МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования   
**«Национальный исследовательский   
Нижегородский государственный университет им. Н.И. Лобачевского»**

**(ННГУ)**

**Институт информационных технологий, математики и механики**

**Кафедра программной инженерии**

Направление подготовки: «Программная инженерия»

Профиль подготовки: «Общий профиль»

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА**

**Тема:**

**«Разработка и реализация Telegram-бота консультанта Dota»**

**Выполнил:** студент группы 381608

Агликов Илья Андреевич

Подпись: ­­

**Научный руководитель:**

к.ф.-м.н., доцент кафедры ПРИН ННГУ,

Шапошников Дмитрий Евгеньевич

Подпись:

Нижний Новгород  
2020

Введение3

1. Обоснование актуальности работы6
2. Описание предметной области7
   1. Постановка задачи8
   2. Анализ имеющихся решений и выводы9
   3. Описание существующих инструментов для решения задачи12
   4. Выбор и обоснование конкретных инструментов13
   5. Описание модели деятельности «как должно быть»14
3. Разработка проектного решения16
   1. Разработка требований16
   2. Проектирование архитектурного дизайна16
   3. Разработка программного кода21
   4. Разработка руководства пользователя27
   5. Коммерциализация28
4. Описание тестирования консультанта29

Заключение30

Список литературы31

Введение

Киберспорт, под которым понимают командные или индивидуальные соревнования на основе видеоигр, является молодой и быстрорастущей отраслью, зародившейся в 1997 году, вместе с основанием первой киберспортивной лиги Cyberathlete Professional League, сделавшей первый турнир по дисциплине Quake. Киберспортивные дисциплины делятся на разные классы в зависимости от навыков, требуемых от киберспортсменов, игровых задач и общих моделей ведения игры. Призовые фонды киберспортивных мероприятий не раз достигали отметку в несколько миллионов долларов. Так, рекордным по призовому фонду мероприятием на данный момент является The International 2019 по компьютерной игре Dota 2, взявший рекордную для киберспорта планку в 34 млн долларов.

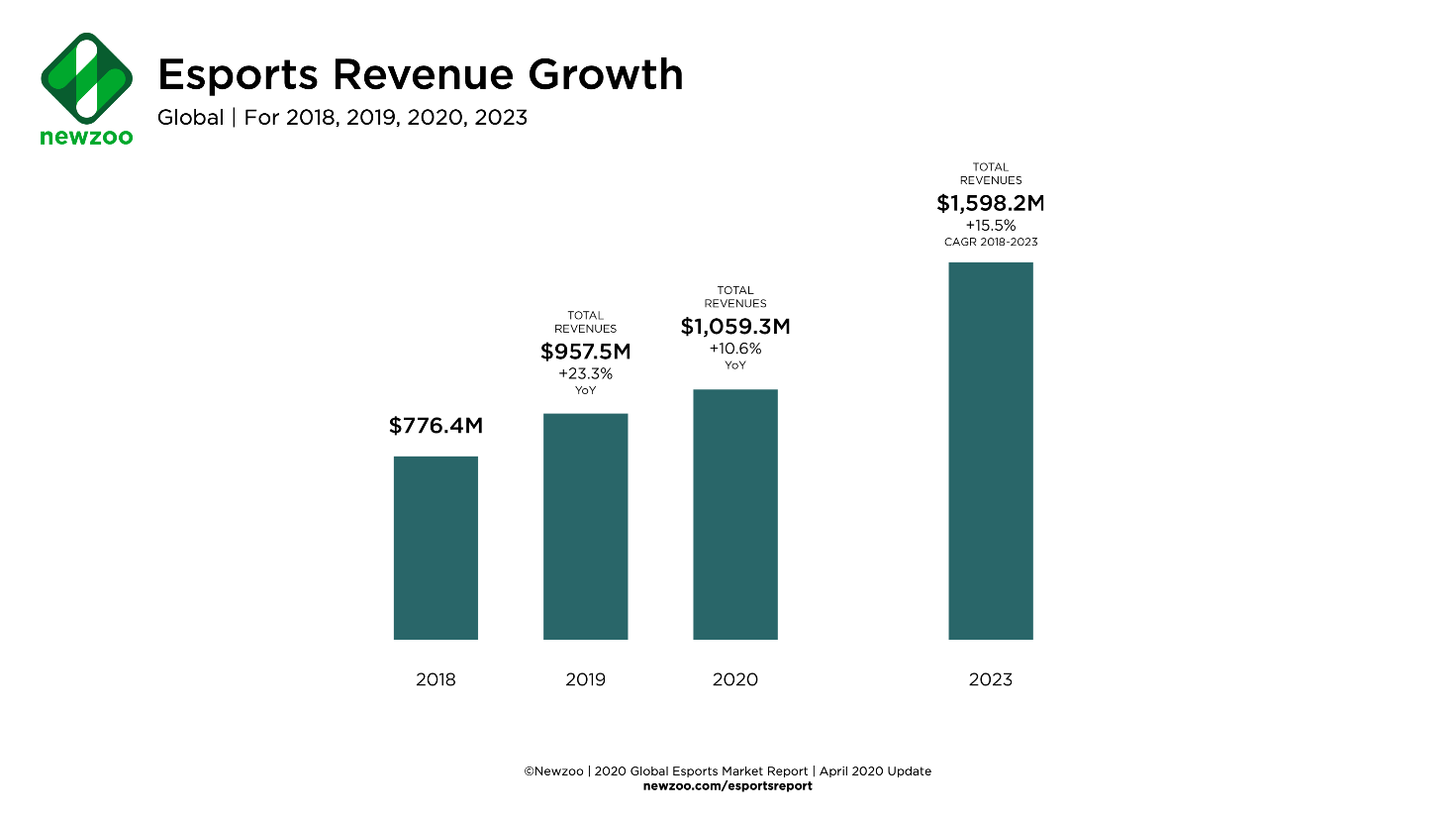


Рис. 1: Newzoo аналитика

Киберспорт достиг признания как одна из спортивных дисциплин благодаря тому, что в нём есть устоявшиеся правила, используются интеллектуальные способности участников и проводят обязательные подготовки. 14 апреля 2018 года Международным олимпийским комитетом компьютерный спорт был признан официальным видом спорта, и в 2024 году будет введён в список Олимпийских игр. Основываясь на сведениях “Newzoo” (наиболее популярном в мире источнике игровой и киберспортивной аналитики), в 2020 году общий доход киберспортивного рынка достигнет 1.1 млрд. долларов, что на 150 млн. долларов больше в сравнении с 2019 годом, а рост киберспортивной аудитории составит 11.7%, сама аудитория составит 495 млн человек.

Одной из наиболее популярных киберспортивных дисциплин в мире является компьютерная игра Dota 2. Это одна из двух наиболее популярных игр в жанре MOBA (Multiplayer Online Battle Arena) (это место с ней делит League of Legends), ежедневное количество игроков в ней в среднем составляет порядка 700 тысяч игроков, и более 8 миллионов активных игроков в месяц. Игра представляет из себя бой двух команд по 5 человек на симметричном игровом поле. Победа той или иной команды достигается двумя факторами: навыками игры самих игроков и грамотно составленными пиками героев (от англ. “Pick” – выбирать; в игре означает выбор конкретного героя, либо всех выбранных героев в команде), которых каждая команда выбирает в начале игры. Все герои в Dota 2 имеют связи в отношении друг друга, которые выражаются в превосходстве одних героев над другими, то есть способностях, полностью нивелирующих способности оппонента, или же, наоборот, они могут усиливать друг друга, обладая хорошо сочетаемыми способностями. Если герой имеет преимущество в отношении другого, то говорят, что он является контр-пиком для этого героя (от англ. “Counter-pick”, что означает выбор хорошо противостоящего героя).

Успех в игре напрямую зависит от правильного выбора героев, ведь даже обладая высокими навыками игры, значительно превышающими вражеские, можно легко проиграть, имея менее подходящий пик. Именно поэтому так важно владеть навыком построения возможных связей между героями, чтобы уметь точно определять правильный выбор. К сожалению, не все игроки в Dota 2 обладают этим навыком даже среди опытных игроков, новички-же просто не могут им обладать по определению. Для этого и было решено создать программу, помогающую пользователям в выборе героев, которая поможет в значительной степени улучшить навыки игроков в подборе и определении правильного пика, что в свою очередь скажется на успехе игры.

Целью данной бакалаврской работы является разработка сервиса, консультирующего игрока в правильном выборе героев в компьютерной игре Dota 2.

1. Обоснование актуальности работы

При выборе между web-сайтом, самостоятельным приложением и чат-ботом был выбран последний, так как он менее зависим от мощности устройства пользователя и предоставляет единый интерфейс для десктопной и мобильной версии. Чат-бот позволит игрокам иметь легковесного и быстрого консультанта, который не будет занимать места в памяти на физическом носителе (в случае приложения).

Актуальность подобного сервиса обусловлена несколькими факторами:

1. Востребованность любых сервисов, предоставляющих консультирующую информацию по игре, растёт прямо пропорционально численности игрового сообщества этой игры. На данный момент Dota 2 является одной из самых популярных MOBA-игр в мире.
2. Сервисов по Dota 2, предоставляющих предложения по выбору героев довольно мало, и нет ни одного решения в виде чат-бота. На данный момент представлены решения в виде сторонних вебсайтов и мобильных приложений.
3. Все решения выбора героев в виде ввода текстовых имён представлены на английском языке и требуют официального имени. Для не знающих английский язык игроков (в контексте данной работы будем рассматривать русскоязычный сегмент) не всегда является известным официальное имя героя, к тому же такие сервисы поддерживают только работу с полным именем, поэтому для пользователя этот метод довольно медлителен в использовании.

2. Описание предметной области

Чат-боты – это программы, имитирующие общение с реальным человеком в соц-сетях или мессенджерах, использующиеся во множестве сфер и направлений. Сейчас они находятся на пике популярности в виду своей полезности и удобности.

Чат-боты предоставляют удобный доступ к информации. Приведём следующий пример. Допустим, есть магазин одежды, человек заходит в чат-бот, и у него выдаётся несколько вариантов диалога. Пользователь может узнать цену по какому-то интересующему его товару, просто посмотреть каталог текущих моделей, которые есть в этом магазине, посмотреть акции, узнать, как добраться до магазина и так далее.

Одним из основных преимуществ чат-бота перед человеком является его круглосуточная доступность, он отвечает всегда и сразу. Также для него не нужно скачивать никаких дополнительных программ, так как площадки, на которых базируется бот, как правило, уже установлены у пользователей. Начало общения с ботом происходит как и с любым другим человеком, нужно просто начать с ним переписываться.

Официально поддерживаются следующие мессенджеры: telegram, VK, facebook и viber. Неофициально – WhatsApp и Instagram. Официальная поддержка означает, что платформа предоставляет инструменты (API) для разработки чат-ботов / взаимодействия с площадкой. Неофициальная – не предоставляет API, но написать бота для неё всё равно возможно, однако такой бот будет имитировать общение с обычным человеком.

Чат-боты используются для:

1. Ответов на вопросы пользователей.
2. Возможность проходить тесты. Например, для определения какой-то характеристики пользователя (для примера с магазином одежды это может быть предпочитаемый пользователем стиль одежды / параметры пользователя).
3. Информационная рассылка.
4. Возможность принимать платежи.
5. Модерация группы / канала. Можно добавить к групповому чату / группе / каналу бота, и бот будет заниматься модерацией, т.е. удалять комментарии с нехорошими словами, либо банить людей, рассылающих спам, но это всё уже зависит от потребности владельца канала и от того, как реализован бот.
6. Просто для развлечений. Например, есть боты с текстовыми квестами и всевозможными играми.

2.1. Постановка задачи

Для выполнения поставленной задачи по реализации такого сервиса выделяются следующие этапы:

1. Провести анализ популярных решений на рынке подобных сервисов, разобраться в их работе.
2. Изучить источники для получения информации о списке доступных на данный момент героев и информации по каждому герою, против кого эти герои слабы.
3. Организовать получение, хранение и обновление вышесказанной информации.
4. Разработать чат-бота, взаимодействующего с пользователем. Бот должен принимать от пользователя такие команды, как, например, «добавить героя в вашу команду» / «добавить героя во вражескую команду».
5. Организовать функционал, используемый чат-ботом для удовлетворения запросов пользователя, а также вывод чат-ботом результата, если он требуется запросом.

2.2. Анализ имеющихся решений и выводы

На данный момент были выделены 3 имеющихся сервиса, предоставляющих помощь в выборе героя:

1. «Dota Plus» - Разработка от официальных разработчиков Dota 2 – Valve. Это единственная встраиваемая непосредственно в игру надстройка, позволяющая прямо во время игры консультировать игрока, подсвечивая тех или иных героев. Помимо неё, невозможно сделать такой же встраиваемый сервис, потому что это затронет файлы игры, что запрещено разработчиками. За нарушение этого запрета игрок получает штраф в steam – крупнейшей цифровой игровой площадке в мире (также созданной Valve). Из-за этой встраиваемости и автоматизации, Dota Plus не обладает недостатками перед другими решениями, является самым удобным из возможных вариантов. Dota Plus является платным и продаётся компанией по типу подписки за 4$ в месяц.
2. «Dotapicker.com» - вебсайт, главной целью которого служит формирование драфта героев (от англ. draft – означает состояние всех выбранных героев). Этот вариант по точности практически не отличается от Dota Plus, однако обладает рядом недостатков. Он имеет два варианта выбора героев: с помощью графического интерфейса и ввода полного имени героя:
   1. Главный минус графического интерфейса в том, что игроку даны 119 небольших изображений персонажей в довольно небольшом окне. Это плохо тем, что заставляет всматриваться более детально в каждый аватар (изображение с героем) героя, и приходится в довольно сжатые сроки (15 секунд на стадию запретов, 40 на выбор каждого героя, 20 на штрафное время, во время которого игрок теряет золото, необходимое на старте) найти из 119 вариантов нужный. Это довольно сложно, часто приходится проходиться глазами не один раз по всем аватарам, так как зачастую нужный герой упускается из-за рассредоточенности внимания.
   2. Метод текстового ввода имени героя тоже имеет несколько недостатков в плане удобства. Приходится вводить полное имя героя, что отнимает время, в виду того, что имена многих героев достаточно длинны и сложны. Особенно это сказывается на игроках, не имеющих достаточно опыта с английским языком. Например, возьмём героя «Ancient apparition». Набрать это имя получается не сразу хотя бы потому, что оно не сразу всплывает в голове, так как все герои в русскоязычном сегментe (а также и во всех других) имеют свои сленговые имена (в русском сегменте он зовётся просто «Аппарат»). Сленговые имена означают неофициальные, придуманные самими игроками для более быстрого или удобного обращения, названия героев. И эти сленговые имена для большинства игроков сбивают написание нужных имён, как, например, для вышесказанного Аппарата. Большинство игроков видят написание его имени «Apparation», что не даёт найти героя, точнее тормозит его поиск.
3. «Помощник дотера для Dota 2» - Приложение для мобильных устройств на базе Android и IOS. Уступает двум вышеописанным вариантам. Оно имеет очень навязчивую рекламу, всплывающую на каждом шагу, чтобы убрать которую, нужно заплатить цену, сопоставимую с покупкой нескольких Dota Plus. Этот помощник дотера имеет в своём распоряжении только графический интерфейс, причём такого же масштаба, что и Dotapicker, однако за счёт того, что используется он на мобильном устройстве, является более удобным, так как расположен ближе к глазам, чем монитор. Способ выбора героев у данного приложения лучше ещё и тем, что герои там разделены на 3 раздела по их соответствующим главным характеристикам, что даёт большую сосредоточенность и сконцентрированность при поиске нужного игроку аватара. Приложение выигрывает у Dotapicker в поиске нужного героя, но проигрывает в самом формировании и управлении драфтом.

Рассмотрев все эти варианты довольно детально и получив опыт использования каждого, можно прийти к следующим выводам. Чтобы разработанное нами приложение имело место и успех на рынке на фоне всех трёх конкурентов:

1. Приложение должно давать высокую скорость поиска и выбора нужных героев.

Данную проблему предполагается решать с помощью системы сленговых имён в текстовом способе ввода. У каждого героя есть свои сленговые имена, разнящиеся от региона к региону, при вводе которого скорость выбора отдельного героя может разниться от секунды до примерно 4, что уже быстрее ввода имён на Dotapicker, а также гораздо быстрее поиска аватара героя в графическом интерфейсе, где процесс поиска отдельного героя может достигать до 30 секунд.

1. Приложение должно выигрывать в ценовой политике у мобильного приложения и Dota Plus.

Предполагается достичь этой цели, сделав чат-бота полностью бесплатным, за исключением одной функции – выбора героев в реальном времени, не требующей от пользователя никаких действий. Само использование этой функции является платным, поэтому сделать её бесплатной просто недопустимо. Более подробно ценовая политика этой функции описана в разделе «3.5. Коммерциализация».

1. Приложение не должно содержать никакой рекламы, а только предоставлять функционал пользователю, иначе реклама может испортить пользовательский опыт.
2. Приложение должно быть легковесным и быстрым. Dotapicker является довольно тяжёлым вебсайтом, который проблематично использовать без задержек на смартфонах среднего сегмента и ниже. Мобильное приложение требует от пользователя выделить место под память на своём смартфоне (для описанного выше помощника требуется 93 мб памяти), также его работа напрямую зависит от мощности используемого устройства.

Предполагается достичь этой цели с помощью мессенджеров. Они легковесны и быстры, есть у подавляющего большинства молодёжи (которая и является основной аудиторией игры). А сам чат-бот вообще не будет замедлять работу пользователя, так как все вычисления и обработки происходят на сервере, где чат-бот будет размещён. При этом пользователю не нужно выделять место на устройстве, так как приложение находится на сервере, а его устройство будет получать доступ к функционалу бота посредством мессенджера.

2.3. Описание существующих инструментов для решения задачи

Перед тем, как приступать к разработке нашего бота, следует выбрать инструментарий и площадку:

1) Предстоит выбрать язык программирования высокого уровня, на котором будет вестись разработка. Ввиду отсутствия объёмных вычислений, за приоритет будем брать время, затраченное на разработку.

2) Площадка, на которой будет развёртываться чат-бот должна предоставлять хороший функционал для бота, быть быстрой, а самое главное – популярной. Рассматриваться будут перечисленные во введении 4 платформы, которые предоставляют официальное API (application programming interface) для взаимодействия с сервисом (написания бота, который будет взаимодействовать с ним). Рассматриваются: VK, telegram, facebook, viber.

3) Также предстоит выбрать какой библиотекой пользоваться для взаимодействия с API. Эта библиотека - своего рода обёртка вокруг предоставляемого API.

2.4. Выбор и обоснование конкретных инструментов

Для выбора платформы, на которой будет базироваться бот, будут рассматриваться какое эта платформа предоставляет API и их популярность для рассматриваемой аудитории. Рассматриваться будет русскоязычный сегмент, так как данная работа ориентируется преимущественно на него. С точки зрения популярности, Viber и Facebook сразу отпадают, так как оба не пользуются популярностью у молодёжи, в частности игроков в Dota 2. Первый за своё управление памятью, которое держит все данные на устройстве пользователя, второй просто не популярен ввиду существования русской альтернативы – VK (так сложилось исторически). Между VK и telegram предстоит выбирать по функционалу, так как оба примерно в равной степени популярны. Telegram имеет более высокую скорость отправки сообщений (она практически моментальная даже при низкой скорости интернета), что делает его для нас более предпочтительным. Также telegram обладает более лучшей безопасностью, как, например, секретные чаты с авто-удалением, пароли на вход и двухэтапная авторизация. Но самое главное – это функционал, который предоставляет telegram для создания чат-ботов, благодаря которому они получаются довольно удобными в управлении. Например, создание пользовательских клавиатур, использование которых сделает разработку в разы более удобной для пользователя.

При выборе языка программирования ключевую роль играет скорость и удобство разработки, в то время как производительность является второстепенной характеристикой, ввиду редких обращений к серверу и отсутствию в необходимости объёмных вычислений. Просмотрев библиотеки для работы с telegram bot API на тех или иных языках программирования, был выбран Python. Сильно ускоряя разработку в сравнении с другими языками (такими как C++, C, Java), он также считается самым популярным языком программирования для написания чат-ботов.

2.5. Описание модели деятельности «как должно быть»

Приложение должно работать примерно по следующей схеме. Пользователь задаёт свои запросы, используя текстовые команды (перечисленные в документации пользователя, что является разделом «помощь» в боте) или графический интерфейс, основанный на пользовательских клавиатурах telegram bot API. Эти запросы обрабатываются отдельными модулями. Всего их два, и нужный выбирается в зависимости от того, чего пользователь хочет. Первый представляет из себя просто набор функций, отвечающих на нужный запрос пользователя и предоставляющий ему интересующую его информацию. Второй отвечает непосредственно за основной функционал бота – это предложения для игрока, основанные на сформированном драфте. Эти модули должны обрабатывать переданные им данные и возвращать приемлемый для пользователя результат.

В первом случае это должна быть информация. Пользователь может передавать в этот модуль id профиля в Dota 2 интересующего игрока, чтобы увидеть какую-то информацию по нему, например, статистику его последних игр, обзор по его героям (на ком он сыграл, у каких героев самый высокий винрейт и так далее). Переданные Dota id должны указывать на аккаунты steam с открытым доступом к просмотру, иначе Dotabuff (крупнейший вебсайт по статистике в Dota 2, используемый в проекте) не сможет получить к ним доступ.

Во втором случае пользователь делает несколько запросов с именами героев, это могут быть любые сленговые имена героев, перечисленные в отдельном файле, также он может удалять отдельных героев, тем самым управляя текущим драфтом, формируемым им в чат-боте.

3. Разработка проектного решения

3.1. Разработка требований

Разрабатываемый программный продукт должен быть оптимизирован, в первую очередь, по производительности, то есть не делать лишних запросов, по возможности хранить ответы на них в хранилище. В качестве этого хранилища была выбрана довольно лёгкая в использовании и управлении встраиваемая СУБД sqlite. Мы можем управлять и взаимодействовать с ней прямиком из кода, с помощью библиотеки sqlite3, являющейся встроенной в Python и не требующей отдельной её установки.

Бот должен отличать команды пользователя и стараться всегда предоставлять ему пользовательскую клавиатуру для удобства. Если пользователь каким-то образом смог выйти из этой клавиатуры (при тестировании такого случая обнаружено не было), то ему достаточно просто ввести команду `/start`, чтобы вернуть её, но в таком случае он также вернётся в начальное меню.

Чат-бот должен подстраиваться под язык пользователя, предоставляя либо англоязычный, либо русскоязычный интерфейс, в зависимости от местонахождения пользователя.

3.2. Проектирование архитектурного дизайна

Весь проект разделён на несколько модулей, каждый из которых выполняет свой определённый функционал и взаимодействует с другими. Модули в проекте:

1. “bot.py” – В этом модуле содержится непосредственно сам бот. Здесь описана вся логика, отвечающая за взаимодействие с пользователем. Более конкретными задачами этого модуля являются:
   1. Принимать от пользователя команды.
   2. Проводить анализ сообщений пользователя и определять, что именно боту нужно сделать.
   3. В зависимости от определённого намерения пользователя, бот должен выполнить тот или иной функционал своими силами или пользуясь помощью других модулей.
2. “config.py” – Конфигурационный файл, содержащий токен для доступа к управлению ботом. Данный файл не предназначен для выкладывания в открытый доступ.
3. “dotabase.py” – Интерфейс одного из двух главных функционалов. Данный модуль отвечает за секцию с получением всякого рода информации (по большей части с вебсайта Dotabuff.com). Предполагается, что пользователю будет доступен функционал по получению:
   1. Статистики конкретного игрока. Для этого нужно указать Dota id этого самого игрока, а также чтобы профиль с указываемым идентификатором имел в настройках открытый доступ, иначе вебсайт, с которого мы получаем информацию не сможет предоставить данные об этом профиле.
   2. Статистику по отдельному герою. Эта статистика может иметь несколько подразделений. Например, это может быть список его соотношений побед относительно других героев, список наиболее часто покупаемых предметов, их винрейт на данном герое, наиболее популярная раскачка героя и так далее. Данный функционал, как и функционал всей секции, планируется наращивать.
4. “dotabuff\_parse.py” – Этот модуль отвечает за несколько вещей: интерфейс для работы с базой данных и парсер самого популярного вебсайта «Dotabuff.com».
   1. Формирование базы данных, в которую входит 120 таблиц: одна для списка официальных имён всех героев, остальные 119 для каждого героя из первой таблицы. Количество этих таблиц может увеличиваться по мере выхода новых героев, об этом уже будет сигнализировать первая таблица). Каждая из этих 119 таблиц показывает относительные показатели доли побед относительно каждого другого героя в Dota 2 за последнее время.
   2. Предоставление методов для получения информации из сформированных таблиц.
   3. Касательно самих данных, из которых формируется база данных, этот модуль:
      1. Делает 120 запросов и получает 120 ответов с вебсайта Dotabuff.com с помощью библиотеки “requests.py”, потом преобразует полученные ответы в доступный для обработки файл формата html с помощью библиотеки lxml.html.
      2. Обрабатывает и ищет нужные данные с полученных html-страниц с помощью технологии Xpath.
5. “dotabuff.db” – Сформированная база данных. Заполняется модулем dotabuff\_parse.py. На момент написания данной работы, база данных хранит в себе 120 таблиц: 1 – имена всех героев в игре (содержит 119 строк с именами героев), 119 – по таблице на каждого героя, в которых указаны все остальные герои с приписанными им значениями – показатель того какую долю побед они имеют в отношении нашего выбранного героя (содержит по 118 строк каждая).
6. “conter\_picker.py” – Интерфейс, который отвечает за второй главный функционал. Этот модуль отвечает за обработку предлагаемых героев на основе уже имеющихся, добавление новых героев в драфт, удаление имеющихся, очистка драфта, показ имеющихся на данный момент в драфте героев, а также показ предлагаемых. Весь модуль состоит из одного класса, который представляет собой сущность драфта.
7. “heroes.json” – Файл формата json, в котором заключены сленговые имена героев.
8. “language.py” – Модуль, отвечающий за локализацию чат-бота. На данный момент в нём представлены два варианта языков: русский и английский. В будущем планируется увеличивать количество языков, поддерживаемых ботом.

Список модулей с их интерфейсами, классами и атрибутами представлен на рисунке 2.

Взаимодействие всех этих модулей представлено на рисунке 3. Пустыми стрелками обозначено просто добавление данных, будь это токен для управления ботом, язык или база данных. Полными стрелками обозначено, что модуль добавляет свой функционал в другой модуль.



Рисунок 2: Полная схема модулей с их интерфейсами и атрибутами

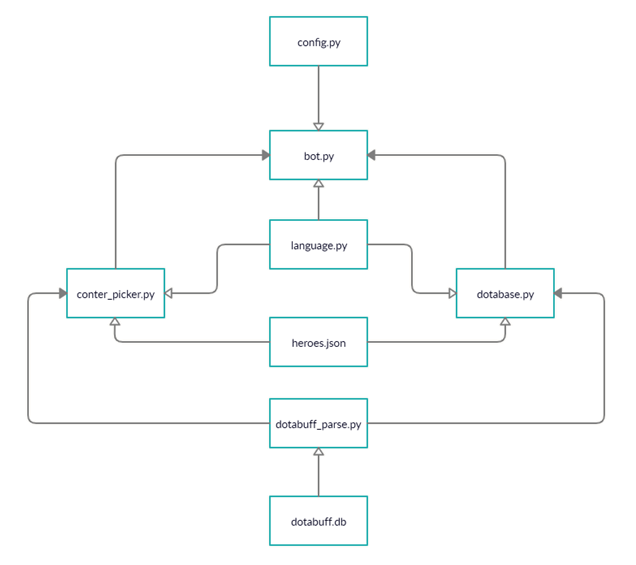


Рисунок 3: Взаимодействие всех модулей.

3.3. Разработка программного кода

Разработку прежде всего следовало начать с задания начального интерфейса пользователя в основном модуле bot.py, то есть непосредственно с начала создания бота, а по мере роста функционала в нём, уже наращивать код в других модулях.

Для начала создаётся модуль bot.py и к нему подключается библиотека pyTelegramBotAPI функцией import telebot, а также подключается файл config.py (import config), чтобы из него достать TOKEN (токен был вынесен в отдельный файл для удобства и конфиденциальности, в случае публикации исходного кода бота). Так как для каждого пользователя не может запускаться отдельная копия программы со своим личным пространством имён и памятью, то определяется несколько глобальных переменных коллекционного типа данных «словарь». В языке Python словарь является неупорядоченным массивом, каждый элемент которого представляет из себя пару «ключ: значение». В каждой такой переменной ключом будет идентификатор конкретного чата с конкретным пользователем, который можно получить из каждого переданного сообщения по пути message.chat.id. Переменных этих всего 4:

1. “lang” – Для каждого пользователя при первом запуске бота определяется его языковой код (message.from\_user.language\_code). На данный момент в боте предоставлены два интерфейса: русскоязычный и англоязычный. Оба они реализованы в файле language.py и представляют собой два класса. Английский является базовым, так как вся разработка первоначально ведётся именно для него. А русский уже является наследником английского, с переопределением атрибутов под свой язык. В будущем при добавлении других языков предполагается также наследоваться от английского с последующим переопределением его атрибутов. В первой функции по сообщению пользователя определяется используемый им язык и создаётся новый элемент в словаре lang (уникальный идентификатор чата с пользователем: язык). В последующем в коде вместо каких-либо текстовых сообщений вызываются конкретные атрибуты конкретного класса. Если для не английского языка этот элемент ещё не был определён, то вызывается этот атрибут в базовом классе. Таким образом имеется единый код, язык интерфейса которого будет автоматически подстраиваться под пользователя.
2. “steam\_id” – Уникальный идентификатор каждого игрока Dota 2. Чтобы получить информацию, связанную с пользователем, необходимо, чтобы этот пользователь имел настройки на открытую видимость аккаунта. Этот id пользователя просят задать при первом использовании какой-либо функции, требующей его. В последствии он может изменить его в любое время. Id используется, например, для вывода информации о конкретном пользователе.
3. “picker” – В этом словаре для каждого пользователя (ключ) хранится экземпляр класса Picker из модуля conter\_picker.py. Экземпляр создаётся при первом использовании функционала по выбору героев для игры.
4. “pick\_status” – Этот словарь является вспомогательным для последней функции в боте – pick\_hero(message). С помощью него мы определяем в какую команду нужно добавить (или из какой удалить) желаемого героя. Параметр задаётся в функциях, вызывающих эту. Данный параметр принимает одно из 4 значений: ‘ally\_pick’, ‘enemy\_pick’, ‘ally\_del’, ‘enemy\_del’.

Бот начинает свою работу с функции ‘start’, в которую пользователь попадает при первом запуске бота или при использовании команды ‘/start’. В этой функции определяется язык интерфейса, очищается текущий пик (если переход в данную функцию был совершён во время формирования драфта) и задаётся пользовательская клавиатура, состоящая из 4 вариантов выбора на старте (показано на рисунке 4).

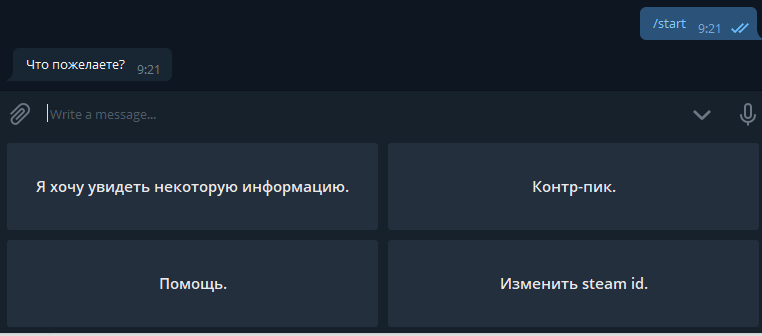


Рисунок 4: Начальный вид меню.

Первые два варианта выбора – ключевой функционал. Первый предоставляет пользователю какую-либо информацию, например, как показано на рисунке 5, это информация по конкретному игроку. Второй – контр-пикер. Третий пункт – своего рода документация, знакомящая пользователя с ботом и дающая краткое описание команд (6). Четвёртый позволяет задать steam\_id впервые, либо изменить уже заданный.

Вся работа бота построена на приёме команд от пользователя и ответе на них. При каждом сообщении пользователя нашей программе посылается структурированный объект, по виду напоминающий файл формата json. При получении такого сообщения, бот проходится по всем функциям, обрамлённым специальным декоратором из библиотеки telebot (или любой другой библиотеки для работы с API telegram) – telebot.TeleBot.message\_handler.

Этот декоратор может различать сообщения четырьмя способами:

1. С помощью определения типа передаваемого в сообщении контента (type). Например, это может быть текст (установлен по умолчанию, если данный параметр ‘type’ не задан), документ, стикер, изображение, аудиозапись, голосовое сообщение и так далее.
2. С помощью обработки контента функцией (func). Как правило, это безымянная lambda-функция. Функция должна возвращать True, чтобы декоратор запустил обрамляемую функцию на выполнение.
3. С помощью регулярных функций (regexp).
4. Непосредственно с помощью команд (commands=[ ]).

В боте используются все 4 способа детерминирования сообщения. Чтобы определить, что пользователь ввёл steam id, используется регулярное выражение на поиск сообщения, состоящего исключительно из цифр. Подавляющее большинство функций определяется с помощью команд и lambda-функций, которые сравнивают введённое (в данной работе это, как правило, переданное сообщение с пользовательской клавиатуры) сообщение с шаблоном из language.py соответствующего языка. При этом, это же большинство функций совмещает в себе командный и функциональный способ определения, что позволяет telegram API. В данном случае, необходимо чтобы хотя бы один из декораторов вернул True. В самом конце модуля расположена функция, декоратор для отлова сообщений, которая не имеет никаких параметров определения (кроме дефолтного текстового типа). Таким образом, в эту функцию попадают все сообщения, которые не прошли ни одну из проверок выше. Задумано, что это функция используется для приёма от пользователя имени героя. В случае, если принятое сообщение не соответствует ни одному из сленговых или официальных имён героев, бот предлагает пользователю попробовать ввести сообщение ещё раз.

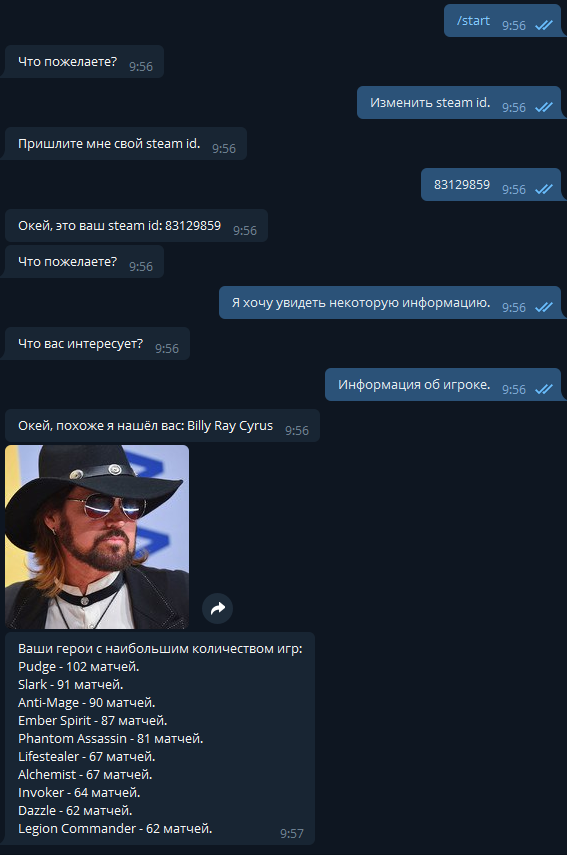
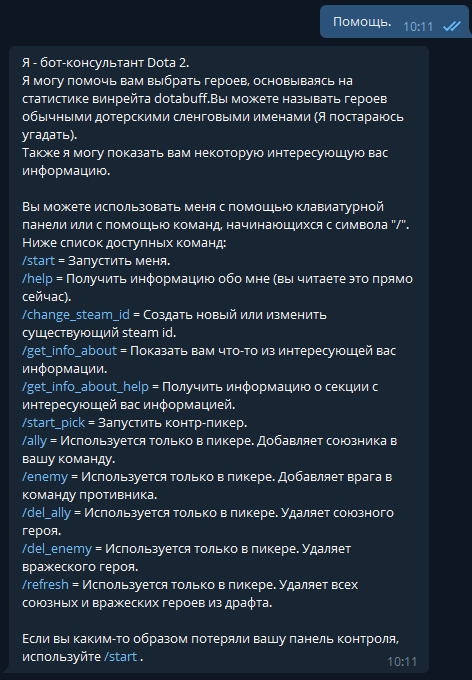
 

Рисунок 5: Информация игрока. Рисунок 6: Вызов документации.

Модуль dotabuff\_parse.py. Этот модуль ответственен за парсинг таблиц относительных винрейтов каждого героя, списка официальных имён всех героев, формирование базы данных на основе полученной информации, а также предоставление интерфейса для доступа к базе данных. Первым делом необходимо получить полный список имён героев. Перечислять их вручную менее надёжно в плане правописания (особенно учитывая немалое количество героев в игре) и лишает возможности автоматизации на случай выхода новых героев. Потом динамически составляются запросы к страницам каждого из героев в полученном списке, парсится нужная информация и динамически формируются таблицы для каждого из полученных героев. Последние упомянутые таблицы имеют вид пар «ключ – значение», где ключом выступает противопоставляемый герой, а значением его показатель побед относительно рассматриваемого. То есть в каждой такой таблице содержатся 118 строк. База данных будет обновляться с периодичностью раз в день. Это позволит давать более актуальные показатели побед между героями, а также не упустить какого-то нового героя или резкого изменения этого соотношения между героями в случае выхода нового обновления игры (что в игре происходит довольно часто).

Предоставляются два интерфейса для взаимодействия с базой данных. Первый даёт нам непосредственно сам список героев. Второй предоставляет таблицу запрошенного героя.

Модуль dotabase.py построен без использования объектно-ориентированного подхода, в нём нечего представлять за сущность. Основная цель этого модуля – предоставить некий набор функций, вызываемых из бота.

Conter\_picker.py содержит ключевой функционал – предоставляет советы по героям, основываясь на текущем драфте, заданном пользователем. Модуль представляет из себя один класс, являющийся сущностью драфта. Этот класс содержит в атрибутах экземпляра текущее состояние драфта, это множество (set, коллекционная не упорядоченная структура данных, по сути является словарём без значений) имеющихся героев и противников; словарь предполагаемого пика, где ключом является имя героя, а значением – его винрейт относительно всех выбранных героев; язык для вывода предлагаемого или текущего пика. Класс предоставляет интерфейсы для добавления и удаления героев. При этом в результате каждого такого изменения меняется и сам словарь предложений. Также в этом модуле содержится функция приёма введённого пользователем имени героя, которая либо находит его по сленговому / полному имени, либо возвращает None, что передаётся в чат-бот как «не найдено», в следствии чего пользователя просят попробовать ввести ещё раз.

3.4. Разработка руководства пользователя

В рамках данной работы, да и в целом в приложениях вида чат-bot, руководство пользователя может быть реализовано только в виде информации, которую пользователь получает непосредственно во время взаимодействия с ботом в чате. По этой причине, руководство пользователя к telegram боту было реализовано в виде вывода информации при вводе команды `/help` или использовании на панели управления кнопки «Help» в англоязычном варианте или «Помощь» в русскоязычном.

Это руководство имеет два варианта:

1. Первый – находится на главной странице панели и предоставляет информацию о боте в целом, а именно: описывает самого бота; сообщает пользователю какой функционал он предоставляет; на основе чего строятся предложения по пику героев; даёт полный список обрабатываемых им команд; сообщает как пользователь может взаимодействовать с ботом.
2. Второй – находится на странице получения различной информации, связанной с Dota 2. Он сообщает пользователю для чего нужен этот раздел, а также описывает каждую доступную функцию. Этот раздел будет активно расширяться по мере наращивания его функционала.

3.5. Коммерциализация

Цена разрабатываемого бота довольно низкая. Так как он не связан с выполнением никаких бизнес-задач, то цена разрабатываемого программного продукта в основном является ценой содержания бота на сервере.

Также, отдельно стоит рассмотреть функцию подбора в реальном времени. Это осуществимо с помощью API к вебсайту “OpenDota.com”, которое позволяет выгружать информацию о текущем матче в реальном времени. Один запрос к этому API стоит 0.00001$.

Можно подсчитать во сколько обойдётся получать такие запросы на протяжении всего пика. Запрос будет делаться каждую секунду, чтобы ничего не пропустить. Первой идёт стадия запрета, не заканчивается раньше времени и длится 15 секунд. Потом игроки поочерёдно выбирают героев. На выбор каждого даётся 40 секунд, плюс 20 секунд штрафного времени. Последнего игрока в расчёт не берём, так как уже нет смысла отслеживать драфт, когда все герои кроме последнего выбраны. Итого мы имеем 60 секунд на 9 игроков, т.е. 540 секунд максимально допустимого времени, что обойдётся в худшем случае в 0.0054$. Так как все эти пики (кроме стадии запрета) могут проходить моментально.

К стоимости программного продукта также можно добавить стоимость отладки и тестирования данной функции. Предполагается сделать эту функции привилегированной, т.е. доступной тем, кто будет оплачивать её.

Её преимущество в сравнении с Dota Plus, который стоит 4$ в месяц в том, что при расходах на оплату этой функции, за 4$ можно позволить себе 740 игр, что в разы превышает количество игр в месяц обычных игроков. Если брать с пользователя 0.000015$ за каждую секунду такого пика, то это обойдётся ему в худшем случае 0.0081$ за игру, или, другими словами, в худшем случае он получит за те же деньги 493 игры в месяц, что является довольно выгодным для игроков, и эта выгодность увеличивается обратно пропорционально частоте игр конкретного пользователя.

4. Описание результатов тестирования консультанта

Контр-пикер чат-бота был предложен для тестирования друзьям и знакомым разработчика, в сумме 12 человек. По отзывам испытателей можно утверждать, что бот довольно удобен в использовании, а затрачиваемое время на выбор отдельных героев было довольно малым и составляло 1 – 3 секунды на каждого героя, что является заслугой системы выбора героя по его сленговому имени. Пользовали нашли список сленговых имён достаточно исчерпывающим, и ни у кого не возникло ситуации, что он не смог выбрать героя, обращаясь к нему по сленговому имени. Также отмечалось удобство учитывания всех избранных на момент выбора героев, а не конкретно отдельно взятого.

Заключение

В ходе выполнения данной выпускной квалификационной работы были получены следующие результаты:

* Изучены официальная документация для разработчиков на языке Python, множество сторонних источников информации по созданию чат-ботов на языке Python, работа с регулярными выражениями, технологией Xpath, работа с подключаемыми модулями requests и sqlite3.
* Изучены предметная область, существующие решения, игра Dota 2.
* Исследованы возможности API платформы telegram, изучена документация программного интерфейса Telegram Bot API.
* Спроектирован и разработан чат-бот для мессенджера telegram с использованием библиотеки pyTelegramBotAPI для работы с API этого мессенджера.

Список литературы

1. Библиотека для работы с Telegram API. Режим доступа:

<https://pypi.org/project/pyTelegramBotAPI/#general-use-of-the-api>

1. С. Наоми – «Python. Экспресс-курс. 3-е изд.»
2. Swaroop Chitlur – «A Byte of Python»
3. Официальная документация Python. Режим доступа: <https://docs.python.org/3/>
4. Полезный гайд по работе с библиотекой pytelegrambotapi. Режим доступа: <https://habr.com/ru/post/448310/>
5. Chat Bots Aren’t a Fad. They’re a Revolution. Chris Messina. Режим доступа: [https://medium.com/chris-messina/the-chat-bot-revolution-is-upon-us-f3fa9e0b380](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://medium.com/chris-messina/the-chat-bot-revolution-is-upon-us-f3fa9e0b380%26amp;sa%3DD%26amp;ust%3D1592125359460000&sa=D&ust=1592125359771000&usg=AFQjCNGVxLUNABnpfBlwflxTzSYUoInofQ)
6. Matthes E. Python Crash Course: A Hands-On, Project-Based Introduction to Programming – 2nd ed. No Starch Press, 2019, pp 78 - 374.
7. Biezly D., Jones B. Python Cookbook – 3rd ed. O'Reilly Media, 2013, pp.20-611.
8. Библиотека Requests: быстрый старт. Режим доступа:

<http://m05quit0.ru/dev/biblioteka-requests-bystryj-start/>

1. eternnoir/pyTelegramBotAPI. Режим доступа:

<https://github.com/eternnoir/pyTelegramBotAPI>

1. Почему Python? Режим доступа:

<https://khashtamov.com/ru/whypython/>

1. Как заработать на создании чат-ботов: опыт белорусского стартапа BotCube. Режим доступа: <https://dev.by/lenta/main/kak-zarabotat-na-sozdaniichat-botov-opyt-belorusskogo-startapa>
2. Технологии создания и применения чат ботов. Режим чтения:

<https://cyberleninka.ru/article/v/tehnologii-sozdaniya-i-primeneniya-chatbotov>

1. Рассуждения другого разработчика о программе контр-пикере для Dota 2. Режим доступа: <https://dota2.ru/blogs/post/3332-moya-novaya-programma-dlya-kontrpikov-princip-ee-raboty-i-nemnogo-rezul/>
2. Open Dota 2 API. Режим доступа:

<https://docs.opendota.com/#section/Introduction>