *Pytorch: https://pytorch.org/*
- *CUDA: https://developer.nvidia.com/cuda-toolkit*

1.5 Overview

This SRS is divided into three subsections:
The first part introduces Purpose, Scope, Definition, Acronyms, Abbreviations, References and Overview of the document.
The second part introduces product perspective, functions, user characteristics, general constraints, and dependencies.
The third part brings specific requirements, such as, external interface requirements, functional requirements, non-functional requirements, design requirements, database requirements.

2 General Description

The following product consists of a machine learning system that will be used to predict users interactions on a web application. The users will communicate with the software, inputting past user actions and obtaining a prediction of the type of action and next action the user will perform.
The results obtained from the system can be later used for many applications: Improve user experience by pre-sending and pre-fetching data, showing recommendations, web caching, etc; and also can be used to predict and recover from errors.

2.1 Product Perspective

This product will be used as an additional component of a particular web application. The product will be deployed separate from the web application and they will communicate via a web API, in order to obtain prediction of the user's intent on the website. It is required that the product is trained using the same type of data (server logs, user client interactions, etc) the web server will send to the product during production, in order to work properly and obtain effective results.

2.2 Product Functions

- Receive user web interactions, parse and convert them to a format ready to be input into a machine learning model.
- Classify different types of user actions.
- Train Neural Network Model to predict users next action and type of action.
- Update the model when new user interactions are received.
- Accept API requests by web server.
- Predict type of interaction and next user action based on data received by server.

2.3 User Characteristics

Two different users will use this product:

- Software developer: This user will be in charge of training/updating the neural network model. The user can input a dataset of new user interactions to the software and it will update itself to be able to predict accurately new user interactions. The software developer can choose whether to use the new data to update the network model or to restart the model to train from scratch using new data.
- Web server: This user will be a server application, will communicate with the software in order to obtain prediction of the users current actions on the website.

2.4 General Constraints

The software can only be used to predict user's actions on an unique website, the website used to train the prediction model. If another website needs to be used, the model needs to be modified and trained again.
Only the web server will have access to the API interface of the software, users of the websit cannot access the software directly.

The proper functioning of the software is limited to the proper functioning of the server where it will be hosted. Additionally, the hosting server will require a graphic card to provide an efficient execution of the prediction and training of the network model.

2.5 Assumptions and Dependencies

- Software Dependencies: The implementation of the software will mainly depend on the following software, which is required to be installed on the server where the software will be deployed:
    - Operating System: Linux
    - Language Interpreters: Node, Python v3
    - Libraries: Pytorch, Pandas, Numpy, ExpressJS
    - DB: MongoDB
- Hardware Dependencies:
    - Graphics Card: Nvidia with at least 4GB RAM
- Assumptions:
    - For the correct use of the software, it is assumed that the format of the data it will receive for prediction/updating will be the same as the format of the data used to initially train the model.

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

The software will interact with the users using a web API with the following endpoints:

- /predict: POST request will accept data in JSON format. The data needs to be a list of user actions and will output a JSON response with the type of action and next action predicted.
- /train/{type}: POST request, depending on the {type} parameter will either update the model or restart the training from zero. The data to be received will be a file with a collection of users actions of the web server. It will just output a confirmation message.

3.1.2 Hardware Interfaces

The only necessary communication with hardware will be the communication with the system's graphics card. This will be handled automatically with a python library (pytorch) and the CUDA interface.

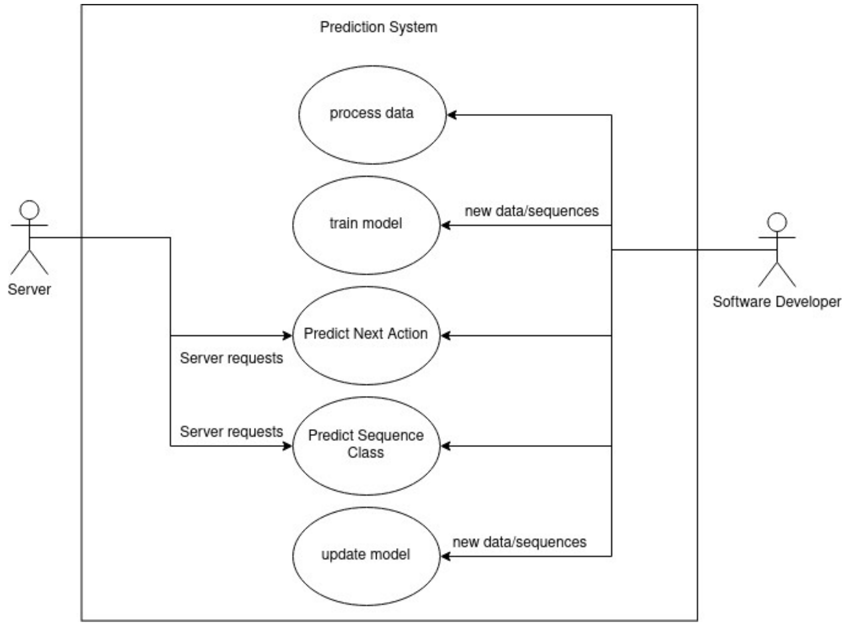
3.1.3 Software Interfaces

- Connection with the CUDA platform will be performed by the pytorch library. This will enable the software to perform processing on the GPU of the system.
- Connection with database MongoDB will be performed on NodeJs using the Mongoose Library.

3.1.4 Communications Interfaces

The software requires a network communication to interact with the users.
The communication will be performed using the HTTP protocol, using the web API previously described.
The communicatino will be restricted uniquely to the ip of the web server that will be using the software.
Any software developer will need to communicate to the software throught the web server.

3.2 Use Cases



Use Case #1	Process Data
Actor	Software Developer
Flow	- Developer collects logs of users' interactions on the website. - Developer submits logs to software via api post request - Developer receives data formatted as input to network model, which later can be used for training
Alternative Flow	- If logs format is invalid, software respond with error.

Use Case #2	Train model
Actor	Software Developer
Flow	- Developer submits file of formatted user interaction data to software via API post request, specifying to train model - Software uses data to train neural network model - Developer receives confirmation of training
Alternative Flow 1	- If data is using the necessary format, respond to actor with format error
Alternative Flow 2	- If actor doesn't have permission to edit model, respond with unauthorized error

Use Case #3	Predict next action
Actor	Software Developer or Web Server
Flow	- Actor collects list of user interactions

	- Actor submits list to software via API post request - Software makes prediction and actor receives possible next actions with confidence levels
Alternative Flow 1	- If data is using the necessary format, respond to actor with format error

Use Case #4	Predict sequence type/class
Actor	Software Developer or Web Server
Flow	- Actor collects list of user interactions - Actor submits list to software via API post request to classify. - Software performs classification - Actor receives possible classes/type of action and confidence
Alternative Flow 1	- If data is using the necessary format, respond to actor with format error

Use Case #5	Update Prediction Model
Actor	Software Developer
Flow	- Developer submits file of formatted user interaction data to software via API post request, specifying to update model - Software uses data and previous data to train neural network model - Developer receives confirmation of training
Alternative Flow 1	- If data is using the necessary format, respond to actor with format error
Alternative Flow 2	- If actor doesn't have permission to edit model, respond with unauthorized error

3.3 Functional Requirements

Req #1	**Req Name:** Submit File Logs	**Use Cases:**1,2,5	**Author:**Gabriel Falcones
Description: User should be able to submits file to server.			
Rationale: In order to collect data to preprocess/train			
Fit Criterion: Data Files should be present on system server after being submitted.			

Req #2	**Req Name:** Validate User	**Use Cases:** 1,2,3,4,5	**Author:**Gabriel Falcones
Description: System must verify user is authorized to update/train model or to request predictions.			
Rationale: Unauthorized users shouldn’t be able to access the system, otherwise can update the model and alter the prediction accuracy			
Fit Criterion: System will reject any request not from authorized server/user.			

Req #3	**Req Name:** Preprocess data	**Use Case:** 1	**Author:**Gabriel Falcones
Description: Parse data of user interactions and create discrete features to be used on the model for training and prediction			
Rationale: The neural network needs discrete cleaned numerical features in order to train model.			
Fit Criterion: User will receive file containing dataset of preprocessed data			
Dependencies: Req#1, Req#2			

Req #4	**Req Name:** Train model	**Use Cases:** 2,5	**Author:** Gabriel Falcones
Description: System will use submitted data to train neural network prediction model.			
Rationale: Model needs to be trained in order to output accurate predictions			
Fit Criterion: The accuracy of the prediction must increase on a test dataset.			
Dependencies: Req #1, Req #2, Req #3			

Req #5	**Req Name:** Predict next action and type	**Use Cases:** 3, 4	**Author:** Gabriel Falcones
Description: System will use submitted data to predict user next action and action type, and their confidence values.			
Rationale:			
Fit Criterion: When user sends action sequence, it will receive a next action and action type as result.			
Dependencies: Req#1, Req#2, Req#3			

Req# 6	**Req Name:** Update model	**Use Cases:** 5	**Author:** Gabriel Falcones
Description: System will train model with new data received from user			
Rationale: System should keep improving for new unseen user actions			
Fit Criterion: Prediction accuracy should increase for a test dataset.			
Dependencies: Req# 1, Req #2, Req #3			

Req # 7	**Req Name:** Restart Model	**Use Cases:** 5	**Author:** Gabriel Falcones
Description: System will train model from zero with new data received from user, ignoring previous train data.			
Rationale: If the previous user's actions recorded are not used anymore, they must be discarded from the model and retrain with new user actions			
Fit Criterion: Prediction accuracy should increase for a test dataset.			
Dependencies: Req# 1, Req #2, Req #3			

3.4 Non-Functional Requirements

Performance: Given an input to be predicted, the software must output a prediction in less than 3s.

Availability: The system must be available whenever the website is available. The system will only be disabled when a software developer sends input to retrain the prediction model.

Security: The system will be protected from retrainig/updating from unauthorized users.

Accuracy: The system will deliver predictions with a confidence greater than 80%.

3.5 Design Constraints

Hardware constraint: In order for the pytorch library to perform training/prediction processing on a graphics card, it needs to be an NVIDIA card with CUDA support.