Bilkent University

Department of Computer Engineering

Senior Design Project

Project short-name: Pandora

High Level Design Report

Taner Baygün – 21300987

Mert Armağan Sarı – 21401861

Berfu Anıl – 21401502

Serhan Gürsoy – 21400840

Ege Yosunkaya – 21402025

Supervisor: Halil Altay Güvenir

Jury Members:

Özgür Ulusoy

Muhammet Mustafa Özdal

Innovation Expert: Cem Çimenbiçer

Website: https://thepandora.github.io/boxproject/

Dec 22, 2017

This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

Table Of Contents

1. Introduction

In this report, we will explain the high-level design of Pandora. We will explain the purpose of our system and design goals of it. Afterwards, we will propose our software and hardware architecture while making use of system decomposition and hardware-software mapping of our system. Lastly, we proceed to investigate the access control and security of the system along with global software control and the boundary conditions of it.

1.1 Purpose of the System

Technology and its impacts on the society are always an interesting topic for many researchers. It is found that with new technological improvements people tend to spend more time with their machines such as smartphones, computers and etc. Since people can't exclude the technology from their lives, they build deeper connections with them [1]. Especially when people want to play games with their friends, machines hook them on screens to give a visual pleasure. Moreover, in a social manner, people stop talking with each other when they play games. The reason is that they need to follow the game's state every minute from their computers or smart phones. So, this creates an anti social environment for many of those who wants to play simple games with their friends. Pandora solves that problem by merging technology with social interactions.

Pandora combines technology with games that consist mimics and gestures. The games in our system are similar to poker, werewolf, liar's dice and etc. which requires analyzing other player's reactions, strategies and body language. Although these games are enjoyable, they have problems when they are implemented for technological platforms. If players want to play poker around a desk with other players, then they will need several items to progress in the game. Some of the problems related to the items can be listed as a physical card deck, cards get damaged by environmental causes, the desk does not shuffle the cards well and the score of the players needs to be written down on a paper. If players want to play poker online, game will prevent players from cheating, bluffing and it is not possible to observe other players' reactions. To enjoy the poker it should be easy to play and should have its nature (cheat, gesture and mimics). Pandora solves this problem. Pandora makes a connection between technology and daily life. While Pandora hosts game materials, players interact by being in the same room. In poker example, Pandora shuffle desk, deliver cards,have scoreboard while only thing a player does is making a move by analyzing other players. By doing that Pandora both provide comfort of the

technology and social interactions by gathering players in the same room and allow them to communicate actively.

1.2 Design Goals

1.2.1 Usability

Since the Pandora device aims to serve and host game rooms among players, the usability is crucial during the progress of the games. The system will notify every player about the current state of the game that is being played. The states should be easy to understand and presented user friendly. The system will serve the owner of the device as an admin of the game. So, the usability is also important for the owner which is another user type for the system. The owner of the device may want to change settings and define new preferences. For this purpose, the interface should be implemented to support various functionalities with easy instructions. There will be a tutorial for the owner of the device to teach how the settings can be changed to complete the tasks.

1.2.2 Accessibility

The Pandora will be developed for technological devices that have a browser and have wifi connector. Pandora will be sold for 35$ (approximately 135 TL). No extra payment is required for the initial game package which consists of Werewolf game. Pandora can support up to 10 users, one of the users should be the owner and the rest are the other players. In order to play a game, they can use smartphones or any other devices that have a browser with a wifi connecter module.

1.2.3 Availability

Pandora has user friendly interface and having its manual distributed with box. Pandora requires admin login to control the system, without admin system will not be able to serve other users. Pandora provides service for users who want to play with pandora both indoor and outdoor. It is recommended not to use outdoor in rainy weather. Pandora has a rechargeable battery that support system for 3 to 4 hours. Pandora broadcasts its own wifi signals for users to connect and not required to connect a network to boot or maintain the system.

1.2.4 Reliability

Reliability of the Pandora is related with both hardware and software. For software part, it is important to run segments and subsystems. Any problem regarding to the operating system may cause Pandora work improperly, therefore, Pandora will prevent users from accessing its operation related areas. For the hardware part, when the battery of Pandora about to run out, Pandora saves current data to the local database and shut itself down so that the loss of any data will be prevented and Pandora will not start until it is charged again.

1.2.5 Performance

To keep the user experience level high, Pandora will work fast after the beginning of the game. Additionally, the box should be fast to keep the connection strong, so that the players are not being disturbed during the gameplay. Player screens will be repeatedly updated by the box therefore it should be fast. The system will be scalable up to 20 users, but for performance issues it will allow 10 users for each session.

1.2.6 Security

For the security measurements, our system will be designed with various encryption techniques. The encryption will be needed for the protection of the static and the dynamic data inside the Pandora box. There will be a settings menu for the owner to change the WiFi password of the box, so that the protection of Pandora can be maintained continuously. The admin user of the games will again be the owner and it is maintained by the password which is defined by the owner. So the admin access to Pandora is also protected.

1.3 Definitions, Acronyms, and Abbreviations

DB: Database

GUI: Graphical User Interface

SQL: Structured Query Language

UI: User Interface

TL: Turkish Lira

1.4 Overview

Email, online chat rooms and social networking sites enable people to keep in touch, but psychologists are concerned that the lack of real human contact may have an adverse effect on people's emotional and social well-being. Aric Sigman, an American psychologist and biologist, says that social networking sites foster social isolation and could put people at risk of developing serious health problems [2]. Imagine a scenario where you want to spend time with your friends and also at the same time you want to have fun with them. A way that includes these two features is rare. You indeed can go and buy a board game. However, board games are only focused on one single game. Hence, if you and your friends get bored of that board game after a certain time, you have to buy a new board game. Furthermore, besides the negative effect of spending extra money, both you and your friends have to spend time to adapt that new game. Based on the complexity and confusing structure of printed materials, this can take hours before you grasp all the complicated rules of games. This approach is time consuming also quite costly even though it has limited variety of games. Pandora Box is designed to solve that problems. By bringing real human contact and technology together pandora will be one of a kind.

With Pandora, you can play games with your friends without wasting your time and money on impractical board games. Thanks to Pandora's user friendly graphical interface, you do not need any piece of paper anymore. Pandora takes advantage of the technological era. To use Pandora, all you require is your phone and Pandora Box. For both owner and player, usage of Pandora is simple. Which is one of the core missions of Pandora, giving people tool to have fun in the easiest way. If you are the owner of the Pandora just turn on your Pandora Box. Setup your new game for your friends and tell them to use their phones and login to game. If you are only the player, simply connect to the Pandora from WiFi, pick your game room and start to have fun with your friends. If you get bored from that game, just start a new game room with totally different game. For beginners to a game, tutorials are available. No more complicated printed manuals. With Pandora, you can even save your game state for continuing later.

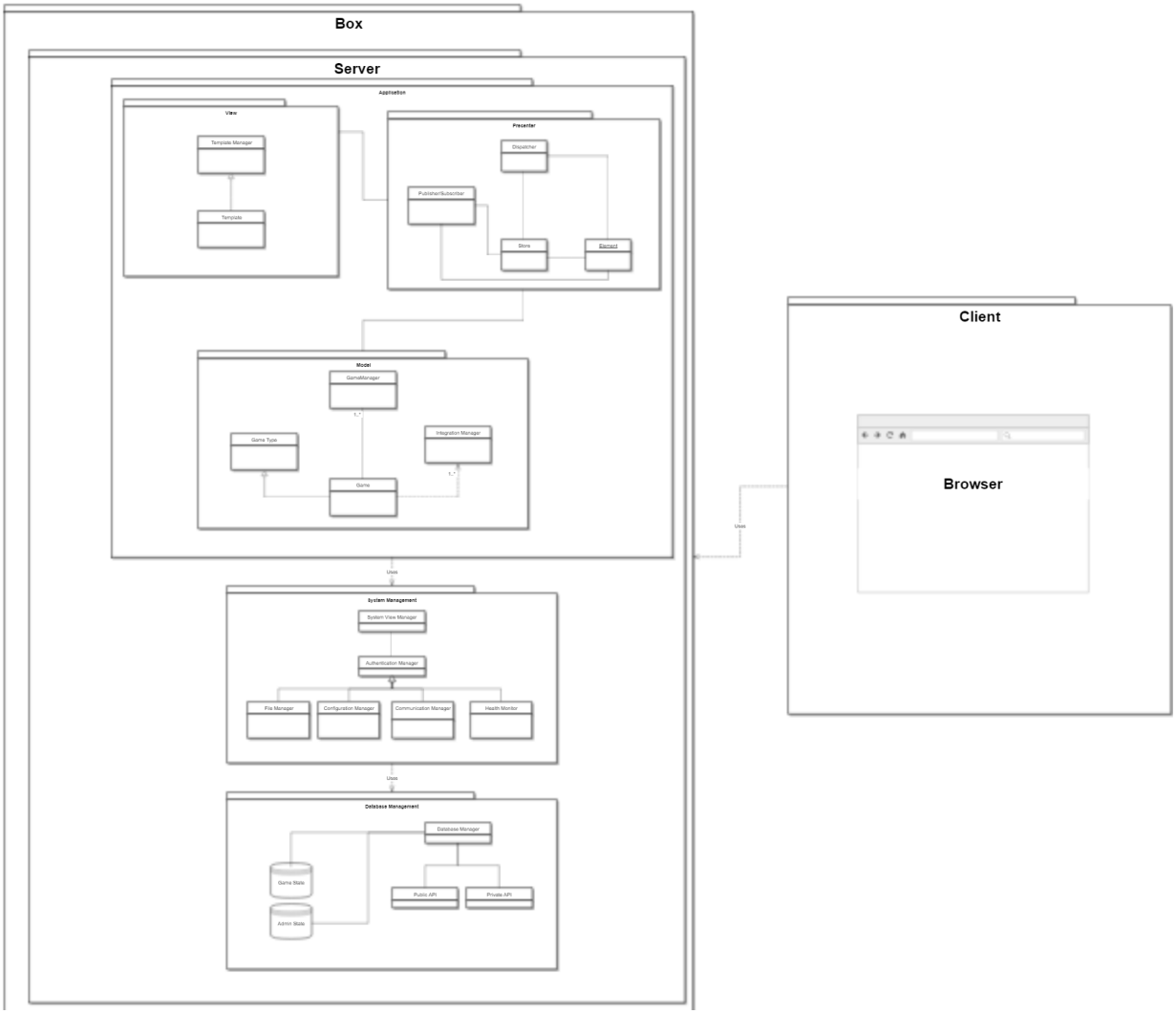
We are also planning to create an Online Web Store for owners of Pandora to browse for new games, bundles and even more creative things that created by us or volunteer developers that developed new games with Pandora's Framework. In this way, there is no need to throw your Pandora Box after years. You can simply download new games and continue having fun with your friends. However, don't forget, Pandora is not actually a game console, it is a tool. Creating fun moments with it totally depends on you and your friends.

2. Proposed Software Architectures

2.1 Overview

In the subsystem decomposition, the subsystem architecture is described in detail. First of all the subsystems, their architectures and responsibilities and their interaction with other subsystems are described. Afterwards, hardware/software mapping of the system is presented to show which hardware parts will be used for what reason. Persistent Data Layer described in detail with design choices and reasonings. In access control and security part the issues with security and how the system design will address them is presented. In global software control, the general flow of the system and how is the flow controlled is explained. At last, in boundary conditions we explicate how are boundary conditions are handled by the system.

2.2 Subsystem Decomposition



2.2.1 Application Subsystem

For the system architecture, we decided to use server/client architecture to model interaction between the browser and the box. The communication will be based on requests from the client (browser) and the responses from the server(the box). In the subsystem we planned to use MVP(Model View Presenter) architecture. Users interact with the game via the presenter layer. Presenter layer manipulates the model layer with respect to the incoming request. Furthermore, presenter responds to the request with a view. In our case, models create views and those views are served with the presenter.

In the architectural design one of our concerns was providing a friendly environment for the creators for designing and creating their games within our system. To achieve this goal, we design the view layer according to our needs. We planned to use flux architecture to construct a model layer. We will create an interface called “Element” for developers to create their own models. A game will be consist of “Element”s that works together. The game logic will be coded inside those elements. The system’s view layer will use the developer created models and create valid views by using them.

Flux Architecture is the backbone for the architectural design of our games. There will be elements which use our Element interface working together to construct a game. There will be documentation about how to use or create Elements for developers. In flux architecture, the state is an immutable object which is updated with actions coming from the action creators (Action creators can be the user or an external system). When the state is updated all the Elements are notified of the change and create new views with the new state. This architecture is suitable for the structures where an event changes various number of views similar to our case.

2.2.2 System Management Subsystem

The system management subsystem has a 2-layer architecture and consists of Hardware Management Layer and System Layer. System layer uses hardware layer to achieve an abstraction level. Moreover, the system layer is the layer with the logic for other subsystems to work. The layer is responsible for all the subsystems to work and heart of the general structure.

Since we use our box as an offline server (or private server) , there will be low level interactions between the software and the hardware of the system. We planned to have a hardware layer to handle the I/O operations , wifi connections or the other operating system or hardware interactions. Hardware Manager uses captive portal software and hotspot software and regulate the interactions with them.

System Management subsystem has crucial duties and should work robustly. The subsystem also responsible for monitoring health conditions (wifi condition, battery life, etc.) of the box and notify the system accordingly.

2.2.3 Database Management Subsystem

Database Management subsystem will be the subsystem that is organising and making database interactions. And the Database Management subsystem exposes two kinds of APIs for other parts of the system to use. One of them is the private API, it will be for the internal use of the system. And the public API for developers to create their games easily with a convenient abstraction.

2.2.4 Future Plans

For further, we planned to have other features such as game market, USB connection with computers, installing or deleting games, etc. Thus, we are trying to plan our architecture capable of implementing new features with ease. The system manager will be the connecting point for all the features. They communicate with each other with the system manager. Moreover, system manager is responsible for creating system views such as login screens, settings screens, etc. (except the game view).

2.3 Hardware/Software Mapping

Our client application in this project is the browser on the user's phone. And the server consists of several subsystems. Pandora system will make use of the browser to render the views into the user's phone and for users to interact with the system. The communication between the Pandora box and the browser will use the HTTP. The Pandora box will serve users HTML documents for browser to render.

The Application component will handle the incoming HTTP requests and serve the generated views by a HTML document. The Application subsystem will use Hardware Management Subsystem and Data Management Subsystem to use hardware and operating system resources.

For further, we are planning to develop a Communication subsystem for handling USB connection with the computer and install/delete games , accessing the software from a remote server etc.

2.4 Persistent Data Management

In our system, the need for data storage is not web-scale, we should only the keep track of system related data (configurations etc.) and saved game statuses. Our first objective is to protect and

save files. We will address it by making a backup plan for data loss such as different storages etc. One for the daily access and the other one for the retrieval.

Moreover, one of our visions for the system is letting other creators to create their games for our platform. Thus the creators would need to access databases and perform saves and loads. To protect the system data and prevent creators to do harm to the system we will implement a persistence layer and expose two kinds of API's:

1. **Public API:**

The public API will be well-documented and available for public use in the website. The public API will be accessible for user created games. The API is planned to have all the necessary functionalities to create a game of any kind.

2. **Private API:**

The private API will be for the system classes and requires an authentication method to use. Private API will be used for development of the box and by the system to change credentials or configurations.

As database system we decided to use a NoSQL database (document-based) for a couple reasons. First, we have different type of data to store in our database such as system configurations, API keys, system information, game save files, etc. we did not want to translate all of them to tables to be stored. Instead, we can store them in a NoSQL database with relatively easy way. Secondly, our game states can be different for each game , constructing a table for each game which would have limited entries (save files) is not desirable. Instead, we can turn every state into a document and store them without constructing tables for each of them. As result our database system will use a document-based database.

2.5 Access Control and Security

The system has two type of users. The security and control policies are different for both users.

1. **Admin (Owner):**

Admin (Owner) has an admin password given him with the box. The system has the default password

already implemented in when the box, thus it is ready to be connected via the password given. All boxes will be produced by a such pre-generated password for owners to join. Owners after logging in as admin to the system can change the password.

Admins have to have the wireless password to have the connection with the system. If not, they will not be able to use the system from their browsers. The default password for wifi is sent via the box as well and can be changed after login as admin.

2. Player (Guest):

Players only need wifi password to join the wireless network of the box to play the game. Username is created for the games that are not related to the users only to the games. Meaning that, games associate usernames with the logged-in players every time a new game created or existing game load. This relating is done by the user, simply typing a new name at the start of a game or clicking one of the existing names in a loading game.

We had two main issues of security and access control , first one is the outsiders who could join the game rooms with connecting the box's wifi. We addressed that issue with wifi password. Therefore only admin can choose who would know the password. The other issue is the controlling the box settings, creating games etc. There should be only one person in the session can control the box. We addressed that issue with admin privileges and sending the admin password to the owner with the box.

2.6 Global Software Control

For the project between procedure-driven and event-driven approach we decided to choose an event-driven control mechanism according to our project's needs. In the project, our system is designed to give response to the user's interactions with the system since we selected server/client model for our project. For example, in a game after system makes a notification about the game status, it goes into the pending state waiting for an event. After the event occurs, event creator might be a user or the system itself, system responses accordingly and starts to wait again.

2.7 Boundary Conditions

2.7.1 Initialization

The initialization of the system consist of two steps, power on the box for the first time and the tutorial. The software inside the box will be ready to use when production of the box finishes. The setup of the software will not be done by the user. However, to teach the basic interactions with the box a tutorial will be set up for the first time connecting the box.

2.7.2 Power On/Off

The system should be robust and prepared for the unexpected power offs. The state of the system as well as the currently playing game (if any) will be saved periodically and a rollback plan will be prepared. The system layer will be responsible for the sanity checks and monitoring the status of hardware resources (e.g. the battery status).

2.7.3 System Failure

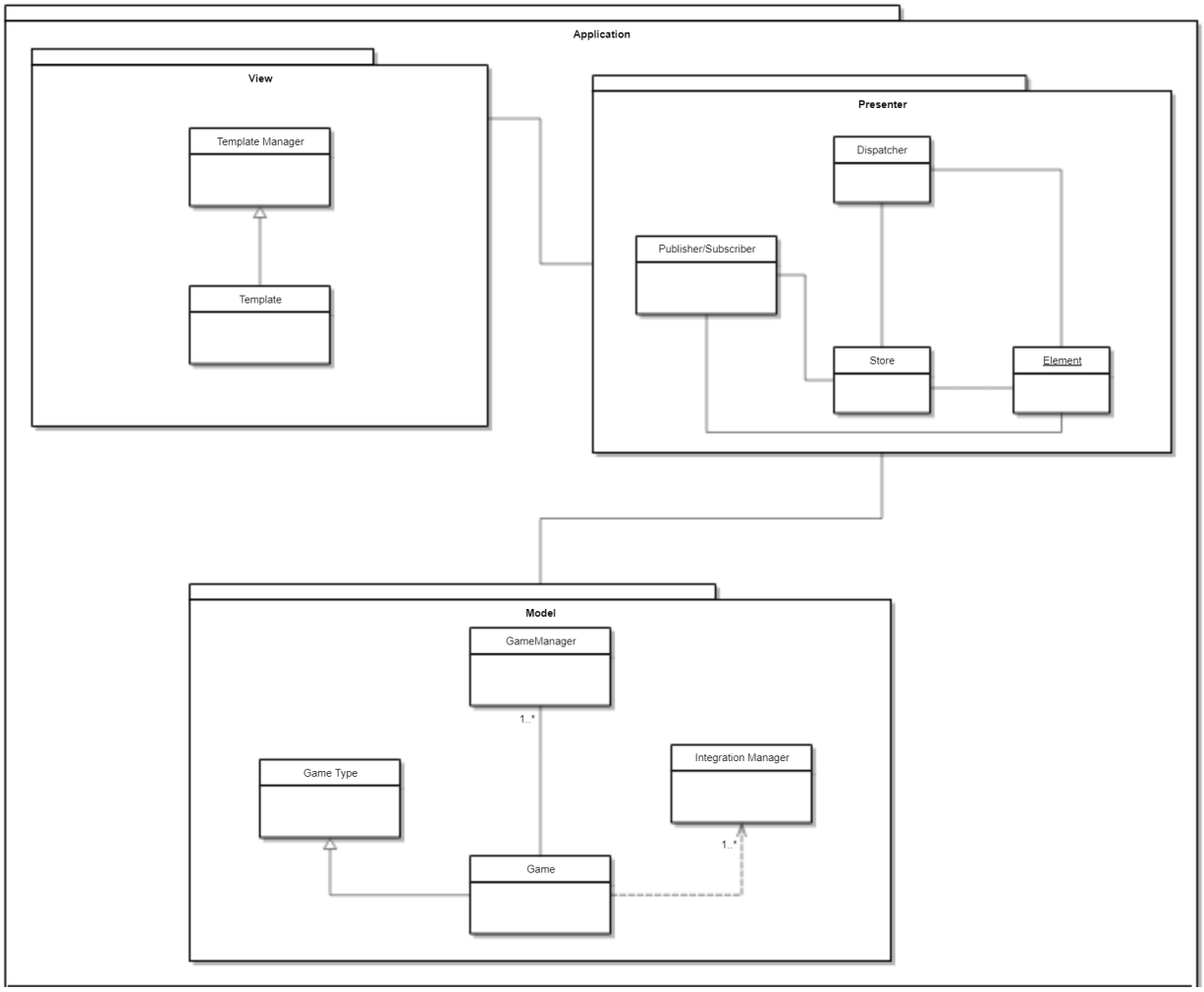
Unfortunately, the system software failures may occur unexpectedly. Because of the fact that our product is a physical product that we cannot access via internet , a computer connection based rescue plan will be implemented. The owner should be able to connect the box to his/her computer via USB and start the rescue sequence from our remote server. We will set up a program in owner's computer and access the box by using that program as a bridge.

2.7.4 Installing Games / Deleting Games

Creating a market and let users develop games for our product is crucial for our project. Thus, in the future we are planning to create a service for installation and deletion of the games. Installing and deleting games should be done by a desktop program where the user connects his/her box to computer , download the bridge software and install/delete games to/from the box with that program's interface.

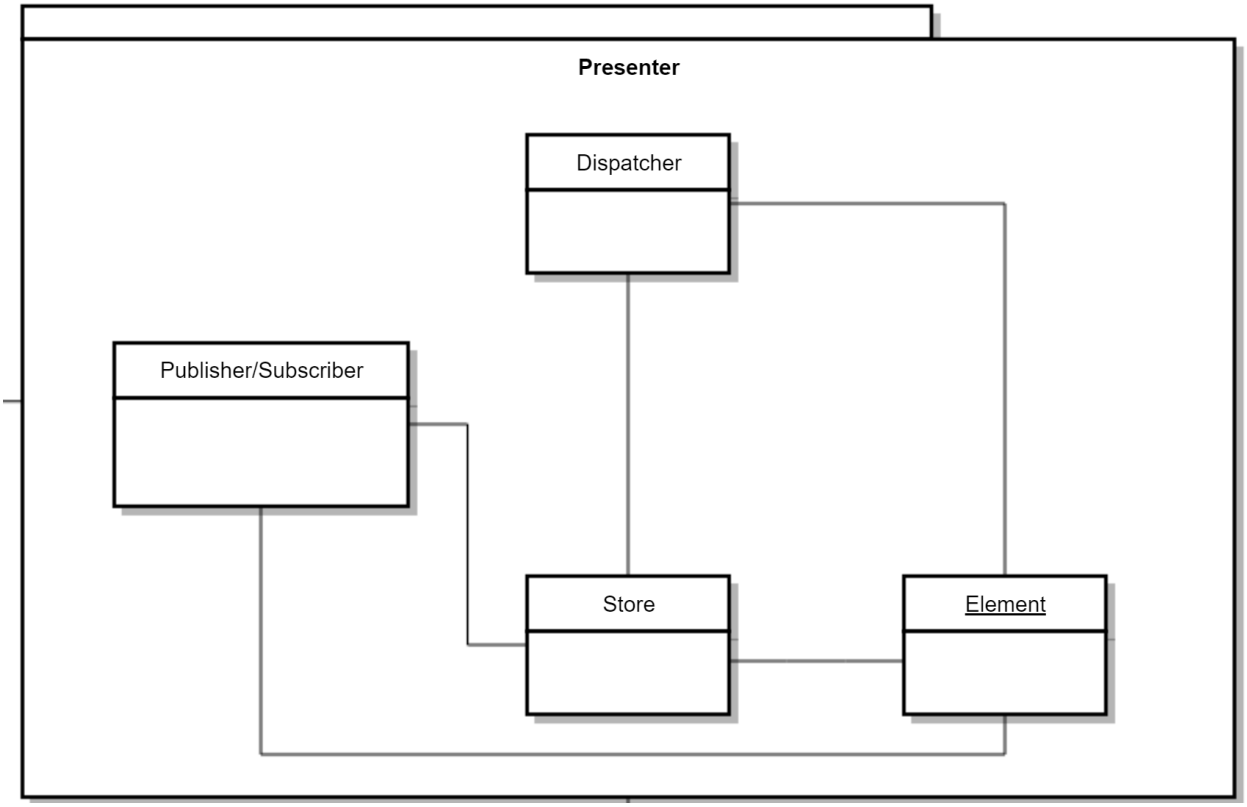
3. Subsystem Services

3.1 Application Subsystem



The application subsystem is responsible for listening HTTP requests coming from the user and responding accordingly. The subsystem uses 3-layer architecture in itself. The layers are presentation layer, logic layer and the data layer.

3.1.1 Presenter Layer



Presenter layer is responsible for creating and modifying the views. The presenter part is rather complicated in our system to let users to create their own games. In our system, all the games are considered as a presenter container. Developers develop their games according to the same structure we provide so that our system can find initialization points , run the game with its logic and handle the game's requests from the box system. We decided to use the flux pattern in this layer. We will provide some utility tools and a base class (called Element) for developers to use and design their games.

Element Class: Element class is the base class for games, the subclasses of the elements creates a game.

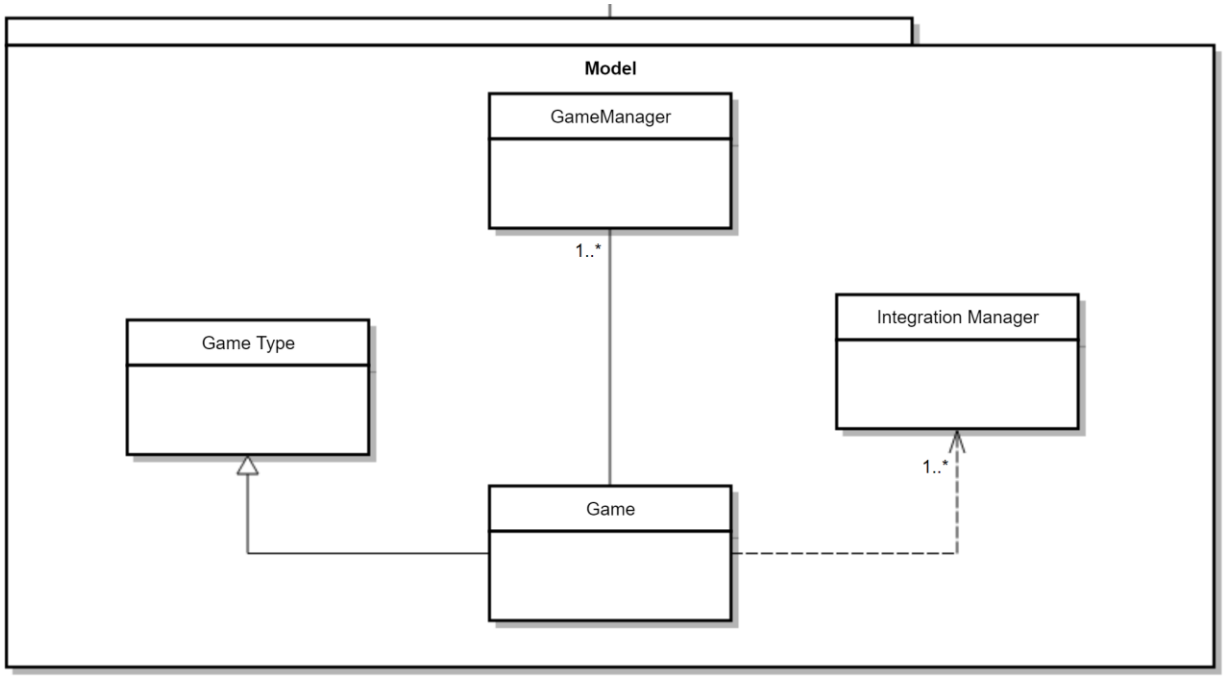
Utility class: Provides some tools for the developers for integrating their games with the system easily.

Dispatcher: In flux design, views create actions that update the store. The dispatcher is responsible for distributing the actions and provides middleware between elements and the store.

Store: The essential of the flux design. Every element uses store to get information about the game state and elements update the store via the actions.

Publisher/Subscriber Interface: Elements update their views according to the changes at the store. They use Publisher/Subscriber pattern to communicate with the store. Elements notified when a change in the Store happens.

3.1.2 Model Layer



Model layer is our application domain.

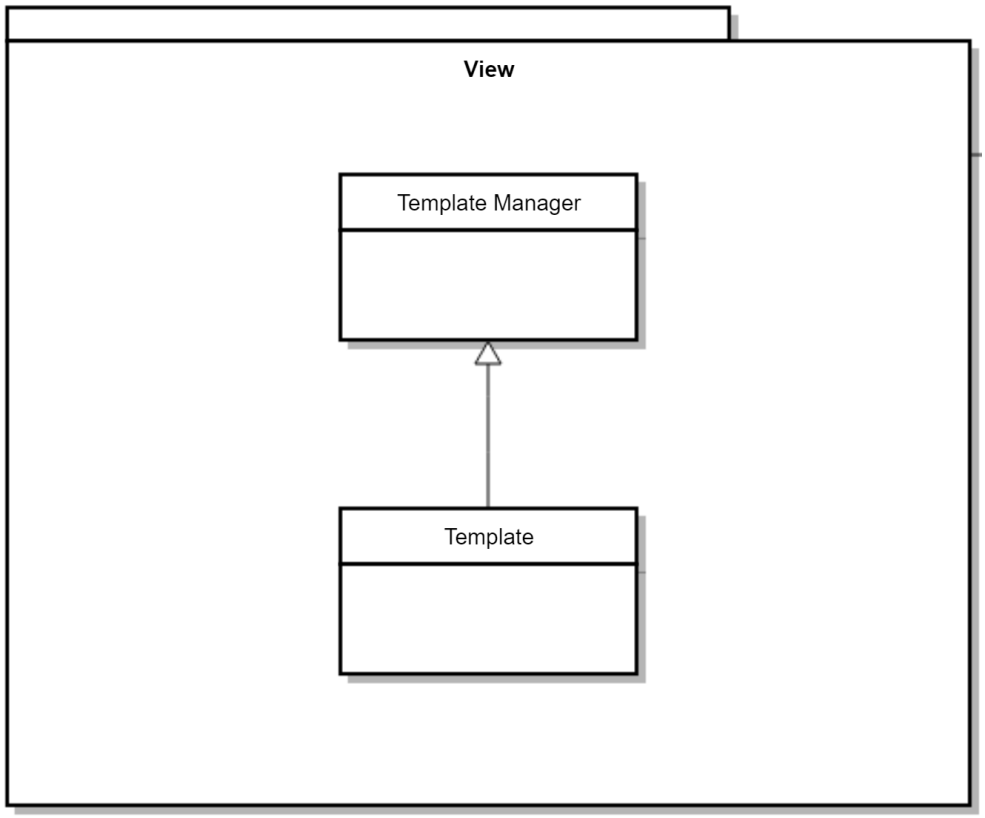
Game Manager: Gives an abstraction to the game objects for them to use the resources of the system. The class is responsible for initializing games , save/load their state and terminating games. Moreover, the class is used for reading the installed game configuration files and creates a GameType object for the each game type.

Integration Manager: Runs the integrity tests for the games and provides results in a meaningful way.

Game Type: Represents a game type in the domain model, consists of fields that every game has common and required to have to be installed. Created one for every game installed.

Game: A wrapper class for the game instances. It represents the game sessions.

3.1.3 View Layer

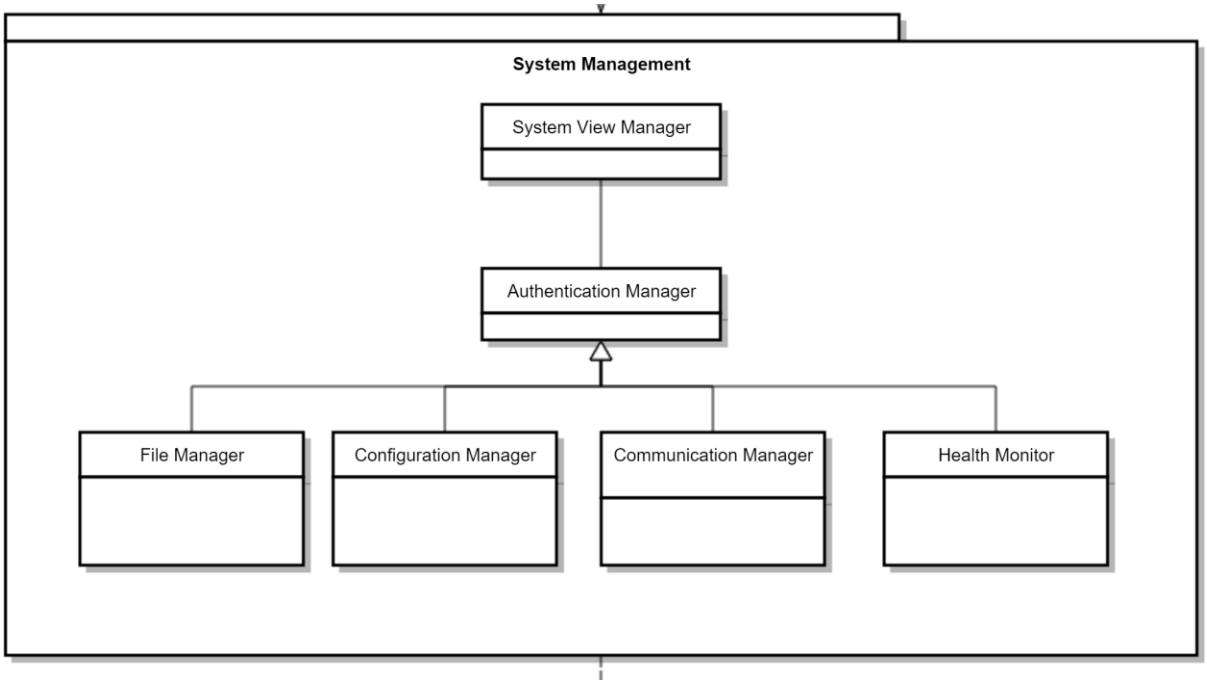


View layer consists of templates. Where the presenter layer fills the dynamic fields according to the current state of the system or the game and returns them as filled static views to the view layer. Then the view layer serves the views as HTML documents.

TemplateManager: The class for creating templates.

Template: Represents the view template of a specific screen or a part of a screen.

3.2 System Management Subsystem



System management subsystem consists of 2 layers, first the hardware layer and the system layer. The system layer has the logic for system to work properly and hardware layer provides an abstraction for the system layer to use.

File Manager: Provides a File System interface for other subsystems to use.

Health Monitor: Uses hardware layer for gathering information and serves it as meaningful data.

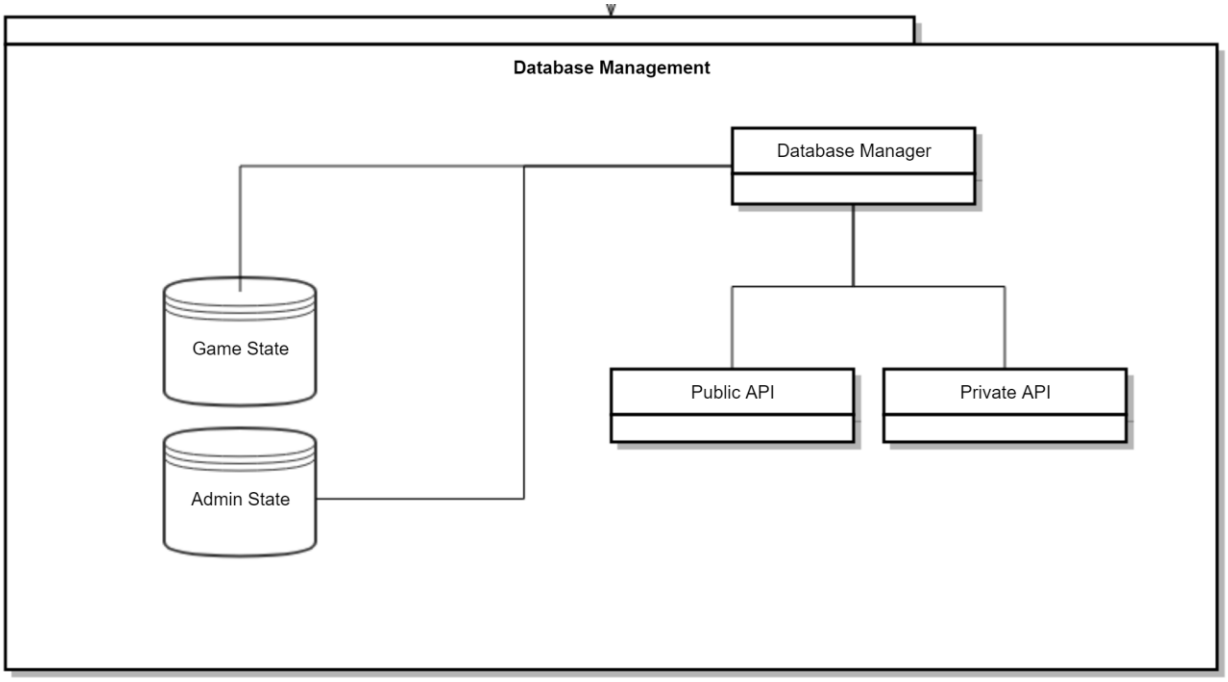
Configuration Manager: Controls the configuration files, stores the system settings , user settings and changes them when needed.

Communication Manager: Controls and regulates the WiFi connections and controls the captive portal.

Authentication Manager: Controls the authentication of the admin.

System View Manager: Provides views for the system interactions with the user.

3.3 Database Management Subsystem



The subsystem is responsible for managing the database software and create an abstraction level for other subsystems to use.

Database Manager: The class that makes requests and retrieves responses from the database. It has a logic for the transactions , data retrieval and aliasing to prevent data loss.

Admin: Represents admin and its credentials.

GameState: Represents a saved game in the database.

Public API: An Application User Interface for other subsystems and developers to use to make basic database operations.

Private API: An Application User Interface for the box, developers to use, works with a key for authentication purposes. It used to get the statistics, access system related documents etc.

4. Glossary

Graphical User Interface: GUI is the interface that interacts with users with dynamic images, buttons and labels. Users can navigate between pages by using the interface with simple operations such as clicking a button. The device responds to the user according to the operation and the given input.

Model View Presenter: Model–view–presenter (MVP) is a derivation of the Model-View-Controller (MVC) architectural pattern, and is used mostly for building user interfaces. In MVP, the presenter assumes the functionality of the "middle-man". In MVP, all presentation logic is pushed to the presenter [3].

5. References

[1] K. Pretz, "Medical Experts Say Addiction to Technology Is a Growing Concern", The Institute, 2016.[Online].Available: http://theinstitute.ieee.org/ieee-roundup/blogs/blog/medical-experts-say-addiction-to-technology-is-a-growing-concern. [Accessed: 04- Nov- 2017].

[2] Sigman, A. (2009). Well connected? The biological implications of 'social networking'. *Biologist*, 56(1), pp.14-20.

[3] "Model–view–presenter", *En.wikipedia.org*, 2017. [Online]. Available: https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93presenter. [Accessed: 20- Dec- 2017].