Project: Black Mirror

Software Requirements Specification

Pre-Alpha 0.0.1

Feb 12th 2019

Floyd Askew

Abu Hassan

Brock Maxwell

Sean Thompson

Prepared for

CSU-CSCI3320—Software Engineering Principles I

Instructor: Junfeng Qu, Ph.D.

Spring 2019

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Description** | **Author** | **Comments** |
| <date> | <Version 1> | <Your Name> | <First Revision> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Document Approval

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

|  |  |  |  |
| --- | --- | --- | --- |
| **Signature** | **Printed Name** | **Title** | **Date** |
|  | <Your Name> | Lead Software Eng. |  |
|  | A. David Doe |  |  |
|  |  |  |  |

**Table of Contents**

Revision History ii

Document Approval ii

1. Introduction 1

1.1 Purpose 1

1.2 Scope 1

1.3 Definitions, Acronyms, and Abbreviations 1

1.4 References 1

1.5 Overview 1

2. General Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Characteristics 2

2.4 General Constraints 2

2.5 Assumptions and Dependencies 2

3. Specific Requirements 2

3.1 External Interface Requirements 3

3.1.1 User Interfaces 3

3.1.2 Hardware Interfaces 3

3.1.3 Software Interfaces 3

3.1.4 Communications Interfaces 3

3.2 Functional Requirements 3

3.2.1 <Functional Requirement or Feature #1> 3

3.2.2 <Functional Requirement or Feature #2> 3

3.3 Use Cases 3

3.3.1 Use Case #1 4

3.3.2 Use Case #2 5

3.4 Classes / Objects 5

3.4.1 <Class / Object #1> 6

3.4.2 <Class / Object #2> 6

3.5 Non-Functional Requirements 6

3.5.1 Performance 6

3.5.2 Reliability 6

3.5.3 Availability 6

3.5.4 Security 6

3.5.5 Maintainability 6

3.5.6 Portability 6

3.6 Inverse Requirements 6

3.7 Design Constraints 7

3.9 Other Requirements 7

4. Analysis Models 7

4.1 Sequence Diagrams 7

5. UI Design 8

6. Change Management Process 9

7. Appendices 9

A.0 Glossary 9

A.1 Appendix 1 //If any others 9

A.2 Appendix 2 9

# 1. Introduction

The introduction to the Software Requirement Specification (SRS) document should provide an overview of the complete SRS document. While writing this document please remember that this document should contain all of the information needed by a software engineer to adequately design and implement the software product described by the requirements listed in this document. (Note: the following subsection annotates are largely taken from the IEEE Guide to SRS).

## 1.1 Purpose

*What is the purpose of this SRS and the (intended) audience for which it is written.*

## 1.2 Scope

*This subsection should:*

*(1) Identify the software product(s) to be produced by name; for example, Host DBMS, Report Generator, etc*

*(2) Explain what the software product(s) will, and, if necessary, will not do*

*(3) Describe the application of the software being specified. As a portion of this, it should:*

*(a) Describe all relevant benefits, objectives, and goals as precisely as possible. For example, to say that one goal is to provide effective reporting capabilities is not as good as saying parameter-driven, user-definable reports with a 2 h turnaround and on-line entry of user parameters.*

*(b) Be consistent with similar statements in higher-level specifications (for example, the System Requirement Specification) , if they exist.What is the scope of this software product.*

## 1.3 Definitions, Acronyms, and Abbreviations

*This subsection should provide the definitions of all terms, acronyms, and abbreviations required to properly interpret the SRS. This information may be provided by reference to one or more appendixes in the SRS or by reference to other documents.*

## 1.4 References

*This subsection should:*

*(1) Provide a complete list of all documents referenced elsewhere in the SRS, or in a separate, specified document.*

*(2) Identify each document by title, report number - if applicable - date, and publishing organization.*

*(3) Specify the sources from which the references can be obtained.*

*This information may be provided by reference to an appendix or to another document.*

## 1.5 Overview

*This subsection should:*

*(1) Describe what the rest of the SRS contains*

*(2) Explain how the SRS is organized.*

# 2. General Description

*This section of the SRS should describe the general factors that affect 'the product and its requirements. It should be made clear that this section does not state specific requirements; it only makes those requirements easier to understand.*

## 2.1 Product Perspective

*This subsection of the SRS puts the product into perspective with other related products or*

*projects. (See the IEEE Guide to SRS for more details).*

## 2.2 Product Functions

This subsection of the SRS should provide a summary of the functions that the software will perform.

## 2.3 User Characteristics

This subsection of the SRS should describe those general characteristics of the eventual users of the product that will affect the specific requirements. (See the IEEE Guide to SRS for more details).

## 2.4 General Constraints

*This subsection of the SRS should provide a general description of any other items that will*

*limit the developer’s options for designing the system. (See the IEEE Guide to SRS for a partial list of possible general constraints).*

## 2.5 Assumptions and Dependencies

This subsection of the SRS should list each of the factors that affect the requirements stated in the SRS. These factors are not design constraints on the software but are, rather, any changes to them that can affect the requirements in the SRS. For example, an assumption might be that a specific operating system will be available on the hardware designated for the software product. If, in fact, the operating system is not available, the SRS would then have to change accordingly.

# 3. Specific Requirements

This will be the largest and most important section of the SRS. The customer requirements will be embodied within Section 2, but this section will give the D-requirements that are used to guide the project’s software design, implementation, and testing.

Each requirement in this section should be:

* Correct
* Traceable (both forward and backward to prior/future artifacts)
* Unambiguous
* Verifiable (i.e., testable)
* Prioritized (with respect to importance and/or stability)
* Complete
* Consistent
* Uniquely identifiable (usually via numbering like 3.4.5.6)

Attention should be paid to the carefuly organize the requirements presented in this section so that they may easily accessed and understood. Furthermore, this SRS is not the software design document, therefore one should avoid the tendency to over-constrain (and therefore design) the software project within this SRS.

## 3.1 External Interface Requirements

### 3.1.1 User Interfaces

### 3.1.2 Hardware Interfaces

### 3.1.3 Software Interfaces

### 3.1.4 Communications Interfaces

## 3.2 Functional Requirements

|  |  |
| --- | --- |
| *Actor* | *Goal* |
| User | Control Specific IoTs Settings |
|  | Login to access personal set of IoTs |
|  | Sign up into application |
|  | Register new IoTs to account |
|  | Deregister inactive IoTs |
|  | Register other accounts to access the same set of IoTs |
|  | Check status of a device |
|  | Set restrictions / rules on specific devices |
|  | Room specific control of IoTs |
|  |  |
| System Application | Alarm after sensor detection |
|  | Notify user of device changes |
|  | Contact Emergency services in high alert situations |
|  | Location awareness of particular devices |
|  | Real-time response to user request |
|  | Compatibility with new IoTs |
|  | Connect user authentication with database |
|  | Realtime update of database |
|  | Realtime virus scanning |
|  | Prevent unauthorized access |
|  | Update status of a device |

**User**

|  |  |
| --- | --- |
| **3.2.1 Control Specific IoTs Settings** | |
| **Purpose** | Allows user the ability to manipulate the status of devices connected to the system |
| **Inputs** | Dependent on type of device |
| **Processing** | Record user input on settings and send out instructions to IoT |
| **Output** | Signal inputted change to selected device and apply |
| **Error Handling** | Notify user of manipulation error and default device to previous setting. |

|  |  |
| --- | --- |
| **3.2.2 Login to access personal set of IoTs** | |
| **Purpose** | Grant access to user to manipulate and monitor their personal set of registered IoTs |
| **Inputs** | User login and password |
| **Processing** | Verify login and password entry |
| **Output** | Grant access to user and display set of IoTs via Home Screen of application |
| **Error Handling** | Notify user of invalid credentials and default back to login screen |

|  |  |
| --- | --- |
| **3.2.3 Sign up into application** | |
| **Purpose** | Grant access to user to create an account |
| **Inputs** | Register user login, password, and email |
| **Processing** | Submit information into user database |
| **Output** | Grant user access to login and register their personal set of IoTs |
| **Error Handling** | Notify user of invalid credentials or a disconnect to database and default back to the sign-up screen |

|  |  |
| --- | --- |
| **3.2.4 Register new IoTs to account** | |
| **Purpose** | Grants user the ability to bind personal IoTs to their account and manipulate them accordingly |
| **Inputs** | Dependent on type of device and manufacturer |
| **Processing** | Update user database and add IoT information to the account for manipulation |
| **Output** | Display IoT setting for user manipulation |
| **Error Handling** | Notify user of failure to add IoT device and return to home screen |

|  |  |
| --- | --- |
| **3.2.5 Deregister inactive IoTs** | |
| **Purpose** | Grants user the ability to deregister personal IoTs from their account |
| **Inputs** | Selection of the IoT and Admin user password or form of confirmation of the deregistering |
| **Processing** | Update user database and remove IoT information to the account |
| **Output** | Return to home screen of application |
| **Error Handling** | Notify user of failure to remove IoT device and return to home screen |

|  |  |
| --- | --- |
| **3.2.6 Register other accounts to access the same set of IoTs** | |
| **Purpose** | Allow the user to grant access to other accounts who can also manipulate certain IoTs |
| **Inputs** | Child account inputs verification code of the parent account |
| **Processing** | Update user database and add IoT information to the account for manipulation |
| **Output** | Return to home screen with updated IoTs information |
| **Error Handling** | Notify parent user of failed attempt to register new account |

|  |  |
| --- | --- |
| **3.2.7 Check status of a device** | |
| **Purpose** | Allows user the ability to check if a device is working as intended or is having issues and give description on what the error issue maybe |
| **Inputs** | Selection of IoT from home screen |
| **Processing** | Retrieve data on selected IoT from database |
| **Output** | Display information regarding the selected IoT’s current state |
| **Error Handling** | Notify user of failure to retrieve IoT data and default back to home screen |

|  |  |
| --- | --- |
| **3.2.8 Set restrictions / rules on specific devices** | |
| **Purpose** | Give user the ability to set specific timer, restrictions, max, and min for settings on a specific IoT |
| **Inputs** | Dependent on type of device and manufacturer |
| **Processing** | Real-time changes to the IoT dependent on the settings |
| **Output** | Haptic/Visual feedback of sliders and buttons of new settings and real-life change of IoT |
| **Error Handling** | Notify user of inability to change settings and default back to home screen |

|  |  |
| --- | --- |
| **3.2.9 Room specific controls of IoTs** | |
| **Purpose** | Provides user the ability to organize and control IoTs into specific portions/rooms of a SmartHome |
| **Inputs** | Input room of home where the IoT is located within the house |
| **Processing** | Place location in home in the database with the specific IoT |
| **Output** | Return to home screen with IoT listed under room tab |
| **Error Handling** | Notify user of the invalid room selection for IoTs and default back to default room selection |

**System Application**

|  |  |
| --- | --- |
| **3.2.10 Update status of a device** | |
| **Purpose** | System runs diagnostic on IoT device to determine if the device is running properly or its issues and record information |
| **Inputs** | N/A |
| **Processing** | Compare status of IoT to last properly working status of IoT |
| **Output** | Notification to the user of any major changes in the status of an IoT |
| **Error Handling** | Notify user of the device errors |

|  |  |
| --- | --- |
| **3.2.11 Alarm after sensor detection** | |
| **Purpose** | Provide user with vital information regarding if any sensor based IoT discovers moment, smoke, or any other trigger for the IoT |
| **Inputs** | Software trigger message from the IoT |
| **Processing** | Decoding the message sent from the IoT |
| **Output** | Notification to the user that there is a detection trigger with a description of what may cause it and its location |
| **Error Handling** | Notify user of the sensor detection feature errors |

|  |  |
| --- | --- |
| **3.2.12 Notify user of device changes** | |
| **Purpose** | Provide user with vital information regarding if any the IoTs receive a massive change in settings or unexpected change in settings |
| **Inputs** | Dependent on device |
| **Processing** | Process and record drastic changes in settings or properties of the IoT |
| **Output** | Notification to the user that there was a change in the device’s setting with a description of what changes were made and by what account |
| **Error Handling** | Notify user of the device changes detection errors |

|  |  |
| --- | --- |
| **3.2.13 Contact Emergency services in high alert situations** | |
| **Purpose** | To provide an extra level of security and safety for users if any IoTs of health or quality of living setting are outside of normal settings |
| **Inputs** | Settings and status of IoTs directly related to emergency services ex smoke detector |
| **Processing** | Process and record status of the IoT and location of the user’s address where the IoT is located |
| **Output** | Contact local emergency services with information on the IoT’s status and user’s address |
| **Error Handling** | - |

|  |  |
| --- | --- |
| **3.2.14 Location awareness of particular devices** | |
| **Purpose** | Understand where the IoT is located and allow IoTs access to locational information if necessary |
| **Inputs** | Input room of home where the IoT is located within the house |
| **Processing** | Assign location to IoT to provide information on the where in the SmartHome a selected IoT is located |
| **Output** | The location of where the selected IoT |
| **Error Handling** | Notify user of no valid location given or found |

|  |  |
| --- | --- |
| **3.2.15 Real-time response to user request** | |
| **Purpose** | Provide real time changes to IoTs according to setting set by users, ex changing of a light’s hue as the user slides the hue control |
| **Inputs** | User setting changes |
| **Processing** | Transfer control changes information to IoT |
| **Output** | Change is applied to IoT |
| **Error Handling** | Notify user of failure to send response/changes to IoT |

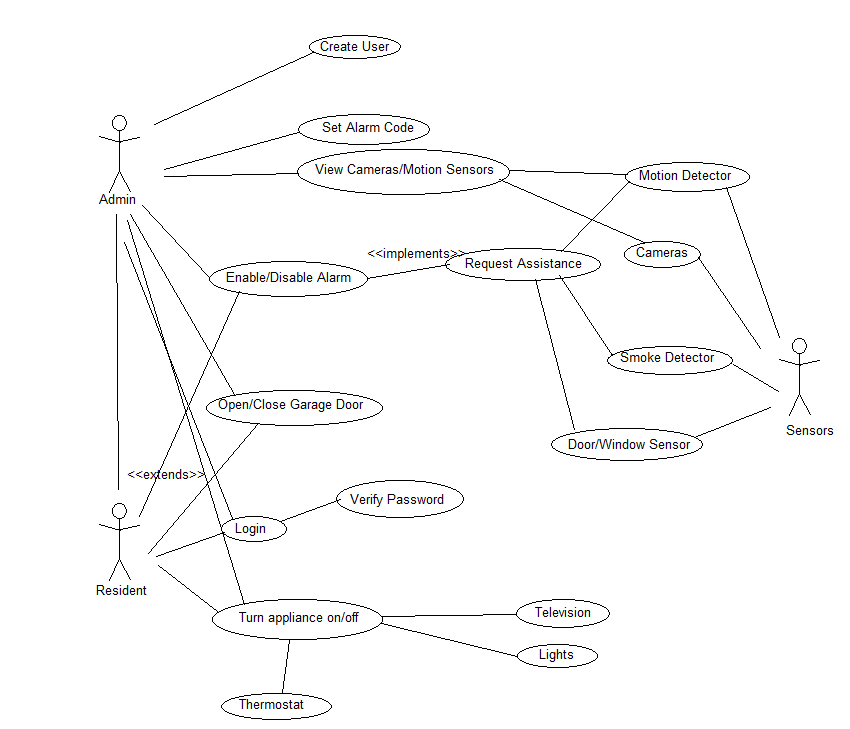
|  |  |
| --- | --- |
| **3.2.16 Compatibility with new IoTs** | |
| **Purpose** | Provides versatility to the software to connect to majority if not all of the different IoT devices to allow users a singular place to access all their systems to their SmartHome |
| **Inputs** | IoT manufacturer, IoT model number, IoT type |
| **Processing** | Compare and register IoT model to compatible models in database |
| **Output** | Notify user of a successful connection between the IoT and software |
| **Error Handling** | Notify user that the IoT is not compatible with the software |

|  |  |
| --- | --- |
| **3.2.17 Connect user authentication with database** | |
| **Purpose** | Bind parents accounts to a specific set of personal IoTs and their children accounts to the same set of IoTs |
| **Inputs** | User login, user password, user email, children accounts information |
| **Processing** | Compare user information to user information stored in database |
| **Output** | Grant or deny access to IoTs given a successful comparison |
| **Error Handling** | Default back to home screen and notify user and software developers of error connecting to user database |

|  |  |
| --- | --- |
| **3.2.18 Realtime virus scanning** | |
| **Purpose** | Provide security to user and IoTs from being able to change or disable device’s settings, user access, and ultimately corrupt IoTs firmware. |
| **Inputs** | Current state of IoT |
| **Processing** | Comparison of IoT files and settings to factory files and settings |
| **Output** | Result of the virus scan and any problematic IoTs |
| **Error Handling** | Notify user of any viruses or problematic IoTs |

|  |  |
| --- | --- |
| **3.2.19 Prevent unauthorized access** | |
| **Purpose** | Provide security to user accounts to prevent access to unauthorized accounts |
| **Inputs** | User login, user password, user email, children accounts information |
| **Processing** | Compare inputted user information to the login information and database registry the user is trying to access / record information on the attempt |
| **Output** | Notify user of possible attempt of unauthorized or unusual activity with account |
| **Error Handling** | Send recorded information to software developers to look into the activity. |

## 3.3 Use Cases



### 3.3.1 Use Case #1

Use case descriptions here

|  |  |
| --- | --- |
| Use Case Name:Login | ID: UC-001 |
| Version: 0.0.1 | Author: Brock Maxwell |
| Primary Actor: Resident | |
| Brief Description:  Resident signs in to account | |
| Trigger: User opens the application | |
| Normal Flow of Events:   1. User opens application 2. User inputs username/password 3. Application verifies username/password 4. User is redirected to the main menu screen | |
| Alternate/Exceptional Flows:   1. User inputs incorrect username/password 2. Application returns to sign in menu and asks user to try again | |
| Pre-Conditions:  User opens application | |
| Post-Conditions:  User is logged into their account and can begin using the application | |

### 3.3.2 Use Case #2

|  |  |
| --- | --- |
| Use Case Name: Create User | ID:UC-002 |
| Version: 0.0.1 | Author: Brock Maxwell |
| Primary Actor: Admin | |
| Brief Description:  Admin creates new resident account | |
| Trigger: Admin enters the create menu and selects ‘new user’ | |
| Normal Flow of Events:   1. User opens the create menu and selects new user 2. User sets username and password for new account 3. User sets permissions for created account 4. New Resident account added to the database | |
| Alternate/Exceptional Flows:   1. Username is already taken 2. Ask the admin to select a new one | |
| Pre-Conditions:  Admin is logged in to the system | |
| Post-Conditions:  A new Resident account has been created | |

### 3.3.2 Use Case #3

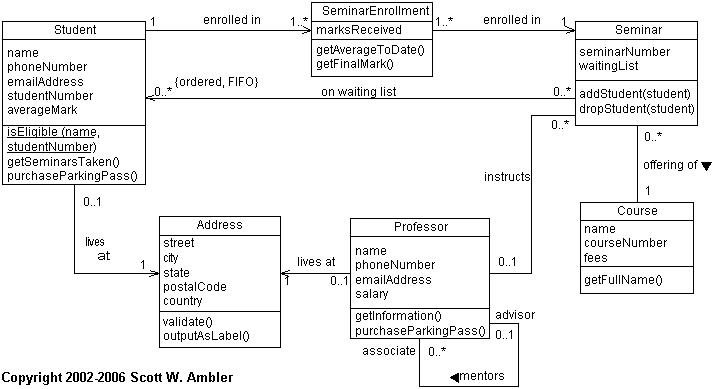
|  |  |
| --- | --- |
| Use Case Name: Turn Appliance On/Off | ID:UC-003 |
| Version: 0.0.1 | Author: Brock Maxwell |
| Primary Actor: Resident | |
| Brief Description:  User …… | |
| Trigger:User starts to use the the system | |
| Normal Flow of Events:   1. User opens management menu 2. User selects appliance to modify 3. User sets the selected appliance to the on/off state | |
| Alternate/Exceptional Flows:   1. Application is unable to connect to the selected appliance 2. Troubleshoot the problem and ask the user to try again | |
| Pre-Conditions:  Appliance is connected to the system and user is logged in | |
| Post-Conditions:  An appliance is turned on or off | |

### 3.3.2 Use Case #4

|  |  |
| --- | --- |
| Use Case Name: Request Assistance | ID:UC-004 |
| Version: 0.0.1 | Author: Brock Maxwell |
| Primary Actor: Resident | |
| Brief Description:  Police/Fire Department alerted to issue | |
| Trigger: Motion Detector, Smoke Detector, Door/Window Sensor, User | |
| Normal Flow of Events:   1. User notices problem and requests assistance 2. Corresponding department is notified | |
| Alternate/Exceptional Flows:   1. Sensor detects problem and signals a request for assistance | |
| Pre-Conditions:  User needs help or sensor detects that help is required | |
| Post-Conditions:  Local authorities are notified that the user’s home needs assistance | |

## 3.4 Classes / Objects

Class diagrams and Object diagrams here.



### 3.4.1 <Class / Object #1>

3.4.1.1 Attributes

3.4.1.2 Functions

<Reference to functional requirements and/or use cases>

### 3.4.2 <Class / Object #2>

…

## 3.5 Non-Functional Requirements

Non-functional requirements may exist for the following attributes. Often these requirements must be achieved at a system-wide level rather than at a unit level. State the requirements in the following sections in measurable terms (e.g., 95% of transaction shall be processed in less than a second, system downtime may not exceed 1 minute per day, > 30 day MTBF value, etc).

### 3.5.1 Performance

### 3.5.2 Reliability

### 3.5.3 Availability

### 3.5.4 Security

### 3.5.5 Maintainability

### 3.5.6 Portability

## 3.6 Inverse Requirements

State any \*useful\* inverse requirements.

## 3.7 Design Constraints

Specify design constrains imposed by other standards, company policies, hardware limitation, etc. that will impact this software project.

## 3.9 Other Requirements

Catchall section for any additional requirements.

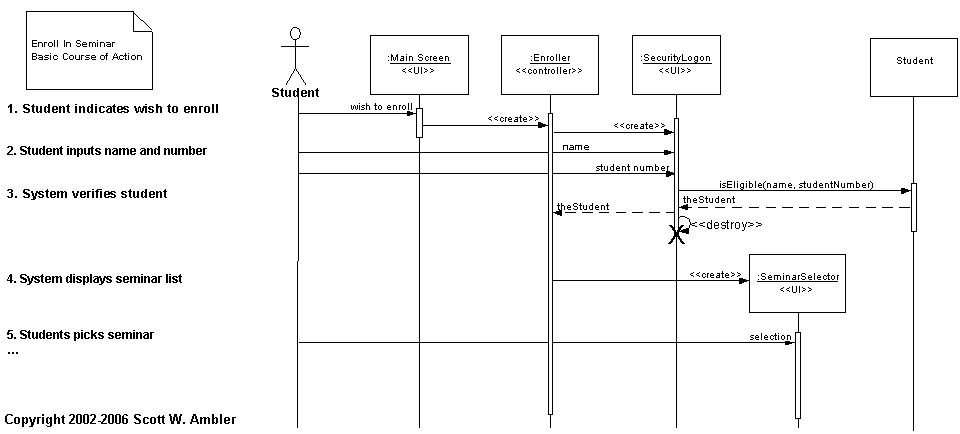
# 4. Analysis Models

List all analysis models used in developing specific requirements previously given in this SRS. Each model should include an introduction and a narrative description. Furthermore, each model should be traceable the SRS’s requirements.

## 4.1 Sequence Diagrams

Draw a sequence diagram for each use case discovered so far.

Example:



# 

# 5. UI Design

#### *For this section, demo the pseudo user interface and navigational paths of each use case*.

#### Eg. Use case -- User Login Mock Screen

**Online Recycler System**

(banner goes here)

(Welcome message and introduction to the system go here)

LogIn

UserID:

Password

User Type

Recycler

Poster

Log In

# 6. Change Management Process

Identify and describe the process that will be used to update the SRS, as needed, when project scope or requirements change. Who can submit changes and by what means, and how will these changes be approved.

# 7. Appendices

Appendices may be used to provide additional (and hopefully helpful) information. If present, the SRS should explicitly state whether the information contained within an appendix is to be considered as a part of the SRS’s overall set of requirements.

*Example Appendices could include (initial) conceptual documents for the software project, marketing materials, minutes of meetings with the customer(s), etc.*

## A.0 Glossary

***lists all the terminology used in this documentation, such as http, admin, student, https, database etc***

## A.1 Appendix 1 //If any others

## A.2 Appendix 2